

# UNCERTAINTY IN ARTIFICIAL INTELLIGENCE





# Uncertainty in Artificial Intelligence

Proceedings of the Thirty-Fourth Conference (2018)

August 6-10, 2018, Monterey, California, USA

**Edited by**

Amir Globerson, Google Inc. and Tel Aviv University  
Ricardo Silva, University College London and The Alan Turing  
Institute

**General Chairs**

Gal Elidan, Google Inc. and The Hebrew University, Israel  
Kristian Kersting, TU Darmstadt, Germany

**Sponsored by**

Google Inc., Microsoft Research, Uber, Disney Research, Berg  
Health, Artificial Intelligence Journal, The Alan Turing Institute,  
Facebook, Noodle.ai

**AUAI Press Corvallis, Oregon**

Cover design © Matt J. Kusner.

Published by AUAI Press for  
Association for Uncertainty in Artificial Intelligence  
<http://auai.org>

Editorial Office:  
P.O. Box 866  
Corvallis, Oregon 97339  
USA

Copyright © 2018 by AUAI Press  
All rights reserved  
Printed in the United States of America

No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means—electronic, mechanical, photocopying, recording, or otherwise—without the prior written permission of the publisher.

ISBN 978-0-9966431-3-9

# Contents

<b>Preface</b>	<b>vii</b>
<b>Organizing Committee</b>	<b>ix</b>
<b>Best Reviewer Awards</b>	<b>xi</b>
<b>Acknowledgments</b>	<b>xiii</b>
<b>Sponsors</b>	<b>xxi</b>
<b>Best Paper Awards</b>	<b>xxiii</b>
<b>1 Proceedings</b>	<b>1</b>
Testing for Conditional Mean Independence with Covariates through Martingale Difference Divergence. <i>Ze Jin, Xiaohan Yan, David S. Matteson</i> . . . . .	1
Analysis of Thompson Sampling for Graphical Bandits Without the Graphs. <i>Fang Liu, Zizhan Zheng, Ness Shroff</i> . . . . .	13
Structured nonlinear variable selection. <i>Magda Gregorova, Alexandros Kalousis, Stephane Marchand-Maillet</i> . . . . .	23
Identification of Strong Edges in AMP Chain Graphs. <i>Jose M. Peña</i> . . . . .	33
A Univariate Bound of Area Under ROC. <i>Siwei Lyu, Yiming Ying</i> . . . . .	43
Efficient Bayesian Inference for a Gaussian Process Density Model. <i>Christian Donner, Manfred Opper</i> . . . . .	53
Comparing Direct and Indirect Temporal-Difference Methods for Estimating the Variance of the Return. <i>Craig Sherstan, Dylan R. Ashley, Brendan Bennett, Kenny Young, Adam White, Martha White, Richard S. Sutton</i> . . . . .	63
How well does your sampler really work? <i>Ryan Turner, Brady Neal</i> . . . . .	73
Learning Deep Hidden Nonlinear Dynamics from Aggregate Data. <i>Yisen Wang, Bo Dai, Lingkai Kong, Sarah Monazam Erfani, James Bailey, Hongyuan Zha</i>	83
Revisiting differentially private linear regression: optimal and adaptive prediction & estimation in unbounded domain. <i>Yu-Xiang Wang</i> . . . . .	93
Imaginary Kinematics. <i>Sabina Marchetti, Alessandro Antonucci</i> . . . . .	104
From Deterministic ODEs to Dynamic Structural Causal Models. <i>Paul K. Rubenstein, Stephan Bongers, Joris M. Mooij, Bernhard Schölkopf</i> . . . . .	114
Frank-Wolfe Optimization for Symmetric-NMF under Simplicial Constraint. <i>Han Zhao, Geoff Gordon</i> . . . . .	124

Learning Time Series Segmentation Models from Temporally Imprecise Labels. <i>Roy Adams, Benjamin M. Marlin</i> . . . . .	135
Multi-Target Optimisation via Bayesian Optimisation and Linear Programming. <i>Alistair Shilton, Santu Rana, Sunil Gupta, Svetha Venkatesh</i> . . . . .	145
Stochastic Learning for Sparse Discrete Markov Random Fields with Controlled Gradient Approximation Error. <i>Sinong Geng, Zhaobin Kuang, Jie Liu, Stephen Wright, David Page</i> . . . . .	156
Active Information Acquisition for Linear Optimization. <i>Shuran Zheng, Bo Waggoner, Yang Liu, Yiling Chen</i> . . . . .	167
Transferable Meta Learning Across Domains. <i>Bingyi Kang, Jiashi Feng</i> . . . . .	177
Learning the Causal Structure of Copula Models with Latent Variables. <i>Ruifei Cui, Perry Groot, Moritz Schauer, Tom Heskes</i> . . . . .	188
$f_{BGD}$ : Learning Embeddings From Positive Unlabeled Data with BGD. <i>Fajie YUAN, Xin Xin, Xiangnan He, Guibing Guo, Weinan Zhang, CHUA Tat-Seng, Joemon Jose</i> . . . . .	198
Soft-Robust Actor-Critic Policy-Gradient. <i>Esther Derman, Daniel J Mankowitz, Timothy A Mann, Shie Mannor</i> . . . . .	208
Constant Step Size Stochastic Gradient Descent for Probabilistic Modeling. <i>Dmitry Babichev, Francis Bach</i> . . . . .	219
Discrete Sampling using Semigradient-based Product Mixtures. <i>Alkis Gotovos, Hamed Hassani, Andreas Krause, Stefanie Jegelka</i> . . . . .	229
Combining Knowledge and Reasoning through Probabilistic Soft Logic for Image Puzzle Solving. <i>Somak Aditya, Yezhou Yang, Chitta Baral, Yiannis Aloimonos</i> . . . . .	238
Nesting Probabilistic Programs. <i>Tom Rainforth</i> . . . . .	249
Scalable Algorithms for Learning High-Dimensional Linear Mixed Models. <i>Zilong Tan, Kimberly Roche, Xiang Zhou, Sayan Mukherjee</i> . . . . .	259
Constraint-based Causal Discovery for Non-Linear Structural Causal Models with Cycles and Latent Confounders. <i>Patrick Forré, Joris M. Mooij</i> . . . . .	269
Marginal Weighted Maximum Log-likelihood for Efficient Learning of Perturb-and-Map models. <i>Tatiana Shpakova, Francis Bach, Anton Osokin</i> . . . . .	279
Variational Inference for Gaussian Processes with Panel Count Data. <i>Hongyi Ding, Young Lee, Issei Sato, Masashi Sugiyama</i> . . . . .	290
A unified probabilistic model for learning latent factors and their connectivities from high- dimensional data. <i>Ricardo Pio Monti, Aapo Hyvarinen</i> . . . . .	300
Improved Stochastic Trace Estimation using Mutually Unbiased Bases. <i>Jack Fitzsimons, Michael Osborne, Stephen Roberts, Joseph Fitzsimons</i> . . . . .	310
Unsupervised Multi-view Nonlinear Graph Embedding. <i>Jiaming Huang, Zhao Li, Vincent W. Zheng, Wen Wen, Yifan Yang, Yuanmi Chen</i> . . . . .	319
Graph-based Clustering under Differential Privacy. <i>Rafael Pinot, Anne Morvan, Florian Yger, Cedric Gouy-Pailler, Jamal Atif</i> . . . . .	329
GaAN: Gated Attention Networks for Learning on Large and Spatiotemporal Graphs. <i>Jiani Zhang, Xingjian Shi, Junyuan Xie, Hao Ma, Irwin King, Dit-yan Yeung</i> . . . . .	339
Causal Learning for Partially Observed Stochastic Dynamical Systems. <i>Søren Wengel Mogensen, Daniel Malinsky, Niels Richard Hansen</i> . . . . .	350
Variational zero-inflated Gaussian processes with sparse kernels. <i>Pashupati Hegde, Markus Heinonen, Samuel Kaski</i> . . . . .	361
KBIn: End-to-End Learning of Knowledge Base Representations with Latent, Relational, and Numerical Features. <i>Alberto Garcia-Duran, Mathias Niepert</i> . . . . .	372
Probabilistic AND-OR Attribute Grouping for Zero-Shot Learning. <i>Yuval Atzmon, Gal Chechik</i> . . . . .	382

Sylvester Normalizing Flows for Variational Inference. <i>Rianne van den Berg, Leonard Hasenclever, Jakub Tomczak, Max Welling</i> . . . . .	393
Holistic Representations for Memorization and Inference. <i>Yunpu Ma, Marcel Hildebrandt, Volker Tresp, Stephan Baier</i> . . . . .	403
Simple and practical algorithms for $\ell_p$ -norm low-rank approximation. <i>Anastasios Kyrillidis</i> . . . . .	414
Quantile-Regret Minimisation in Infinitely Many-Armed Bandits. <i>Arghya Roy Chaudhuri, Shivaram Kalyanakrishnan</i> . . . . .	425
Variational Inference for Gaussian Process Models for Survival Analysis. <i>Minyoung Kim, Vladimir Pavlovic</i> . . . . .	435
A Cost-Effective Framework for Preference Elicitation and Aggregation. <i>Zhibing Zhao, Haoming Li, Junming Wang, Jeffrey O. Kephart, Nicholas Mattei, Hui Su, Lirong Xia</i> . . . . .	446
Incremental Learning-to-Learn with Statistical Guarantees. <i>Giulia Denevi, Carlo Ciliberto, Dimitris Stamos, Massimiliano Pontil</i> . . . . .	457
Bandits with Side Observations: Bounded vs. Logarithmic Regret. <i>Rmy Degenne, Evrard Garcelon, Vianney Perchet</i> . . . . .	467
Sampling and Inference for Beta Neutral-to-the-Left Models of Sparse Networks. <i>Benjamin Bloem-Reddy, Adam Foster, Emile Mathieu, Yee Whye Teh</i> . . . . .	477
Clustered Fused Graphical Lasso. <i>Yizhi Zhu, Oluwasanmi Koyejo</i> . . . . .	487
Unsupervised Learning of Latent Physical Properties Using Perception-Prediction Networks. <i>David Zheng, Vinson Luo, Jiajun Wu, Joshua Tenenbaum</i> . . . . .	497
Subsampled Stochastic Variance-Reduced Gradient Langevin Dynamics. <i>Difan Zou, Pan Xu, Quanquan Gu</i> . . . . .	508
Finite-State Controllers of POMDPs using Parameter Synthesis. <i>Sebastian Junges, Nils Jansen, Ralf Wimmer, Tim Quatmann, Leonore Winterer, Joost-Pieter Katoen, Bernd Becker</i> . . . . .	519
Identification of Personalized Effects Associated With Causal Pathways. <i>Ilya Shpitser, Eli Sherman</i> . . . . .	530
Fast Counting in Machine Learning Applications. <i>Subhadeep Karan, Matthew Eichhorn, Blake Hurlburt, Grant Iraci, Jaroslaw Zola</i> . . . . .	540
A Dual Approach to Scalable Verification of Deep Networks. <i>Krishnamurthy Dvijotham, Robert Stanforth, Sven Gowal, Timothy Mann, Pushmeet Kohli</i>	550
Understanding Measures of Uncertainty for Adversarial Example Detection. <i>Lewis Smith, Yarin Gal</i> . . . . .	560
Causal Discovery in the Presence of Measurement Error. <i>Tineke Blom, Anna Klimovskaia, Sara Magliacane, Joris M. Mooij</i> . . . . .	570
IDK Cascades: Fast Deep Learning by Learning not to Overthink. <i>Xin Wang, Yujia Luo, Daniel Crankshaw, Alexey Tumanov, Fisher Yu, Joseph E. Gonzalez</i>	580
Learning Fast Optimizers for Contextual Stochastic Integer Programs. <i>Vinod Nair, Dj Dvijotham, Iain Dunning, Oriol Vinyals</i> . . . . .	591
Differential Analysis of Directed Networks. <i>Min Ren, Dabao Zhang</i> . . . . .	601
Sparse-Matrix Belief Propagation. <i>Reid Bixler, Bert Huang</i> . . . . .	611
Sequential Learning under Probabilistic Constraints. <i>Amirhossein Meisami, Henry Lam, Chen Dong, Abhishek Pani</i> . . . . .	621
Abstraction Sampling in Graphical Models. <i>Filjor Broka, Rina Dechter, Alexander Ihler, Kalev Kask</i> . . . . .	632
Meta Reinforcement Learning with Latent Variable Gaussian Processes. <i>Steindor Saemundsson, Katja Hofmann, Marc Peter Deisenroth</i> . . . . .	642
Non-Parametric Path Analysis in Structural Causal Models. <i>Junzhe Zhang, Elias Bareinboim</i> . . . . .	653

Stochastic Layer-Wise Precision in Deep Neural Networks. <i>Griffin Lacey, Graham W. Taylor, Shawki Areibi</i> . . . . .	663
Estimation of Personalized Effects Associated With Causal Pathways. <i>Razieh Nabi, Phyllis Kanki, Ilya Shpitser</i> . . . . .	673
High-confidence error estimates for learned value functions. <i>Touqir Sajed, Wesley Chung, Martha White</i> . . . . .	683
Combinatorial Bandits for Incentivizing Agents with Dynamic Preferences. <i>Tanner Fiez, Shreyas Sekar, Liyuan Zheng, Lillian Ratliff</i> . . . . .	693
Sparse Multi-Prototype Classification. <i>Vikas K. Garg, Lin Xiao, Ofer Dekel</i> . . . . .	704
Fast Stochastic Quadrature for Approximate Maximum-Likelihood Estimation. <i>Nico Piatkowski, Katharina Morik</i> . . . . .	715
Finite-sample Bounds for Marginal MAP. <i>Qi Lou, Rina Dechter, Alexander Ihler</i> . . . . .	725
Acyclic Linear SEMs Obey the Nested Markov Property. <i>Ilya Shpitser, Robin Evans, Thomas S. Richardson</i> . . . . .	735
A Unified Particle-Optimization Framework for Scalable Bayesian Sampling. <i>Changyou Chen, Ruiyi Zhang, Wenlin Wang, Bai Li, Liqun Chen</i> . . . . .	746
An Efficient Quantile Spatial Scan Statistic for Finding Unusual Regions in Continuous Spatial Data with Covariates. <i>Travis Moore, Weng-Keen Wong</i> . . . . .	756
Stable Gradient Descent. <i>Yingxue Zhou, Sheng Chen, Arindam Banerjee</i> . . . . .	766
Learning to select computations. <i>Frederick Callaway, Sayan Gul, Paul M. Krueger, Thomas L. Griffiths, Falk Lieder</i> . . . . .	776
Per-decision Multi-step Temporal Difference Learning with Control Variates. <i>Kristopher De Asis, Richard S. Sutton</i> . . . . .	786
The Indian Buffet Hawkes Process to Model Evolving Latent Influences. <i>Xi Tan, Vinayak Rao, Jennifer Neville</i> . . . . .	795
Battle of Bandits. <i>Aadirupa Saha, Aditya Gopalan</i> . . . . .	805
Adaptive Stochastic Dual Coordinate Ascent for Conditional Random Fields. <i>Rémi Le Priol, Alexandre Piché, Simon Lacoste-Julien</i> . . . . .	815
Adaptive Stratified Sampling for Precision-Recall Estimation. <i>Ashish Sabharwal, Yexiang Xue</i> . . . . .	825
Fast Kernel Approximations for Latent Force Models and Convolved Multiple-Output Gaussian processes. <i>Cristian Guarnizo, Mauricio Álvarez</i> . . . . .	835
Fast Policy Learning through Imitation and Reinforcement. <i>Ching-An Cheng, Xinyan Yan, Nolan Wagener, Byron Boots</i> . . . . .	845
Hyperspherical Variational Auto-Encoders. <i>Tim Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, Jakub M. Tomczak</i> . . . . .	856
Dissociation-Based Oblivious Bounds for Weighted Model Counting. <i>Li Chou, Wolfgang Gatterbauer, Vibhav Gogate</i> . . . . .	866
Averaging Weights Leads to Wider Optima and Better Generalization. <i>Pavel Izmailov, Dmitrii Podoprikin, Timur Garipov, Dmitry Vetrov, Andrew Gordon Wilson</i> . . . . .	876
Block-Value Symmetries in Probabilistic Graphical Models. <i>Gagan Madan, Ankit Anand, Mausam, Parag Singla</i> . . . . .	886
Max-margin learning with the Bayes factor. <i>Rahul G. Krishnan, Arjun Khandelwal, Rajesh Ranganath, David Sontag</i> . . . . .	896
Densified Winner Take All (WTA) Hashing for Sparse Datasets. <i>Beidi Chen, Anshumali Shrivastava</i> . . . . .	906
Lifted Marginal MAP Inference. <i>Vishal Sharma, Noman Ahmed Sheikh, Happy Mittal, Vibhav Gogate, Parag Singla</i> . . . . .	917



PAC-Reasoning in Relational Domains. <i>Ondrej Kuzelka, Yuyi Wang, Jesse Davis, Steven Schockaert</i> . . . . .	927
Pure Exploration of Multi-Armed Bandits with Heavy-Tailed Payoffs. <i>Xiaotian Yu, Han Shao, Michael R. Lyu, Irwin King</i> . . . . .	937
Counterfactual Normalization: Proactively Addressing Dataset Shift Using Causal Mechanisms. <i>Adarsh Subbaswamy, Suchi Saria</i> . . . . .	947
Decentralized Planning for Non-dedicated Agent Teams with Submodular Rewards in Uncertain Environments. <i>Pritee Agrawal, Pradeep Varakantham, William Yeoh</i> . . . . .	958
A Forest Mixture Bound for Block-Free Parallel Inference. <i>Neal Lawton, Greg Ver Steeg, Aram Galstyan</i> . . . . .	968
Causal Identification under Markov Equivalence. <i>Amin Jaber, Jiji Zhang, Elias Bareinboim</i> . . . . .	978
The Variational Homoencoder: Learning to learn high capacity generative models from few examples. <i>Luke B. Hewitt, Maxwell I. Nye, Andreea Gane, Tommi Jaakkola, Joshua B. Tenenbaum</i> . . . . .	988
Probabilistic Collaborative Representation Learning for Personalized Item Recommendation. <i>Aghiles Salah, Hady W. Lauw</i> . . . . .	998
Reforming Generative Autoencoders via Goodness-of-Fit Hypothesis Testing. <i>Aaron Palmer, Dipak Dey, Jinbo Bi</i> . . . . .	1009
Towards Flatter Loss Surface via Nonmonotonic Learning Rate Scheduling. <i>Sihyeon Seong, Yegang Lee, Youngwook Kee, Dongyoon Han, Junmo Kim</i> . . . . .	1020
A Lagrangian Perspective on Latent Variable Generative Models. <i>Shengjia Zhao, Jiaming Song, Stefano Ermon</i> . . . . .	1031
Bayesian optimization and attribute adjustment. <i>Stephan Eismann, Daniel Levy, Rui Shu, Stefan Bartzsch, Stefano Ermon</i> . . . . .	1042
Join Graph Decomposition Bounds for Influence Diagrams. <i>Junkyu Lee, Alexander Ihler, Rina Dechter</i> . . . . .	1053
Causal Discovery with Linear Non-Gaussian Models under Measurement Error: Structural Identifiability Results. <i>Kun Zhang, Mingming Gong, Joseph Ramsey, Kayhan Batmanghelich, Peter Spirtes, Clark Glymour</i> . . . . .	1063



# Preface

The Conference on Uncertainty in Artificial Intelligence (UAI) is the premier international conference on research related to representation, inference, learning and decision making in the presence of uncertainty within the field of Artificial Intelligence. This volume contains all papers that were accepted for the 34th UAI Conference, held in Monterey, California, from August 6 to 10, 2018.

Papers appearing in this conference were subjected to a rigorous review process. A total of 337 papers were reviewed by at least 3 reviewers each. Of these, 104 papers were accepted, for an acceptance rate of close to 31%. We are very grateful to the program committee and senior program committee members for their diligent efforts. We are confident that the proceedings, like past UAI conference proceedings, will become an important archival reference for the field.

We are pleased to announce that the Best Paper Award was awarded to Krishnamurthy Dvijotham, Robert Stanforth, Sven Gowal, Timothy Mann and Pushmeet Kohli for their paper “A Dual Approach to Scalable Verification of Deep Networks.” The Best Student Paper Award was awarded to Amin Jaber, Jiji Zhang and Elias Bareinboim for their paper “Causal Identification under Markov Equivalence.” We are grateful to the members of the best paper committee: Thomas Richardson, Ilya Shpitser and David Sontag.

In addition to the presentation of technical papers, we were very pleased to have four distinguished invited speakers at UAI 2018: Michael C. Frank (Stanford University), Joelle Pineau (McGill University and Facebook), Stuart Russell (UC Berkeley) and Raquel Urtasun (University of Toronto and Uber).

The UAI 2018 tutorials program, chaired by Shakir Mohamed, consisted of four invited tutorials: “Tackling Data Scarcity in Deep Learning” by Anima Anandkumar (Caltech and Amazon AI) and Zachary Lipton (Carnegie Mellon University), “Recent Progress in the Theory of Deep Learning” by Tengyu Ma (Facebook and Stanford University), “Bayesian Approaches for Blackbox Optimization” by Matt Hoffman (DeepMind), and “Machine Reading” by Sebastian Riedel (UCL), Johannes Welbl (UCL) and Dirk Weissenborni (German Research Center for Artificial Intelligence).

UAI 2018 also hosted three workshops, chaired by Yarin Gal: Safety, Risk and Uncertainty in RL organized by Emma Brunskill (Stanford), Audrey Durand (McGill), Vincent Franois (McGill), Daniel (Zhaohan) Guo (CMU), Joelle Pineau (McGill), and Guillaume Rabusseau (McGill); The 7th Causal Inference Workshop organized by Bryant Chen (IBM), Panos Toulis (University of Chicago) and Alexander Volfovsky (Duke University); and Uncertainty in Deep Learning organized by Andrew Wilson (Cornell), Balaji Lakshminarayanan (Deepmind), Dustin Tran (Columbia, Google) and Matt Hoffman (Google).

Following the success of last years event, UAI 2018 continued to hold MLTrain, a hands-on training session on modern machine learning technologies organized by Nikolaos Vasiloglou. This year we partnered with the Linqs team from UC Santa Cruz and with the Pyro team from Uber ATG and teach UAI participants probabilistic programming. We covered the fundamentals of modeling with Probabilistic Soft Logic a new language that redefines the way we blend human expertise with machine learning.

*Amir Globerson and Ricardo Silva (Program Co-Chairs)  
Gal Elidan and Kristian Kersting (General Chair)*



# Organizing Committee

## **General Chairs**

Gal Elidan, Google Inc. and The Hebrew University, Israel  
Kristian Kersting, TU Darmstadt, Germany

## **Program Chairs**

Amir Globerson, Google Inc. and Tel Aviv University  
Ricardo Silva, University College London and The Alan Turing Institute

## **Workshops Chair**

Yarin Gal, University of Oxford

## **Tutorial Chair**

Shakir Mohamed, DeepMind, UK

## **Sponsoring and Social Media Chair**

Nikolaos Vasiloglou, Ismion, USA  
Alexander Ihler, University of California, Irvine, USA

## **Local Arrangements Chair**

Stefano Ermon, Stanford University

## **Proceedings Chair**

Matt J. Kusner, University of Oxford and The Alan Turing Institute

## **Registration Chair**

Alejandro Molina, TU Darmstadt

## **Webmaster and OpenReview Chair**

Yin Cheng Ng, University College London

## **OpenReview Team**

Michael Spector, University of Massachusetts Amherst



# Best Reviewer Awards

Jacob Andreas  
Elias Bareinboim  
Olivier Buffet  
Tom Claassen  
Aditya Deshpande  
Justin Domke  
Robin Evans  
Yuan-Ting Hu  
Daniel Lowd  
Daniel Malinsky  
Brandon Malone  
Nick Ruoizzi  
Scott Sanner  
Ilya Shpitser  
Berk Ustun  
Greg ver Steeg  
Roi Weiss  
Thomas Werner  
Raymond Yeh





# Acknowledgments

The success of UAI depends greatly on the efforts of many individuals who volunteer their time to provide expert and detailed reviews of submitted papers. In particular, the Program Committee and Senior Program Committee for UAI 2018 were responsible for generating reviews and recommendations for the submissions to the conference. Each submitted paper was reviewed by at least 3 members of the Program Committee. The Senior Program Committee then assessed the individual reviews for each paper, moderated discussion among Program Committee members if needed, and generated meta-reviews and recommendations for the program chairs. We are extremely grateful for the efforts of all of the individuals listed below.

## Senior Program Committee

Adrian Weller	University of Cambridge
Alex Schwing	University of Illinois, Urbana Champaign
Andriy Mnih	DeepMind
Andrew Wilson	Cornell University
Cassio de Campos	Queen's University Belfast
Dale Schuurmans	University of Alberta
Daniel Roy	University of Toronto
David Sontag	MIT
Doina Precup	McGill University
Fabio Cozman	Universidade de Sao Paulo
Fabio Cuzzolin	Oxford Brookes University
Guy Van den Broek	University of California, Los Angeles
Honglak Lee	Google
Jennifer Neville	Purdue University
Jonathan Berant	Tel Aviv University
Jonas Peters	University of Copenhagen
Jun Zhu	Tsinghua University
Kun Zhang	Carnegie Mellon University
Le Song	College of Computing, Georgia Institute of Technology
Marloes Maathuis	Swiss Federal Institute of Technology
Martin Zinkevich	Google
Michalis Titsias	Athens University of Economics and Business
Ofer Meshi	Google
Peter Grünwald	Leiden University, Dept. of Mathematics
Pradeep Ravikumar	Carnegie Mellon University
Roger Grosse	Department of Computer Science, University of Toronto
Sanjoy Dasgupta	UC San Diego
Sivan Sabato	Ben Gurion University of the Negev
Sinead Williamson	University of Texas, Austin
Steffen Grunewalder	Lancaster University
Sujay Sanghavi	University of Texas, Austin
Thomas Richardson	University of Washington
Tamir Hazan	Technion
Vibhav Gogate	University of Texas, Dallas
Yaron Singer	Harvard University
Yishay Mansour	Tel Aviv University

## Program Committee

Aditi Raghunathan	Stanford
Aditya Deshpande	University of Illinois, Urbana Champaign
Alan Malek	Massachusetts Institute of Technology
Alessandro Antonucci	IDSIA
Alessio Benavoli	IDSIA
Alexandre Bouchard-Cote	University of British Columbia
Ali Ghodsi	University of Waterloo
Alon Brutzkus	Tel Aviv University
Anders Madsen	Aalborg University
Andrey Kolobov	Microsoft
Angel Lee	8 Du Box Inc
Angelika Kimmig	Cardiff University
Antti Hyttinen	University of Helsinki, Helsinki Institute for Information Technology
Aonan Zhang	Columbia
April Liu	Shanghai University of Finance and Economics
Aritanan Gruber	University of ABC, São Paulo State
Arjen Hommersom	Open University of the Netherlands
Arthur Choi	University of California, Los Angeles
Arvind Agarwal	International Business Machines
Aurelie Lozano	IBM Research
Balaji Lakshminarayanan	Google DeepMind
Ben Athiwaratkun	Cornell University
Benjamin Guedj	INRIA
Berk Ustun	Harvard University
Bernie Wang	Tufts University
Biwei Huang	Carnegie Mellon University
Brandon Malone	NEC
Branislav Kveton	Adobe Research
Brian Rutenber	Charles River Analytics
Brian Westerweel	Radboud University
Changhe Yuan	Queens College
Changyou Chen	State University of New York, Buffalo
Chen Liang	Northwestern
Chengtao Li	Massachusetts Institute of Technology
Cho-Jui Hsieh	University of California, Davis
Chongxuan Li	Tsinghua University
Chris Maddison	University of Oxford, DeepMind
Christian Shelton	University of California, Riverside
Christoph Dann	Carnegie Mellon University
Christopher Srinivasa	University of Toronto
Christopher Tosh	University of California, San Diego
Ciara Pike-Burke	Lancaster University
Colin Graber	University of Illinois, Urbana Champaign
Cong Chen	City University of New York
Cory Butz	University of Regina
Creagh Briercliffe	University of British Columbia
Dan Garber	Technion
Daniel Kumor	Purdue University
Daniel Lowd	University of Oregon
Daniel Malinsky	Johns Hopkins University
Danielle Belgrave	Microsoft
David Alvarez-Melis	Massachusetts Institute of Technology
David Arbour	Facebook
David Barber	University College London
David Belanger	University of Massachusetts, Amherst
David Jensen	University of Massachusetts, Amherst
David Page	University of Wisconsin, Madison
David Pennock	Microsoft

David Pynadath	Massachusetts Institute of Technology
Debarun Bhattacharjya	Stanford University
Denis Mau	Universidade de Sao Paulo
Dmitrii Podoprikin	Lomonosov Moscow State University
Dominik Rothenhaeusler	Swiss Federal Institute of Technology
Dustin Tran	Columbia University
Elad Eban	Google
Elad Mezuman	Google
Eleni Triantafillou	University of Toronto
Elias Bareinboim	Purdue University
Emilija Perkovic	Swiss Federal Institute of Technology
Erik Ordentlich	Oath
Eunho Yang	Korea Advanced Institute of Science & Technology
Eyal Amir	University of Illinois, Urbana Champaign
Fabio Stella	University of Milan-Bicocca
Farhad Anaraki	University of Colorado, Boulder
Francisco J. R. Ruiz	University of Cambridge
Frederic Sala	Stanford University
Frederick Eberhardt	California Institute of Technology
Fredrik Daniel Johansson	Massachusetts Institute of Technology
Gavin Whitaker	University College London
Geoff Pleiss	Cornell University
George Tucker	Google
Gerardo Simari	Universidad Nacional del Sur
Giorgio Corani	IDSIA
Giorgos Borboudakis	University of Crete
Greg Ver Steeg	Information Sciences Institute
Guodong Zhang	University of Toronto
Hang Su	Tsinghua University
Hao Zhang	Carnegie Mellon University
Hong Chang	Institute of Computing Technology, Chinese Academy of Sciences
Hyunjik Kim	DeepMind
Idan Schwartz	Technion
Ilya Shpitser	Johns Hopkins University
Ioannis Tsamardinos	University of Crete
Jack Baker	University of Edinburgh
Jacob Andreas	University of California Berkeley
Jacob R Gardner	Cornell University
James Cussens	University of York
James Foulds	University of Maryland, Baltimore County
James M. Robins	Harvard School of Public Health
James Robins	Harvard University
Jesse Hostetler	Oregon State University
Jianfei Chen	Tsinghua University
Jiji Zhang	Lingnan University
Jinwoo Shin	KAIST
Jirka Vomlel	Czech Academy of Sciences
Joe Suzuki	Osaka University
Johannes Textor	Radboud Universiteit Nijmegen
Jonas Mueller	Massachusetts Institute of Technology
Jonas Vlasselaer	KU Leuven
José A. Gámez	University of Castilla - La Mancha
José M. Puerta	University of Castilla - La Mancha
Jose Pea	Linköping University
Joseph Ramsey	Carnegie Mellon University
Julian McAuley	UC San Diego
Junhyuk Oh	University of Michigan
Junxiang Chen	Northeastern University
Junzhe Zhang	Purdue University
Justin Domke	University of Massachusetts, Amherst

Kacper Chwialkowski	University College London
Kar Wai Lim	National University of Singapore
Karan Goel	Carnegie Mellon University
Karthika Mohan	University of California Berkeley
Karthikeyan Shanmugam	IBM Research
Kevin Small	Amazon
Kieran Campbell	University of British Columbia
Kim-Leng Poh	National University of Singapore
L. Enrique Sucar	IIE
Lajanugen Logeswaran	University of Michigan
Lan Du	Monash University
Leonard Poon	The Education University of Hong Kong
Linbo Wang	Harvard University
Linda C. van der Gaag	Utrecht University
Lloyd T Elliott	University of Oxford
Lluis Godo	Artificial Intelligence Research Institute
M. Julia Flores	University of Castilla - La Mancha
Malik Magdon-Ismaïl	Rensselaer Polytechnic Institute
Manuel Gomez-Olmedo	University of Granada
Manuel Luque	UNED
Marco Valtorta	University of South Carolina
Maria Vanina Martinez	Universidad Nacional del Sur en Bahia Blanca
Mario Casaro	Onera - The French Aerospace Lab
Mark Bartlett	University of York
Mark van der Wilk	Prowler.io
Martin Mladenov	TU Dortmund University, Germany
Maruan Al-Shedivat	Carnegie Mellon University
Matej Balog	University of Cambridge
Mathias Niepert	NEC
Mathieu Sinn	University of Luebeck
Matin Plajner	Czech Academy of Sciences
Matthew Johnson	Google
Maurice Diesendruck	University of Texas, Austin
Mengye Ren	University of Toronto
Michael Gimelfarb	University of Toronto
Michael Zhang	University of Texas, Austin
Mikko Koivisto	University of Helsinki
Milan Studený	UTIA, Czech Academy of Sciences
Ming Yuan	Columbia University
Mingming Gong	University of Pittsburgh
Minos Garofalakis	Yahoo Research
Misha Chertkov	LANL
Mladen Kolar	University of Chicago
Mohammad Ali Javidian	University of South Carolina
Mohammad Emtiyaz Khan	RIKEN
Nahla Ben Amor	Institut Suprieur de Gestion
Nan Ye	Queensland University of Technology
Narges Razavian	New York University
Nicholas Ruoizzi	University of Texas, Dallas
Nicolas Drougard	ISAE - Supaero
Nicolo Colombo	University College London
Niklas Pfister	Swiss Federal Institute of Technology
Nimar Arora	University of California Berkeley
Nir Rosenfeld	Harvard University
Ole Mengshoel	Carnegie Mellon University
Olivier Buffet	INRIA
Oluwasanmi O Koyejo	University of Illinois, Urbana Champaign
Or Sharir	The Hebrew University of Jerusalem
Pang Wei Koh	Stanford University
Paola Vicard	Università Roma Tre

Paul Weng	Shanghai Jiaotong University
Pavel Izmailov	Cornell University
Pengtao Xie	Carnegie Mellon University
Pengzhi Gao	Petuum Inc
Philipp Geiger	Max Planck Institute for Intelligent Systems
Philippe Leray	University of Nantes
Piyush Rai	IIT Kanpur
Qi Lou	University of California, Irvine
Quanquan Gu	University of Virginia
Radu Marinescu	IBM
Rafael Rumi	University of Almeria
Rahul G Krishnan	Massachusetts Institute of Technology
Ran Gilad-Bachrach	Microsoft
Raymond Yeh	University of Illinois, Urbana Champaign
Renjie Liao	University of Toronto
Reza Babanezhad Harikandeh	University of British Columbia
Robert Castelo	Universitat Pompeu Fabra
Robin Evans	University of Oxford
Roi Livni	Princeton University
Roi Weiss	Weizmann Institute
Roland Ramsahai	University of Cambridge
Rong Ge	Duke University
Ru He	Amazon
Ruben Villegas	University of Michigan
Ruichu Cai	Guangdong University of Technology
Ruitong Huang	Borealis AI
Russell Almond	Florida State University
Ruth Urner	York University
Saifuddin Syed	University of British Columbia
Sam Corbett-Davies	Stanford University
Sameer Singh	University of California, Irvine
Samuel Kaski	Helsinki Institute for Information Technology
Sara Magliacane	IBM
Scott Sanner	University of Toronto
Sebastien Destercke	Université de technologie de Compiègne
Shang-Tse Chen	Georgia Institute of Technology
Shay Moran	Institut for Advanced Study, Princeton
Shohei Shimizu	Shiga University
Shuai Li	The Chinese University of Hong Kong
Shuai Li	University of Cambridge
Siamak Ravanbakhsh	University of British Columbia
Silja Renooij	Utrecht University
Sinong Geng	Wisconsin Institute for Discovery
Sohrab Salehi	University of British Columbia
Sriraam Natarajan	Indiana University
Stefano Ermon	Stanford University
Steffen Michels	Radboud University
Stephen Bach	Brown University
Stephen Page	Lancaster University
Steven Holtzen	University of California, Los Angeles
Subhojyoti Mukherjee	IIT Madras
Sumeet Katariya	University of Wisconsin-Madison
Suriya Gunasekar	Toyota Technological Institute at Chicago
Sren Wengel Mogensen	University of Copenhagen
Tahrima Rahman	University of Texas, Dallas
Tameem Adel	University of Cambridge
Taneli Mielikainen	Oath
Teemu Roos	University of Helsinki
Thang D Bui	University of Cambridge
Theofanis Karaletsos	Uber

Thijs van Ommen	Universiteit van Amsterdam
Tianjiao Chu	University of Pittsburgh
Tiberio Caetano	Ambiata
Tingting Zhao	University of British Columbia
Tom Claassen	Radboud University
Tom Heskes	Radboud University
Tomas Singliar	Amazon
Tomas Werner	Czech Technical Univeresity in Prague
Tomer Koren	Google
Tristan de Boer	Radboud University
Truong-Huy Dinh Nguyen	Fordham University
Tyler Lu	Google
Uri Shalit	New York University
Valentin Koch	Radboud University
Vanessa Didelez	Leibniz Institute, University of Bremen
Victor Veitch	Columbia University
Václav Kratochvíl	Czech Academy of Sciences
Wannes Meert	KU Leuven
Wei Ping	Baidu Silicon Valley AI Lab
Weiwei Liu	The University of New South Wales
Wen Wei Loh	Ghent University
William Cheung	Hong Kong Baptist University
William Herlinds	Carnegie Mellon University
Xiangyang Liu	University of Maryland, College Park
Xiaotong Yuan	NUIST
Y. Sam Wang	University of Washington
Yan Liu	University of Southern California
Yang Xiang	University of Guelph
Yasin Abbasi-Yadkori	Adobe
Yexiang Xue	Cornell University
Yifeng Zeng	Teesside University
Yining Wang	Carnegie Mellon University
Yishay Mansour	Tel Aviv University
Yoav Itzhak Wald	Google
Yong Ren	Tsinghua University
Yoon Kim	Harvard University
Yu Zhang	The Hong Kong University of Science and Technology
Yuan Yang	Petuum Inc
Yuan-Ting Hu	University of Illinois, Urbana Champaign
Yuhong Guo	Carleton University
Yujia Li	Google
Yutian Chen	DeepMind
Zachary Lipton	CMU
Zhaobin Kuang	University of Wisconsin-Madison
Zi Wang	Massachusetts Institute of Technology
Zied Eloudi	ISG Tunis

## Additional Acknowledgments

A number of other people have made significant contributions towards making UAI 2018 possible. We acknowledge and thank:

- The OpenReview team:

Andrew McCallum	University of Massachusetts Amherst
Melisa Bok	University of Massachusetts Amherst
Pamela Mandler	University of Massachusetts Amherst
Michael Spector	University of Massachusetts Amherst

- The following student scholarship volunteers:

Hesham Alghodhaifi	University of Michigan
Yuval Atzmon	Bar Ilan University
Rui-Yi Zhang	Duke University
Falorsi Luca	University of Amsterdam
Nicola De Cao	University of Amsterdam
Bingyi Kang	National University of Singapore
Tim Davidon	University of Amsterdam
Min Ren	Purdue University
Pavel Izmailov	Cornell University
Neal Lawton	University of Southern California
Gagan Madan	Indian Institute of Technology Delhi
Neal Jean	Stanford University
Jiaming Song	Stanford University
Pritee Agrawal	Singapore Management University, Singapore
Yisen Wang	Tsinghua University
Yingxue Zhou	University of Minnesota
Junming Wang	Rensselaer Polytechnic Institute
Tanner Fiez	University of Washington
Qi Lou	University of California, Irvine
Thomas Ng	UC Santa Cruz
Aadirupa Saha	Indian Institute of Science Bangalore
Arghya Roy Chaudhuri	Indian Institute of Technology Bombay
Jiani Zhang	The Chinese University of Hong Kong
Sren Wengel Mogensen	University of Copenhagen
Vishal Sharma	Indian Institute of Technology Delhi
Zilong Tan	Duke University
Noman Ahmed Sheikh	Sprinklr
Subhadeed Karan	University at Buffalo
Zhibing Zhao	Rensselaer Polytechnic Institute
Fang Liu	The Ohio State University
Shengjia Zhao	Stanford University





# Sponsors

We gratefully acknowledge the generous support of our sponsors, including support for best paper awards and travel scholarships. Without our sponsors' support it would not be feasible to organize a conference such as UAI 2018 without charging much higher registration fees.

## Gold Sponsors



## Bronze Sponsors



## Startup Sponsor





# Best Paper Awards

## **Best Paper Award**

*A Dual Approach to Scalable Verification of Deep Networks*

Robert Stanforth, Sven Gowal, Timothy Mann and Pushmeet Kohli

## **Best Student Paper Award**

*Causal Identification under Markov Equivalence*

Amin Jaber, Jiji Zhang and Elias Bareinboim



# Proceedings

---

# Testing for Conditional Mean Independence with Covariates through Martingale Difference Divergence

---

**Ze Jin\***

Department of Statistical Science  
Cornell University  
Ithaca, NY 14850

**Xiaohan Yan**

Department of Statistical Science  
Cornell University  
Ithaca, NY 14850

**David S. Matteson†**

Department of Statistical Science  
Cornell University  
Ithaca, NY 14850

## Abstract

A crucial problem in statistics is to decide whether additional variables are needed in a regression model. We propose a new multivariate test to investigate the conditional mean independence of  $Y$  given  $X$  conditioning on some known effect  $Z$ , i.e.,  $E(Y|X, Z) = E(Y|Z)$ . Assuming that  $E(Y|Z)$  and  $Z$  are linearly related, we reformulate an equivalent notion of conditional mean independence through transformation, which is approximated in practice. We apply the martingale difference divergence (Shao and Zhang, 2014) to measure conditional mean dependence, and show that the estimation error from approximation is negligible, as it has no impact on the asymptotic distribution of the test statistic under some regularity assumptions. The implementation of our test is demonstrated by both simulations and a financial data example.

## 1 INTRODUCTION

Testing (conditional) dependence and conditional mean dependence plays an important role in statistics with various applications, including variable selection (Székely and Rizzo, 2014; Park et al., 2015; Zhang et al., 2015; Yan and Bien, 2018), feature screening (Li et al., 2012; Shao and Zhang, 2014; Yan et al., 2017), and graphical models (Gan et al., 2018; Li and McCormick, 2017; Li et al., 2018). Both areas attracted tremendous attention in the last two decades, as datasets have increased in size and dimension. Let  $X \in \mathbb{R}^p$ ,  $Y \in \mathbb{R}^q$ ,  $Z \in \mathbb{R}^r$  be the

three random vectors of interest, and denote pairwise independence by  $\perp$ .

Measures of (conditional) dependence have been extensively studied. Székely et al. (2007) proposed distance covariance (dCov) to capture the non-linear and non-monotone pairwise dependence between  $X$  and  $Y$ , and  $\text{dCov} = 0$  if and only if pairwise independence ( $X \perp Y$ ) holds. Jin and Matteson (2017) extended distance covariance to mutual dependence measures (MDMs), which have been applied to independent component analysis in Jin and Matteson (2018). To capture the conditional dependence between  $X$  and  $Y$  given  $Z$ , Székely and Rizzo (2014) generalized distance covariance to partial distance covariance (pdCov), however,  $\text{pdCov} = 0$  is not equivalent to conditional independence ( $X \perp Y | Z$ ), and neither one implies the other. Wang et al. (2015) extended distance covariance to conditional distance covariance (CDCov) using kernel estimators, and  $\text{CDCov} = 0$  if and only if conditional independence holds. Under a linear assumption between  $X, Y$  and  $Z$ , Fan et al. (2015) converted testing conditional independence to testing independence, and applied distance covariance to measure the dependence of estimated variables. Moreover, inter-temporal conditional dependence is known as Granger causality in time series analysis. Hiemstra and Jones (1994), Su and White (2007), and Chen and Hong (2012) each introduced non-parametric tests for non-linear Granger causality based on conditional probabilities and characteristic functions.

Likewise, various measures of conditional mean dependence have been broadly developed as well. Testing the conditional mean independence of  $Y$  given  $X$ , i.e.,

$$H_0 : E(Y|X) = E(Y) \text{ a.s.}, \quad H_A : \text{o.w.} \quad (1)$$

provides insight on whether  $X$  contributes to the conditional mean of  $Y$ . Shao and Zhang (2014) generalized distance covariance to martingale difference divergence (MDD), and  $\text{MDD} = 0$  if and only if (1) holds. Testing

---

\*Corresponding author. Email address: zj58@cornell.edu.

†Research support from an NSF Award (DMS-1455172), a Xerox PARC Faculty Research Award, and Cornell University Atkinson Center for a Sustainable Future (AVF-2017).

the conditional mean independence of  $Y$  given  $X$  conditioning on some known effect  $Z$ , i.e.,

$$H_0 : E(Y|X, Z) = E(Y|Z) \text{ a.s.}, \quad H_A : o.w. \quad (2)$$

sheds light on whether  $X$  contributes to the conditional mean of  $Y$  when taking known dependence on  $Z$  into account. Park et al. (2015) generalized martingale difference divergence to partial martingale difference divergence (pMDD), however, pMDD = 0 is not equivalent to (2). Fan and Li (1996), Lavergne and Vuong (2000), and Ait-Sahalia et al. (2001) each introduced non-parametric tests for (2) using kernel estimators of conditional expectations. Assuming a linear model between  $Y$  and  $(X, Z)$ , Lan et al. (2014) generalized the classical partial F-test (Chatterjee and Hadi, 2015) to a partial covariance-based (pcov) test for (2) in the high-dimensional setting, and Tang et al. (2017) further proposed a hybrid test for (2) through finding the most predictive covariate based on both maximum-type and sum-type statistics. Conditional mean independence conditioning on lagged covariates is known as Granger causality *in mean* in time series analysis. Raïssi et al. (2011) proposed a parametric test for linear Granger causality in mean based on vector autoregressive (VAR) models, and Hong et al. (2009) introduced a non-parametric test for non-linear Granger causality in mean based on cross-correlations.

In this paper, we focus on testing conditional mean independence with covariates and develop a method to test (2) for two main reasons. As Cook and Li (2002) state, regression analysis is mostly concerned with the conditional mean of the response given the predictors, which makes testing conditional mean independence more appealing than testing conditional independence. Further, it is very common in practice that some given covariates  $Z$  have been known to affect the conditional mean of  $Y$ . In this situation, we aim to determine whether  $X$  has marginal effect on the conditional mean of  $Y$  in the presence of  $Z$ , and decide whether  $X$  should be included to model the conditional mean of  $Y$  along with  $Z$ . In general, testing (2) is more useful than testing (1), but requires more careful handling.

We first simplify testing (2) to testing conditional mean independence through a transformation. Let  $V = Y - E(Y|Z) \in \mathbb{R}^q$ , and  $U = (X, Z) \in \mathbb{R}^{p+r}$ . Then  $E(V) = 0$ , and  $E(V|U) = E(Y|X, Z) - E(Y|Z)$ . As a result, we obtain an equivalent hypothesis test to (2) as

$$H_0 : E(V|U) = E(V) = 0 \text{ a.s.}, \quad H_A : o.w. \quad (3)$$

which is conditional mean independence of  $V$  given  $U$ . Thus, we consider the MDD with  $U$  and  $V$  to investigate (3). However, there are two problems to solve when we apply MDD to  $U$  and  $V$ . First,  $V$  needs to be estimated

since it is unobserved. We will replace  $V$  by its estimate  $\widehat{V}$  in calculating MDD. Second, we need to confirm that the estimation error of  $\widehat{V}$  is negligible, i.e., MDD with  $\widehat{V}$  is close enough to that with  $V$ , such that  $\widehat{V}$  may be used for inference instead of  $V$ .

The rest of this paper is organized as follows. In Section 2, we give a brief overview of martingale difference divergence. In Section 3, we estimate  $V$  based on the assumption that  $E(Y|Z)$  is a linear function of  $Z$ , and prove that the estimation of  $V$  does not affect the asymptotic distribution of martingale difference divergence under some regularity conditions. We present simulation results in Section 4, followed by a real data analysis in Section 5<sup>1</sup>. Finally, we summarize our work in Section 6.

The following notation is used throughout this paper. Let  $\{(X_i, Y_i, Z_i) : i = 1, \dots, n\}$  be an i.i.d. sample from the joint distribution  $F_{X,Y,Z}$ . When  $A$  is a matrix, the element of  $A$  at row  $k$  and column  $\ell$  is denoted by  $A(k, \ell)$ . When  $A$  is a vector, the element of  $A$  at index  $k$  is denoted by  $A(k)$ . The Frobenius norm of matrix  $A \in \mathbb{R}^{p \times q}$  is denoted by  $\|A\|_F$ . The Euclidean norm of vector  $X \in \mathbb{R}^p$  is denoted by  $|X|$ . The weighted  $\mathcal{L}_2$  norm  $\|\cdot\|_w$  of any complex-valued function  $\eta(t)$ ,  $t \in \mathbb{R}^p$  is defined by  $\|\eta(t)\|_w^2 = \int_{\mathbb{R}^p} |\eta(t)|^2 w(t) dt$  where  $|\eta(t)|^2 = \eta(t)\overline{\eta(t)}$ ,  $\overline{\eta(t)}$  is the complex conjugate of  $\eta(t)$ , and  $w(t)$  is any positive weight function under which the integral exists. Furthermore, *a.s.* is an abbreviation of almost surely.

## 2 MARTINGALE DIFFERENCE DIVERGENCE

Shao and Zhang (2014) proposed martingale difference divergence to capture the conditional mean dependence (in any form) of  $Y \in \mathbb{R}^q$  given  $X \in \mathbb{R}^p$ .

The non-negative martingale difference divergence for  $X$  and  $Y$ ,  $MDD(Y|X)$  is defined by its square

$$\begin{aligned} MDD^2(Y|X) &= \|E(Ye^{i\langle s, X \rangle}) - E(Y)E(e^{i\langle s, X \rangle})\|_{w_p}^2 \\ &\triangleq \int_{\mathbb{R}^p} |E(Ye^{i\langle s, X \rangle}) - E(Y)E(e^{i\langle s, X \rangle})|^2 w_p(s) ds, \end{aligned}$$

where the weight  $w_p(s) = c_p |s|^{1+p}$ , with  $c_p = \frac{\pi^{(1+p)/2}}{\Gamma((1+p)/2)}$ , and  $\Gamma$  is the gamma function. If  $E(|X|^2 + |Y|^2) < \infty$ , then  $MDD(Y|X) = 0$  if and only if  $E(Y|X) = E(Y)$  holds *a.s.*

The non-negative empirical martingale difference diver-

<sup>1</sup>See CRAN for an accompanying R package `EDMeasure` (Jin et al., 2018).

gence  $\text{MDD}_n(Y|X)$  is analogously defined by

$$\text{MDD}_n^2(Y|X) = \frac{1}{n^2} \sum_{i,j=1}^n A_{ij}B_{ij},$$

where  $A_{ij} = a_{ij} - \bar{a}_{i.} - \bar{a}_{.j} + \bar{a}_{..}$ ,  $\bar{a}_{i.} = \frac{1}{n} \sum_{j=1}^n a_{ij}$ ,  $\bar{a}_{.j} = \frac{1}{n} \sum_{i=1}^n a_{ij}$ ,  $\bar{a}_{..} = \frac{1}{n^2} \sum_{i,j=1}^n a_{ij}$ ,  $a_{ij} = |X_i - X_j|$ , and similarly for  $B_{ij}$  with  $b_{ij} = \frac{1}{2}|Y_i - Y_j|^2$ .

The consistency and weak convergence of  $\text{MDD}_n(Y|X)$  are derived as follows. If  $E(|X| + |Y|^2) < \infty$ , we have (i)  $\text{MDD}_n(Y|X) \xrightarrow[n \rightarrow \infty]{a.s.} \text{MDD}(Y|X)$ ; (ii) under  $H_0$  :

$$E(Y|X) = E(Y) \text{ a.s.}, n\text{MDD}_n^2(Y|X) \xrightarrow[n \rightarrow \infty]{\mathcal{D}} \|\zeta(s)\|_{w_p}^2,$$

where  $\zeta(\cdot)$  is a complex-valued zero-mean Gaussian process whose covariance function depends on  $F_{X,Y}$ ; (iii) under  $H_A : o.w.$ ,  $n\text{MDD}_n^2(Y|X) \xrightarrow[n \rightarrow \infty]{a.s.} \infty$ . Utilizing the nice properties of MDD, we next propose our test for (3).

### 3 METHODOLOGY

Inspired by the linear assumption to simplify the conditional dependence structure in Fan et al. (2015), we assume that the conditional expectation  $E(Y|Z)$  is a linear function of  $Z$ , simplifying the conditional mean dependence structure. As a result, we can decompose  $Y$  into the conditional expectation and reminder as

$$Y = E(Y|Z) + [Y - E(Y|Z)] \triangleq BZ + V,$$

where  $B \in \mathbb{R}^{q \times r}$ ,  $V \in \mathbb{R}^q$ . Then we have  $E(V|Z) = 0$ , and  $E(V) = 0$ . Similarly, the  $i$ th sample counterpart is  $Y_i = E(Y_i|Z_i) + V_i \triangleq BZ_i + V_i$ ,  $i = 1, \dots, n$ .

Suppose  $\hat{B}$  is the ordinary least squares (OLS) estimator of  $B$  when regressing  $Y$  on  $Z$ . We will then replace  $B$  with  $\hat{B}$  to estimate  $E(Y_i|Z_i)$  as  $\hat{E}(Y_i|Z_i) = \hat{B}Z_i$ , and  $V_i$  as  $\hat{V}_i = Y_i - \hat{B}Z_i = (B - \hat{B})Z_i + V_i$ . When estimating  $B$  via the OLS,  $Z$  is implicitly assumed to have full column rank. In case  $Z$  is high-dimensional, i.e.,  $r > n$ , we can estimate  $B$  by the penalized least squares (PLS) similar to Fan et al. (2015), including ridge (Hoerl and Kennard, 1970) and lasso (Tibshirani, 1996).

We now construct a test for (3) based on  $\text{MDD}_n^2(\hat{V}|U)$  and its counterparts using permutation samples, then calculate the empirical p-value following the permutation in Park et al. (2015). Because the samples are independent, but with an unspecified distribution, permutation tests are a convenient tool for inference. We will later show in Theorem 2 that the asymptotic distribution of  $n\text{MDD}_n^2(\hat{V}|U)$  depends on an unknown underlying distribution, which justifies the use of permutation tests. To measure the conditional mean dependence of  $V$  given  $U$ , we first compute the test statistic  $\text{MDD}_n^2(\hat{V}|U)$

from the sample  $\{(\hat{V}_i, U_i) : i = 1, \dots, n\}$ , where  $U_i = (X_i, Z_i)$ . That is,  $\text{MDD}_n^2(\hat{V}|U)$  depends on the i.i.d. sample  $\{(X_i, Y_i, Z_i) : i = 1, \dots, n\}$ . Next we draw  $B$  permutation samples of size  $n$  as  $\{(X_i^*, Y_i, Z_i) : i = 1, \dots, n\}$ , where only the sample of  $X$  is permuted in order to approximate the sampling distribution. For each permutation sample, we calculate the test statistic  $\text{MDD}_{n,b}^2(\hat{V}|U)$ ,  $b = 1, \dots, B$ . Then the empirical p-value is given by

$$\hat{p} = \frac{\sum_{b=1}^B \mathbf{1} \left\{ \text{MDD}_{n,b}^2(\hat{V}|U) \geq \text{MDD}_n^2(\hat{V}|U) \right\}}{B}.$$

When  $H_0$  is false,  $\text{MDD}_n^2(\hat{V}|U)$  tends to be large while  $\text{MDD}_{n,b}^2(\hat{V}|U)$  tends to be small. As a result, the empirical p-value is expected to be very small, leading to a rejection of  $H_0$ . We name the proposed test linear martingale difference divergence (LinMDD). To justify our LinMDD test, it remains to validate that  $\text{MDD}_n^2(\hat{V}|U)$  is close enough to  $\text{MDD}_n^2(V|U)$ , i.e., the estimation error in  $\hat{V}$  is negligible for the sampling distribution of the test statistic, focusing on the asymptotic case. To begin with, we introduce some regularity conditions to derive the asymptotic distribution of  $\text{MDD}_n^2(\hat{V}|U)$ .

**Condition 1.** *There exist constants  $0 < c_1, c_2, c_3 < \infty$ , such that  $E(|U_i - U_j|^2) = c_1$ ,  $i \neq j$ ;  $E(|U_i - U_j||U_i - U_k|) = c_2$ ,  $i \neq j \neq k$ ;  $E(|U_i - U_j||U_k - U_\ell|) = c_3$ ,  $i \neq j \neq k \neq \ell$ .*

**Condition 2.** *There exists constant  $0 < c_4 < \infty$ , such that  $E[(Z_i(t) - Z_j(t))^2(Z_i(s) - Z_j(s))^2] \leq c_4$ ,  $i \neq j$ ,  $\forall t, s$ .*

**Condition 3.** *There exists constant  $0 < c_5 < \infty$ , such that  $E[(Z_i(t) - Z_j(t))^2(V_i(s) - V_j(s))^2] \leq c_5$ ,  $i \neq j$ ,  $\forall t, s$ .*

**Condition 4.**  $\|\hat{B} - B\|_F = O_p(n^{-1/2})$ .

*Remark.* Condition 4 can be derived from the bounded density of  $|V_i - V_j|$  and non-heavy tails of  $Z_i(t)$  and  $V_i(t)$  according to Fan et al. (2015) and Fan et al. (2011).

Through a similar derivation to Theorem 2 of Fan et al. (2015), we justify the choice of using  $\text{MDD}_n^2(\hat{V}|U)$  in place of  $\text{MDD}_n^2(V|U)$  by the following lemma and theorems. Lemma 1 shows that the difference between  $\text{MDD}_n^2(\hat{V}|U)$  and  $\text{MDD}_n^2(V|U)$  is negligible as the sample size increases. The proof of Lemma 1 can be found in Appendix 6.

**Lemma 1.** *If  $Y = BZ + V$  and Conditions 1-4 hold, we have*

$$\text{MDD}_n^2(\hat{V}|U) - \text{MDD}_n^2(V|U) = O_p(n^{-3/2}).$$

Consequently, the consistency and weak convergence of  $\text{MDD}_n(\hat{V}|U)$  follow from Lemma 1 and are summarized in Theorem 1 and 2 below.



**Theorem 1** (Consistency). *If  $Y = BZ + V$  and Conditions 1-4 hold, we have*

$$MDD_n(\widehat{V}|U) \xrightarrow[n \rightarrow \infty]{\mathcal{P}} MDD(V|U).$$

**Theorem 2** (Weak convergence). *If  $Y = BZ + V$  and Conditions 1-4 hold, under  $H_0$ , we have*

$$nMDD_n^2(\widehat{V}|U) \xrightarrow[n \rightarrow \infty]{\mathcal{D}} \|\zeta(s)\|_{w_p}^2,$$

where  $\zeta(\cdot)$  denotes the complex-valued Gaussian random process corresponding to the asymptotic distribution of  $nMDD_n^2(V|U)$ . Under  $H_A$ , we have

$$nMDD_n^2(\widehat{V}|U) \xrightarrow[n \rightarrow \infty]{\mathcal{P}} \infty.$$

According to Theorem 1,  $MDD_n(\widehat{V}|U)$  converges to the same population statistic  $MDD(V|U)$  as  $MDD_n(V|U)$ , and thus it can serve to measure the conditional mean dependence of  $V$  given  $U$ . In addition,  $nMDD_n^2(\widehat{V}|U)$  and  $nMDD_n^2(V|U)$  have the same asymptotic distribution stated in Theorem 2, which establishes the effectiveness of LinMDD test, as we approximate the limiting distribution of  $nMDD_n^2(V|U)$  using  $nMDD_n^2(\widehat{V}|U)$  in LinMDD test. In Section 4 and Section 5, we will present the finite-sample performance of our LinMDD test through simulations and a real data example, respectively.

## 4 SIMULATION STUDIES

To evaluate the performance of our LinMDD test, we adopt the simulation setup in Lavergne and Vuong (2000), and compare our test to the pMDD test (Park et al., 2015), pdCov test (Székely and Rizzo, 2014), and pcov test (Lan et al., 2014) as benchmarks. All tests are implemented as permutation tests with permutation size  $B = 500$ , in which we only permute the sample of  $X$  to approximate the distribution of the test statistic.

We generate data from the underlying model

$$Y = -Z + b \cdot Z^3 + f(X) + \epsilon,$$

where  $Z \sim N(0, 1)$ ,  $X \sim \mathcal{N}(0, 1)$ ,  $\epsilon \sim \mathcal{N}(0, 4)$ , and  $Z, X, \epsilon$  are independent. We test the null hypothesis  $H_0 : E(Y|X, Z) = E(Y|Z)$  a.s. with significance level  $\alpha \in \{0.05, 0.1\}$ , and examine the empirical size and power of each test. We run 1000 replications with sample size  $n \in \{20, 30, 50, 70, 100\}$  for each specific model.

**Model 1** (Linear  $Z$ , linear  $X$ ).  $b = 0$ ,  $f(X) = cX$  where  $c \in \{0, \frac{2}{3}, 1, \frac{3}{2}\}$ .

**Model 2** (Linear  $Z$ , non-linear  $X$ ).  $b = 0$ ,  $f(X) = \sin(c\pi X)$  where  $c \in \{\frac{1}{4}, \frac{1}{3}, \frac{1}{2}\}$ . We omit  $c = 0$  as it is exactly the same as  $c = 0$  in Model 1.

From Figure 1, the empirical size of all tests is around 0.05 (0.1). The empirical power of all tests increases as  $n$  increases. For the linear  $X$  case, the empirical power of all tests is higher when  $c$  is larger, since the signal-to-noise ratio increases. Moreover, the empirical power of the LinMDD and pcov tests is consistently higher than that of the other tests, because the linear assumption is valid, and only LinMDD and pcov tests are designed for linear  $Z$ . For the non-linear  $X$  case, the LinMDD test still outperforms the other tests, while the performance of the pcov test degrades as  $c$  increases, because the LinMDD test is designed for non-linear  $X$  while pcov test is suitable only for linear  $X$ .

**Model 3** (Nonlinear  $Z$ , linear  $X$ ).  $b = 1$ ,  $f(X) = cX$  where  $c \in \{0, \frac{2}{3}, 1, \frac{3}{2}\}$ .

**Model 4** (Nonlinear  $Z$ , non-linear  $X$ ).  $b = 1$ ,  $f(X) = \sin(c\pi X)$  where  $c \in \{\frac{1}{4}, \frac{1}{3}, \frac{1}{2}\}$ . We omit  $c = 0$  as it is exactly the same as  $c = 0$  in Model 3.

From Figure 2, the empirical size of all tests is around 0.05 (0.1). For the linear  $X$  case, the empirical power of the LinMDD and pcov tests is competitive with but not always higher than that of the other tests. The reason is that the linear dependence of  $Y$  on  $Z$  is violated while the other tests do not rely it. For the non-linear  $X$  case, we similarly find that the performance of the pcov test degrades as  $c$  increases. The simulation results show that our LinMDD test achieves competitive and often better performance than the others in these situations. Next, we apply the proposed LinMDD test on a real dataset.

## 5 FINANCIAL DATA APPLICATION

In finance, the capital asset pricing model (CAPM) was proposed by Sharpe (1964), Lintner (1965), and Mossin (1966) to describe the stock returns through the market risk as

$$r_t = \alpha + \beta_1 m_t,$$

where  $r_t$  is the excess stock return (in excess the risk-free return), and  $m_t$  is the excess market return at time  $t$ . Fama and French (1993) added size and value factors to the CAPM, and proposed the Fama–French three-factor model as

$$r_t = \alpha + \beta_1 m_t + \beta_2 \text{SMB}_t + \beta_3 \text{HML}_t,$$

where SMB (small minus big) and HML (high minus low) account for stocks with small/big market capitalization and high/low book-to-market ratio, respectively. Fama and French (2015) further added profitability and investment factors to the three-factor model, and extended it to the Fama–French five-factor model as

$$r_t = \alpha + \beta_1 m_t + \beta_2 \text{SMB}_t + \beta_3 \text{HML}_t + \beta_4 \text{RMW}_t + \beta_5 \text{CMA}_t,$$

where RMW (robust minus weak) and CMA (conservative minus aggressive) further account for stocks with robust/weak operating profitability and conservative/aggressive investment, respectively.

We collect the annual risk-free returns and Fama–French five factors<sup>2</sup>, and the annual returns of Boeing (BA) stock<sup>3</sup> in the past 53 years between 1964 and 2016. The time series and histograms of excessive BA stock returns and Fama–French five factors are depicted in Figure 3.

### 5.1 CAPM VS. FAMA–FRENCH THREE-FACTOR MODEL

First, we are curious whether the size and value factors should be added to the CAPM, i.e., whether SMB and HML in the Fama–French three-factor model contribute to the expectation of excess stock returns given the market risk. Thus, we test  $H_0 : E(Y|X, Z) = E(Y|Z)$  *a.s.*, where  $X_t = (\text{SMB}_t, \text{HML}_t)$ ,  $Y_t = r_t$ , and  $Z_t = (1, m_t)$ .

We apply our LinMDD test to the data with  $n = 53$  and  $B = 500$ . Our p-value is 0.072, while the p-values are 0.012 (pMDD), 0.092 (pdCov) and 0.096 (pcov) using competing tests. As a result, we reject  $H_0$  with significance level  $\alpha = 0.1$ , and conclude that SMB and HML help determine the excess returns of BA stock in the presence of the market risk. Our results align with the research in finance that the Fama–French three-factor model remarkably outperforms the CAPM in explaining excess stock returns.

### 5.2 FAMA–FRENCH THREE-FACTOR MODEL VS. FIVE-FACTOR MODEL

Similarly, we are interested in whether the profitability and investment factors should be further added to the Fama–French three-factor model, i.e., whether RMW and CMA in the Fama–French five-factor model contribute to the description of excess stock returns given the other three factors. Hence, we test  $H_0 : E(Y|X, Z) = E(Y|Z)$  *a.s.*, in which  $X_t = (\text{RMW}_t, \text{CMA}_t)$ ,  $Y_t = r_t$ , and  $Z_t = (1, m_t, \text{SMB}_t, \text{HML}_t)$ .

We apply our LinMDD test to the data with  $n = 53$  and  $B = 500$ , and its p-value is 0.360, while the p-values are 0.358 (pMDD), 0.878 (pdCov) and 0.768 (pcov) using competing tests. As a result, we fail to reject  $H_0$  with significance level  $\alpha = 0.1$ , and conclude that RMW and CMA are unable to help determine the excess re-

turns of BA stock in the presence of the other three factors. Our results align with the research in finance that the Fama–French five-factor model has yet to be proven as a significant improvement over the three-factor model in describing excess stock returns.

### 5.3 FAMA–FRENCH FOUR-FACTOR MODEL VS. FIVE-FACTOR MODEL

Fama and French (2015) showed that the value factor HML becomes redundant when profitability and investment factors are added to the Fama–French three-factor model, because HML is fully captured by its exposures to the other four factors, especially RMW and CMA. To validate this argument, we test  $H_0 : E(Y|X, Z) = E(Y|Z)$  *a.s.*, where  $X_t = \text{HML}_t$ ,  $Y_t = r_t$ , and  $Z_t = (1, m_t, \text{SMB}_t, \text{RMW}_t, \text{CMA}_t)$ .

We apply our LinMDD test to the data with  $n = 53$  and  $B = 500$ . Our p-value is 0.218, while the p-values are 0.438 (pMDD), 0.540 (pdCov) and 0.858 (pcov) using competing tests. As a result, we fail to reject  $H_0$  with significance level  $\alpha = 0.1$ , and conclude that HML cannot help explain the excess returns of BA stock in the presence of the other four factors. Our results demonstrate that HML is redundant for describing excess stock returns in the Fama–French five-factor model.

## 6 CONCLUSION

In this paper, we propose a new test, LinMDD, for the null hypothesis  $H_0 : E(Y|X, Z) = E(Y|Z)$  *a.s.* by investigating an equivalent one  $H_0 : E(V|U) = E(V) = 0$  *a.s.*, derived from a transformation involving the conditional expectation. When applying martingale difference divergence (Shao and Zhang, 2014) to test  $H_0 : E(V|U) = E(V) = 0$  *a.s.*, we make two major contributions.

- (1) Since  $V$  is unobservable, we estimate  $V$  based on the assumption that  $E(Y|Z)$  is a linear function of  $Z$ , simplifying the conditional mean dependence structure.
- (2) We prove that the estimation error in  $\hat{V}$  is negligible for the asymptotic distribution of the test statistic. Thus, we can replace  $V$  with  $\hat{V}$  in the test statistic for inference in large samples.

We implement the LinMDD test as a permutation test following Park et al. (2015), and compare it with existing tests in various simulation studies. The LinMDD test consistently outperforms existing tests when its linear assumption is valid, and it achieves competitive results with existing tests even when its linear assumption is violated.

<sup>2</sup>Download data at [http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data\\_library.html](http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html).

<sup>3</sup>Download data using `get.hist.quote` in the R package `tseries` (Trapletti and Hornik, 2017).

To illustrate the practical value of the LinMDD test, we compare the CAPM, the Fama–French three-factor and five-factor models by applying LinMDD test to the financial data. We find that the Fama–French three-factor outperforms the CAPM, while the Fama–French five-factor is not a significant improvement over the three-factor model when explaining the excess annual returns of a major stock. Moreover, we validate the statement that the value factor is redundant in the Fama–French five-factor model (Fama and French, 2015) using the LinMDD test.

The relaxation of the linear assumption is an important topic for future research. Our method will become more general if the linear assumption of conditional mean dependence can be generalized to a non-linear one, using non-parametric regression (local regression, splines) instead of linear regression in the estimation of conditional mean. In addition, the high-dimensional setting regarding  $Z$  where  $r > n$  is an interesting direction to consider as well.

## APPENDIX

### PROOF OF LEMMA 1

*Proof.* We define  $T$

$$\begin{aligned} &= n\text{MDD}_n^2(\widehat{V}|U) - n\text{MDD}_n^2(V|U) \\ &= \frac{1}{2n} \sum_{i,j} [(F_{ij} - \frac{1}{n} \sum_k F_{kj} - \frac{1}{n} \sum_k F_{ik} + \frac{1}{n^2} \sum_{k,\ell} F_{k\ell}) \\ &\quad \times (E_{ij} - \frac{1}{n} \sum_k E_{kj} - \frac{1}{n} \sum_k E_{ik} + \frac{1}{n^2} \sum_{k,\ell} E_{k\ell})], \end{aligned}$$

where  $F_{ij} = |\widehat{V}_i - \widehat{V}_j|^2 - |V_i - V_j|^2$ ,  $E_{ij} = |U_i - U_j|$ .

We apply Taylor expansion to  $|\widehat{V}_t - \widehat{V}_s|^2$  at  $V_t - V_s$  in terms of  $f(x) = x^T x$ ,  $f'(x) = 2x^T$ , then there exists  $\lambda \in (0, 1)$ , such that  $F_{ij}$

$$\begin{aligned} &= 2[\lambda(\widehat{V}_i - \widehat{V}_j) + (1 - \lambda)(V_i - V_j)]^T (\widehat{V}_i - \widehat{V}_j - V_i + V_j) \\ &= 2[\lambda(Z_i - Z_j)^T (B - \widehat{B})^T (B - \widehat{B})(Z_i - Z_j) \\ &\quad + (V_i - V_j)^T (B - \widehat{B})(Z_i - Z_j)]. \end{aligned}$$

Thus, we have  $T = T_1 + T_2$ , where  $T_1$

$$\begin{aligned} &= \frac{\lambda}{n} \sum_{i,j} [(G_{ij} - \frac{1}{n} \sum_k G_{kj} - \frac{1}{n} \sum_k G_{ik} + \frac{1}{n^2} \sum_{k,\ell} G_{k\ell}) \\ &\quad \times (E_{ij} - \frac{1}{n} \sum_k E_{kj} - \frac{1}{n} \sum_k E_{ik} + \frac{1}{n^2} \sum_{k,\ell} E_{k\ell})], \\ &G_{ij} = (Z_i - Z_j)^T (B - \widehat{B})^T (B - \widehat{B})(Z_i - Z_j), \end{aligned}$$

and  $T_2$

$$\begin{aligned} &= \frac{1}{n} \sum_{i,j} [(H_{ij} - \frac{1}{n} \sum_k H_{kj} - \frac{1}{n} \sum_k H_{ik} + \frac{1}{n^2} \sum_{k,\ell} H_{k\ell}) \\ &\quad \times (E_{ij} - \frac{1}{n} \sum_k E_{kj} - \frac{1}{n} \sum_k E_{ik} + \frac{1}{n^2} \sum_{k,\ell} E_{k\ell})], \end{aligned}$$

$$H_{ij} = (V_i - V_j)^T (B - \widehat{B})(Z_i - Z_j).$$

First, we will show (i)  $T_1 = O_p(n^{-1})$ .

After a simple calculation, we have

$$\begin{aligned} &\frac{1}{n} \sum_{i,j} (G_{ij} - \frac{1}{n} \sum_k G_{kj} - \frac{1}{n} \sum_k G_{ik} + \frac{1}{n^2} \sum_{k,\ell} G_{k\ell}) E_{ij} \\ &= \text{tr}[\frac{1}{n} \sum_{i,j} |U_i - U_j| (G_{ij} - \frac{1}{n} \sum_k G_{kj} - \frac{1}{n} \sum_k G_{ik} \\ &\quad + \frac{1}{n^2} \sum_{k,\ell} G_{k\ell})] \\ &= \text{tr}[(B - \widehat{B})^T (B - \widehat{B})M], \end{aligned}$$

where  $M = \frac{1}{n} \sum_{i,j} |U_i - U_j| S_{ij}$ , and

$$S_{ij} = R_{ij} - \frac{1}{n} \sum_k R_{kj} - \frac{1}{n} \sum_k R_{ik} + \frac{1}{n^2} \sum_{k,\ell} R_{k\ell},$$

$R_{ij} = (Z_i - Z_j)(Z_i - Z_j)^T$ ,  $R_{ij} = R_{ji}$ ,  $S_{ij} = S_{ji}$ , then

$$\begin{aligned} &\text{E}[(M(t, s))^2] \\ &= \text{E}[\frac{1}{n^2} (\sum_{i,j} |U_i - U_j| S_{ij}(t, s))^2] \\ &= \text{E}\{\text{E}[\frac{1}{n^2} (\sum_{i,j} |U_i - U_j| S_{ij}(t, s))^2 | U_i, \forall i]\} \\ &= \text{E}[\frac{2c_1}{n^2} \sum_{i \neq j} (S_{ij}(t, s))^2 \\ &\quad + \frac{2c_2}{n^2} \sum_{i \neq j \neq k} (S_{ij}(t, s) S_{ik}(t, s) + S_{ij}(t, s) S_{kj}(t, s)) \\ &\quad + \frac{c_3}{n^2} \sum_{i \neq j \neq k \neq \ell} S_{ij}(t, s) S_{k\ell}(t, s)], \end{aligned}$$

where  $c_1 = \text{E}(|U_i - U_j|^2)$ ,  $i \neq j$ ;  $c_2 = \text{E}(|U_i - U_j| |U_i - U_k|)$ ,  $i \neq j \neq k$ ;  $c_3 = \text{E}(|U_i - U_j| |U_k - U_\ell|)$ ,  $i \neq j \neq k \neq \ell$ .

Considering that  $\text{E}[(R_{ij}(t, s))^2] = \text{E}[(Z_i - Z_j)_t^2 (Z_i - Z_j)_s^2] \leq c_4$ ,  $i \neq j$ ,  $\forall t, s$ , we have  $\text{E}[(R_{ij}(t, s))^2] = O(1)$ , which implies  $\text{E}[(S_{ij}(t, s))^2] = O(1)$ , and thus  $\text{E}[\frac{1}{n^2} \sum_{i \neq j} (S_{ij}(t, s))^2] = O(1)$ .

After a simple calculation, we have  $\sum_i S_{ij}(t, s) = 0$ ,  $\sum_j S_{ij}(t, s) = 0$ ,  $\sum_i \sum_j S_{ij}(t, s) = 0$ , and

$$\begin{aligned}
& \sum_{i \neq j \neq k} S_{ij}(t, s) S_{ik}(t, s) \\
&= \sum_i (S_{ii}(t, s))^2 - \sum_{i \neq j} (S_{ij}(t, s))^2, \\
& \sum_{i \neq j \neq k} S_{ii}(t, s) S_{jk}(t, s) \\
&= \sum_i (S_{ii}(t, s))^2 - \sum_{i \neq j} S_{ii}(t, s) S_{jj}(t, s), \\
& \sum_{i \neq j \neq k \neq \ell} S_{ij}(t, s) S_{k\ell}(t, s) \\
&= -2 \sum_{i \neq j \neq k} [S_{ii}(t, s) S_{jk}(t, s) + S_{ij}(t, s) S_{ik}(t, s) \\
&+ S_{ij}(t, s) S_{kj}(t, s)] \\
&- \sum_{i \neq j} [4S_{ii}(t, s) S_{ij}(t, s) + S_{ii}(t, s) S_{jj}(t, s) \\
&+ 2(S_{ij}(t, s))^2] - \sum_i (S_{ii}(t, s))^2,
\end{aligned}$$

we have

$$\begin{aligned}
& \mathbb{E}\left[\frac{1}{n^2} \sum_{i \neq j \neq k} S_{ij}(t, s) S_{ik}(t, s)\right] = O(1), \\
& \mathbb{E}\left[\frac{1}{n^2} \sum_{i \neq j \neq k} S_{ij}(t, s) S_{kj}(t, s)\right] = O(1), \\
& \mathbb{E}\left[\frac{1}{n^2} \sum_{i \neq j \neq k} S_{ii}(t, s) S_{jk}(t, s)\right] = O(1), \\
& \mathbb{E}\left[\frac{1}{n^2} \sum_{i \neq j \neq k \neq \ell} S_{ij}(t, s) S_{k\ell}(t, s)\right] = O(1).
\end{aligned}$$

Therefore,  $\mathbb{E}[(M(t, s))^2] = O(1)$ .

Applying Chebyshev's inequality to  $M(t, s)$ , we have

$$P(|M(t, s) - \mu| \geq k\sigma) \leq 1/k^2,$$

where  $\mu = \mathbb{E}[M(t, s)]$ ,  $\sigma^2 = \text{Var}[M(t, s)]$ . As a result,  $M(t, s) = O_p(1)$ .

Given that  $\|\widehat{B} - B\|_F = O_p(n^{-1/2})$ , we have

$$\begin{aligned}
& \frac{1}{n} \sum_{i,j} (G_{ij} - \frac{1}{n} \sum_k G_{kj} - \frac{1}{n} \sum_k G_{ik} + \frac{1}{n^2} \sum_{k,\ell} G_{k\ell}) E_{ij} \\
&= \text{tr}[(B - \widehat{B})^T (B - \widehat{B}) M] \\
&= pq^2 O_p(n^{-1}) O_p(1) \\
&= O_p(n^{-1}).
\end{aligned}$$

Similarly, we have

$$\begin{aligned}
& \frac{1}{n} \sum_{i,j} (G_{ij} - \frac{1}{n} \sum_k G_{kj} - \frac{1}{n} \sum_k G_{ik} + \frac{1}{n^2} \sum_{k,\ell} G_{k\ell}) E_{kj}, \\
& \frac{1}{n} \sum_{i,j} (G_{ij} - \frac{1}{n} \sum_k G_{kj} - \frac{1}{n} \sum_k G_{ik} + \frac{1}{n^2} \sum_{k,\ell} G_{k\ell}) E_{ik}, \\
& \frac{1}{n} \sum_{i,j} (G_{ij} - \frac{1}{n} \sum_k G_{kj} - \frac{1}{n} \sum_k G_{ik} + \frac{1}{n^2} \sum_{k,\ell} G_{k\ell}) E_{k\ell}
\end{aligned}$$

are all  $O_p(n^{-1})$ . Therefore,  $T_1 = O_p(n^{-1})$ .

Analogous to (i), we can show (ii)  $T_2 = O_p(n^{-1/2})$ . The only differences are

$$\begin{aligned}
& \frac{1}{n} \sum_{i,j} (H_{ij} - \frac{1}{n} \sum_k H_{kj} - \frac{1}{n} \sum_k H_{ik} + \frac{1}{n^2} \sum_{k,\ell} H_{k\ell}) E_{ij} \\
&= \text{tr}[(B - \widehat{B}) M],
\end{aligned}$$

where  $M$  is defined similarly with  $R_{ij} = (Z_i - Z_j)(V_i - V_j)^T$ , and  $\mathbb{E}[(R_{ij}(t, s))^2] = \mathbb{E}[(Z_i - Z_j)_t^2 (V_i - V_j)_s^2] \leq c_5$ ,  $i \neq j$ ,  $\forall t, s$ , and

$$\begin{aligned}
& \frac{1}{n} \sum_{i,j} (H_{ij} - \frac{1}{n} \sum_k H_{kj} - \frac{1}{n} \sum_k H_{ik} + \frac{1}{n^2} \sum_{k,\ell} H_{k\ell}) E_{ij} \\
&= \text{tr}[(B - \widehat{B}) M] \\
&= pq O_p(n^{-1/2}) O_p(1) \\
&= O_p(n^{-1/2}),
\end{aligned}$$

and therefore  $T_2 = O_p(n^{-1/2})$ .

As a conclusion,  $T = T_1 + T_2 = O_p(n^{-1/2})$ .  $\square$

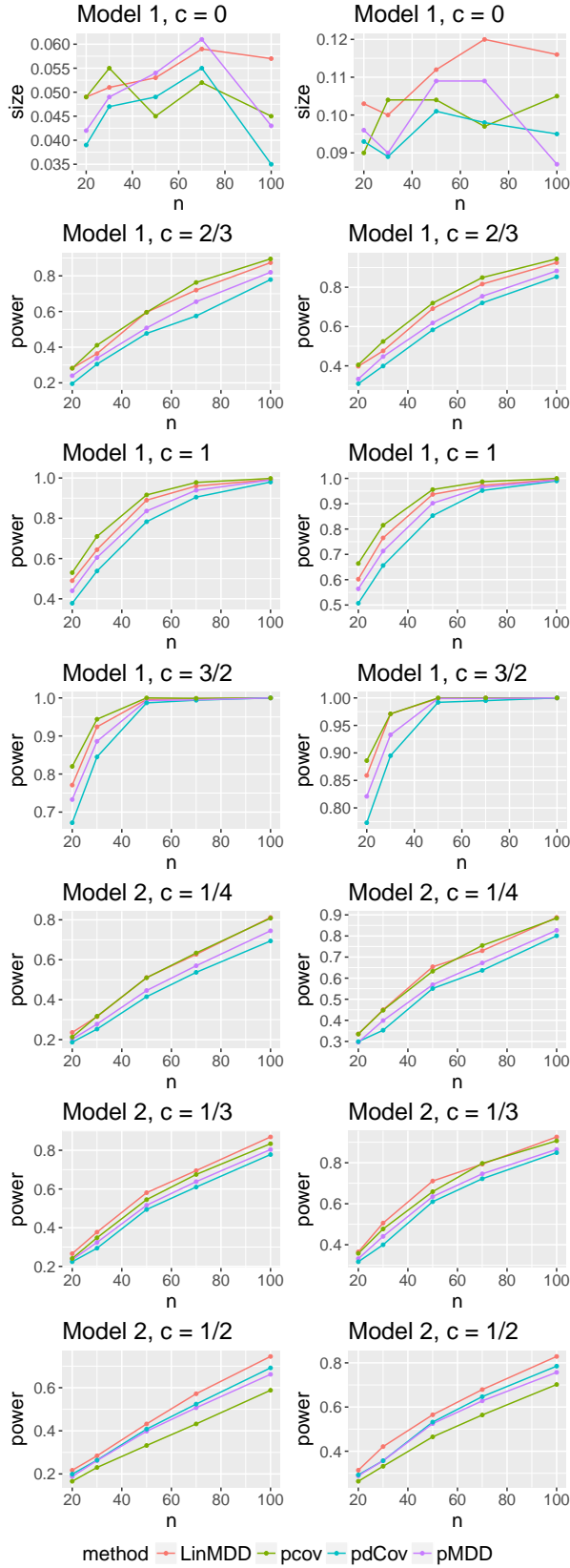


Figure 1: Empirical size and power of 1000 replications with  $B = 500$  for Model 1 & 2.

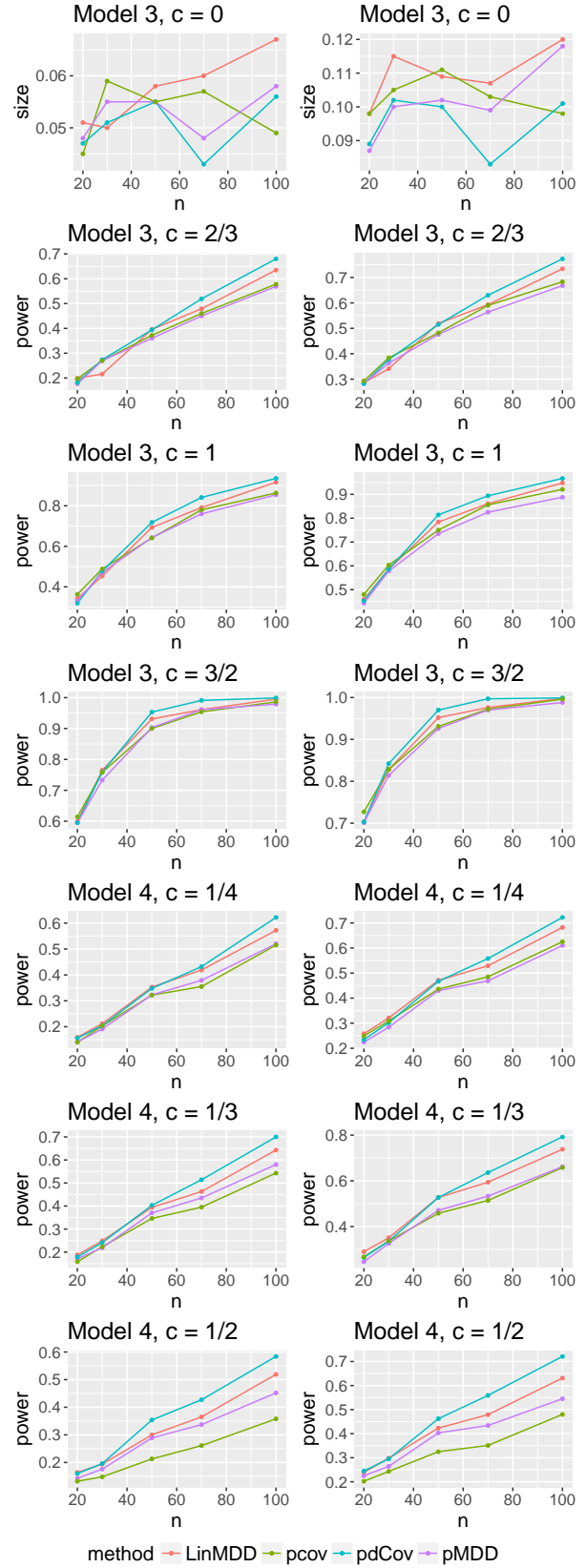


Figure 2: Empirical size and power of 1000 replications with  $B = 500$  for Model 3 & 4.

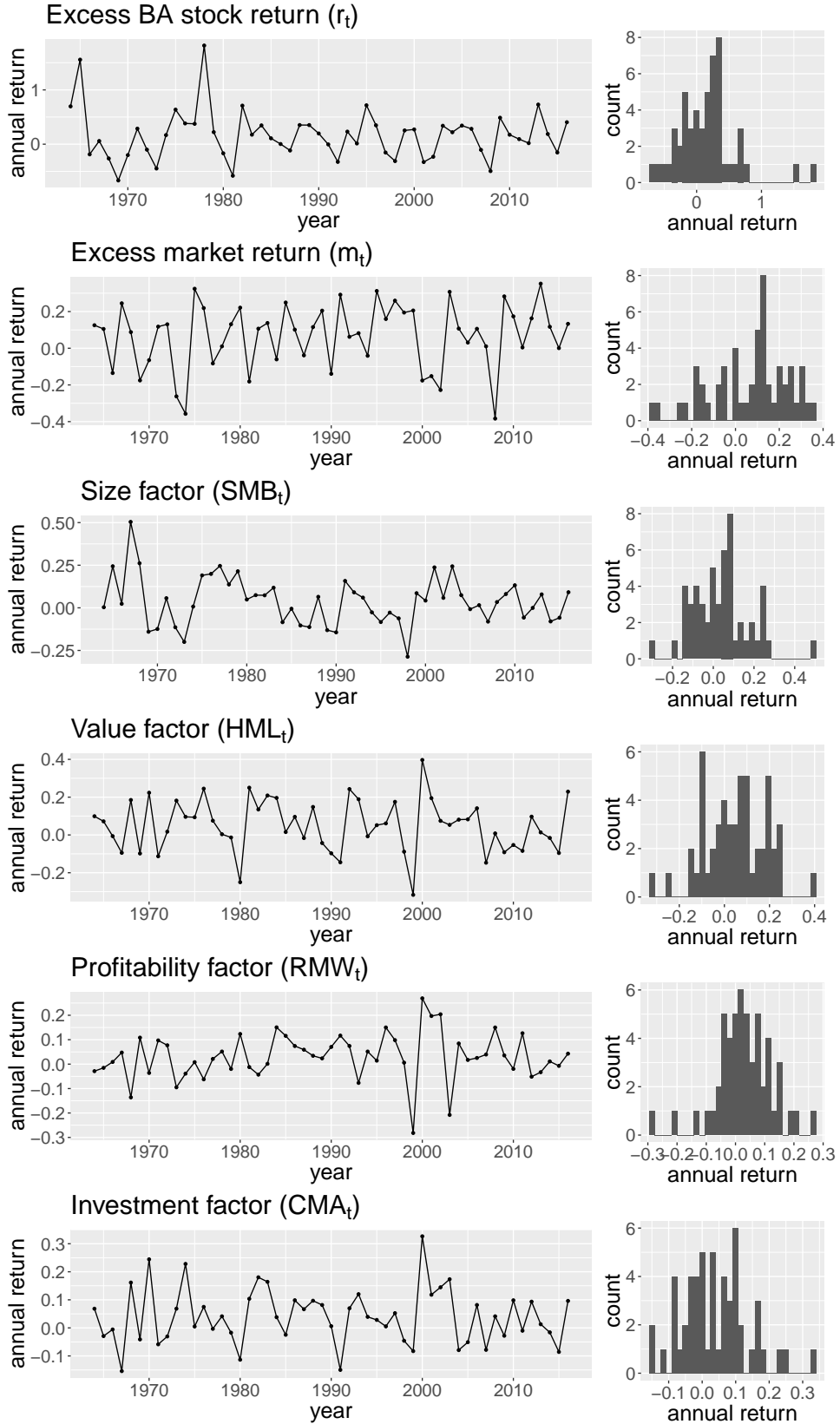


Figure 3: Time series and histograms of excess BA stock returns ( $r_t$ ), excess market returns ( $m_t$ ), size factors ( $SMB_t$ ), value factors ( $HML_t$ ), profitability factors ( $RMW_t$ ), and investment factors ( $CMA_t$ ) between 1964 and 2016.

## References

- Y. Aït-Sahalia, P. J. Bickel, and T. M. Stoker. Goodness-of-fit tests for kernel regression with an application to option implied volatilities. *Journal of Econometrics*, 105(2):363–412, 2001.
- S. Chatterjee and A. S. Hadi. *Regression analysis by example*. John Wiley & Sons, 2015.
- B. Chen and Y. Hong. Testing for the markov property in time series. *Econometric Theory*, 28(1):130–178, 2012.
- R. D. Cook and B. Li. Dimension reduction for conditional mean in regression. *Annals of Statistics*, pages 455–474, 2002.
- E. F. Fama and K. R. French. Common risk factors in the returns on stocks and bonds. *Journal of financial economics*, 33(1):3–56, 1993.
- E. F. Fama and K. R. French. A five-factor asset pricing model. *Journal of financial economics*, 116(1):1–22, 2015.
- J. Fan, Y. Liao, and M. Mincheva. High dimensional covariance matrix estimation in approximate factor models. *Annals of statistics*, 39(6):3320, 2011.
- J. Fan, Y. Feng, and L. Xia. A conditional dependence measure with applications to undirected graphical models. *arXiv preprint arXiv:1501.01617*, 2015.
- Y. Fan and Q. Li. Consistent model specification tests: omitted variables and semiparametric functional forms. *Econometrica: Journal of the econometric society*, pages 865–890, 1996.
- L. Gan, N. N. Narisetty, and F. Liang. Bayesian regularization for graphical models with unequal shrinkage. *Journal of the American Statistical Association*, (just-accepted), 2018.
- C. Hiemstra and J. D. Jones. Testing for linear and nonlinear granger causality in the stock price-volume relation. *The Journal of Finance*, 49(5):1639–1664, 1994.
- A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.
- Y. Hong, Y. Liu, and S. Wang. Granger causality in risk and detection of extreme risk spillover between financial markets. *Journal of Econometrics*, 150(2):271–287, 2009.
- Z. Jin and D. S. Matteson. Generalizing distance covariance to measure and test multivariate mutual dependence. *arXiv preprint arXiv:1709.02532*, 2017.
- Z. Jin and D. S. Matteson. Independent component analysis via energy-based and kernel-based mutual dependence measures. *arXiv preprint arXiv:1805.06639*, 2018.
- Z. Jin, S. Yao, D. S. Matteson, and X. Shao. *EDMeasure: Energy-Based Dependence Measures*, 2018. R package version 1.2.
- W. Lan, H. Wang, and C.-L. Tsai. Testing covariates in high-dimensional regression. *Annals of the Institute of Statistical Mathematics*, 66(2):279–301, 2014.
- P. Lavergne and Q. Vuong. Nonparametric significance testing. *Econometric Theory*, 16(4):576–601, 2000.
- R. Li, W. Zhong, and L. Zhu. Feature screening via distance correlation learning. *Journal of the American Statistical Association*, 107(499):1129–1139, 2012.
- Z. Li and T. H. McCormick. An expectation conditional maximization approach for gaussian graphical models. *arXiv preprint arXiv:1709.06970*, 2017.
- Z. R. Li, T. H. McCormick, and S. J. Clark. Bayesian joint spike-and-slab graphical lasso. *arXiv preprint arXiv:1805.07051*, 2018.
- J. Lintner. The valuation of risk assets and the selection of risky investments in stock portfolios and capital budgets. *The review of economics and statistics*, pages 13–37, 1965.
- J. Mossin. Equilibrium in a capital asset market. *Econometrica: Journal of the econometric society*, pages 768–783, 1966.
- T. Park, X. Shao, and S. Yao. Partial martingale difference correlation. *Electronic Journal of Statistics*, 9(1):1492–1517, 2015.
- H. Raïssi et al. Testing linear causality in mean when the number of estimated parameters is high. *Electronic journal of statistics*, 5:507–533, 2011.
- X. Shao and J. Zhang. Martingale difference correlation and its use in high-dimensional variable screening. *Journal of the American Statistical Association*, 109(507):1302–1318, 2014.
- W. F. Sharpe. Capital asset prices: A theory of market equilibrium under conditions of risk. *The journal of finance*, 19(3):425–442, 1964.
- L. Su and H. White. A consistent characteristic function-based test for conditional independence. *Journal of Econometrics*, 141(2):807–834, 2007.
- G. J. Székely and M. L. Rizzo. Partial distance correlation with methods for dissimilarities. *The Annals of Statistics*, 42(6):2382–2412, 2014.
- G. J. Székely, M. L. Rizzo, and N. K. Bakirov. Measuring and testing dependence by correlation of distances. *The annals of statistics*, 35(6):2769–2794, 2007.
- Y. Tang, H. J. Wang, and E. Barut. Testing for the presence of significant covariates through conditional marginal regression. *Biometrika*, 2017.

- R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.
- A. Trapletti and K. Hornik. *tseries: Time Series Analysis and Computational Finance*, 2017. R package version 0.10-42.
- X. Wang, W. Pan, W. Hu, Y. Tian, and H. Zhang. Conditional distance correlation. *Journal of the American Statistical Association*, 110(512):1726–1734, 2015.
- X. Yan and J. Bien. Rare feature selection in high dimensions. *arXiv preprint arXiv:1803.06675*, 2018.
- X. Yan, J. Bien, et al. Hierarchical sparse modeling: A choice of two group lasso formulations. *Statistical Science*, 32(4):531–560, 2017.
- B. Zhang, N. Mohammed, V. Dave, and M. A. Hasan. Feature selection for classification under anonymity constraint. *arXiv preprint arXiv:1512.07158*, 2015.



---

# Analysis of Thompson Sampling for Graphical Bandits Without the Graphs

---

**Fang Liu**

The Ohio State University  
Columbus, Ohio 43210  
liu.3977@osu.edu

**Zizhan Zheng**

Tulane University  
New Orleans, LA 70118  
zzheng3@tulane.edu

**Ness Shroff**

The Ohio State University  
Columbus, Ohio 43210  
shroff.11@osu.edu

## Abstract

We study multi-armed bandit problems with graph feedback, in which the decision maker is allowed to observe the neighboring actions of the chosen action, in a setting where the graph may vary over time and is never fully revealed to the decision maker. We show that when the feedback graphs are undirected, the original Thompson Sampling achieves the optimal (within logarithmic factors) regret  $\tilde{O}\left(\sqrt{\beta_0(G)T}\right)$  over time horizon  $T$ , where  $\beta_0(G)$  is the average independence number of the latent graphs. To the best of our knowledge, this is the first result showing that the original Thompson Sampling is optimal for graphical bandits in the undirected setting. A slightly weaker regret bound of Thompson Sampling in the directed setting is also presented. To fill this gap, we propose a variant of Thompson Sampling, that attains the optimal regret in the directed setting within a logarithmic factor. Both algorithms can be implemented efficiently and do not require the knowledge of the feedback graphs at any time.

## 1 INTRODUCTION

Multi-Armed Bandits (MAB) models are quintessential models for sequential decision making. In the classical MAB setting, at each time, a policy must choose an action from a set of  $K$  actions with unknown probability distributions. Choosing an action  $i$  at time  $t$  reveals a random reward  $X_i(t)$  drawn from the probability distribution of action  $i$ . The goal is to find policies that minimize the expected loss due to uncertainty about actions' distributions over a given time horizon  $T$ .

In this work, we consider an important variant of bandit problems, called graphical bandits, where choosing an action  $i$  not only generates a reward from action  $i$ , but also reveals observations for a subset of the remaining actions. Graphical bandits are also known as bandits with graph-structured feedback or bandits with side-observations, in which the feedback model is specified by a sequence  $\{G_t\}_{t \geq 1}$  of feedback graphs. Each feedback graph  $G_t$  is a directed graph whose nodes correspond to the actions. An arc<sup>1</sup>  $(i, j)$  in the graph indicates that the agent observes the reward of action  $j$  if action  $i$  is chosen in that round.

Motivating examples for situations where side observations are available include viral marketing and online pricing. Consider the viral marketing problem, where a decision maker wants to find the user with the maximum influence in an online social network (e.g., Facebook) to offer a promotion (Carpentier and Valko [2016]). Each time the decision maker offers a promotion to a user, it also has an opportunity to survey the user's neighbors in the network regarding their potential interest in the same offer. This is possible when the online network has an additional survey feature that generates "side observations". For example, when user  $i$  is offered a promotion, her neighbors may be queried as follows: "User  $i$  was recently offered a promotion. Would you also be interested in the offer?". Here, choosing an action in the graphical bandit problem corresponds to choosing a user in the network and side-observations across actions are captured by the links in the social network.

Consider another example in the online pricing problem, where a seller is selling goods on the Internet. In each round, the seller announces a price for the product. Then, a buyer arrives and decides whether or not to purchase the product based on its private value. A purchase takes place if and only if the announced price is no more than

---

<sup>1</sup>We also use the notation  $i \rightarrow j$  to represent an arc from node  $i$  to node  $j$  for simplicity.

Table 1: Comparison of the existing algorithms

Algorithm	Reference	Graph	Undirected	Directed
Non-stochastic graphical bandits				
ExpBan	Mannor and Shamir [2011]	Informed	$O\left(\sqrt{\chi(G)T \log \bar{K}}\right)$	
ELP	Mannor and Shamir [2011]	Informed	$O\left(\sqrt{\beta_0(G)T \log \bar{K}}\right)$	$O\left(\sqrt{\chi(G)T \log \bar{K}}\right)$
Exp3-SET	Alon <i>et al.</i> [2013]	Uninformed	$O\left(\sqrt{\beta_0(G)T \log \bar{K}}\right)$	$O\left(\sqrt{mas(G)T \log \bar{K}}\right)$
Exp3-DOM	Alon <i>et al.</i> [2013]	Informed	$O\left(\sqrt{\beta_0(G)T \log(KT)} \log K\right)$	
Exp3.G	Alon <i>et al.</i> [2015]	Uninformed	$O\left(\sqrt{\beta_0(G)T} \log(KT)\right)$	
Exp3-IX	Kocák <i>et al.</i> [2014]	Uninformed	$O\left(\sqrt{\beta_0(G)T \log K \log(KT)}\right)$	
Stochastic graphical bandits				
Cohen’s Algo. 1	Cohen <i>et al.</i> [2016]	Without	$O\left(\sqrt{\beta_0(G)T \log \bar{K} \log(KT)}\right)$	
IDS-N/IDSN-LP	Liu <i>et al.</i> [2018]	Informed	$O\left(\sqrt{\chi(G)T \log \bar{K}}\right)$	
TS-N	Liu <i>et al.</i> [2018]	Without	$O\left(\sqrt{\chi(G)T \log \bar{K}}\right)$	
TS-N	this paper	Without	$O\left(\sqrt{\beta_0(G)T \log \bar{K}}\right)$	$O\left(\sqrt{mas(G)T \log \bar{K}}\right)$
TS-U	this paper	Without	$O\left(\sqrt{\beta_0(G)T \log K \log(KT)}\right)$	

its private value. At the end of the round, the seller observes whether or not the buyer purchased the product at the announced price. If the buyer purchases the product, then the seller knows that the buyer would have bought the product at any lower price. Otherwise, the seller knows that the buyer would not have bought the product at any higher price. Here, actions in the graphical bandit problem corresponds to the prices that the seller can choose. The feedback graph is a directed graph over the prices that a price is connected to a lower (higher) price if and only if they are both below (above) the private value of the buyer.

Graphical bandits have been studied in both non-stochastic (adversarial) domain by Mannor and Shamir [2011]; Alon *et al.* [2013, 2015]; Kocák *et al.* [2014], and stochastic domain by Caron *et al.* [2012]; Baccapatnam *et al.* [2014, 2017]; Tossou *et al.* [2017]; Liu *et al.* [2018]. Regret bounds as a function of combinatorial properties of the feedback graphs are characterized in different settings: undirected graphs vs directed graphs, time-invariant graphs vs time-variant graphs.

However, most of the existing works mentioned above require prior knowledge of the feedback graphs for their algorithms to run. These algorithms fall into either the informed setting (where the algorithms have access to the graph structure before making decisions) or the uninformed setting (where the algorithms have access to the graph for performing their updates after decisions).

The assumption that the feedback graph is disclosed to the decision maker does not hold in many real-world applications. For example, in the viral marketing problem, the third-party decision maker is not allowed to have the knowledge of the social network in order to protect the privacy of the users. In the online pricing problem, the private value of the buyer is never revealed to the seller. Thus the feedback graph is never disclosed to the seller. This motivates us to study the graphical bandits in a setting with limited information, where the feedback graphs are *never fully revealed* to the decision maker.

In this work, we study the graphical bandits without the graphs in a general setting, where the graphs are allowed to be time-variant and directed. Moreover, the only feedback available to the decision maker at the end of each round is the out-neighborhood of the chosen action in the latent graph, along with the rewards associated with the observed actions. Generally speaking, our results show that Thompson Sampling algorithms (introduced by Thompson [1933]) can achieve a regret bound of the form  $\tilde{O}\left(\sqrt{\beta_0(G)T}\right)$ , where  $\beta_0(G)$  is the average independence number<sup>2</sup> of the latent graphs, that is optimal within logarithmic factors. More specifically, we make the following contributions to graphical bandits without knowledge of the feedback graphs. (Table 1 summarizes the main results.)

<sup>2</sup>See Section 2.2 for a brief review of the combinatorial properties of graphs.

- We develop a problem-independent Bayesian regret bound for the vanilla Thompson Sampling algorithm (TS-N) for graphical bandits without the graphs. In the undirected setting, where the latent graphs are undirected, we show that TS-N obtains the optimal (within logarithmic factors) regret bound of  $O\left(\sqrt{\beta_0(G)T \log K}\right)$ , where  $\beta_0(G)$  is the average independence number of the latent graphs (Corollary 1). Our regret bound is much sharper than the form of  $O\left(\sqrt{\chi(G)T \log K}\right)$  that was shown by Liu *et al.* [2018], where  $\chi(G)$  is the average clique cover number of the latent graphs, as  $\beta_0(G) \leq \chi(G)$  in general. As far as we know, this is the first result showing that Thompson Sampling, without knowledge of the graph, can attain the optimal regret within logarithmic factors.
- In the directed setting, where the graphs are allowed to be directed, we show that TS-N achieves  $O\left(\sqrt{\text{mas}(G)T \log K}\right)$  regret in expectation, where  $\text{mas}(G)$  is the average maximal acyclic subgraph number of the latent graphs (Corollary 2). As a byproduct, our regret bounds for TS-N provide improved regret bounds for information directed sampling algorithms (IDS-N and IDSN-LP algorithms) proposed by Liu *et al.* [2018].
- We propose a variant of the Thompson Sampling algorithm, TS-U, that achieves a regret bound of  $O\left(\sqrt{\beta_0(G)T \log K \log(KT)}\right)$  for both the undirected and directed setting (Corollary 3). The regret bound of TS-U is optimal within logarithmic factors, and sharper than the state-of-the-art algorithm proposed by Cohen *et al.* [2016]. Our results offer a recipe for practitioners to choose algorithms for graphical bandits without the graphs. If the latent graphs are known to be undirected, one can choose TS-N for the best regret guarantee. Otherwise, TS-U is the choice with the best guarantee.

## 1.1 RELATED WORK

Graphical bandits were introduced in the non-stochastic domain by Mannor and Shamir [2011]. They propose the ExpBan algorithm that works in the time-invariant and informed setting, with the regret bound depending on the clique cover number. They also propose the ELP algorithm, that replaces the uniform distribution of Exp3 algorithm (proposed by Auer *et al.* [2002b]) with a distribution that maximizes the minimum probability to observe an action. An optimal (within logarithmic factors) regret bound of ELP is shown in the undirected setting.

However, the regret bound of the ELP depends on the

clique cover number in the directed setting. These results are improved by Alon *et al.* [2013]. They show that the vanilla Exp3 algorithm without mixing uniform distribution (Exp3-SET) achieves the same (but improved in the directed setting) regret bound as ELP, even in the uninformed setting. In the informed setting, they propose the Exp3-DOM algorithm, which is a variant of Exp3 algorithm with mixing uniform distribution over the dominating set of the feedback graph, achieves  $\tilde{O}\left(\sqrt{\beta_0(G)T}\right)$  regret. This regret bound is further attained by Exp3.G (Alon *et al.* [2015]) and Exp3-IX (Kocák *et al.* [2014]) in the uninformed setting. The Exp3.G algorithm is a variant of Exp3-DOM where it replaces the dominating set with the universal set. The Exp3-IX algorithm uses a novel implicit exploration idea. However, these algorithms still require the knowledge of the feedback graphs for performing updates after the decisions in the uninformed setting.

Graphical bandits have also been considered in the stochastic domain by Caron *et al.* [2012], who propose a natural variant of upper confidence bounds (introduced by Auer *et al.* [2002a]) algorithm (UCB-N) and provide a problem-dependent regret guarantee depending on the clique cover number. This result is improved by Buccapatnam *et al.* [2014] in the informed and time-invariant setting. Policies proposed by Buccapatnam *et al.* [2014], namely  $\epsilon_t$ -greedy-LP and UCB-LP, are shown to be asymptotically optimal, both in terms of the graph structure and time.

However, all of the afore-mentioned algorithms do not apply when the feedback graphs vary over the time and are never fully disclosed. Recently, researchers have developed new algorithms for graphical bandits in the setting with limited information, where the feedback graphs are time-variant, directed and never revealed to the decision maker. Cohen *et al.* [2016] propose an elimination-based algorithm that achieves the  $\tilde{O}\left(\sqrt{\beta_0(G)T}\right)$  regret bound. Tossou *et al.* [2017] analyze the Bayesian regret performance of Thompson Sampling for the graphical bandits and provide a regret bound depending on the maximal clique cover number of the latent graphs. This result is improved to a regret bound depending on the average clique cover number by Liu *et al.* [2018]. In this work, we provide sharper regret bounds for the vanilla Thompson Sampling (TS-N) and propose a variant of Thompson Sampling (TS-U) that obtains a better (within logarithmic factor) regret bound than the algorithm developed by Cohen *et al.* [2016].

Other related partial feedback models include label efficient bandit in Audibert and Bubeck [2010] and prediction with limited advice in Seldin *et al.* [2014], where side observations are limited by a budget. Graphical

bandits with Erdős-Rényi random graphs are studied by Kocák *et al.* [2016a]; Chen *et al.* [2016]; Liu *et al.* [2018]. Graphical bandits with noisy observations are studied by Kocák *et al.* [2016b]; Wu *et al.* [2015]. A survey of the graphical bandits refers to Valko [2016].

## 2 PROBLEM FORMULATION

### 2.1 STOCHASTIC BANDIT MODEL

We consider a Bayesian formulation of the stochastic  $K$ -armed bandit problem in which uncertainties are modeled as random variables. At each time  $t \in \mathbb{N}$ , a decision maker chooses an action  $A_t$  from a finite action set  $\mathcal{K} = \{1, \dots, K\}$  and receives the corresponding random reward  $Y_{t,A_t}$ . Without loss of generality, we assume the space of possible rewards  $\mathcal{Y} = [0, 1]$ . Note that the results in this work can be extended to the case where reward distributions are sub-Gaussian. There is a random variable  $Y_{t,a} \in \mathcal{Y}$  associated with each action  $a \in \mathcal{K}$  and  $t \in \mathbb{N}$ . We assume that  $\{Y_{t,a}, \forall a \in \mathcal{K}\}$  are independent for each time  $t$ . Let  $\mathbf{Y}_t \triangleq (Y_{t,a})_{a \in \mathcal{K}}$  be the vector of random variables at time  $t \in \mathbb{N}$ . The true reward distribution  $p^*$  is a distribution over  $\mathcal{Y}^{\mathcal{K}}$ , which is randomly drawn from the family of distributions  $\mathcal{P}$  and unknown to the decision maker. Conditioned on  $p^*$ ,  $(\mathbf{Y}_t)_{t \in \mathbb{N}}$  is an independent and identically distributed sequence with each element  $\mathbf{Y}_t$  sampled from the distribution  $p^*$ .

Let  $A^* \in \arg \max_{a \in \mathcal{K}} \mathbb{E}[Y_{t,a} | p^*]$  be the true optimal action conditioned on  $p^*$ . Then the  $T$  period regret of the decision maker is the expected difference between the total rewards obtained by an oracle that always chooses the optimal action and the accumulated rewards up to time horizon  $T$ . Formally, we study the expected regret

$$\mathbb{E}[R(T)] = \mathbb{E} \left[ \sum_{t=1}^T Y_{t,A^*} - Y_{t,A_t} \right], \quad (1)$$

where the expectation is taken over the randomness in the action sequence  $(A_1, \dots, A_T)$  and the outcomes  $(\mathbf{Y}_t)_{t \in \mathbb{N}}$  and over the prior distribution over  $p^*$ . This notion of regret is also known as *Bayesian regret*.

### 2.2 GRAPH FEEDBACK MODEL

In this problem, we assume the existence of side observations, which are described by a graph  $G_t = (\mathcal{K}, \mathcal{E}_t)$  over the action set for each time  $t$ . The graph  $G_t$  may be directed or undirected and can be dependent on time  $t$ . At each time  $t$ , the decision maker observes the reward  $Y_{t,A_t}$  for playing action  $A_t$  as well as the outcome  $Y_{t,a}$  for each action  $a \in \{a \in \mathcal{K} | (A_t, a) \in \mathcal{E}_t\}$ . Note that it becomes the classical bandit feedback setting when the

graph is empty (i.e., no edge exists) and it becomes the full-information (expert) setting when the graph is complete for all time  $t$ . Note that the graph  $G_t$  is *never fully revealed* to the decision maker.

Let  $\mathbf{G}_t \in \mathbb{R}^{K \times K}$  be the adjacent matrix that represents the deterministic graph feedback structure  $G_t$ . Let  $\mathbf{G}_t(i, j)$  be the element at the  $i$ -th row and  $j$ -th column of the matrix. Then  $\mathbf{G}_t(i, j) = 1$  if there exists an edge  $(i, j) \in \mathcal{E}_t$  and  $\mathbf{G}_t(i, j) = 0$  otherwise. Note that we assume  $\mathbf{G}_t(i, i) = 1$  for any  $i \in \mathcal{K}$ .

**Definition 1.** (Clique cover number) A *clique* of a graph  $G = (\mathcal{K}, \mathcal{E})$  is a subset  $S \subseteq \mathcal{K}$  such that the sub-graph formed by  $S$  and  $\mathcal{E}$  is a complete graph. A *clique cover* of a graph  $G = (\mathcal{K}, \mathcal{E})$  is a partition of  $\mathcal{K}$ , denoted by  $\mathcal{C}$ , such that  $S$  is a clique for each  $S \in \mathcal{C}$ . The cardinality of the smallest clique cover is called the *clique cover number*, which is denoted by  $\chi(G)$ .

**Definition 2.** (Independence number) An *independent set* of a graph  $G = (\mathcal{K}, \mathcal{E})$  is a subset  $S \subseteq \mathcal{K}$  such that no two  $i, j \in \mathcal{K}$  are connected by an edge in  $\mathcal{E}$ . The cardinality of a largest independent set is the *independence number* of  $G$ , denoted by  $\beta_0(G)$ .

Note that the independence number of a directed graph is equivalent to that of the undirected graph by ignoring arc orientation. We can also lift the notion of independence number of an undirected graph to directed graph through the notion of maximum acyclic subgraphs.

**Definition 3.** (Maximum acyclic subgraphs) An *acyclic subgraph* of  $G = (\mathcal{K}, \mathcal{E})$  is any graph  $G' = (\mathcal{K}', \mathcal{E}')$  such that  $\mathcal{K}' \subseteq \mathcal{K}$ , and  $\mathcal{E}' = \mathcal{E} \cap (\mathcal{K}' \times \mathcal{K}')$ , with no directed cycles. The cardinality of the largest such  $\mathcal{K}'$  is the *maximum acyclic subgraphs number*, denoted by  $mas(G)$ .

Note that  $mas(G) \geq \beta_0(G)$  in general. The equality holds when the graph  $G$  is undirected. In this work, we slightly abuse the notation of the above graph numbers and use  $\chi(G_t)$  and  $\chi(\mathbf{G}_t)$  interchangeably since  $\mathbf{G}_t$  fully characterizes the graph structure  $G_t$ .

### 2.3 RANDOMIZED POLICIES

We define all random variables with respect to a probability space  $(\Omega, \mathcal{F}, \mathbb{P})$ . Consider the filtration  $(\mathcal{F}_t)_{t \in \mathbb{N}}$  such that  $\mathcal{F}_t \subseteq \mathcal{F}$  is the  $\sigma$ -algebra generated by the observation history  $O_{t-1}$ . The observation history  $O_t$  includes all decisions, rewards and side observations from time 1 to time  $t$ . For each time  $t$ , the decision maker chooses an action based on the history  $O_{t-1}$  and possibly some randomness. Any policy of the decision maker can be viewed as a *randomized policy*  $\pi$ , which is an  $\mathcal{F}_t$ -adapted sequence  $(\pi_t)_{t \in \mathbb{N}}$ . For each time  $t$ , the decision maker chooses an action randomly according to

$\pi_t(\cdot) = \mathbb{P}(A_t = \cdot | \mathcal{F}_t)$ , which is a probability distribution over  $\mathcal{K}$ . Let  $\mathbb{E}[R(T, \pi)]$  be the Bayesian regret defined by (1) when the decisions  $(A_1, \dots, A_T)$  are chosen according to  $\pi$ .

Uncertainty about  $p^*$  induces uncertainty about the true optimal action  $A^*$ , which is described by a prior distribution  $\alpha_1$  of  $A^*$ . Let  $\alpha_t$  be the posterior distribution of  $A^*$  given the history  $O_{t-1}$ , i.e.,  $\alpha_t(\cdot) = \mathbb{P}(A^* = \cdot | \mathcal{F}_t)$ . Then,  $\alpha_{t+1}$  can be updated by Bayes rule given  $\alpha_t$ , decision  $A_t$ , reward  $Y_{t,A_t}$  and side observations. The *Shannon entropy* of  $\alpha_t$  is defined as  $H(\alpha_t) \triangleq -\sum_{i \in \mathcal{K}} \alpha_t(i) \log(\alpha_t(i))$ . We slightly abuse the notion of  $\pi_t$  and  $\alpha_t$  such that they represent distributions (or functions) over the finite set  $\mathcal{K}$  as well as vectors in a simplex  $\mathcal{S} \subset \mathbb{R}^K$ . Note that  $\mathcal{S} = \{\pi \in \mathbb{R}^K \mid \sum_{i=1}^K \pi(i) = 1, \pi(i) \geq 0, \forall i \in \mathcal{K}\}$ .

Let  $\Delta_t$  be the instantaneous regret vector such that the  $i$ -th coordinate,  $\Delta_t(i) \triangleq \mathbb{E}[Y_{t,A^*} - Y_{t,i} | \mathcal{F}_t]$ , is the expected regret of playing action  $i$  at time  $t$ . Let  $g_t$  be the information gain vector such that the  $i$ -th coordinate,  $g_t(i) = \mathbb{E}[H(\alpha_t) - H(\alpha_{t+1}) | \mathcal{F}_t, A_t = i]$ , is the expected information gain of playing action  $i$  at time  $t$ . Note that the information gain of playing action  $i$  consists of that of observing the reward  $Y_{t,i}$  and possibly some side observations. We define the information gain of observing action  $a$  (i.e.,  $Y_{t,a}$ ) as  $h_t(a) \triangleq I_t(A^*; Y_{t,a})$ , which is the *mutual information* under the posterior distribution between random variables  $A^*$  and  $Y_{t,a}$ . Let  $D(\cdot || \cdot)$  be the *Kullback-Leibler* divergence between two distributions<sup>3</sup>. By the definition of mutual information, we have that  $I_t(A^*; Y_{t,a}) \triangleq$

$$D(\mathbb{P}((A^*, Y_{t,a}) \in \cdot | \mathcal{F}_t) || \mathbb{P}(A^* \in \cdot | \mathcal{F}_t) \mathbb{P}(Y_{t,a} \in \cdot | \mathcal{F}_t)). \quad (2)$$

At each time  $t$ , a randomized policy updates  $\alpha_t$ , and makes a decision according to a sampling distribution  $\pi_t$ . For any randomized policy, we define the *information ratio* (Russo and Van Roy [2016]) of sampling distribution  $\pi_t$  at time  $t$  as

$$\Psi_t(\pi_t) \triangleq (\pi_t^T \Delta_t)^2 / (\pi_t^T g_t). \quad (3)$$

Note that  $\pi_t^T \Delta_t$  is the expected instantaneous regret of the sampling distribution  $\pi_t$ , and  $\pi_t^T g_t$  is the expected information gain of the sampling distribution  $\pi_t$ . So the information ratio  $\Psi_t(\pi_t)$  measures the “energy” cost (which is the square of the expected instantaneous regret) per bit of information acquired.

<sup>3</sup>If  $P$  is absolutely continuous with respect to  $Q$ , then  $D(P||Q) = \int \log\left(\frac{dP}{dQ}\right) dP$ , where  $\frac{dP}{dQ}$  is the Radon-Nikodym derivative of  $P$  w.r.t.  $Q$ .

---

### Algorithm 1 TS-N algorithm

---

**Input:** time horizon  $T$

**for**  $t$  **from** 1 **to**  $T$  **do**

**Updating statistics:** compute  $\alpha_t$  accordingly.

**Generating policy:**  $\pi_t = \alpha_t$ .

**Sampling:** sample  $A_t$  according to  $\pi_t$ , play action  $A_t$  and receive reward  $Y_{t,A_t}$ .

**Observations:** observe  $Y_{t,a}$  if  $(A_t, a) \in \mathcal{E}_t$ , where  $G_t = (\mathcal{K}, \mathcal{E}_t)$  is the latent graph.

**end for**

---

## 2.4 THOMPSON SAMPLING

Thompson Sampling algorithm simply samples actions according to the posterior probability that they are optimal. In particular, actions are chosen randomly at time  $t$  according to the sampling distribution  $\pi_t = \alpha_t$ . This conceptually elegant policy can be efficiently implemented. Consider the case where  $\mathcal{P} = \{p_\theta\}_{\theta \in \Theta}$  is some parametric family of distributions. The true reward distribution  $p^*$  is indexed by  $\theta^* \in \Theta$  in the sense that  $p^* = p_{\theta^*}$ . Practical implementations of Thompson Sampling consist of two steps. First, an index  $\hat{\theta}_t \sim \mathbb{P}(\theta^* \in \cdot | \mathcal{F}_t)$  is sampled from the posterior distribution. Then, the algorithm selects the action  $A_t = \arg \max_{a \in \mathcal{K}} \mathbb{E}[Y_{t,a} | \theta^* = \hat{\theta}_t]$  that would be optimal if the sampled parameter were the true parameter. Given the observation of playing  $A_t$ , the posterior distribution is updated by Bayes’ rule.

## 3 VANILLA THOMPSON SAMPLING

In this section, we show that a vanilla Thompson Sampling algorithm, TS-N as shown in Algorithm 1, obtains optimal regret (within a logarithmic factor) in the undirected setting. A slightly weaker regret bound in the directed setting is also presented.

TS-N is the Thompson Sampling algorithm for graphical bandits such that  $\pi_t = \alpha_t$ , where  $\alpha_t$  is updated based on all the observations available, without additional modifications. It naturally keeps the information ratio  $\Psi_t(\pi_t)$  bounded as well as balances between having low expected instantaneous regret (a.k.a. exploitation) and obtaining knowledge about the optimal action (a.k.a. exploration). If the information ratio is bounded, then the expected regret is bounded in terms of the maximum amount of information one could expect to acquire, which is at most the entropy of the prior distribution of  $A^*$ , i.e.,  $H(\alpha_1)$ . First, we bound the information ratio of

TS-N in terms of the key quantity

$$Q_t(\pi_t) = \sum_{i \in \mathcal{K}} \frac{\pi_t(i)}{\sum_{j: j \xrightarrow{t} i} \pi_t(j)}. \quad (4)$$

Note that  $j \xrightarrow{t} i$  represents an arc  $(j, i)$  in graph  $G_t$ .

**Proposition 1.** *If  $\pi_t = \alpha_t$ , then the information ratio satisfies  $\Psi_t(\pi_t) \leq \frac{1}{2}Q_t(\pi_t)$  almost surely.*

Proposition 1 is a tight bound for the information ratio of the vanilla Thompson Sampling (Thompson [1933]). If the graph is empty (i.e., there is no edges in the graph), then the quantity  $Q_t(\pi_t)$  equals to  $K$ . If the graph is complete, then the quantity  $Q_t(\pi_t)$  equals to 1. These recover the information ratio bounds shown by Russo and Van Roy [2016]. Also, the quantity  $Q_t(\pi_t)$  is upper bounded by clique cover number  $\chi(G_t)$  as one can separate the sum by cliques and dropping the weights out of the clique. This recovers the information ratio bound shown by Liu *et al.* [2018]. Proposition 1 allows us to show a tighter regret bound of Thompson Sampling for graphical bandits. Next, we bound the regret of TS-N in terms of the quantity  $Q_t(\pi_t)$ .

**Theorem 1.** *The regret of TS-N satisfies*

$$\mathbb{E}[R(T, \pi)] \leq \sqrt{\frac{1}{2} \sum_{t=1}^T \mathbb{E}[Q_t(\alpha_t)] H(\alpha_1)}. \quad (5)$$

Note that the entropy is bounded, i.e.,  $H(\alpha_1) \leq \log K$ . It has been shown by Mannor and Shamir [2011]; Alon *et al.* [2014] that the quantity  $Q_t(\pi_t)$  is related to the graph numbers irrespective of the choice of the distribution  $\pi_t$ . The following graph-theoretic result shows that the quantity  $Q_t(\pi_t)$  is bounded by the independence number of the latent graph if the graph is undirected.

**Lemma 1.** (Lemma 3 in Mannor and Shamir [2011]) *Let  $G = (\mathcal{K}, \mathcal{E})$  be an undirected graph. For any distribution  $\pi$  over  $\mathcal{K}$ ,*

$$\sum_{i \in \mathcal{K}} \frac{\pi(i)}{\sum_{j: j \xrightarrow{t} i} \pi(j)} \leq \beta_0(G). \quad (6)$$

The following regret result of TS-N follows immediately from Theorem 1 and Lemma 1.

**Corollary 1.** *In the undirected setting, the regret of TS-N satisfies*

$$\mathbb{E}[R(T, \pi)] \leq \sqrt{\frac{1}{2} \sum_{t=1}^T \beta_0(G_t) H(\alpha_1)}. \quad (7)$$

As far as we know, this is the best regret bound for graphical bandits without the graphs. First, an information-theoretic lower bound of graphical bandits has been

shown by Mannor and Shamir [2011]; Alon *et al.* [2014] to be  $\Omega(\sqrt{\beta_0(G)T})$ . So Corollary 1 shows that TS-N obtains the optimal regret (within a logarithmic factor) in the undirected setting. Moreover, the bound proven in Corollary 1 is tighter than the  $O(\sqrt{\chi(G)TH(\alpha_1)})$  bound of TS-N shown by Liu *et al.* [2018] as  $\beta_0(G) \leq \chi(G)$ . At last, Corollary 1 shows that TS-N enjoys better regret bound than Cohen's Algorithm 1 developed by Cohen *et al.* [2016] in the undirected setting, both of which do not require the knowledge of the feedback graphs.

We now turn to the directed setting. The following graph-theoretic result shows that the quantity  $Q_t(\pi_t)$  is upper-bounded by the maximum acyclic subgraph number if the graph is directed.

**Lemma 2.** (Lemma 10 in Alon *et al.* [2014]) *Let  $G = (\mathcal{K}, \mathcal{E})$  be a directed graph. For any distribution  $\pi$  over  $\mathcal{K}$ ,*

$$\sum_{i \in \mathcal{K}} \frac{\pi(i)}{\sum_{j: j \xrightarrow{t} i} \pi(j)} \leq \text{mas}(G). \quad (8)$$

The following regret result of TS-N follows immediately from Theorem 1 and Lemma 2.

**Corollary 2.** *In the directed setting, the regret of TS-N satisfies*

$$\mathbb{E}[R(T, \pi)] \leq \sqrt{\frac{1}{2} \sum_{t=1}^T \text{mas}(G_t) H(\alpha_1)}. \quad (9)$$

The bound proven in Corollary 2 is tighter than the  $O(\sqrt{\chi(G)TH(\alpha_1)})$  bound of TS-N shown by Liu *et al.* [2018] since  $\text{mas}(G) \leq \chi(G)$ . However, there is a gap between the lower bound and the upper bound shown in Corollary 2. Though  $\beta_0(G) = \text{mas}(G)$  when the graph is undirected, the gap between them can be large in general directed graphs. For example, consider a directed graph  $G_0 = (\mathcal{K}, \mathcal{E})$  such that  $\text{arc}(i, j) \in \mathcal{E}$  if and only if  $i \leq j$ . It is clear that  $\beta_0(G_0) = 1$  and  $\text{mas}(G_0) = K$ . This leads us to consider a more sophisticated randomized policy. In the next section, we show that a modified Thompson Sampling algorithm results in an optimal (within a logarithmic factor) regret bound in the general setting.

**Remark 1.** *The bounds proven in Corollary 1 and 2 hold for IDS-N and IDSN-LP developed by Liu *et al.* [2018] since the information ratio of IDS-N and IDSN-LP are bounded by the information ratio of TS-N almost surely. So our results also provide tighter bounds for IDS-N and IDSN-LP algorithms. Note that IDS-N and IDSN-LP algorithms require prior knowledge of the feedback graphs.*

---

**Algorithm 2** TS-U algorithm

---

**Input:** time horizon  $T$  and parameter  $\epsilon \in [0, 1]$

**for**  $t$  **from** 1 **to**  $T$  **do**

**Updating statistics:** compute  $\alpha_t$  accordingly.

**Generating policy:**  $\pi_t = (1 - \epsilon)\alpha_t + \epsilon/K$ .

**Sampling:** sample  $A_t$  according to  $\pi_t$ , play action  $A_t$  and receive reward  $Y_{t,A_t}$ .

**Observations:** observe  $Y_{t,a}$  if  $(A_t, a) \in \mathcal{E}_t$ , where  $G_t = (\mathcal{K}, \mathcal{E}_t)$  is the latent graph.

**end for**

---

## 4 THOMPSON SAMPLING WITH EXPLORATION

In this section, we show that a mixture of Thompson Sampling and uniform sampling, TS-U as shown in Algorithm 2, obtains the optimal regret within a logarithmic factor in the directed setting.

TS-U algorithm is a variant of Thompson Sampling with explicit exploration that allows the algorithm to explore some suboptimal actions with large out-degrees. The collected side observations from the uniform sampling allows us to capture the latent graph information, thus yielding a regret bound in terms of the independence number.

As shown in Algorithm 2, TS-U is a randomized policy such that  $\pi_t = (1 - \epsilon)\alpha_t + \epsilon/K$  for some parameter  $\epsilon \in [0, 1]$ . The implementation and computation of TS-U is quite efficient. At each time  $t$ , TS-U algorithm plays vanilla Thompson Sampling with probability  $1 - \epsilon$  and plays uniform sampling with probability  $\epsilon$ . While the expected information gain diminishes, the expected instantaneous regret is bounded away from zero due to uniform sampling. Thus, the classical analysis of bounding the information ratio  $\Psi_t(\pi_t)$  does not work any more. Fortunately, the linear form of  $\pi_t$  allows us to bound the regret of TS-U by the regret due to uniform sampling plus the regret from Thompson Sampling, as shown in Theorem 2. Indeed, our techniques work for any variant of Thompson Sampling that is a linear combination of  $\alpha_t$  and some other distributions.

**Theorem 2.** *The regret of TS-U satisfies*

$$\mathbb{E}[R(T, \pi)] \leq \epsilon T + \sqrt{\frac{1}{2} \sum_{t=1}^T \mathbb{E}[Q_t(\pi_t)] H(\alpha_1)}. \quad (10)$$

Theorem 2 shows that the regret of TS-U consists of two parts, the regret from the uniform sampling and the regret from the Thompson Sampling. Note that the regret from the Thompson Sampling takes into account the information gain from the uniform sampling as the term

$Q_t(\pi_t)$  depends on  $\pi_t$  rather than  $\alpha_t$ . This allows us to use the following graph-theoretic result to bound the term  $Q_t(\pi_t)$ , thus the regret of TS-U, by the independence number of the latent graph.

**Lemma 3.** (Lemma 5 in Alon *et al.* [2015]) *Let  $G = (\mathcal{K}, \mathcal{E})$  be a directed graph. For any distribution  $\pi$  over  $\mathcal{K}$  such that  $\pi(i) \geq \eta$  for all  $i \in \mathcal{K}$  for some constant  $0 < \eta < 0.5$ . Then*

$$\sum_{i \in \mathcal{K}} \frac{\pi(i)}{\sum_{j: j \xrightarrow{t} i} \pi(j)} \leq 4\beta_0(G) \log \left( \frac{4K}{\beta_0(G)\eta} \right). \quad (11)$$

Lemma 3 shows that the quantity  $Q_t(\pi_t)$  can be bounded by the independence number of the directed graph if  $\pi_t$  is bounded away from zero. The uniform sampling part of TS-U allows the sampling distribution to satisfy this condition. By Theorem 2 and taking  $\eta = \epsilon/K$  in Lemma 3, we have the following result.

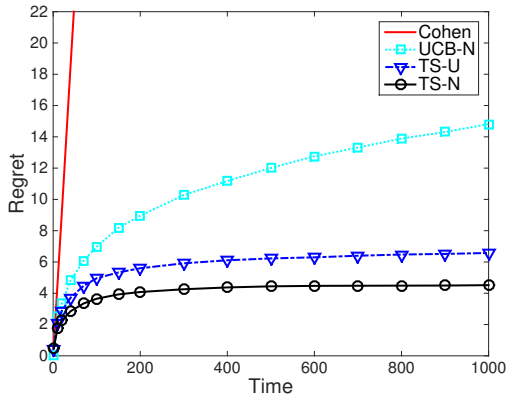
**Corollary 3.** *If  $\epsilon = 1/\sqrt{T}$ , then the regret of TS-U satisfies*

$$\mathbb{E}[R(T, \pi)] = O \left( \sqrt{\log(KT) \sum_{t=1}^T \beta_0(G_t) H(\alpha_1)} \right). \quad (12)$$

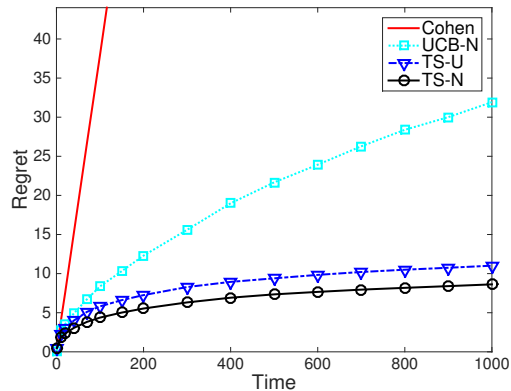
Comparing the regret bound in Corollary 3 to the lower bound,  $\Omega(\sqrt{\beta_0(G)T})$ , the TS-U algorithm obtains the optimal regret within a logarithmic factor in the general setting. Moreover, Corollary 3 shows that TS-U enjoys a sharper (by a logarithmic factor) regret bound than Cohen's Algorithm 1 developed by Cohen *et al.* [2016] in the directed setting, both of which do not require the knowledge of the feedback graphs. As far as we know, this is the best-known regret bound for graphical bandits without the graphs. Finally, note that a comparison between Corollary 1 and Corollary 3 reveals that a symmetric observation system (i.e., undirected feedback graphs) enjoys better regret than an asymmetric observation system as the regret bound of TS-N is sharper by a logarithmic factor than the bound in Corollary 3 in the undirected setting.

**Remark 2.** *We present TS-U algorithm with fixed exploration rate  $\epsilon$  for simplicity. It is easy to verify that the regret result of TS-U still holds if one uses some appropriate decreasing exploration rate sequence  $\{\epsilon_t\}$ . For example, when  $\epsilon_t = 1/t$ , Theorem 2 holds by replacing the term  $\epsilon T$  with  $\log T$ . Then the regret result of the corresponding TS-U algorithm follows. In practice, we recommend practitioners to use decreasing exploration rates.*

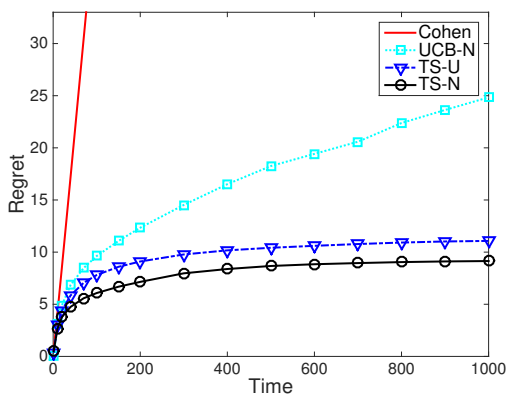
**Remark 3.** *In the informed setting (i.e., when the feedback graph are revealed to the decision maker before the*



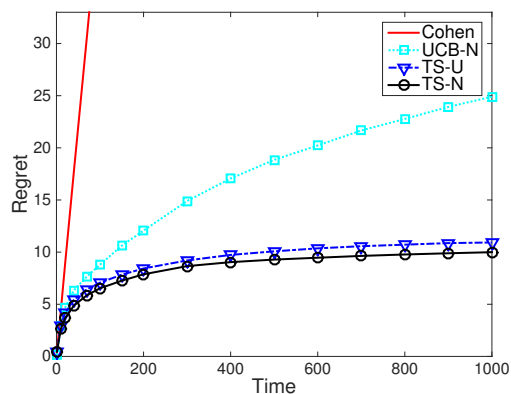
(a) Time-invariant graphs



(a) Time-invariant graphs



(b) Time-variant graphs



(b) Time-variant graphs

Figure 1: Regret comparison under undirected graph feedback

Figure 2: Regret comparison under directed graph feedback

decisions), one may propose a variant of TS-U such that it restricts the exploration set to the dominating set of the feedback graph. In other words, one may replace uniform sampling over all the actions with uniform sampling over only the dominating set. The regret bound in Corollary 3 still holds by a variant of Lemma 3 shown in Alon et al. [2014].

## 5 NUMERICAL RESULTS

This section presents numerical results from experiments that evaluate the effectiveness of Thompson Sampling based policies in comparison to UCB-N and Cohen’s algorithm. We consider the classical Beta-Bernoulli bandit problem with independent actions. The reward of each action  $i$  is a Bernoulli( $\mu_i$ ) random variable and  $\mu_i$  is independently drawn from Beta(1,1). The implementations of TS-N and TS-U are shown in Algorithms 3 and 4. In the experiment, we set  $K = 5$ ,  $T = 1000$  and  $\epsilon_t = 1/t$  as suggested by Remark 2. All the regret results

are averaged over 1000 trials.

Figure 1 presents the cumulative regret results under undirected graph feedback. For the time-invariant case shown in Figure 1a, we use a graph with 2 cliques, presented in Figure 3. Figure 2 presents the cumulative regret results under directed graph feedback. For the time-invariant case shown in Figure 2a, we use the graph  $G = (\mathcal{K}, \mathcal{E})$  such that  $(i, j) \in \mathcal{E}$  if and only if  $i \leq j$ , presented in Figure 4. For the time-variant cases shown in Figures 1b and 2b, the sequences of graphs are generated by the Erdős-Rényi model with parameter  $p_t^4$  drawn from the uniform distribution over  $[0, 0.2]$ .

We find that TS-N and TS-U outperform the alternative algorithms, which is consistent with the empirical observation in the bandit feedback setting (Chapelle and Li [2011]). However, TS-N has better empirical performance in the tested settings even though we have

<sup>4</sup>For each time  $t$  and each pair  $(i, j)$ ,  $(i, j) \in \mathcal{E}_t$  with probability  $p_t$



---

**Algorithm 3** TS-N (Bernoulli case)

---

**Input:** time horizon  $T$   
 For each arm  $i$ , set  $S_i = 1$  and  $F_i = 1$   
**for**  $t$  **from** 1 **to**  $T$  **do**  
   For each arm  $i$ , sample  $\theta_i$  from  $\text{Beta}(S_i, F_i)$ .  
   Play action  $A_t = \arg \max_{i \in \mathcal{K}} \theta_i$ .  
   **for** all  $a \in \mathcal{K}$  such that  $(A_t, a) \in \mathcal{E}_t$  **do**  
      $S_a = S_a + Y_{t,a}$  and  $F_a = F_a + 1 - Y_{t,a}$ .  
   **end for**  
**end for**

---



---

**Algorithm 4** TS-U (Bernoulli case)

---

**Input:** time horizon  $T$  and  $\{\epsilon_t\}_{t \geq 1}$   
 For each arm  $i$ , set  $S_i = 1$  and  $F_i = 1$   
**for**  $t$  **from** 1 **to**  $T$  **do**  
   Sample  $\beta_t$  from uniform distribution over  $[0, 1]$ .  
   **if**  $\beta_t < \epsilon_t$  **then**  
     Play action  $A_t$  drawn uniformly from  $\mathcal{K}$ .  
   **else**  
     For each arm  $i$ , sample  $\theta_i$  from  $\text{Beta}(S_i, F_i)$ .  
     Play action  $A_t = \arg \max_{i \in \mathcal{K}} \theta_i$ .  
   **end if**  
   **for** all  $a \in \mathcal{K}$  such that  $(A_t, a) \in \mathcal{E}_t$  **do**  
      $S_a = S_a + Y_{t,a}$  and  $F_a = F_a + 1 - Y_{t,a}$ .  
   **end for**  
**end for**

---

proven a better regret bound for TS-U under directed feedback graphs. The average regrets of Cohen’s algorithm are dramatically larger than that of Thompson Sampling based policies. For this reason, parts of Cohen’s algorithm are omitted from Figures 1 and 2.

## 6 CONCLUSION

We have provided regret analysis of Thompson Sampling for graphical bandits without knowing the feedback graphs at any time. We show that the regret of TS-N is bounded by  $O\left(\sqrt{\text{mas}(G)T \log K}\right)$  in the general setting. In the undirected setting,  $\text{mas}(G) = \beta_0(G)$ , and the resulting regret bound is optimal up to a logarithmic factor. As far as we know, this is the first result that shows that Thompson Sampling, even without the knowledge of the graph, can attain the optimal regret in the graphical bandits. As a byproduct, our analysis for TS-N provide improved regret bounds for information directed sampling algorithms (IDS-N and IDS-N-LP algorithms) proposed by Liu *et al.* [2018] in the informed setting.

We have proposed a variant of Thompson Sampling, TS-U, that mixes Thompson Sampling with

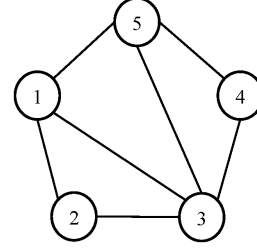


Figure 3: Graph structure for the experiment under time-invariant and undirected graph

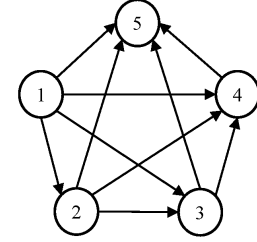


Figure 4: Graph structure for the experiment under time-invariant and directed graph

uniform sampling. This modification allows the algorithm to capture the graph structure and obtain  $O\left(\sqrt{\beta_0(G)T \log K \log(KT)}\right)$  regret bound in the directed setting, which is optimal within a logarithmic factor. Our results offer a recipe for practitioners to choose algorithms for graphical bandits without knowledge of the graphs. If the latent graphs are known to be undirected, one can choose TS-N for the best regret guarantee. Otherwise, TS-U is the choice with the best guarantee in the general (directed) setting.

## Acknowledgements

This work has been supported in part by a grant from DTRA grant HDTRA1-14-1-0058 and a grant from the Board of Regents of the State of Louisiana LEQSF(2017-19)-RD-A-15. This work has also been supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT), (2017-0-00692, Transport-aware Streaming Technique Enabling Ultra Low-Latency AR/VR Services).

## References

- Noga Alon, Nicolo Cesa-Bianchi, Claudio Gentile, and Yishay Mansour. From bandits to experts: A tale of domination and independence. In *Advances in Neural Information Processing Systems*, pages 1610–1618, 2013.
- Noga Alon, Nicolo Cesa-Bianchi, Claudio Gentile, Shie Mannor, Yishay Mansour, and Ohad Shamir. Non-stochastic multi-armed bandits with graph-structured feedback. *arXiv preprint arXiv:1409.8428*, 2014.
- Noga Alon, Nicolo Cesa-Bianchi, Ofer Dekel, and Tomer Koren. Online learning with feedback graphs: Beyond bandits. In *COLT*, pages 23–35, 2015.
- Jean-Yves Audibert and Sébastien Bubeck. Regret bounds and minimax policies under partial monitoring. *Journal of Machine Learning Research*, 11(Oct):2785–2836, 2010.
- Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.
- Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2002.
- Swapna Buccapatnam, Atilla Eryilmaz, and Ness B. Shroff. Stochastic bandits with side observations on networks. *SIGMETRICS Perform. Eval. Rev.*, 42(1):289–300, June 2014.
- Swapna Buccapatnam, Fang Liu, Atilla Eryilmaz, and Ness B Shroff. Reward maximization under uncertainty: Leveraging side-observations on networks. *arXiv preprint arXiv:1704.07943*, 2017.
- S. Caron, B. Kveton, M. Lelarge, and S. Bhagat. Leveraging side observations in stochastic bandits. In *UAI*, pages 142–151. AUAI Press, 2012.
- Alexandra Carpentier and Michal Valko. Revealing graph bandits for maximizing local influence. In *International Conference on Artificial Intelligence and Statistics*, pages 10–18, 2016.
- Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. In *Advances in neural information processing systems*, pages 2249–2257, 2011.
- Wei Chen, Yajun Wang, Yang Yuan, and Qinshi Wang. Combinatorial multi-armed bandit and its extension to probabilistically triggered arms. *Journal of Machine Learning Research*, 17(50):1–33, 2016.
- Alon Cohen, Tamir Hazan, and Tomer Koren. Online learning with feedback graphs without the graphs. *CoRR*, abs/1605.07018, 2016.
- Tomáš Kocák, Gergely Neu, Michal Valko, and Rémi Munos. Efficient learning by implicit exploration in bandit problems with side observations. In *Advances in Neural Information Processing Systems*, pages 613–621, 2014.
- Tomáš Kocák, Gergely Neu, and Michal Valko. Online learning with erdős-rényi side-observation graphs. In *Uncertainty in Artificial Intelligence*, 2016.
- Tomáš Kocák, Gergely Neu, and Michal Valko. Online learning with noisy side observations. In *AISTATS*, pages 1186–1194, 2016.
- Fang Liu, Swapna Buccapatnam, and Ness Shroff. Information directed sampling for stochastic bandits with graph feedback. In *AAAI*, 2018.
- Shie Mannor and Ohad Shamir. From bandits to experts: On the value of side-observations. In *NIPS*, pages 684–692, 2011.
- Daniel Russo and Benjamin Van Roy. An information-theoretic analysis of thompson sampling. *Journal of Machine Learning Research*, 17(68):1–30, 2016.
- Yevgeny Seldin, Peter Bartlett, Koby Crammer, and Yasin Abbasi-Yadkori. Prediction with limited advice and multiarmed bandits with paid observations. In *International Conference on Machine Learning*, pages 280–287, 2014.
- William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.
- Aristide Tossou, Christos Dimitrakakis, and Devdatt Dubhashi. Thompson sampling for stochastic bandits with graph feedback. In *AAAI Conference on Artificial Intelligence*, 2017.
- Michal Valko. *Bandits on graphs and structures*. PhD thesis, École normale supérieure de Cachan-ENS Cachan, 2016.
- Yifan Wu, András György, and Csaba Szepesvári. Online learning with gaussian payoffs and side observations. In *Advances in Neural Information Processing Systems*, pages 1360–1368, 2015.

---

# Structured nonlinear variable selection

---

**Magda Gregorová**  
Geneva School of Business Administration, HES-SO, Switzerland  
University of Geneva, Switzerland

**Alexandros Kalousis**

**Stéphane Marchand-Maillet**  
University of Geneva  
Switzerland

## Abstract

We investigate structured sparsity methods for variable selection in regression problems where the target depends nonlinearly on the inputs. We focus on general nonlinear functions not limiting a priori the function space to additive models. We propose two new regularizers based on partial derivatives as nonlinear equivalents of group lasso and elastic net. We formulate the problem within the framework of learning in reproducing kernel Hilbert spaces and show how the variational problem can be reformulated into a more practical finite dimensional equivalent. We develop a new algorithm derived from the ADMM principles that relies solely on closed forms of the proximal operators. We explore the empirical properties of our new algorithm for Nonlinear Variable Selection based on Derivatives (NVSD) on a set of experiments and confirm favourable properties of our structured-sparsity models and the algorithm in terms of both prediction and variable selection accuracy.

## 1 INTRODUCTION

We are given a set of  $n$  input-output pairs  $\{(\mathbf{x}^i, y^i) \in (\mathcal{X} \times \mathcal{Y}) : \mathcal{X} \subseteq \mathbb{R}^d, \mathcal{Y} \subseteq \mathbb{R}, i \in \mathbb{N}_n\}$  sampled i.i.d. according to an unknown probability measure  $\rho$ . Our task is to learn a regression function  $f : \mathcal{X} \rightarrow \mathcal{Y}$  with minimal expected squared error loss  $\mathcal{L}(f) = \mathbb{E}(y - f(\mathbf{x}))^2 = \int (y - f(\mathbf{x}))^2 d\rho(\mathbf{x}, y)$ .

We follow the standard theory of regularised learning where  $\hat{f}$  is learned by minimising the regularised empirical squared error loss  $\hat{\mathcal{L}}(f) = \frac{1}{n} \sum_i^n (y^i - f(\mathbf{x}^i))^2$

$$\hat{f} = \underset{f}{\operatorname{argmin}} \hat{\mathcal{L}}(f) + \tau \mathcal{R}(f) . \quad (1)$$

In the above,  $\mathcal{R}(f)$  is a suitable penalty typically based on some prior assumption about the function space (e.g. smoothness), and  $\tau > 0$  is a suitable regularization hyper-parameter. The principal assumption we consider in this paper is that the function  $f$  is sparse with respect to the original input space  $\mathcal{X}$ , that is it depends only on  $l \ll d$  input variables.

Learning with variable selection is a well-established and rather well-explored problem in the case of linear models  $f(\mathbf{x}) = \sum_a^d x_a w_a$ , e.g. Hastie et al. (2015). The main ideas from linear models have been successfully transferred to additive models  $f(\mathbf{x}) = \sum_a^d f_a(x_a)$ , e.g. Ravikumar et al. (2007), Bach (2009), Koltchinskii and Yuan (2010), and Yin et al. (2012), or to additive models with interactions  $f(\mathbf{x}) = \sum_a^d f_a(x_a) + \sum_{a < b}^d f_{a,b}(x_a, x_b)$ , e.g. Lin and Zhang (2006) and Tyagi et al. (2016).

However, sparse modelling of general non-linear functions is more intricate. A promising stream of works focuses on the use of non-linear (conditional) cross-covariance operators arising from embedding probability measures into Hilbert function spaces, e.g. Yamada et al. (2014) and Chen et al. (2017).

In this work, we follow an alternative approach proposed in Rosasco et al. (2013) based on partial derivatives and develop new regularizers to promote structured sparsity with respect to the original input variables. We stress that our objective here is not to learn new data representations nor learn sparse models in some latent feature space, e.g. Gurram and Kwon (2014). Nor is it to learn models sparse in the data instances (in the sense of support vectors, e.g. Chan et al. (2007)). We aim at selecting the relevant input variables, the relevant dimensions of the input vectors  $\mathbf{x} \in \mathbb{R}^d$ .

After a brief review of the regularizers used in Rosasco et al. (2013) for individual variable selection in non-linear model learning (similar in spirit to lasso Tibshi-

rani (1996)) we propose two extensions motivated by the linear structured-sparsity learning literature. Using suitable norms of the partial derivatives we propose the non-linear versions of the group lasso Yuan and Lin (2006) and the elastic net Zou and Hastie (2005).

We pose our problem into the framework of learning in the reproducing kernel Hilbert space (RKHS). We extend the representer theorem to show that the minimiser of (1) with our new regularizers  $\mathcal{R}(f)$  can be conveniently written as a linear combination of kernel functions and their partial derivatives evaluated over the training set.

We further propose a new reformulation of the equivalent finite dimensional learning problem, which allows us to develop a new algorithm (NVSD) based on the Alternating Direction Method of Multipliers (ADMM) Boyd (2010). This is a generic algorithm that can be used (with small alterations) for all regularizers we discuss here. At each iteration, the algorithm needs to solve a single linear problem, perform a proximal step resulting in a soft-thresholding operation, and do a simple additive update of the dual variables. Unlike Rosasco et al. (2013), which uses approximations of the proximal operator, our algorithm is based on proximals admitting closed forms for all the discussed regularizers, including the one suggested previously in Rosasco et al. (2013). Furthermore, by avoiding the approximations in the proximal step, the algorithm directly provides also the learned sparsity patterns over the training set (up to the algorithmic convergence precision).

We explore the effect of the proposed regularizers on model learning on synthetic and real-data experiments, and confirm the superior performance of our methods in comparison to a range of baseline methods when learning structured-sparse problems. Finally, we conclude by discussing the advantages and shortcomings of the current proposal and outline some directions for future work.

## 2 REGULARIZERS FOR VARIABLE SELECTION

In Rosasco et al. (2013) the authors propose to use the partial derivatives of the function with respect to the input vector dimensions  $\{\partial_a f : a \in \mathbb{N}_d\}$  to construct a regularizer promoting sparsity. The partial derivative evaluated at an input point  $\partial_a f(\mathbf{x})$  is the rate of change of the function at that point with respect to  $x_a$  holding the other input dimensions fixed. Intuitively, when the function does not depend on an input variable (input dimension  $a$ ), its evaluations do not change with changes in the input variable:  $\partial_a f(\mathbf{x}) = 0$  at all points  $\mathbf{x} \in \mathcal{X}$ . A natural measure of the size of the partial derivatives

across the space  $\mathcal{X}$  is the  $L_2$  norm

$$\|\partial_a f\|_{L_2} = \sqrt{\int_{\mathcal{X}} |\partial_a f(\mathbf{x})|^2 d\rho_{\mathbf{x}}(\mathbf{x})} \quad (2)$$

**Remark 1.** *At this point we wish to step back and make a link to the linear models  $f(\mathbf{x}) = \sum_a^d x_a w_a$ . The partial derivatives with respect to any of the  $d$  dimensions of the input vector  $\mathbf{x}$  are the individual elements of the  $d$ -dimensional parameter vector  $\mathbf{w}$ ,  $\partial_a f(\mathbf{x}) = w_a$ , and this at every point  $\mathbf{x} \in \mathcal{X}$ . For the linear model we thus have  $\|\partial_a f\|_{L_2} = |w_a|$ . Sparsity inducing norms or constraints operating over the parameter vectors  $\mathbf{w}$  can therefore be seen as special cases of the same norms and constraints imposed on the partial-derivative norms (2).*

### 2.1 SPARSITY INDUCING NORMS

The sparsity objective over a vector  $\mathbf{v} \in \mathbb{R}^d$  can be cast as the minimization of the  $\ell_0$  norm  $\|\mathbf{v}\|_0 = \#\{a = 1, \dots, d : v_a \neq 0\}$  which counts the number of non-zero elements of the vector. Since it is well known from the linear sparse learning literature that finding the  $\ell_0$  solutions is computationally difficult in higher dimensions (NP-hard, Weston et al. (2003)), the authors in Rosasco et al. (2013) suggest to use its tightest convex relaxation, the  $\ell_1$  norm  $\|\mathbf{v}\|_1 = \sum_a^d |v_a|$ . They apply the  $\ell_1$  norm over the partial-derivative norms (2) so that the lasso-like sparsity regularizer in (1) is

$$\mathcal{R}^L(f) = \sum_{a=1}^d \|\partial_a f\|_{L_2} \quad (3)$$

In this paper we explore two extensions inspired by the linear sparse learning, opening the doors to many of the other sparsity and structured sparsity inducing norms that have been proposed in the abundant literature on this topic. Namely, we focus here on the structured sparsity induced by the mixed  $\ell_1/\ell_2$  norm known in the context of linear least squares as the group lasso Yuan and Lin (2006). For a vector  $\mathbf{v}$  composed of  $G$  groups  $\mathbf{v}_g$  (non-overlapping but not necessarily consecutive) with  $p_g$  number of elements each, the mixed  $\ell_1/\ell_2$  norm is  $\|\mathbf{v}\|_{1,2} = \sum_g^G p_g \|\mathbf{v}_g\|_2$ . The corresponding group-lasso-like regularizer to be used in (1) is

$$\mathcal{R}^{GL}(f) = \sum_{g=1}^G p_g \sqrt{\sum_{a \in g} \|\partial_a f\|_{L_2}^2} \quad (4)$$

Second, we look at the elastic net penalty proposed initially in Zou and Hastie (2005). This uses a convex combination of the  $\ell_1$  and square of the  $\ell_2$  norm and has been

shown to have better selection properties over the vanilla  $\ell_1$  norm regularization in the presence of highly correlated features. Unlike the  $\ell_1$  penalty, the combined elastic net is also strictly convex. The corresponding elastic-net-like regularizer to be used in (1) is

$$\mathcal{R}^{EN}(f) = \mu \sum_{a=1}^d \|\partial_a f\|_{L_2} + (1 - \mu) \sum_{a=1}^d \|\partial_a f\|_{L_2}^2, \quad \mu \in [0, 1]. \quad (5)$$

## 2.2 EMPIRICAL VERSIONS OF REGULARIZERS

A common problem of the regularizers introduced above is that in practice they cannot be evaluated due to the unknown probability measure  $\rho_x$  on the input space  $\mathcal{X}$ . Therefore instead of the partial-derivative norms defined in expectation in (2)

$$\|\partial_a f\|_{L_2} = \sqrt{\mathbb{E}|\partial_a f(\mathbf{x})|^2} \quad (6)$$

we use their sample estimates replacing the expectation by the training sample average

$$\|\partial_a f\|_{2_n} = \sqrt{\frac{1}{n} \sum_i |\partial_a f(\mathbf{x}^i)|^2}. \quad (7)$$

This corresponds to the move from expected loss to the empirical loss introduced in section 1 and is enabled by the i.i.d. sample assumptions.

In result, the regression function is learned from the empirical version of (1)

$$\hat{f} = \operatorname{argmin}_{f \in \mathcal{F}} \hat{\mathcal{L}}(f) + \tau \hat{\mathcal{R}}(f), \quad (8)$$

where  $\hat{\mathcal{R}}(f)$  are the empirical analogues of the regularizers (3), (4) and (5) replacing the population partial-derivative norms  $\|\partial_a f\|_{L_2}$  by their sample estimates  $\|\partial_a f\|_{2_n}$ . The function space  $\mathcal{F}$  is discussed next.

## 3 LEARNING IN RKHS

In this paper, the hypothesis space  $\mathcal{F}$  within which we learn the function  $f$  is a reproducing kernel Hilbert space (RKHS). We recall (e.g. Saitoh and Sawano (2016)) that a RKHS is a function space  $\mathcal{F}$  of real-valued functions over  $\mathcal{X}$  endowed with an inner product  $\langle \cdot, \cdot \rangle_{\mathcal{F}}$  and the induced norm  $\|\cdot\|_{\mathcal{F}}$  that is uniquely associated with a positive semidefinite kernel  $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ . The kernel  $k$  has the reproducing property  $\langle k_{\mathbf{x}}, f \rangle_{\mathcal{F}} = f(\mathbf{x})$  and, in particular,  $\langle k_{\mathbf{x}}, k_{\mathbf{x}'} \rangle_{\mathcal{F}} = k(\mathbf{x}, \mathbf{x}')$ , where  $k_{\mathbf{x}} \in \mathcal{F}$  is the kernel section centred at  $\mathbf{x}$  such that  $k_{\mathbf{x}}(\mathbf{x}') = k(\mathbf{x}, \mathbf{x}')$

for any two  $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ . Furthermore, the space  $\mathcal{F}$  is the completion of the linear span of the functions  $\{k_{\mathbf{x}} : \mathbf{x} \in \mathcal{X}\}$ .

In addition to these fairly well known properties of the RKHS and its kernel, the author in Zhou (2008) has shown that if  $k$  is continuous and sufficiently smooth the kernel partial-derivative functions belong to the RKHS and have a partial-derivative reproducing property. More specifically, we define the kernel partial-derivative function  $[\partial_a k_{\mathbf{x}}] : \mathcal{X} \rightarrow \mathbb{R}$  as

$$[\partial_a k_{\mathbf{x}}](\mathbf{x}') = \frac{\partial}{\partial x_a} k(\mathbf{x}, \mathbf{x}') \quad \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}. \quad (9)$$

The function  $[\partial_a k_{\mathbf{x}}] \in \mathcal{F}$  has the reproducing property  $\langle [\partial_a k_{\mathbf{x}}], f \rangle_{\mathcal{F}} = \partial_a f(\mathbf{x})$ . In particular  $\langle [\partial_a k_{\mathbf{x}}], k_{\mathbf{x}'} \rangle_{\mathcal{F}} = \partial_a k_{\mathbf{x}'}(\mathbf{x})$  and  $\langle [\partial_a k_{\mathbf{x}}], [\partial_b k_{\mathbf{x}'}] \rangle_{\mathcal{F}} = \frac{\partial^2}{\partial x_a \partial x'_b} k(\mathbf{x}, \mathbf{x}')$ .

**Remark 2.** *Since the notation above may seem somewhat knotty at first, we invite the reader to appreciate the difference between the function  $[\partial_a k_{\mathbf{x}}]$  and the partial derivative of the kernel section with respect to the  $a$ th dimension  $\partial_a k_{\mathbf{x}}$ . Clearly,  $[\partial_a k_{\mathbf{x}}](\mathbf{x}') \neq \partial_a k_{\mathbf{x}}(\mathbf{x}')$  for any  $\mathbf{x} \neq \mathbf{x}' \in \mathcal{X}$ . However, due to the symmetry of the kernel we do have  $[\partial_a k_{\mathbf{x}}](\mathbf{x}') = \partial_a k_{\mathbf{x}'}(\mathbf{x}) = \frac{\partial}{\partial x_a} k(\mathbf{x}, \mathbf{x}')$ .*

## 3.1 SOLUTION REPRESENTATION

The variational (infinite-dimensional) problem (8) is difficult to handle as is. However, it has been previously shown for a multitude of RKHS learning problems that their solutions  $\hat{f}$  can be expressed as finite linear combinations of the kernel evaluations over the training data Argyriou and Dinuzzo (2014). This property, known as *representer theorem*, renders the problems amenable to practical computations.

**Proposition 1.** *The minimising solution  $\hat{f}$  of the variational problem*

$$\hat{f} = \operatorname{argmin}_{f \in \mathcal{F}} \hat{\mathcal{L}}(f) + \tau \hat{\mathcal{R}}(f) + \nu \|f\|_{\mathcal{F}}^2, \quad (10)$$

where  $\tau, \nu \geq 0$  and  $\hat{\mathcal{R}}(f)$  is any of the empirical versions of the three formulations (3), (4), (5) can be represented as

$$\hat{f} = \sum_i \alpha_i k_{\mathbf{x}^i} + \sum_i \sum_a \beta_{ai} [\partial_a k_{\mathbf{x}^i}]. \quad (11)$$

The proof (available in the appendix) follows the classical approach (e.g. Schölkopf et al. (2001)) of decomposition of  $\mathcal{F}$  into the space spanned by the representation and its orthogonal complement.

The proposition extends the representer theorem of Rosasco et al. (2013) to the new regularizers (4) and (5).

Note that we included the induced Hilbert norm  $\|f\|_{\mathcal{F}}$  into (10) as a useful generalization that reduces to our original problem (8) if  $\nu = 0$ . On the other hand, when  $\tau = 0$  we recover a classical kernel regression problem which is known to have another simpler representation consisting just of the first term in (11).

## 4 ALGORITHM

In this section we describe the new algorithm we developed to solve problem (10) with the three sparse regularizers introduced in section 2. The algorithm is versatile so that it requires only small alterations in specific steps to move from one regularizer to the other. Importantly, unlike the algorithm proposed in Rosasco et al. (2013) for solving only the lasso-like problem, our algorithm does not need to rely on proximal approximations since all the proximal steps can be evaluated in closed forms. Our algorithm also directly provides values of the partial derivatives of the learned function indicating the learned sparsity.

### 4.1 FINITE DIMENSIONAL FORMULATION

To be able to develop a practical algorithm we first need to reformulate the variational optimisation problem (10) into its finite dimensional equivalent. For this we introduce the following objects: the  $n$ -long vector  $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_n]^T$ , the  $dn$ -long vector  $\boldsymbol{\beta} = [\beta_{11}, \dots, \beta_{1n}, \beta_{21}, \dots, \beta_{dn}]^T$ , the  $n \times n$  symmetric PSD kernel matrix  $\mathbf{K}$  such that  $K_{ij} = k(\mathbf{x}^i, \mathbf{x}^j)$ , the  $n \times n$  (non-symmetric) kernel derivative matrices  $\mathbf{D}^a$  and  $\tilde{\mathbf{D}}^a$ ,  $a \in \mathbb{N}_d$  such that  $D_{ij}^a = [\partial_a k_{\mathbf{x}^i}](\mathbf{x}^j) = \partial_a k_{\mathbf{x}^j}(\mathbf{x}^i) = \tilde{D}_{ji}^a$ , the  $n \times n$  (non-symmetric) kernel 2nd derivative matrices  $\mathbf{L}^{ab}$ ,  $a, b \in \mathbb{N}_d$  such that  $\mathbf{L}_{ij}^{ab} = \frac{\partial^2}{\partial x_i^a \partial x_j^b} k(\mathbf{x}^i, \mathbf{x}^j) = \frac{\partial}{\partial x_j^b} [\partial_a k_{\mathbf{x}^i}](\mathbf{x}^j) = \mathbf{L}_{ji}^{ba}$ . Further, we need the following concatenations:

$$\mathbf{D} = \begin{bmatrix} \mathbf{D}^1 \\ \dots \\ \mathbf{D}^d \end{bmatrix} \quad \mathbf{L}^a = [\mathbf{L}^{a1} \dots \mathbf{L}^{ad}] \quad \mathbf{L} = \begin{bmatrix} \mathbf{L}^1 \\ \dots \\ \mathbf{L}^d \end{bmatrix}$$

and specifically for the groups  $g$  in  $\mathcal{R}^{GL}$  the partitions

$$\ddot{\mathbf{D}}^g = \begin{bmatrix} \mathbf{D}^{g_1} \\ \dots \\ \mathbf{D}^{g_{p_g}} \end{bmatrix} \quad \dot{\mathbf{L}}^g = \begin{bmatrix} \mathbf{L}^{g_1} \\ \dots \\ \mathbf{L}^{g_{p_g}} \end{bmatrix},$$

where the subscripts  $g_i$  are the corresponding indexes of the input dimensions.

**Proposition 2.** *The variational problem (10) is equivalent to the finite dimensional problem*

$$\operatorname{argmin}_{\boldsymbol{\alpha}, \boldsymbol{\beta}} \mathcal{J}1(\boldsymbol{\alpha}, \boldsymbol{\beta}) + \tau \mathcal{J}2(\boldsymbol{\alpha}, \boldsymbol{\beta}) + \nu \mathcal{J}3(\boldsymbol{\alpha}, \boldsymbol{\beta}), \quad (12)$$

where

$$\begin{aligned} \mathcal{J}1(\boldsymbol{\alpha}, \boldsymbol{\beta}) &= \frac{1}{n} \|\mathbf{y} - \mathbf{K}\boldsymbol{\alpha} - \mathbf{D}^T \boldsymbol{\beta}\|_2^2 \\ \mathcal{R}^L : \mathcal{J}2(\boldsymbol{\alpha}, \boldsymbol{\beta}) &= \frac{1}{\sqrt{n}} \sum_a^d \|\mathbf{D}^a \boldsymbol{\alpha} + \mathbf{L}^a \boldsymbol{\beta}\|_2 \\ \mathcal{R}^{GL} : \mathcal{J}2(\boldsymbol{\alpha}, \boldsymbol{\beta}) &= \frac{1}{\sqrt{n}} \sum_g^G p_g \|\ddot{\mathbf{D}}^g \boldsymbol{\alpha} + \dot{\mathbf{L}}^g \boldsymbol{\beta}\|_2 \\ \mathcal{R}^{EN} : \mathcal{J}2(\boldsymbol{\alpha}, \boldsymbol{\beta}) &= \frac{\mu}{\sqrt{n}} \sum_a^d \|\mathbf{D}^a \boldsymbol{\alpha} + \mathbf{L}^a \boldsymbol{\beta}\|_2 \\ &\quad + \frac{1-\mu}{n} \sum_a^d \|\mathbf{D}^a \boldsymbol{\alpha} + \mathbf{L}^a \boldsymbol{\beta}\|_2^2 \\ \mathcal{J}3(\boldsymbol{\alpha}, \boldsymbol{\beta}) &= \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} + 2\boldsymbol{\alpha}^T \mathbf{D}^T \boldsymbol{\beta} + \boldsymbol{\beta}^T \mathbf{L} \boldsymbol{\beta} \end{aligned}$$

The proof (available in the appendix) is based on the finite dimensional representation (11) of the minimising function, and the kernel and derivative reproducing properties stated in section 3.

The problem reformulation (12) is instructive in terms of observing the roles of the kernel and the derivative matrices and is reminiscent of the classical finite dimensional reformulation of Hilbert-norm regularised least squares. However, for the development of our algorithm we derive a more convenient equivalent form.

**Proposition 3.** *The variational problem (10) is equivalent to the finite dimensional problem*

$$\operatorname{argmin}_{\boldsymbol{\omega}} \frac{1}{n} \|\mathbf{y} - \mathbf{F}\boldsymbol{\omega}\|_2^2 + \tau \mathcal{J}(\boldsymbol{\omega}) + \nu \boldsymbol{\omega}^T \mathbf{Q} \boldsymbol{\omega}, \quad (13)$$

where

$$\begin{aligned} \mathcal{R}^L : \mathcal{J}(\boldsymbol{\omega}) &= \frac{1}{\sqrt{n}} \sum_a^d \|\mathbf{Z}^a \boldsymbol{\omega}\|_2 \\ \mathcal{R}^{GL} : \mathcal{J}(\boldsymbol{\omega}) &= \frac{1}{\sqrt{n}} \sum_g^G p_g \|\ddot{\mathbf{Z}}^g \boldsymbol{\omega}\|_2 \\ \mathcal{R}^{EN} : \mathcal{J}(\boldsymbol{\omega}) &= \frac{\mu}{\sqrt{n}} \sum_a^d \|\mathbf{Z}^a \boldsymbol{\omega}\|_2 + \frac{1-\mu}{n} \sum_a^d \|\mathbf{Z}^a \boldsymbol{\omega}\|_2^2, \end{aligned}$$

with

$$\boldsymbol{\omega} = \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} \quad \mathbf{F} = [\mathbf{K} \mathbf{D}^T] \quad \mathbf{Z}^a = [\mathbf{D}^a \mathbf{L}^a] \quad \mathbf{Q} = \begin{bmatrix} \mathbf{K} & \mathbf{0} \\ 2\mathbf{D} & \mathbf{L} \end{bmatrix}$$

The proof is trivial using (12) as an intermediate step.

### 4.2 DEVELOPMENT OF GENERIC ALGORITHM

Problem (13) is convex though its middle part  $\mathcal{J}(\boldsymbol{\omega})$  is non-differentiable for all three discussed regularizers. In-

deed, it is the singularities of the norms at zero points that yield the sparse solutions. A popular approach for solving convex non-differentiable problems is the proximal gradient descent Parikh and Boyd (2013). At every step it requires evaluating the proximal operator defined for any function  $f : \mathbb{R}^m \rightarrow \mathbb{R}^m$  and any vector  $\mathbf{v} \in \mathbb{R}^m$  as

$$\text{prox}_f(\mathbf{v}) = \underset{\mathbf{x}}{\operatorname{argmin}} f(\mathbf{x}) + \frac{1}{2} \|\mathbf{x} - \mathbf{v}\|_2^2. \quad (14)$$

However, proximal operators for the functions  $\mathcal{J}$  in (13) do not have closed forms or fast methods for solving which makes the proximal gradient descent algorithm difficult to use.

We therefore propose to introduce a linearizing change of variables  $\mathbf{Z}^a \boldsymbol{\omega} = \boldsymbol{\varphi}_a$  and cast the problem in a form amenable for the ADMM method Boyd (2010)

$$\min \mathcal{E}(\boldsymbol{\omega}) + \tau \mathcal{I}(\boldsymbol{\varphi}), \quad \text{s.t. } \mathbf{Z}\boldsymbol{\omega} - \boldsymbol{\varphi} = 0. \quad (15)$$

In the above

$$\boldsymbol{\varphi} = \begin{bmatrix} \boldsymbol{\varphi}_1 \\ \dots \\ \boldsymbol{\varphi}_d \end{bmatrix}, \quad \mathbf{Z} = \begin{bmatrix} \mathbf{Z}^1 \\ \dots \\ \mathbf{Z}^d \end{bmatrix},$$

(or concatenation of the double-dot version for the group structure),  $\mathcal{E} : \mathbb{R}^{n+nd} \rightarrow \mathbb{R}$  is the convex differentiable function

$$\mathcal{E}(\boldsymbol{\omega}) = \frac{1}{n} \|\mathbf{y} - \mathbf{F}\boldsymbol{\omega}\|_2^2 + \nu \boldsymbol{\omega}^T \mathbf{Q} \boldsymbol{\omega},$$

and  $\mathcal{I} : \mathbb{R}^{nd} \rightarrow \mathbb{R}$  is the convex non-differentiable function corresponding to each regularizer such that  $\mathcal{I}(\boldsymbol{\varphi}) = \mathcal{J}(\boldsymbol{\omega})$  for every  $\mathbf{Z}\boldsymbol{\omega} = \boldsymbol{\varphi}$ .

At each iteration the ADMM algorithm consists of the following three update steps (the standard approach of augmented Lagrangian with  $\boldsymbol{\lambda}$  as the scaled dual variable and  $\kappa$  as the step size):

$$S1 : \boldsymbol{\omega}^{(k+1)} = \underset{\boldsymbol{\omega}}{\operatorname{argmin}} \mathcal{E}(\boldsymbol{\omega}) + \frac{\kappa}{2} \|\mathbf{Z}\boldsymbol{\omega} - \boldsymbol{\varphi}^{(k)} + \boldsymbol{\lambda}^{(k)}\|_2^2$$

$$S2 : \boldsymbol{\varphi}^{(k+1)} = \underset{\boldsymbol{\varphi}}{\operatorname{argmin}} \tau \mathcal{I}(\boldsymbol{\varphi}) + \frac{\kappa}{2} \|\mathbf{Z}\boldsymbol{\omega}^{(k+1)} - \boldsymbol{\varphi} + \boldsymbol{\lambda}^{(k)}\|_2^2$$

$$S3 : \boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} + \mathbf{Z}\boldsymbol{\omega}^{(k+1)} - \boldsymbol{\varphi}^{(k+1)}$$

The first step  $S1$  is a convex quadratic problem with a closed form solution

$$S1 : (\nu \mathbf{Q} + \nu \mathbf{Q}^T + 2n^{-1} \mathbf{F}^T \mathbf{F} + \kappa \mathbf{Z}^T \mathbf{Z}) \boldsymbol{\omega}^{(k+1)} = 2n^{-1} \mathbf{F}^T \mathbf{y} + \kappa \mathbf{Z}^T (\boldsymbol{\varphi}^{(k)} - \boldsymbol{\lambda}^{(k)})$$

By comparing with (14) we observe that the second step  $S2$  is a proximal update. The advantage of our problem reformulation and our algorithm is that this has a closed form for all the three discussed regularizers.

**Proposition 4.** *The proximal problem in step  $S2$  is decomposable by the  $d$  partitions of vector  $\boldsymbol{\varphi}$  (or  $G$  partition in case of the group structure) and the minimising solution is*

$$\mathcal{R}^L : \boldsymbol{\varphi}_a^{(k+1)} = (\mathbf{Z}^a \boldsymbol{\omega}^{(k+1)} + \boldsymbol{\lambda}_a^{(k)}) \left( 1 - \frac{\tau}{\kappa \sqrt{n} \|\mathbf{Z}^a \boldsymbol{\omega}^{(k+1)} + \boldsymbol{\lambda}_a^{(k)}\|_2} \right)_+$$

$$\mathcal{R}^{GL} : \boldsymbol{\varphi}_g^{(k+1)} = (\ddot{\mathbf{Z}}^g \boldsymbol{\omega}^{(k+1)} + \ddot{\boldsymbol{\lambda}}_g^{(k)}) \left( 1 - \frac{\tau p_g}{\kappa \sqrt{n} \|\mathbf{Z}^g \boldsymbol{\omega}^{(k+1)} + \boldsymbol{\lambda}_g^{(k)}\|_2} \right)_+$$

$$\mathcal{R}^{EN} : \boldsymbol{\varphi}_a^{(k+1)} = \frac{\mathbf{Z}^a \boldsymbol{\omega}^{(k+1)} + \boldsymbol{\lambda}_a^{(k)}}{2\tau(1-\mu)/(\kappa n) + 1} \left( 1 - \frac{\tau \mu}{\kappa \sqrt{n} \|\mathbf{Z}^a \boldsymbol{\omega}^{(k+1)} + \boldsymbol{\lambda}_a^{(k)}\|_2} \right)_+$$

Here  $(v)_+ = \min(0, v)$  is the thresholding operator.

The decomposability comes from the additive structure of  $\mathcal{I}$ . The derivation follows similar techniques as used for classical  $\ell_1$  and  $\ell_2$  proximals.<sup>1</sup>

### 4.3 PRACTICAL IMPLEMENTATION

In practice, the  $\mathbf{Q}$ ,  $\mathbf{F}$  and  $\mathbf{Z}$  matrices are precomputed in a preprocessing step and passed onto the algorithm as inputs. The matrices are directly computable using the kernel function  $k$  and its first and second order derivatives evaluated at the training points (following the matrix definitions introduced in section 4.1).

The algorithm converges to a global minimum by the standard properties of ADMM. In our implementation (available at [https://bitbucket.org/dmmlgeneva/nvsd\\_uai2018/](https://bitbucket.org/dmmlgeneva/nvsd_uai2018/)) we follow a simple updating rule Boyd (2010, sec. 3.4.1) for the step size  $\kappa$ . We use inexact minimization for the most expensive step  $S1$ , gradually increasing the number of steepest descent steps, each with complexity  $\mathcal{O}((nd)^2)$ .

Furthermore, we use  $S2$  to get the values of the training sample partial-derivative norms defined in equation (7) as  $\|\partial_a f^{(k)}\|_{2_n} = \|\boldsymbol{\varphi}_a^{(k)}\|_2 / \sqrt{n}$ . The sparsity pattern is obtained by examining for which of the dimensions  $a \in \mathbb{N}_d$  the norm is zero  $\|\partial_a f^{(k)}\|_{2_n} = 0$ .

<sup>1</sup>For  $\mathcal{R}^{EN}$  it is more practical to add the quadratic term into  $\mathcal{E}(\boldsymbol{\omega})$  in  $S1$  and use the corresponding scaled version of the  $\mathcal{R}^L$  proximal in  $S2$ .

## 5 EMPIRICAL EVALUATION

We conducted a set of synthetic and real-data experiments to document the efficacy of our structured-sparsity methods and the new algorithm under controlled and more realistic conditions. We compare our methods NVSD(L), NVSD(GL) and NVSD(EN) in terms of their predictive accuracy and their selection ability to the simple (non-sparse) kernel regularised least squares (Krls), to the sparse additive model (SpAM) of Ravikumar et al. (2007), to the non-linear cross-covariance-based method using the Hilbert Schmidt Independence Criterion in a lasso-like manner (HSIC) of Yamada et al. (2014), and to the derivative-based lasso-like method (Denovas) of Rosasco et al. (2013).<sup>2</sup> We compared also to simple mean and linear sparse and non-sparse models. All of these performed considerably worse than the non-linear models and therefore are not listed in the summary results. For all the sparse kernel methods we consider a two-step debiasing procedure based on variable selection via the base algorithm followed by a simple kernel regularised least squares on the selected variables.<sup>3</sup>

### 5.1 SYNTHETIC EXPERIMENTS

We motivate each synthetic experiment by a realistic story-line and explain the data generating process here below. In all the synthetic experiments we fix the input dimension to  $d = 18$  with only 6 input variables  $\{1, 2, 3, 7, 8, 9\}$  relevant for the model and the other 12 irrelevant.

**E1** In the first experiment we focus on the NVSD(GL) which assumes the input variables can be grouped a priori by some domain knowledge (e.g. each group describes a *type* of input data such as a different biological process) and the groups are expected to be completely in or out of the model. The input variables are generated independently from a standard normal distribution and they are grouped by three into 6 groups. The output is generated from the 1st and the 3rd group as

$$y = \sum_{i=1}^3 \sum_{j=i}^3 \sum_{k=j}^3 x_i x_j x_k + \sum_{q=7}^9 \sum_{r=q}^9 \sum_{s=r}^9 x_q x_r x_s + \epsilon ,$$

with  $\epsilon \sim N(0, 0.01)$ . For learning we fix the kernel to 3rd order polynomial.

<sup>2</sup>For HSIC and Denovas we used the author’s code, for SpAM the R implementation of Zhao et al. (2014). For all algorithms we kept the default settings.

<sup>3</sup>This is native to Denovas and necessary for HSIC which otherwise does not produce a predictive model.

**E2** In the second experiment we do not assume any a priori grouping of the variables. Instead some of the variables are strongly correlated (perhaps relating to a single phenomenon), a case for NVSD(EN). The input variables are generated similarly as in E1 but with the pairs  $\{1, 7\}$ ,  $\{2, 8\}$  and  $\{3, 9\}$  strongly correlated (Pearson’s population correlation coefficient 0.95). The remaining (irrelevant) input variables are also pair-wise correlated and the output is generated as

$$y = \sum_{i,j,k=1}^3 x_i x_j x_k + \sum_{q,r,s=7}^9 x_q x_r x_s + \epsilon ,$$

with  $\epsilon \sim N(0, 0.01)$ . For learning we fix the kernel to 3rd order polynomial.

**E3** In the third experiment we assume the inputs are noisy measurements of some true phenomenon (e.g. repeated measurements, measurements from multiple laboratories) for which there is no reason to prefer one over the other in the model. We first generate the true data  $z_i \sim N(0, 1)$ ,  $i = 1, \dots, 6$  and use these to generate the outputs as

$$y = 10(z_1^2 + z_3^2)e^{-2(z_1^2 + z_3^2)} + \epsilon ,$$

with  $\epsilon \sim N(0, 0.01)$ . We then generate the noisy measurements that will be used as inputs for the learning: for each  $z_i$  we create three noisy measurements  $x_{ij} = z_i + N(0, 0.1)$ ,  $j = 1, 2, 3$  (a group for the NVSD(GL) method); the input vector is the concatenation of all  $x_{ij}$  so that from the 18 long concatenated input vector  $\mathbf{x}$  again only the set  $\{1, 2, 3, 7, 8, 9\}$  of the dimensions is relevant for predicting the output  $y$ . For learning we fix the kernel to Gaussian with width  $\sigma = 4$ .

**Remark 3.** *In all the synthetic experiments we use the same experimental protocol. We split the data into train sets varying the size in  $n = \{30, 50, 70, 90, 110\}$ , a validation set of length 1000, and a test set of length 1000. We train the models over the train sets and use the validation set to select the regularization hyper-parameters (and therefore the models) based on the minimal validation MSE. We use dense grids of 50 points for the  $\tau$  search (automatically established by the algorithm) and 5 points grid for  $\mu \in \{0.1, \dots, 0.9\}$ . Complete settings (also for the baseline methods) are detailed in the replication files publicly available at [https://bitbucket.org/dmmlgeneva/nvds\\_uai2018/](https://bitbucket.org/dmmlgeneva/nvds_uai2018/).*

We report the average results across 50 independent replications of the experiments in table 1. We measure



Table 1: Results of Synthetic Experiments

Train size		30	50	70	90	110	
E1	RMSE	Krls	12.79	11.66	10.99	10.43	9.80
		SpAM	11.41	9.47	8.66	8.22	7.75
		HSIC	11.37	10.00	8.58	7.28	5.68
		Denovas	11.66	10.87	12.37	13.28	11.78
		NVSD(L)	11.55	10.22	9.36	7.90	7.13
		NVSD(GL)	<b>9.92</b>	<b>7.89</b>	<b>6.34</b>	<b>1.94</b>	<b>2.41</b>
E1	Selection error	Krls	0.67	0.67	0.67	0.67	0.67
		SpAM	0.54	0.56	0.59	0.57	0.58
		HSIC	0.50	0.48	0.42	0.35	0.32
		Denovas	0.49	0.50	0.53	0.67	0.73
		NVSD(L)	0.49	0.47	0.48	0.39	0.32
		NVSD(GL)	<b>0.28</b>	<b>0.24</b>	<b>0.22</b>	<b>0.05</b>	<b>0.11</b>
E2	RMSE	Krls	27.69	24.83	22.53	19.14	18.04
		SpAM	31.24	29.21	29.25	27.11	26.03
		HSIC	21.74	15.50	12.02	9.42	7.67
		Denovas	24.23	34.33	17.51	8.89	11.20
		NVSD(L)	21.24	16.59	11.79	8.61	7.35
		NVSD(EN)	<b>17.53</b>	<b>10.05</b>	<b>5.67</b>	<b>4.29</b>	<b>3.29</b>
E2	Selection error	Krls	0.67	0.67	0.67	0.67	0.67
		SpAM	0.57	0.55	0.49	0.52	0.46
		HSIC	0.52	0.42	0.42	0.35	0.32
		Denovas	0.46	0.54	0.40	0.30	0.26
		NVSD(L)	0.46	0.43	0.36	0.31	0.29
		NVSD(EN)	<b>0.35</b>	<b>0.20</b>	<b>0.14</b>	<b>0.09</b>	<b>0.08</b>
E3	RMSE	Krls	0.65	0.55	0.54	0.53	0.50
		SpAM	0.51	0.49	0.47	0.47	0.46
		HSIC	0.52	0.47	0.45	0.44	0.43
		Denovas	0.55	0.51	0.50	0.51	0.50
		NVSD(L)	0.51	0.44	0.44	0.41	0.34
		NVSD(GL)	0.51	<b>0.41</b>	<b>0.39</b>	<b>0.33</b>	0.31
NVSD(EN)	<b>0.50</b>	0.43	0.42	0.36	<b>0.30</b>		
E3	Selection error	Krls	0.67	0.67	0.67	0.67	0.67
		SpAM	0.65	0.61	0.60	0.58	0.59
		HSIC	0.59	0.51	0.53	0.47	0.44
		Denovas	0.49	0.45	0.47	0.45	0.41
		NVSD(L)	0.33	0.30	0.40	0.34	0.23
		NVSD(GL)	<b>0.26</b>	<b>0.20</b>	<b>0.24</b>	<b>0.15</b>	<b>0.14</b>
NVSD(EN)	0.30	0.33	0.35	0.25	0.16		

Best results in bold; underlined when structured-sparsity methods significantly better than all other methods using Wilcoxon signed-rank test at 5% significance level.

the prediction accuracy by the root mean squared error (RMSE) over the test sets and the selection accuracy by the Tanimoto distance between the true sparsity and the learned sparsity patterns (section 4.3).

Our structured-sparsity methods clearly outperform all the non-structured sparse learning methods achieving better prediction accuracy based on more precise variable selection, typically with statistically significant differences. Also, the prediction and selection accuracy generally increases (errors decrease) for larger training sample sizes suggesting our methods are well-behaved in

terms of the standard statistical learning paradigms. In the E3 experiment, NVSD(GL) performs the best having the benefit of the prior knowledge of the variable groupings. Remarkably, NVSD(EN) follows closely after even without such prior information, learning about the groups of correlated variables from the data when building the model.

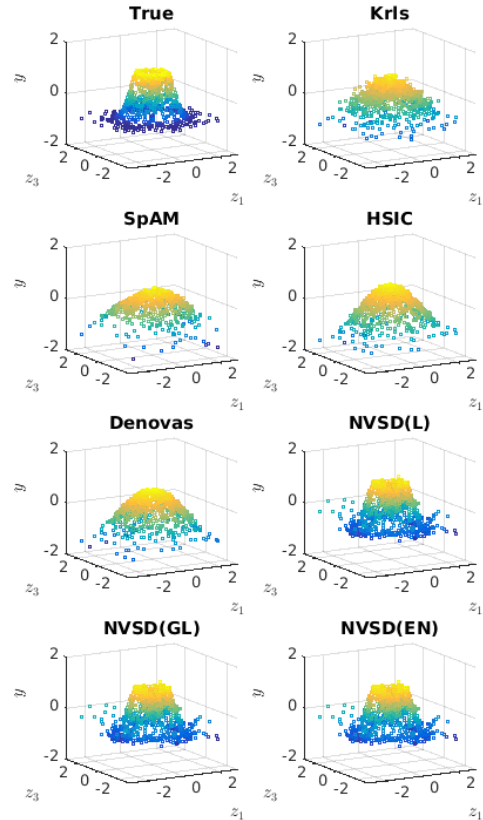


Figure 1: Predictions for the E3 experiment over the test data. We picked an example for the model trained with 110 instances (the 17th replication) which illustrates well the advantage our NVSD methods have over the baselines in capturing the True complex non-linear structure.

Krls can only learn full models and therefore performs rather poorly on these by-construction sparse problems. From the other three baselines, HSIC typically achieves the second best results (after our NVSD methods). SpAM is not particularly suitable for the non-additive structures of our experiments. Finally, in all the experiments our NVSD(L) outperforms Denovas though they share the same lasso-like problem formulation. We attribute this to our new algorithm developed in section 4 which, unlike Denovas, does not rely on approximations of the proximal operators.

## 5.2 REAL-DATA EXPERIMENTS

For the real-data experiments we used a collection of regression datasets from UCI Lichman (2013) and LIACC<sup>4</sup> repositories listed in table 2.

Table 2: Real Datasets Description

Code	Name	Inputs	Test Size	Source
AI	Airfoil Self Noise	5	700	UCI
BH	Boston Housing	10	200	UCI
CC	Concrete Compressive	8	450	UCI
EN	Energy Efficiency	8	300	UCI
CP	Computer Activity	21	1000	LIACC
EL	F16 Elevators	17	1000	LIACC
KN	Kynematics	8	1000	LIACC

We report the average results across 50 replications of the experiments in tables 3 and 4. We use RMSE over the test data for measuring the prediction accuracy. For the real datasets we do not know the ground-truth sparsity patterns. Instead of measuring the selection error we therefore count the number of input variables selected by each method. Krls has no selection ability, its support size is hence equal to the total number of input variables in each problem.

**Remark 4.** *We followed similar experimental protocol as for the synthetic experiments. We fixed the training sample size for all experiments to 100 instances and used 200-1000 instances for the validation and test sets (depending on the total number of available observations). We pre-processed the data by normalizing the inputs and centering the outputs. For all the experiments we used a Gaussian kernel with the width set to the median distance calculated over the nearest 20 neighbours, and the 3rd order polynomial kernel. With the exception of the EN dataset, the Gaussian kernel yielded better results and was therefore kept for the final evaluation. Full details of the settings can be found in the replication files publicly available at [https://bitbucket.org/dmmlgeneva/nvds\\_uai2018/](https://bitbucket.org/dmmlgeneva/nvds_uai2018/).*

Results in table 3 are for the original data for which we have no prior knowledge about possible variable groupings. Therefore we only use the non-structured methods and our NVSD(EN) that do not rely on any such prior information.

Our NVSD methods learned sparse non-linear models achieving better or comparable results than the baselines in 4 out of the 5 experiments (BH, CP, EN, EL). For CC reducing the number of input dimensions does not seem to bring any advantages and the methods tend to learn full

Table 3: Results of Real-data Experiments

Experiment	BH	CP	CC	EN	EL	
RMSE	Krls	4.00	12.27	8.70	1.83	5.10
	SpAM	4.33	~	12.70	~	~
	HSIC	4.02	9.39	8.73	<b>1.19</b>	9.07
	Denovas	4.02	9.21	12.07	3.02	6.01
	NVSD(L)	3.96	8.43	<b>8.67</b>	1.50	<u>4.81</u>
	NVSD(EN)	<b>3.93</b>	<b>7.88</b>	8.70	1.20	<b>4.67</b>
Support size	Krls	10.00	21.00	8.00	8.00	17.00
	SpAM	9.00	~	2.82	~	~
	HSIC	6.12	8.26	5.88	5.08	0.00
	Denovas	8.80	4.76	4.38	4.96	10.52
	NVSD(L)	8.20	3.78	7.36	7.26	14.06
	NVSD(EN)	8.06	4.58	7.98	6.66	13.00

Best results in bold; underlined when NVSD methods significantly better than all the baselines using Wilcoxon signed-rank test at 5% significance level. For several experiments SpAM finished with errors.

models. For several experiments SpAM finished with errors and therefore the results in the table are missing.

To explore the performance and benefits of NVSD(GL) method we had to construct variable groups that could potentially help the model learning. We adopted two strategies:

1. For CP and EL datasets we constructed the groups based on the NVSD(EN) results. For CP we grouped together the 5 most often selected variables across the 50 replications of the experiment and created 3 other groups from the remaining variables. For EL we created five groups by 3-4 elements putting together variables with similar frequencies of occurrence in the support of the learned NVSD(EN) models over the 50 replications.
2. For AI, CC, and KN datasets we doubled the original input data dimensions by complementing the input data by a copy of each input variable with permuted instance order. We then constructed two groups, the first over the original data, the second over the permuted copy.

Table 4 confirms that our NVSD(GL) is able to use the grouping information based on prior knowledge to select better, more relevant subset of variables than the non-structured baselines. Thanks to this it achieves significantly better prediction accuracy in all the experiments.

## 6 CONCLUSIONS AND FUTURE WORK

In this work we addressed the problem of variable selection in non-linear regression problems. We followed

<sup>4</sup><http://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html>

Table 4: Results of Real-data Experiments with Groups

Experiment	AI	CP	CC	KN	EL	
RMSE	Krls	5.08	12.27	10.34	2.07	5.10
	SpAM	~	~	13.31	2.20	~
	HSIC	4.64	9.39	9.29	2.05	9.07
	Denovas	5.12	9.21	11.49	2.10	6.01
	NVSD(L)	<u>4.45</u>	8.43	9.58	2.03	<u>4.81</u>
	NVSD(GL)	<b>4.16</b>	<b>7.43</b>	<b>8.79</b>	<b>1.96</b>	<b>4.76</b>
Support size	Krls	10.00	21.00	16.00	16.00	17.00
	SpAM	~	~	2.60	11.32	~
	HSIC	5.08	8.26	6.16	11.82	0.00
	Denovas	5.94	4.76	6.96	9.72	10.52
	NVSD(L)	4.76	3.78	8.16	13.58	14.06
	NVSD(GL)	5.00	5.84	8.00	11.84	13.82

Best results in bold; underlined when NVSD methods significantly better than all the baselines using Wilcoxon signed-rank test at 5% significance level. For several experiments SpAM finished with errors.

up from the work of Rosasco et al. (2013) arguing for the use of partial derivatives as an indication of the pertinence of an input variable for the model. Extending the existing work, we proposed two new derivative-based regularizers for learning with structured sparsity in non-linear regression similar in spirit to the linear elastic net and group lasso.

After posing the problems into the framework of RKHS learning, we designed a new NVSD algorithm for solving these. Unlike the previously proposed Denovas our new algorithm does not rely on proximal approximations. This is most likely the main reason why our NVSD(L) method achieved systematically better predictive performance than Denovas on a broad set of experiments. We also empirically demonstrated the advantages our structured sparsity methods NVSD(GL) and NVSD(EN) bring for learning tasks with a priori known group structures or correlation in the inputs.

These promising results point to questions requiring further attention:

Our NVSD algorithm achieves better results in terms of prediction accuracy than Denovas, however, at the cost of longer training times. Its  $\mathcal{O}((nd)^2)$  complexity is not favourable for scaling in neither instances nor dimensions. Exploring avenues for speeding up, possibly along the lines of random features construction, is certainly an important next step in making the algorithm operational for more practical real-life problems.

The method is based on the partial-derivative arguments and therefore assumes the functions (and therefore the kernels) are at least 2nd order differentiable (and square-integrable). We use here the polynomial and Gaussian

kernel as the most commonly used examples. What other properties of the kernels are necessary to ensure good performance and how the methods could be extended to other, more complex kernels are relevant questions.

The full problem formulation (e.g. equation (10) in proposition 1) combines the sparse regularizers with the function Hilbert-norm. This combination has been proposed in Rosasco et al. (2013) to ensure that the regularization part of the problem is strongly convex and the problem is well-posed in terms of the generalization properties.

However, interactions of the Hilbert norm with the sparsity inducing regularizers of section 2 and the effects on the learning and selection properties are not yet fully clear. Empirically (from Rosasco et al. (2013) and our own experiments) the models are often little sensitive to variations in  $\nu^5$ .

In addition, the  $\mathcal{R}^{EN}$  regularizer is already strongly convex even without the Hilbert norm. To what degree combining it with the Hilbert norm is necessary to guarantee good generalization for outside the training needs to be further investigated. So does its behaviour and the possible improvements it can bring when learning from inputs with non-linear dependencies. In view of the above considerations, our paper is posing the motivations, foundations and principles for further studies on partial derivative-based regularizations.

### Acknowledgements

This work was partially supported by the research projects HSTS (ISNET) and RAWFIE #645220 (H2020). The computations were performed at University of Geneva on the Baobab and Whales clusters.

<sup>5</sup>We fix it based on a small subset of replications instead of including it into the full hyper-parameter search.

## References

- Argyriou, Andreas and Francesco Dinuzzo (2014). “A Unifying View of Representer Theorems”. In: *International Conference on Machine Learning (ICML)*.
- Bach, Francis (2009). “High-Dimensional Non-Linear Variable Selection through Hierarchical Kernel Learning”. In: *ArXiv 0909.0844*.
- Boyd, Stephen (2010). “Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers”. In: *Foundations and Trends in Machine Learning* 3.1.
- Chan, Antoni B, Nuno Vasconcelos, and Gert R G Lanckriet (2007). “Direct convex relaxations of sparse SVM”. In: *International Conference on Machine Learning (2007)*.
- Chen, Jianbo et al. (2017). “Kernel Feature Selection via Conditional Covariance Minimization”. In: *Advances in Neural Information Processing Systems (NIPS)*.
- Gurram, Prudhvi and Heesung Kwon (2014). “Optimal sparse kernel learning in the empirical kernel feature space for hyperspectral classification”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 7.4, pp. 1217–1226.
- Hastie, Trevor, Robert Tibshirani, and Martin Wainwright (2015). *Statistical Learning with Sparsity: The Lasso and Generalizations*. CRC Press.
- Koltchinskii, Vladimir and Ming Yuan (2010). “Sparsity in multiple kernel learning”. In: *Annals of Statistics* 38.6, pp. 3660–3695.
- Lichman, M. (2013). “UCI Machine Learning Repository”. In: <http://archive.ics.uci.edu/ml>.
- Lin, Yi and Hao Helen Zhang (2006). “Component selection and smoothing in multivariate nonparametric regression”. In: *Annals of Statistics* 34.5, pp. 2272–2297.
- Parikh, Neal and Stephen Boyd (2013). “Proximal algorithms”. In: *Foundations and Trends in Optimization* 1.3.
- Ravikumar, Pradeep et al. (2007). “Spam: Sparse additive models”. In: *Advances in Neural Information Processing Systems (NIPS)*.
- Rosasco, Lorenzo, S Villa, and S Mosci (2013). “Non-parametric sparsity and regularization”. In: *Journal of Machine Learning Research* 14, pp. 1665–1714.
- Saitoh, Saburo and Yoshihiro Sawano (2016). *Theory of Reproducing Kernels and Applications*. Springer.
- Schölkopf, Bernhard, Ralf Herbrich, and Alex J Smola (2001). “A Generalized Representer Theorem”. In: *COLT/EuroCOLT*.
- Tibshirani, Robert (1996). “Regression Shrinkage and Selection via the Lasso”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 58.1, pp. 267–288.
- Tyagi, Hemant, Andreas Krause, and Z Eth (2016). “Efficient Sampling for Learning Sparse Additive Models in High Dimensions”. In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*.
- Weston, Jason et al. (2003). “Use of the Zero-Norm with Linear Models and Kernel Methods”. In: *Journal of Machine Learning Research* 3, pp. 1439–1461.
- Yamada, Makoto et al. (2014). “High-dimensional feature selection by feature-wise kernelized Lasso.” In: *Neural Computation* 26.1, pp. 185–207.
- Yin, Junming, Xi Chen, and Eric P Xing (2012). “Group Sparse Additive Models”. In: *International Conference on Machine Learning (ICML)*.
- Yuan, Ming and Yi Lin (2006). “Model selection and estimation in regression with grouped variables”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68.1, pp. 49–67.
- Zhao, Tuo et al. (2014). *CRAN - Package SAM*.
- Zhou, Ding Xuan (2008). “Derivative reproducing properties for kernel methods in learning theory”. In: *Journal of Computational and Applied Mathematics* 220.1-2, pp. 456–463.
- Zou, Hui and Trevor Hastie (Apr. 2005). “Regularization and variable selection via the elastic net”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67.2, pp. 301–320.

---

# Identification of Strong Edges in AMP Chain Graphs

---

Jose M. Peña

Department of Computer and Information Science  
Linköping University  
58183 Linköping, Sweden

## Abstract

The essential graph is a distinguished member of a Markov equivalence class of AMP chain graphs. However, the directed edges in the essential graph are not necessarily strong or invariant, i.e. they may not be shared by every member of the equivalence class. Likewise for the undirected edges. In this paper, we develop a procedure for identifying which edges in an essential graph are strong. We also show how this makes it possible to bound some causal effects when the true chain graph is unknown.

## 1 INTRODUCTION

In most practical applications, the data available consists of observations. Therefore, it can rarely single out the true causal model. At best, it identifies the Markov equivalence class that contains the true causal model. In this paper, we represent causal models with the help of AMP chain graphs (Andersson et al., 2001). As argued by Peña (2016), these graphs are suitable for representing causal linear models with additive Gaussian noise. Intuitively, the directed subgraph of a chain graph represents the causal relations in the domain, and the undirected subgraph represents the dependence structure of the noise terms. Additive noise is a rather common assumption in causal discovery (Peters et al., 2017), mainly because it produces tractable models which are useful for gaining insight into the system under study. Note also that linear structural equation models, which have extensively been studied for causal effect identification (Pearl, 2009), are additive noise models.

In order to represent the equivalence class of chain graphs identified from the observations at hand, we typically use a distinguished member of it. In the literature, there are two distinguished members: The essential

graph (Andersson and Perlman, 2006), and the largest deflagged graph (Roverato and Studený, 2006). In general, they do not coincide: The essential graph is a deflagged graph (Andersson and Perlman, 2006, Lemma 3.2) but not necessarily the largest in the equivalence class (Andersson et al., 2001, p. 57). Unfortunately, the directed edges in either of the two representatives are not necessarily strong,<sup>1</sup> i.e. they may not be shared by every member of the equivalence class. Likewise for the undirected edges. In this paper, we use essential graphs to represent equivalence classes of chain graphs. And we develop a procedure for identifying which edges in an essential graph are strong. Note that while we assume that the true chain graph is unknown, its corresponding essential graph can be obtained from observational data as follows. First, learn a chain graph as shown by Peña (2014, 2016) and Peña and Gómez-Olmedo (2016) and, then, transform it into an essential graph as shown by Sonntag and Peña (2015, Section 3).

Identifying the strong edges in an essential graph is important because it makes it possible to identify causal paths from data even though the data may not be able to single out the true chain graph: Simply output every directed path in the essential graph that consists of only strong edges. Of course, the true chain graph may have additional causal paths. Identifying the strong edges in an essential graph is also important because it allows to efficiently bound some causal effects of the form  $p(y|do(x))$  where  $X$  and  $Y$  are singletons. The simplest way to bound such a causal effect consists in enumerating all the chain graphs that are equivalent to the essential graph and, then, computing the causal effect for each of them from the observational data by adjusting for the appropriate variables. Although we know how to enumerate the equivalent chain graphs (Sonntag and Peña, 2015, Theorem 3), this method may be inefficient for all but small domains. Instead, we show in this paper how the knowledge of the strong edges in an essential graph

---

<sup>1</sup>The term invariant or essential is also used in the literature.

allows to enumerate the adjusting sets without enumerating the equivalent chain graphs explicitly.

The rest of the paper is organized as follows. Section 2 introduces some preliminaries. Section 3 presents our algorithm to identify strong edges in an essential graph. Section 4 presents our procedure to bound causal effects when the true chain graph is unknown but its corresponding essential graph is known. Section 5 closes the paper with some discussion and lines of future research.

## 2 PRELIMINARIES

All the graphs and probability distributions in this paper are defined over a finite set  $V$  unless otherwise stated. All the graphs contain at most one edge between a pair of nodes. The elements of  $V$  are not distinguished from singletons.

The parents of a set of nodes  $X$  of a graph  $G$  is the set  $Pa_G(X) = \{A \mid A \rightarrow B \text{ is in } G \text{ with } B \in X\}$ . The children of  $X$  is the set  $Ch_G(X) = \{A \mid B \rightarrow A \text{ is in } G \text{ with } B \in X\}$ . The neighbors of  $X$  is the set  $Ne_G(X) = \{A \mid A - B \text{ is in } G \text{ with } B \in X\}$ . The adjacents of  $X$  is the set  $Ad_G(X) = \{A \mid A \rightarrow B, B \rightarrow A \text{ or } A - B \text{ is in } G \text{ with } B \in X\}$ . The descendants of  $X$  is the set  $De_G(X) = \{A \mid B \rightarrow \dots \rightarrow A \text{ is in } G \text{ with } B \in X\}$ . A route from a node  $V_1$  to a node  $V_n$  in  $G$  is a sequence of (not necessarily distinct) nodes  $V_1, \dots, V_n$  such that  $V_i \in Ad_G(V_{i+1})$  for all  $1 \leq i < n$ . A route is called a cycle if  $V_n = V_1$ . A cycle has a chord if two non-consecutive nodes of the cycle are adjacent in  $G$ . A cycle is called semidirected if it is of the form  $V_1 \rightarrow V_2 \dashrightarrow \dots \dashrightarrow V_n$  where  $\dashrightarrow$  is a short for  $\rightarrow$  or  $-$ . A chain graph (CG) is a graph with (possibly) directed and undirected edges, and without semidirected cycles. A set of nodes of a CG  $G$  is connected if there exists a route in  $G$  between every pair of nodes in the set and such that all the edges in the route are undirected. A chain component of  $G$  is a maximal connected set. Note that the chain components of  $G$  can be sorted topologically, i.e. for every edge  $A \rightarrow B$  in  $G$ , the component containing  $A$  precedes the component containing  $B$ . A set of nodes of  $G$  is complete if there is an undirected edge between every pair of nodes in the set. Moreover, a node is called simplicial if its neighbors are a complete set.

We now recall the interpretation of CGs due to Andersson et al. (2001), also known as AMP CGs.<sup>2</sup> A node

<sup>2</sup>Andersson et al. (2001) interpret CGs via the so-called augmentation criterion. Levitz et al. (2001, Theorem 4.1) introduce the so-called p-separation criterion and prove its equivalence to the augmentation criterion. Peña (2016, Theorem 2) introduce the route-based criterion that we use in this paper and prove its equivalence to the p-separation criterion.

$B$  in a route  $\rho$  in a CG  $G$  is called a triplex node in  $\rho$  if  $A \rightarrow B \leftarrow C$ ,  $A \rightarrow B - C$ , or  $A - B \leftarrow C$  is a subroute of  $\rho$ . Moreover,  $\rho$  is said to be  $Z$ -open with  $Z \subseteq V$  when (i) every triplex node in  $\rho$  is in  $Z$ , and (ii) every non-triplex node in  $\rho$  is outside  $Z$ . Let  $X, Y$  and  $Z$  denote three disjoint subsets of  $V$ . When there is no  $Z$ -open route in  $G$  between a node in  $X$  and a node in  $Y$ , we say that  $X$  is separated from  $Y$  given  $Z$  in  $G$  and denote it as  $X \perp_G Y \mid Z$ . The statistical independences represented by  $G$  are the separations  $X \perp_G Y \mid Z$ . A probability distribution  $p$  is Markovian with respect to  $G$  if the independences represented by  $G$  are a subset of those in  $p$ . If the two sets of independences coincide, then  $p$  is faithful to  $G$ . Two CGs are Markov equivalent if the sets of distributions that are Markovian with respect to each CG are the same. If a CG has an induced subgraph of the form  $A \rightarrow B \leftarrow C$ ,  $A \rightarrow B - C$  or  $A - B \leftarrow C$ , then we say that the CG has a triplex  $(A, B, C)$ . Two CGs are Markov equivalent if and only if they have the same adjacencies and triplexes (Andersson et al., 2001, Theorem 5).

**Lemma 1.** *Two CGs  $G$  and  $H$  are Markov equivalent if and only if they represent the same independences.*

*Proof.* The if part is trivial. To see the only if part, note that Levitz et al. (2001, Theorem 6.1) prove that there are Gaussian distributions  $p$  and  $q$  that are faithful to  $G$  and  $H$ , respectively. Moreover,  $p$  is Markovian with respect to  $H$ , because  $G$  and  $H$  are Markov equivalent. Likewise for  $q$  and  $G$ . Therefore,  $G$  and  $H$  must represent the same independences.  $\square$

### 2.1 ESSENTIAL GRAPHS

The essential graph (EG)  $G^*$  is a distinguished member of a class of equivalent CGs. Specifically, an edge  $A \rightarrow B$  is in  $G^*$  if and only if  $A \rightarrow B$  is in some member of the class and  $A \leftarrow B$  is in no member of the class. An algorithm (without proof of correctness) for constructing the EG from any other member of the equivalence class has been developed by Andersson and Perlman (2004, Section 7). An alternative algorithm with proof of correctness has been developed by Sonntag and Peña (2015, Section 3). The latter algorithm can be seen in Tables 1 and 2. A perpendicular line at the end of an edge such as in  $\dashrightarrow$  or  $\dashleftarrow$  represents a block, and it means that the edge cannot be oriented in that direction. Note that the ends of some of the edges in the rules in Table 2 are labeled with a circle such as in  $\dashrightarrow$  or  $\dashleftarrow$ . The circle represents an unspecified end, i.e. a block or nothing. The modifications in the consequents of the rules consist in adding some blocks. Note that only the blocks that appear in the consequents are added, i.e. the circled ends do not get modified. In line 2 of Table 1, any such set  $S$  will do. For instance, if  $B \notin De_G(A)$ , then

Table 1: Algorithm for constructing the EG.

In: A CG $G$ .	
Out: The EG $G^*$ in the equivalence class of $G$ .	
1	For each ordered pair of non-adjacent nodes $A$ and $B$ in $G$
2	Set $S_{AB} = S_{BA} = S$ such that $A \perp_G B   S$
3	Let $G^*$ denote the undirected graph that has the same adjacencies as $G$
4	Apply the rules R1-R4 to $G^*$ while possible
5	Replace every edge $A - B$ in every cycle in $G^*$ that is of length greater than three, chordless, and without blocks with $A \dashv B$
6	Apply the rules R2-R4 to $G^*$ while possible
7	Replace every edge $A \dashv B$ and $A \vdash B$ in $G^*$ with $A \rightarrow B$ and $A - B$ , respectively

Table 2: Rules in the algorithm in Table 1. The antecedents represent induced subgraphs.

R1:		$\Rightarrow$	
R2:		$\Rightarrow$	
R3:		$\Rightarrow$	
R4:		$\Rightarrow$	

let  $S = Ne_G(A) \cup Pa_G(A \cup Ne_G(A))$ , otherwise let  $S = Ne_G(B) \cup Pa_G(B \cup Ne_G(B))$ . In line 5, that the cycle has no blocks means that the ends of the edges in the cycle have no blocks. Note that the rule R1 is not used in line 6, because it will never fire after its repeated application in line 4. Finally, note that  $G^*$  may have edges without blocks after line 6.

### 3 STRONG EDGES

We say that a directed edge in a CG is strong if it appears in every equivalent CG. Likewise for undirected edges. Therefore, strong edges are features of a class of equivalent CGs. Clearly, strong directed edges correspond to directed edges in the EG of the equivalence

class. However, the opposite is not true. Likewise for strong undirected edges. For an example, consider the EG  $A \rightarrow B \leftarrow C - D$ . The naive way to detect which edges in an EG are strong consists in generating all the CGs in the equivalence class and, then, recording the shared edges. Since there may be many CGs in the equivalence class, enumerating them in an efficient manner is paramount, but challenging. In truth, it suffices to enumerate what we call the minimally oriented CGs in order to identify the strong directed edges and, then, find one maximally oriented CG to identify the strong undirected edges. We prove these claims in Section 3.1. Although there are typically considerably fewer minimally oriented CGs, enumerating them in an efficient manner seems challenging too. That is why we present in Section 3.2 an algorithm that does not rely on enumerating CGs or minimally oriented CGs.

#### 3.1 MINIMALLY AND MAXIMALLY ORIENTED CGs

Given a CG  $G$ , merging two of its chain components  $U$  and  $L$  implies replacing the edge  $A \rightarrow B$  with  $A - B$  for all  $A \in U$  and  $B \in L$ . We say that a merging is feasible when

1.  $L \subseteq Ch_G(X)$  for all  $X \in Pa_G(L) \cap U$ ,
2.  $Pa_G(L) \cap U$  is a complete set,
3.  $Pa_G(Pa_G(L) \cap U) \subseteq Pa_G(Y)$  for all  $Y \in L$ , and
4.  $De_G(U) \cap Pa_G(L) = \emptyset$ .

A feasible merging of two chain components of a CG results in an equivalent CG (Sonntag and Peña, 2015, Lemma 2). If a CG does not admit any feasible merging, then we call it minimally oriented. Note that several equivalent minimally oriented CGs may exist, e.g.  $A \rightarrow B - C$  and  $A - B \leftarrow C$ . Note also that an EG is not necessarily a minimally oriented CG, e.g.  $A \rightarrow B \leftarrow C$ . If the directed edges of a CG are a subset of the directed edges of a second CG (with the same orientation), then we say that the former is larger than the latter.

**Lemma 2.** *The minimally oriented CGs in an equivalence class are the maximally large CGs in the class, and vice versa.*

*Proof.* Clearly, a maximally large CG must be minimally oriented because, otherwise, it admits a feasible merging which results in a larger CG, which is a contradiction. On the other hand, let  $G$  be a minimally oriented CG, and assume to the contrary that there is a CG  $H$  that is equivalent but larger than  $G$ . Specifically, let  $G$  have an edge  $A \rightarrow B$  whereas  $H$  has an edge  $A - B$ . Consider a

topological ordering of the chain components of  $G$ . We say that an edge  $X \rightarrow Y$  precedes an edge  $Z \rightarrow W$  in  $G$  if the chain component of  $X$  precedes the chain component of  $Z$  in the ordering, or if both chain components coincide and the chain component of  $Y$  precedes the chain component of  $W$  in the ordering. Assume without loss of generality that no other edge that is directed in  $G$  but undirected in  $H$  precedes the edge  $A \rightarrow B$  in  $G$ . Let  $U$  and  $L$  denote the chain components of  $A$  and  $B$ , respectively. Clearly, all the directed edges from  $U$  to  $L$  in  $G$  must be undirected in  $H$  because, otherwise,  $H$  has a semidirected cycle. However, this implies a contradiction. To see it, recall that  $G$  is a minimally oriented CG and, thus, merging  $U$  and  $L$  in  $G$  is not feasible. If condition 1 fails, then  $G$  has an induced subgraph  $X \rightarrow Y - Z$  where  $X \in U$  and  $Y, Z \in L$ , whereas  $H$  has an induced subgraph  $X - Y - Z$ . However, this implies that  $G$  and  $H$  are not equivalent, since  $G$  has a triplex  $(X, Y, Z)$  that  $H$  has not.

If condition 2 fails but condition 1 holds, then  $G$  has an induced subgraph  $X \rightarrow Y \leftarrow Z$  where  $X, Z \in U$  and  $Y \in L$ , whereas  $H$  has an induced subgraph  $X - Y - Z$ . However, this implies that  $G$  and  $H$  are not equivalent, since  $G$  has a triplex  $(X, Y, Z)$  that  $H$  has not.

If condition 3 fails but condition 1 holds, then  $G$  has an induced subgraph  $Z \rightarrow X \rightarrow Y$  where  $X \in U, Y \in L$  and  $Z \in V \setminus (U \cup L)$ , whereas  $H$  has an induced subgraph  $Z \rightarrow X - Y$ . However, this implies that  $G$  and  $H$  are not equivalent, since  $H$  has a triplex  $(Z, X, Y)$  that  $G$  has not. Note that  $Z \rightarrow X$  is in  $H$  because  $Z \rightarrow X$  precedes  $X \rightarrow Y$  and thus  $A \rightarrow B$  in  $G$ .

Finally, if condition 4 fails but condition 1 holds, then  $G$  has a subgraph of the form  $X \rightarrow Y \leftarrow \dots \leftarrow Z \leftarrow X' \leftarrow \dots \leftarrow X$  where  $X, X' \in U, Y \in L$  and  $Z \in V \setminus (U \cup L)$ , whereas  $H$  has a subgraph of the form  $X - Y - \dots - Z - X$ . To see it, note that any other option results in a semidirected cycle because, recall,  $H$  is larger than  $G$ . However, this is a contradiction because  $X' \rightarrow Z$  precedes  $X \rightarrow Y$  and thus  $A \rightarrow B$  in  $G$ .  $\square$

The following result follows from the previous lemma.

**Theorem 1.** *A directed edge is strong if and only if it is in every minimally oriented CG in the equivalence class.*

Finally, one may think that an undirected edge that is in every minimally oriented CG in the equivalence class is strong. But this is not true. For an example, consider the equivalence class represented by the EG  $A - B$ . Instead, an undirected edge is strong if and only if it is in any maximally oriented CG in the equivalence class (Sonntag and Peña, 2015, Theorems 4 and 5). Formally, a maximally oriented CG is a CG that does not admit any

Table 3: Algorithm to label strong edges in an EG. It replaces line 7 of the algorithm in Table 1.

7	Label every edge $X \dashv Y$ as strong in $G^*$
8	For each edge $X \dashv Y$ in $G^*$
9	Set $H = G^*$
10	Replace $X \dashv Y$ in $H$ with $X \dashv Y$
11	Apply the rules R2-3 to $H$ while possible
12	If $G^*$ has an induced subgraph $A \dashv B \dashv C$ whereas $H$ has $A \dashv B \dashv C$ then
13	Label $X \dashv Y$ as strong in $G^*$
14	Replace every edge $X \dashv Y$ and $X \dashv Y$ in $G^*$ with $X \rightarrow Y$ and $X - Y$ , respectively

feasible split, which is the inverse operation of the feasible merge operation described before. Alternatively, we can say that if the minimally oriented CGs are the maximally large CGs in an equivalence class, then the maximally oriented CGs are the minimally large (Sonntag and Peña, 2015, Lemma 13). Note that several equivalent maximally oriented CGs may exist (e.g.,  $A \rightarrow B$  and  $A \leftarrow B$ ) but all of them have the same undirected edges (Sonntag and Peña, 2015, Theorems 4 and 5). Note also that an EG is not necessarily a maximally oriented CG, e.g.  $A - B$ .

### 3.2 ENUMERATION-FREE ALGORITHM

Although the minimally and maximally oriented CGs in an equivalence class can be obtained by repeatedly performing feasible splits and merges (Sonntag and Peña, 2015, Theorem 3), the approach outlined above for identifying strong edges via enumeration may be inefficient for all but small domains. Hence, Table 3 presents an alternative algorithm that does not rely on enumerating the CGs or the minimally oriented CGs in the equivalence class. The new algorithm replaces line 7 in Table 1. In other words, the new algorithm postpones orienting edges until line 14, and in lines 7-13 it identifies which of the future directed and undirected edges are strong. Line 7 identifies the strong undirected edges, whereas lines 8-13 identify the strong directed edges. To do the latter, the algorithm tries to build a CG  $H$  that is equivalent to  $G^*$  and contains an edge  $X - Y$ . If this fails, then  $X \rightarrow Y$  is strong. Specifically, line 10 forces the edge between  $X$  and  $Y$  to be undirected in  $H$  by blocking the end at  $Y$ . Line 11 computes other blocks that follow from the new block at  $Y$ . After line 11,  $H$  can be oriented as indicated in line 14 without creating a semidirected cycle or a triplex that is not in  $G^*$ . Finally, line 12 checks if every triplex in  $G^*$  is in  $H$ . If not,  $X - Y$  is incompatible with some triplex in  $G^*$ , which implies that  $X \rightarrow Y$  is strong in  $G^*$ . We prove the correctness of the algorithm below.

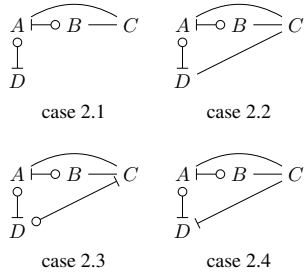


**Lemma 3.** After line 11,  $H$  does not have any induced subgraph of the form  $A \overleftarrow{\circ} B \rightarrow C$ .

*Proof.* The proof is an adaptation of the proof of Lemma 5 by Peña (2014). Assume to the contrary that the lemma does not hold. We interpret the execution of lines 10-11 as a sequence of block additions and, for the rest of the proof, one particular sequence of these block additions is fixed. Fixing this sequence is a crucial point upon which some important later steps of the proof are based. Since there may be several induced subgraphs of  $H$  of the form under study after lines 10-11, let us consider any of the induced subgraphs  $A \overleftarrow{\circ} B \rightarrow C$  that appear first during the execution of lines 10-11 and fix it for the rest of the proof. Note that  $H$  has no such induced subgraph after line 9 (Sonntag and Peña, 2015, Lemma 9). Now, consider the following cases.

**Case 1** Assume that  $A \overleftarrow{\circ} B$  is in  $H$  due line 10. However, this implies that  $H$  had an induced subgraph  $A \overleftarrow{\circ} B \rightarrow C$  before line 10, which is a contradiction (Sonntag and Peña, 2015, Lemma 9).

**Case 2** Assume that  $A \overleftarrow{\circ} B$  is in  $H$  due to R2 in line 11. Then, after line 11,  $H$  has an induced subgraph of one of the following forms:



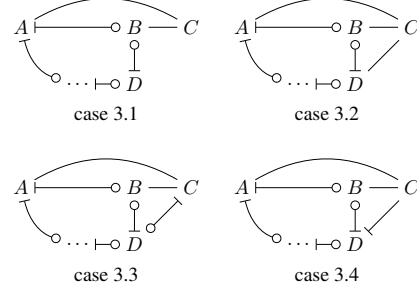
**Case 2.1** If  $A \notin S_{CD}$  then  $A \rightarrow C$  is in  $H$  by R1 in line 4 of Table 1, else  $A \leftarrow C$  is in  $H$  by R2. Either case is a contradiction.

**Case 2.2** Note that  $D \overleftarrow{\circ} A \rightarrow C$  cannot be an induced subgraph of  $H$  after line 11 because, otherwise, it would contradict the assumption that  $A \overleftarrow{\circ} B \rightarrow C$  is one of the first induced subgraph of that form that appeared during the execution of lines 10-11. So, this case is impossible.

**Case 2.3** Note that  $A \rightarrow C$  is in  $H$  by R3, which is a contradiction.

**Case 2.4** If  $C \notin S_{BD}$  then  $B \leftarrow C$  is in  $H$  by R1 in line 4 of Table 1, else  $B \rightarrow C$  is in  $H$  by R2. Either case is a contradiction.

**Case 3** Assume that  $A \overleftarrow{\circ} B$  is in  $H$  due to R3 in line 11. Then, after line 11,  $H$  had a subgraph of one of the following forms, where possible additional edges between  $C$  and internal nodes of the route  $A \overleftarrow{\circ} \dots \overleftarrow{\circ} D$  are not shown:



Note that  $C$  cannot belong to the route  $A \overleftarrow{\circ} \dots \overleftarrow{\circ} D$  because, otherwise, R3 could not have been applied since the cycle  $A \overleftarrow{\circ} \dots \overleftarrow{\circ} D \rightarrow B \rightarrow A$  would not have been chordless.

**Case 3.1** If  $B \notin S_{CD}$  then  $B \rightarrow C$  is in  $H$  by R1 in line 4 of Table 1, else  $B \leftarrow C$  is in  $H$  by R2. Either case is a contradiction.

**Case 3.2** Note that  $D \overleftarrow{\circ} B \rightarrow C$  cannot be an induced subgraph of  $H$  after line 11 because, otherwise, it would contradict the assumption that  $A \overleftarrow{\circ} B \rightarrow C$  is one of the first induced subgraph of that form that appeared during the execution of lines 10-11. So, this case is impossible.

**Case 3.3** Note that  $B \rightarrow C$  is in  $H$  by R3, which is a contradiction.

**Case 3.4** Note that  $C$  cannot be adjacent to any node of the route  $A \overleftarrow{\circ} \dots \overleftarrow{\circ} D$  besides  $A$  and  $D$  and, thus,  $A \leftarrow C$  is in  $H$  by R3. To see it, assume to the contrary that  $C$  is adjacent to some nodes  $E_1, \dots, E_n \neq A, D$  of the route  $A \overleftarrow{\circ} \dots \overleftarrow{\circ} D$ . Assume without loss of generality that  $E_i$  is closer to  $A$  in the route than  $E_{i+1}$  for all  $1 \leq i < n$ . Now, note that  $E_n \overleftarrow{\circ} C$  must be in  $H$  by R3. This implies that  $E_{n-1} \overleftarrow{\circ} C$  must be in  $H$  by R3. By repeated application of this argument, we can conclude that  $E_1 \overleftarrow{\circ} C$  must be in  $H$  and, thus,  $A \leftarrow C$  must be in  $H$  by R3, which is a contradiction.

□

**Lemma 4.** After line 11, every chordless cycle  $\rho : V_1, \dots, V_n = V_1$  in  $H$  that has an edge  $V_i \leftarrow V_{i+1}$  also has an edge  $V_j \rightarrow V_{j+1}$ .

*Proof.* The proof is an adaptation of the proof of Lemma 6 by Peña (2014). Assume for a contradiction that  $\rho$  is of the length three such that  $V_1 \vdash V_2$  occur and neither  $V_2 \dashv V_3$  nor  $V_1 \dashv V_3$  occur. Note that  $V_2 \dashv V_3$  cannot occur either because, otherwise,  $V_1 \dashv V_3$  or  $V_1 \vdash V_3$  must occur by R3. Since the former contradicts the assumption, then the latter must occur. However, this implies that  $V_1 \vdash V_2$  must occur by R3, which contradicts the assumption. Similarly,  $V_1 \dashv V_3$  cannot occur either. Then,  $\rho$  is of one of the following forms:

$$V_1 \vdash V_2 \dashv V_3 \quad V_1 \vdash V_2 \circ \dashv V_3 \quad V_1 \vdash V_2 \vdash V_3$$

The first form is impossible by Lemma 3. The second form is impossible because, otherwise,  $V_2 \circ \dashv V_3$  would occur by R3. The third form is impossible because, otherwise,  $V_1 \dashv V_3$  would be occur by R3. Thus, the lemma holds for cycles of length three.

Assume for a contradiction that  $\rho$  is of length greater than three and has an edge  $V_i \dashv V_{i+1}$  but no edge  $V_j \dashv V_{j+1}$ . Note that if  $V_l \dashv V_{l+1} \circ \dashv V_{l+2}$  is a subroute of  $\rho$ , then either  $V_{l+1} \dashv V_{l+2}$  or  $V_{l+1} \vdash V_{l+2}$  is in  $\rho$  by R1 and R2. Since  $\rho$  has no edge  $V_j \dashv V_{j+1}$ ,  $V_{l+1} \dashv V_{l+2}$  is in  $\rho$ . By repeated application of this reasoning together with the fact that  $\rho$  has an edge  $V_i \dashv V_{i+1}$ , we can conclude that every edge in  $\rho$  is  $V_k \dashv V_{k+1}$ . Then, by repeated application of R3, observe that every edge in  $\rho$  is  $V_k \vdash V_{k+1}$ , which contradicts the assumption.  $\square$

**Lemma 5.** *After line 11,  $H$  can be oriented as indicated in line 14 without creating a semidirected cycle.*

*Proof.* Assume to the contrary that the orientation produces a semidirected cycle  $\rho : V_1, \dots, V_n$ . Note that  $\rho$  must have a chord because, otherwise,  $\rho$  is impossible by Lemma 4. Specifically, let the chord be between  $V_i$  and  $V_j$  with  $i < j$ . Then, divide  $\rho$  into the cycles  $\rho_L : V_1, \dots, V_i, V_j, \dots, V_n = V_1$  and  $\rho_R : V_i, \dots, V_j, V_i$ . Note that  $\rho_L$  or  $\rho_R$  is a semidirected cycle but shorter than  $\rho$ . By repeated application of this reasoning, we can conclude that the orientation produces a chordless semidirected cycle, which contradicts Lemma 4.  $\square$

**Lemma 6.** *After line 11,  $H$  can be oriented as indicated in line 14 without creating a triplex that is not in  $G^*$ .*

*Proof.* We call pretriplex to an induced subgraph of  $G^*$  or  $H$  that results in a triplex when  $G^*$  or  $H$  are oriented as indicated in line 14. Note that  $G^*$  and  $H$  have the same pretriplexes after line 9. Assume to the contrary that after line 11  $H$  has a pretriplex that is not in  $G^*$ . Assume that the spurious pretriplex is created in line 10 when  $A \dashv B$  becomes  $A \vdash B$ . Then, after line 11  $H$

has a pretriplex (1)  $A \vdash B \dashv C$  or (2)  $C \dashv A \vdash B$ . Case (1) implies that  $H$  has actually an induced subgraph  $A \vdash B \vdash C$  by R2, which is a contradiction. To see that R2 is applicable, note that  $B \in S_{AC}$  because  $G^*$  does not have a triplex  $(A, B, C)$ . Case (2) implies that  $H$  has actually an induced subgraph  $C \vdash A \vdash B$  by R2, which again is a contradiction. As before, R2 is clearly applicable. Finally, assume that the spurious pretriplex is created in line 11. Then, after line 11  $H$  has an induced subgraph (1)  $A \vdash B \dashv C$ , (2)  $A \dashv B \dashv C$  or (3)  $A \dashv B \vdash C$ . However, this implies that  $H$  has actually an induced subgraph  $A \vdash B \vdash C$  or  $A \dashv B \vdash C$  by R2, which again is a contradiction. As before, R2 is clearly applicable.  $\square$

**Lemma 7.** *After line 14, the undirected edges in  $G^*$  that had no blocks after line 7 are not strong.*

*Proof.* The proof is an adaptation of the proof of Theorem 11 by Sonntag and Peña (2015). Let  $F$  denote the graph that contains all and only the edges of  $G^*$  resulting from the replacements in line 14, and let  $U$  denote the graph that contains the rest of the edges of  $G^*$  after line 14. Note that all the edges in  $U$  are undirected and they had no blocks when line 14 was to be executed. Therefore,  $U$  has no cycle of length greater than three that is chordless by line 5. In other words,  $U$  is chordal. Then, we can orient all the edges in  $U$  without creating triplexes nor directed cycles by using, for instance, the maximum cardinality search (MCS) algorithm (Koller and Friedman, 2009, p. 312). Consider any such orientation of the edges in  $U$  and denote it  $D$ . Now, add all the edges in  $D$  to  $F$ . As we show below, this last step does not create any triplex or semidirected cycle in  $F$ :

- It does not create a triplex  $(A, B, C)$  in  $F$  because, otherwise,  $A \dashv B \circ \dashv C$  must exist in  $G^*$  when line 14 was to be executed, which implies that  $A \dashv B$  or  $A \circ \dashv B$  was in  $G^*$  by R1 or R2 when line 14 was to be executed, which contradicts that  $A \dashv B$  is in  $U$ .
- Assume to the contrary that it does create a semidirected cycle  $\rho$  in  $F$ . We can assume without loss of generality that  $\rho$  is chordless because if it has a chord between  $V_i$  and  $V_j$  with  $i < j$ . Then, divide  $\rho$  into the cycles  $\rho_L : V_1, \dots, V_i, V_j, \dots, V_n = V_1$  and  $\rho_R : V_i, \dots, V_j, V_i$ . Note that  $\rho_L$  or  $\rho_R$  is a semidirected cycle but shorter than  $\rho$ . By repeated application of this reasoning, we can conclude that  $F$  has a chordless semidirected cycle.

Since  $D$  has no directed cycles,  $\rho$  must have a  $\dashv$  or  $\vdash$  edge when line 14 was to be executed. The former case is impossible (Sonntag and Peña, 2015,

Lemma 10). The latter case implies that  $A - B \vdash C$  must exist in  $G^*$  when line 14 was to be executed, which implies that  $A$  and  $C$  are adjacent in  $G^*$  because, otherwise,  $A \vdash B$  or  $A \dashv B$  was in  $G^*$  by R1 or R2 when line 14 was to be executed, which contradicts that  $A - B$  is in  $U$ . Then,  $A \vdash C$  or  $A \dashv C$  exists in  $G^*$  when line 14 was to be executed (Sonntag and Peña, 2015, Lemma 9), which implies that  $A \vdash B$  or  $A \dashv B$  was in  $G^*$  by R3 when line 14 was to be executed, which contradicts that  $A - B$  is in  $U$ .

Consequently,  $F$  is a CG that is Markov equivalent to  $G$ . Finally, let us recall how the MCS algorithm works. It first unmarks all the nodes in  $U$  and, then, iterates through the following step until all the nodes are marked: Select any of the unmarked nodes with the largest number of marked neighbors and mark it. Finally, the algorithm orients every edge in  $U$  away from the node that was marked earlier. Clearly, any node may get marked first by the algorithm because there is a tie among all the nodes in the first iteration, which implies that every edge may get oriented in any of the two directions in  $D$  and thus in  $F$ . Therefore, either orientation of every edge of  $U$  occurs in some CG  $F$  that is Markov equivalent to  $G$ . Then, every edge of  $U$  must be a strong undirected edge in  $G^*$ .  $\square$

**Theorem 2.** *Table 3 identifies all and only the strong edges in  $G^*$ .*

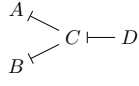
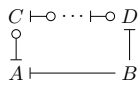


*Proof.* By definition of EG, the edges in  $G^*$  with blocks on both ends in line 7 correspond to strong undirected edges in  $G^*$  after line 14. Moreover, the edges in  $G^*$  with no blocks in line 7 correspond to non-strong undirected edges in  $G^*$  after line 14, by Lemma 7.

After line 11,  $H$  can be oriented as indicated in line 14 without creating semidirected cycles by Lemma 5, and without creating a triplex that is not in  $G^*$  by Lemma 6. Therefore, if  $H$  can be oriented as indicated in line 14 without destroying any of the triplexes in  $G^*$ , then the algorithm has found a CG that is Markov equivalent to  $G^*$  and such that  $X \rightarrow Y$  is in  $G^*$  but  $X - Y$  is in the CG found and, thus,  $X \rightarrow Y$  is non-strong in  $G^*$ . Otherwise,  $X \rightarrow Y$  is strong in  $G^*$ . This is checked in line 12.  $\square$

The algorithm in Table 3 may be sped up with the help of the rules in Table 4. S1-3 should be run while possible before line 8, and S4-6 should be run while possible after line 8 to propagate the labellings due to line 13 in the previous iteration.

**Corollary 1.** *Applying the rules in Table 4 to an EG  $G^*$  correctly identifies strong directed edges in  $G^*$ .*

Table 4: Rules for accelerating the search for strong directed edges in an EG. The antecedents represent induced subgraphs.

S1:		$\Rightarrow C \vdash D$ is strong
S2:	$A \vdash B \vdash C$	$\Rightarrow A \vdash B$ is strong
S3:		$\Rightarrow A \vdash B$ is strong
S4:	$A - B \vdash C$ and $A \vdash B$ is strong	$\Rightarrow B \vdash C$ is strong
S5:		$\Rightarrow A \vdash B$ is strong and $C \vdash B$ is strong
S6:		$\Rightarrow A \vdash B$ is strong and $A \vdash C$ is strong

*Proof.* Consider any member  $G$  of the equivalence class of  $G^*$ . Consider the rule S1. Since  $G^*$  has a triplex  $(A, C, B)$  after line 14,  $G$  must have an edge  $A \rightarrow C$  or  $B \rightarrow C$ . In either case  $G$  must also have an edge  $C \rightarrow D$ , since  $G^*$  has not a triplex  $(A, C, D)$  or  $(B, C, D)$ .

Consider the rule S2. Since  $G^*$  has a triplex  $(A, B, C)$  after line 14 and  $G$  has an edge  $B - C$  due to the blocks at  $B$  and  $C$ , then  $G$  must also have an edge  $A \rightarrow B$ .

Consider the rule S3. Assume to the contrary that  $G$  has an edge  $A - B$ . Then,  $G$  must have an edge  $D \rightarrow B$  since  $G^*$  has a triplex  $(A, B, D)$  after line 14. However, this implies that  $G$  has a semidirected cycle due to the blocks in the antecedent of the rule, which is a contradiction.

Consider the rule S4. Since  $G^*$  has not a triplex  $(A, B, C)$  after line 14 and  $G$  has an edge  $A \rightarrow B$  because it is strong, then  $G$  must also have an edge  $B \rightarrow C$ .

Consider the rule S5. Since  $G$  has an edge  $C \rightarrow B$  because it is strong, then  $G$  must also have an edge  $A \rightarrow B$  to avoid having a semidirected cycle, because either  $A \rightarrow C$  or  $A - C$  is in  $G$  due to the blocks in the antecedent of the rule. The rule S6 can be proven similarly.  $\square$

The rules in Table 4 are by no means complete, i.e. there may be strong edges that the rules alone do not detect. Thus, additional rules can be created. We doubt though that a complete set of concise rules can be produced. The difficulty lies in the disjunctive nature of some labellings. For instance, let an EG  $G^*$  have induced subgraphs  $A \rightarrow C \leftarrow B$ ,  $A \rightarrow C \rightarrow \dots \rightarrow D \rightarrow E$  and  $B \rightarrow C \rightarrow \dots \rightarrow D \rightarrow E$ . Since  $G^*$  has no triplex in  $A \rightarrow C \rightarrow \dots \rightarrow D \rightarrow E$ , if a member  $G$  of the equivalence class of  $G^*$  has an edge  $A \rightarrow C$  then it has an edge  $D \rightarrow E$ . Similarly, if  $G$  has an edge  $B \rightarrow C$  then it has an edge  $D \rightarrow E$ . Then,  $G$  has an edge  $D \rightarrow E$  because it has an edge  $A \rightarrow C$  or  $B \rightarrow C$ , since  $G^*$  and thus  $G$  has a triplex  $(A, C, B)$ . Therefore,  $D \rightarrow E$  is strong. Although it is easy to produce a rule for this example, many more such disjunctive examples exist and we do not see any way to produce concise rules for all of them.

## 4 CAUSAL EFFECT BOUNDS

When the true CG is unknown, a causal effect of the form  $p(y|do(x))$  with  $X, Y \in V$  cannot be computed, but it can be bounded as follows:

1. Obtain all the CGs that are Markov equivalent to the true one by running the learning algorithm developed by Peña (2014, 2016) or Peña and Gómez-Olmedo (2016).
2. Compute the causal effect for each CG obtained as follows. Like in a Bayesian network, any causal effect in a CG  $G$  is computable uniquely from observed quantities (i.e. it is identifiable) by adjusting for the appropriate variables. Specifically,

$$p(y|do(x)) = \int p(y|x, z)p(z)dz$$

where  $Z = Ne_G(X) \cup Pa_G(X \cup Ne_G(X))$  and  $Y \notin Z$ . The role of  $Z$  is to block every non-causal path in  $G$  between  $X$  and  $Y$ . We call  $Z$  the adjusting set in  $G$ .

Unfortunately, the learning algorithm in step 1 may be too time consuming for all but small domains. At least, this is the conclusion that follows from the experimental results reported by Sonntag et al. (2015) for a similar algorithm for learning Lauritzen-Wermuth-Frydenberg CGs. Instead, we propose the following alternative approach:

- 1'. Learn the EG  $G^*$  corresponding to the true CG from data as follows. First, learn a CG from data as shown by Peña (2014, 2016) and Peña and Gómez-Olmedo (2016) and, then, transform it into an EG as shown by Sonntag and Peña (2015, Section 3).

- 2'. Enumerate all the CGs that are Markov equivalent to  $G^*$  as shown by Sonntag and Peña (2015, Theorem 3).
- 3'. Compute the causal effect for each CG enumerated as shown above.

This approach has successfully been applied when the causal models are represented by other graphical models than CGs (Hyttinen et al., 2015; Malinsky and Spirtes, 2016; Maathuis et al., 2009). The experimental results reported by Peña and Gómez-Olmedo (2016) indicate that the learning algorithm in step 1' scales to medium sized domains. However, the enumeration in step 2' may be too time consuming for all but small domains. Alternatively, we may try to enumerate the adjusting sets in the equivalent CGs without enumerating these explicitly. Specifically, we know that all the adjusting sets are subsets of  $Ad_{G^*}(X) \cup Ad_{G^*}(Ad_{G^*}(X))$ , because all the equivalent CGs have the same adjacencies as  $G^*$ . Therefore, we can adjust for every subset of  $Ad_{G^*}(X) \cup Ad_{G^*}(Ad_{G^*}(X))$  to obtain bounds for the causal effect of interest. True that some of these subsets are not valid adjusting sets in the sense that they do not correspond to any of the equivalent CGs. However, this does not make the bounds invalid, just more loose. The rest of the section studies a case where all and only the valid adjusting sets can be enumerated efficiently.

Assume that we believe a priori that the dependencies in the domain at hand are due to causal rather than non-causal relationships. Then, we believe a posteriori that the true CG is a maximally oriented CG, because such CGs have the fewest undirected edges in the equivalence class of the EG  $G^*$  learned from the data in step 1'. Moreover, recall from Section 3.1 that all of them have the same undirected edges. Therefore, we can bound the causal effect  $p(y|do(x))$  by modifying the latter framework above so that only maximally oriented CGs are enumerated in step 2'. A maximally oriented CG that is equivalent to  $G^*$  can be obtained from  $G^*$  by repeatedly performing feasible splits (Sonntag and Peña, 2015, Theorem 3). Unfortunately, this enumeration method may be inefficient for all but small domains. Instead, we show below how to enumerate the adjusting sets in the maximally oriented CGs that are equivalent to  $G^*$  without enumerating these explicitly.

Given a node  $X \in V$ , we define  $St_{G^*}(X) = \{A|A - X \text{ is a strong edge in } G^*\}$  and  $Nst_{G^*}(X) = \{A|A - X \text{ is a non-strong edge in } G^*\}$ . Given a set  $S \subseteq Nst_{G^*}(X)$ , we let  $G_{S \rightarrow X}^*$  denote the graph that is obtained from  $G^*$  by replacing the edge  $A - X$  with  $A \rightarrow X$  for all  $A \in S$ , and replacing the edge  $A - X$  with  $A \leftarrow X$  for all  $A \in Nst_{G^*}(X) \setminus S$ . Moreover, we say that  $G_{S \rightarrow X}^*$  is locally valid if  $G_{S \rightarrow X}^*$  does not have any triplex  $(A, X, B)$  that

is not in  $G^*$ . The next theorem proves that producing the adjusting sets in the equivalent maximally oriented CGs simplifies to produce locally valid sets.

**Theorem 3.**  $G_{S \rightarrow X}^*$  is locally valid if and only if there is a maximally oriented CG  $G$  that is equivalent to  $G^*$  and such that  $Ne_G(X) = St_{G^*}(X)$  and  $Pa_G(X) = Pa_{G^*}(X) \cup S$ , which implies that the adjusting set in  $G$  is  $St_{G^*}(X) \cup Pa_{G^*}(X \cup St_{G^*}(X)) \cup S$ .

*Proof.* The proof is an adaptation of the proof of Lemma 3.1 by Maathuis et al. (2009). The if part is trivial. To prove the only if part, note first that  $S \cup X$  is a complete set because, otherwise,  $G_{S \rightarrow X}^*$  would not be locally valid.

Let  $G$  denote the graph that contains all and only the non-strong undirected edges in  $G^*$ . Recall from Lemma 7 that these edges had no blocks when line 14 in Table 3 was to be executed. Therefore,  $G$  is chordal by line 5 in Table 1. We now show that we can orient the edges of  $G$  without creating triplexes or directed cycles and such that  $Pa_G(X) = S$ . Specifically, we show that there is a perfect elimination sequence that ends with  $X$  followed by the nodes in  $S$ . Orienting the edges of  $G$  according to this sequence produces the desired graph. If  $G$  is complete, then the sequence clearly exists. If  $G$  is not complete, then note that  $G$  has at least two non-adjacent simplicial nodes (Jensen and Nielsen, 2007, Theorem 4.1). Note that one of them is outside of  $S \cup X$  because, as shown above, the latter is a complete set. Take that node as the first node in the sequence. Note moreover that the subgraph of  $G$  induced by the rest of the nodes is chordal. Therefore, we can repeat the previous step to select the next node in the sequence until we obtain the desired perfect elimination sequence.

Finally, consider the oriented  $G$  obtained in the previous paragraph, and add to it all the directed edges and strong undirected edges in  $G^*$ . We now prove that  $G$  is the desired CG in the theorem. First, note that  $G$  is maximally oriented because all the undirected edges in it are strong in  $G^*$ . Second, note that if  $G^*$  has a triplex  $(A, B, C)$  then  $A \dashv B \dashv C$  must be in  $G^*$  when line 14 was to be executed, which implies that neither of the edges in the triplex is non-strong undirected in  $G^*$ , which implies that  $G$  has a triplex  $(A, B, C)$ . Third, note that  $G$  does not have a triplex  $(A, B, C)$  that is not in  $G^*$  because, otherwise, the triplex should have been created as a product of the perfect elimination sequence above. This is possible only if  $A - B - C$  or  $A - B \dashv C$  exists in  $G^*$  when line 14 was to be executed. The former case is impossible by definition of perfect elimination sequence. The latter case implies that  $A \dashv B$  or  $A \dashv C$  was in  $G^*$  by R1 or R2 when line 14 was to be executed, which contradicts that  $A - B$  was a non-strong undirected edge in  $G^*$ .

Fourth, assume to the contrary that  $G$  has a semidirected cycle  $\rho : V_1, \dots, V_n$ . We can assume without loss of generality that  $\rho$  is chordless because if it has a chord between  $V_i$  and  $V_j$  with  $i < j$ . Then, divide  $\rho$  into the cycles  $\rho_L : V_1, \dots, V_i, V_j, \dots, V_n = V_1$  and  $\rho_R : V_i, \dots, V_j, V_i$ . Note that  $\rho_L$  or  $\rho_R$  is a semidirected cycle but shorter than  $\rho$ . By repeated application of this reasoning, we can conclude that  $G$  has a chordless semidirected cycle. Note that it follows from the paragraph above that  $\rho$  cannot consist of just non-strong undirected edges in  $G^*$ . Then, it includes some edge that was  $A \dashv B$  or  $A \dashv C$  when line 14 was to be executed. The former alternative is impossible (Sonntag and Peña, 2015, Lemma 10). The latter alternative implies that  $A \dashv B - C$  must exist in  $G^*$  when line 14 was to be executed, which implies that  $A$  and  $C$  are adjacent in  $G^*$  because, otherwise,  $B \dashv C$  or  $B \dashv C$  was in  $G^*$  by R1 or R2 when line 14 was to be executed, which contradicts that  $B - C$  is a non-strong undirected edge in  $G^*$ . Then,  $A \dashv C$  or  $A \dashv C$  exists in  $G^*$  when line 14 was to be executed (Sonntag and Peña, 2015, Lemma 9), which implies that  $B \dashv C$  or  $B \dashv C$  was in  $G^*$  by R3 when line 14 was to be executed, which contradicts that  $B - C$  is a non-strong undirected edge in  $G^*$ .  $\square$

The procedure outlined above can be simplified as follows.

**Corollary 2.**  $St_{G^*}(X) = \emptyset$  or  $Nst_{G^*}(X) = \emptyset$ .

*Proof.* Assume the contrary. Then,  $G^*$  has a subgraph  $A \dashv X - B$  when line 14 in Table 3 is to be executed. Then,  $A$  and  $B$  are adjacent in  $G^*$  because, otherwise, the edge  $X - B$  would have some block by R1 or R2. However, this implies that the edge  $A - B$  has some block by Lemma 3, which implies that  $X - B$  has some block by R3. This is a contradiction.  $\square$

## 5 DISCUSSION

In this paper, we have presented an algorithm to identify the strong edges in an EG. We have also shown how this makes it possible to compute bounds of causal effects under the assumption that the true CG is unknown but maximally oriented. In the future, we would like to derive a similar result for minimally oriented CGs. Moreover, as mentioned in the introduction, an EG is a deflagged graph but not necessarily the largest in the equivalence class. Therefore, an EG may contain a directed edge where the largest deflagged graph has an undirected edge. Then, the algorithm in Table 3 may be improved by consulting the largest deflagged graph before trying labeling a directed edge as strong. An algorithm for constructing this graph exists (Roverato and Studený, 2006).

## References

- Andersson, S. A., Madigan, D. and Perlman, M. D. Alternative Markov Properties for Chain Graphs. *Scandinavian Journal of Statistics*, 28:33-85, 2001.
- Andersson, S. A. and Perlman, M. D. Characterizing Markov Equivalent Classes for AMP Chain Graph Models. Technical Report 453, University of Washington, 2004. Available at <http://www.stat.washington.edu/www/research/reports/2004/tr453.pdf>.
- Andersson, S. A. and Perlman, M. D. Characterizing Markov Equivalent Classes for AMP Chain Graph Models. *The Annals of Statistics*, 34:939-972, 2006.
- Koller, D. and Friedman, N. *Probabilistic Graphical Models*. MIT Press, 2009.
- Hyttinen, A., Eberhardt, F. and Järvisalo, M. Do-calculus when the True Graph is Unknown. In *Proceedings of the 31th Conference on Uncertainty in Artificial Intelligence*, 395-404, 2015.
- Jensen, F. V. and Nielsen, T. D. *Bayesian Networks and Decision Graphs*. Springer Verlag, 2007.
- Levitz, M., Perlman M. D. and Madigan, D. Separation and Completeness Properties for AMP Chain Graph Markov Models. *The Annals of Statistics*, 29:1751-1784, 2001.
- Maathuis, M. H., Kalisch, M. and Bühlmann, P. Estimating High-Dimensional Intervention Effects from Observational Data. *The Annals of Statistics*, 37:3133-3164, 2009.
- Malinsky, D. and Spirtes, P. Estimating Causal Effects with Ancestral Graph Markov Models. In *Proceedings of the 8th International Conference on Probabilistic Graphical Models*, 299-309, 2016.
- Pearl, J. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2009.
- Peña, J. M. Learning AMP Chain Graphs and some Marginal Models Thereof under Faithfulness. *International Journal of Approximate Reasoning*, 55:1011-1021, 2014.
- Peña, J. M. Alternative Markov and Causal Properties for Acyclic Directed Mixed Graphs. In *Proceedings of the 32nd Conference on Uncertainty in Artificial Intelligence*, 577-586, 2016.
- Peña, J. M. and Gómez-Olmedo, M. Learning Marginal AMP Chain Graphs under Faithfulness Revisited. *International Journal of Approximate Reasoning*, 68:108-126, 2016.
- Peters, J., Janzing, D. and Schölkopf, B. *Elements of Causal Inference: Foundations and Learning Algorithms*. The MIT Press, 2017.
- Roverato, A. and Studený, M. A Graphical Representation of Equivalence Classes of AMP Chain Graphs. *Journal of Machine Learning Research*, 7:1045-1078, 2006.
- Sonntag, D. and Peña, J. M. Chain Graph Interpretations and their Relations Revisited. *International Journal of Approximate Reasoning*, 58:39-56, 2015.
- Sonntag, D., Järvisalo, M., Peña, J. M. and Hyttinen, A. Learning Optimal Chain Graphs with Answer Set Programming. In *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence*, 822-831, 2015.

---

# A Univariate Bound of Area Under ROC

---

Siwei Lyu

Yiming Ying

Computer Science Department   Mathematics Department  
University at Albany, State University at New York, USA  
{slyu,yying}@albany.edu

## Abstract

Area under ROC (AUC) is an important metric for binary classification and bipartite ranking problems. However, it is difficult to directly optimize AUC as a learning objective, so most existing algorithms are based on optimizing a surrogate loss to AUC. One significant drawback of these surrogate losses is that they require pairwise comparisons among training data, which leads to slow running time and increasing local storage for online learning. In this work, we describe a new surrogate loss based on a reformulation of AUC risk, which does not require pairwise comparison but rankings of the predictions. We further show that the ranking operation can be avoided, and the learning objective obtained based on this surrogate enjoys linear complexity in time and storage. We perform experiments to demonstrate the effectiveness of the online and batch algorithms for AUC optimization based on the proposed surrogate loss.

## 1 INTRODUCTION

The *area under receiver operating characteristics curves* (AUC) is a useful quantitative metric for assessing the performance of binary classification and bipartite ranking algorithms [1, 2]. However, there are two factors make AUC difficult to be used directly as a learning objective to train classification or ranking algorithms. The foremost is due to the discontinuous indicator function in the definition of the AUC (c.f. Eq.(1)), which makes direct minimization of the AUC in general an NP-hard problem [4]. As such, most existing AUC learning algorithm replace the indicator function with surrogates that are continuous and convex upper-bounds of the AUC.

The second issue with the AUC is the requirement of pairwise comparison between all positive and negative examples in training data. This leads to algorithms with a running time complexity that is quadratic in the number of training data, and a space complexity that is linear of the training data. For batch algorithms, this means slow running time as we need to compare all pairs of positive/negative examples, and for online learning, this means ever increasing local storage as we need to store all previously seen data for the pairwise comparisons. Both are undesirable when applying these algorithms to large-scale datasets.

In this work, we describe a new surrogate loss to AUC that has a linear time complexity and constant space complexity. This new loss is based on an equivalent formulation of AUC based on ranking the prediction scores, which obviates pairwise comparisons. We further show that the ranking operation can be replaced with an equivalent optimization problem, and the learning objective affords a simple form that has a bounding relation with AUC. Furthermore, we show that the new loss has a close relation with the SVM learning objective, which sheds light on the previous observations of the effectiveness of the SVM on optimizing AUC [5, 6, 7, 8]. The new surrogate loss leads naturally to an online AUC optimization method with simple (projected) stochastic sub-gradient steps. Experimental evaluations on several standard benchmark datasets show that learning objective formed from this new loss achieves performance in par with other widely used AUC surrogates, with a significant reduction in running time and storage requirement.

## 2 DEFINITIONS

To facilitate subsequent description, we first review the definition of AUC in the context of binary classification. Assume we are given a set of data  $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ , with  $y_i \in \{-1, +1\}$  and  $\mathbf{x}_i \in \mathcal{R}^d$ . We denote  $\mathcal{I}^+ = \{i|y_i =$

$+1\}$  and  $\mathcal{I}^- = \{i|y_i = -1\}$  as the sets of indices of positive and negative examples, respectively, with  $N^+ = |\mathcal{I}^+|$  and  $N^- = |\mathcal{I}^-|$ , and  $N^+ + N^- = N$ . Define  $\mathbf{I}$  as the *indicator function*:  $\mathbf{I}_a = 1$  if  $a$  is true and 0 otherwise. A parametric binary classifier  $c_{\mathbf{w},\theta} : \mathcal{R}^d \mapsto \{-1, +1\}$ , constructed as

$$c_{\mathbf{w},\theta}(\mathbf{x}) = 2\mathbf{I}_{f_{\mathbf{w}}(\mathbf{x}) \geq \theta} - 1 = \text{sign}(f_{\mathbf{w}}(\mathbf{x}) - \theta),$$

maps an example to the class label, where  $f_{\mathbf{w}} : \mathcal{R}^d \mapsto \mathcal{R}$  (with  $\mathbf{w} \in \mathcal{R}^m$  being the parameter) is the prediction function and  $\theta \in \mathcal{R}$  is the classification threshold. We denote  $c_i = f_{\mathbf{w}}(\mathbf{x}_i)$  as the *prediction score* of the  $i^{\text{th}}$  example ( $i = 1, \dots, N$ ). For simplicity, we assume there are no ties in the prediction scores, *i.e.*,  $c_i \neq c_j$  for  $i \neq j$ , though this condition will be relaxed later.

Given a threshold  $\theta$ , negative examples with prediction scores greater than  $\theta$  are false positives, and the false positive rate is given by  $\tau_{FP} = |\mathbf{I}_{c_i > \theta \wedge i \in \mathcal{I}^-}|/N^-$ . Correspondingly, positive examples with prediction scores greater or equal to  $\theta$  are true positives, and the true positive rate is given by  $\tau_{TP} = |\mathbf{I}_{c_i \geq \theta \wedge i \in \mathcal{I}^+}|/N^+$ . Then the receiver operation curve (ROC) is defined as the curve formed by the pair  $(\tau_{FP}, \tau_{TP})$  with  $\theta \in (-\infty, \infty)$ . With this definition, ROC is a curve confined to  $[0, 1] \times [0, 1]$  and connecting  $(0, 0)$  to  $(1, 1)$ . AUC then corresponds to the area enclosed by the ROC curve of the classifier.

It is more conveniently computed in closed form using the Wilcoxon-Mann-Whitney (WMW) statistic [3], as  $A = \frac{1}{N^+N^-} \sum_{i \in \mathcal{I}^+} \sum_{j \in \mathcal{I}^-} \mathbf{I}_{c_i > c_j}$ . In this work, we use the *AUC risk*, which is defined as

$$L_{\text{AUC}} = 1 - A = \frac{1}{N^+N^-} \sum_{i \in \mathcal{I}^+} \sum_{j \in \mathcal{I}^-} \mathbf{I}_{c_i < c_j}. \quad (1)$$

Note that  $L_{\text{AUC}}$  takes values in  $[0, 1]$  and corresponds to the fraction of pairs of positive and negative predictions that are ranked incorrectly, *i.e.*, a positive example with lower prediction score than a negative example, so  $L_{\text{AUC}} = 0$  indicates perfect classification/ranking. In addition,  $L_{\text{AUC}}$  is independent of threshold  $\theta$ , and only concerns with the overall performance of the predictor  $f_{\mathbf{w}}$ . Hence, we aim to learn a prediction function  $f_{\mathbf{w}}$  that minimizes  $L_{\text{AUC}}$ , from which we can choose  $\theta$  to construct classifier  $c_{\mathbf{w},\theta}(\mathbf{x})$ .

### 3 RELATED WORKS

Most existing works for either batch or online algorithms for AUC optimization (*e.g.*, [9, 10]) minimize surrogates to the true AUC risk, which are usually in the form of convex upper-bounds to the indicator function in Eq.(1). Specifically, denoting the prediction scores for  $\mathbf{x}_i$  and  $\mathbf{x}_j$  as  $c_i$  and  $c_j$ , respectively, the surrogate loss function

takes the form as  $\frac{1}{N^+N^-} \sum_{i \in \mathcal{I}^+} \sum_{j \in \mathcal{I}^-} \ell(c_i - c_j)$ , and common choices for  $\ell$  include

1. the hinge loss [10],  $\ell_h(c_i, c_j) = [1 - (c_i - c_j)]_+$ , where  $[a]_+ = \max\{0, a\}$  is the hinge function,
2. the squared hinge loss [11, 9],  $\ell_{sh}(c_i, c_j) = [1 - (c_i - c_j)]_+^2$ ,
3. the logistic loss,  $\ell_{lg} = \log_2(1 + e^{c_i - c_j})$ ,
4. and the rank-boost loss [12],  $\ell_e(c_i, c_j) = e^{c_i - c_j}$ .

All these surrogates are nonnegative, monotonic decreasing and satisfy  $\ell(c_i, c_j) = 1$  when  $c_i = c_j$ . One significant problem with these surrogates is that they all rely on pairwise comparisons between positive and negative training examples, which lead to algorithms with quadratic running time complexity. For large datasets, such quadratic running time will significantly slow down the training process, and the pairwise comparisons prohibit efficient online learning algorithms for AUC optimization.

One exception is the work of [11], which shows that the squared hinge surrogate of AUC risk,  $\ell_{sh}(c_i, c_j)$ , affords an equivalent saddle point reformulation. An online stochastic gradient descent method is then developed based on this reformulation that has complexities  $O(N)$  in time and  $O(1)$  in space. However, there are two issues of this method that this work aims to improve on. First, the original surrogate loss still requires pairwise comparison, and to decouple them, one needs to introduce auxiliary variables for a saddle point reformulation. In contrast, our surrogate loss obviates pairwise comparison all together. Second, our surrogate loss reduces to a minimization problem, which is easier to analyze and implement than the saddle point reformulation of [11].

In parallel with methods directly optimizing AUC, empirical observations suggest that learning objectives not designed for AUC optimization (*e.g.*, SVM or boosting) can achieve low AUC risk [5, 6, 7, 8]. For instance, in [6], a generalized SVM approach was developed that is able to optimize multivariate non-linear performance measures in polynomial time, including AUC. However, when assessed with respect to the AUC, the superiority of the direct AUC optimization approach over standard SVMs seemed less convincing. The work of [7] many performance measures for binary classification are compared experimentally, and it was found that maximum margin methods such as boosting and SVMs yield excellent performance when measured with AUC. In [5] it was shown that optimizing standard SVMs leads to maximizing the AUC in the special (trivial) case when the given data is separable. As a perfect separation implies a zero AUC risk. The work [13] uses the rank-equivalent



definition of AUC to derive a hinge rank loss and shows that it is analogous to the SVM objective. However, no explicit relation between the SVM objective and AUC or AUC surrogates are established in previous works.

Further along this line, several studies have provided results on the consistency of the univariate losses to AUC risk, *i.e.*, in the asymptotic sense, minimizing the univariate losses under certain conditions may also lead to the minimization of AUC risk [14, 15], and a similar analysis is conducted for binary surrogate losses to AUC risk in [16]. These analyses show that univariate losses such as the  $\ell_2$ , squared hinge and exponential losses are consistent with AUC risk, yet the widely used hinge loss in SVM are inconsistent. This seems to put in question whether minimizing AUC risk based on pairwise comparisons is really warranted. However, these studies are still of limited in practice due to several reasons. First, they can not explain the observation that the SVM algorithm which is based on the hinge loss, oftentimes leads to good performance when evaluated with AUC risk, though it is not theoretically consistent with AUC risk. In addition, these analysis does not reveal a direct relation between the univariate losses and AUC risk, and it is more illustrative if some bounding relation between them can be revealed. Furthermore, these analyses may not be as relevant in practice, as the learning objective in actual algorithms is usually combined with extra terms such as the regularizers.

## 4 METHOD

In this section, we start with an equivalent definition of AUC risk, which does not require pairwise comparisons of positive and negative examples. From this equivalent definition, we establish our AUC surrogate loss and its equivalent form for efficient optimization.

### 4.1 AUC Risk Without Pairwise Comparison

Besides the WMW statistics, Eq.(1), there exists another equivalent formulation of AUC risk (and AUC itself), which depends on the ranking of the prediction scores instead of all pairwise comparisons of the prediction scores of the positive and negative examples [4, 13]. To explain this equivalent form of AUC risk, we first introduce several additional notations. For simplicity, we assume there are no ties in the prediction scores, *i.e.*,  $c_i \neq c_j$  for  $i \neq j$ , though this condition will be relaxed later.

We denote  $(c_1^\uparrow, \dots, c_N^\uparrow)$  as the result of sorting  $(c_1, \dots, c_N)$  in ascending order, *i.e.*,  $c_1^\uparrow < c_2^\uparrow < \dots < c_N^\uparrow$ . Moreover, let  $r_i^+ \in \{1, \dots, N\}$  ( $i = 1, \dots, N^+$ ) be the *rank* of the  $i^{\text{th}}$  positive example encountered in the or-



Figure 1: An illustration of the ranking definition of the AUC. Note that in this case, we have  $N^+ = 7$ ,  $N^- = 6$ , and  $(r_1^+, r_2^+, r_3^+, r_4^+, r_5^+, r_6^+, r_7^+) = (4, 6, 7, 8, 9, 11, 13)$ . For the positive example highlighted with circle, it is the second positive example in the ordered list, and it is outranked by two negative examples (shown by arrows). So its contribution to AUC risk is  $N^- + i - r_i^+ = 6 + 2 - 6 = 2$ . Repeating for all 7 positive examples the total wrong pairs is  $3 + 2 + 2 + 2 + 1 + 0 = 12$  and AUC risk is  $\frac{12}{6 \times 7} = \frac{2}{7}$ , which is the same as computed with Eq.(1).

dered list  $(c_1^\uparrow, \dots, c_N^\uparrow)$  starting from the beginning. With a slight abuse of notation, let  $c_i^{\uparrow+}$  be the corresponding value of the  $i^{\text{th}}$  positive example in the ordered list  $(c_1^\uparrow, \dots, c_N^\uparrow)$ , *i.e.*,  $c_i^{\uparrow+} = c_{r_i^+}^\uparrow$ . An example illustrating these definitions is given in Fig.1. These definitions immediately lead to the following simple result that will be important subsequently.

**Lemma 1.** For  $i = 1, \dots, N^+$ , we have

$$r_i^+ \leq N^- + i, \quad c_{N^-+i}^\uparrow \geq c_i^{\uparrow+}.$$

Proof of Lemma 1 is provided in the Appendix A.

With these definitions, AUC risk can be defined using the rankings of the predictions [4], which is equivalent to the definition based on the WMW statistics as given in Eq.(1). The intuition behind this equivalent form is a different way to count the number of reverse ordered pairs of positive and negative examples, which is illustrated with the numerical example in Figure 1.

**Lemma 2** ([4]). When there is no ties in training data, *i.e.*,  $c_i \neq c_j$  for  $i \neq j$ , we have

$$\begin{aligned} L_{AUC} &= \frac{1}{N^+N^-} \sum_{i=1}^{N^+} (N^- + i - r_i^+) \\ &= 1 + \frac{N^++1}{2N^-} - \frac{1}{N^+N^-} \sum_{i=1}^{N^+} r_i^+. \end{aligned} \quad (2)$$

Proof of Lemma 2 is provided in the Appendix A. Note that  $\sum_{i=1}^{N^+} (N^- + i)$  corresponds to (trivially) the sum of the indices of the largest  $N^+$  (top- $N^+$ ) elements in the ranked list of prediction scores, and  $\sum_{i=1}^{N^+} r_i^+$  is the sum of the indices of positive examples in the ranked list of predictions. As such, AUC risk as defined in Eq.(2) is proportional to the difference between the two sums.

This gives another intuitive explanation of AUC risk: in a perfect separable case, when the prediction scores of all the positive examples rank higher than those of the negative examples, *i.e.*, all prediction scores of positive examples have ranks  $N^- + 1, \dots, N$  in the ordered list, AUC risk is zero. In the more general cases, AUC risk measures how the rankings of the prediction scores deviate from this ideal case.

## 4.2 Univariate Bound on AUC risk

The equivalent form of AUC risk of Eq.(2) inspires a new surrogate loss based on the values of the sorted prediction scores  $(c_1^\uparrow, \dots, c_N^\uparrow)$ . To be specific, let us define a new quantity

$$\tilde{L} = \frac{1}{N^+N^-} \sum_{i=N^++1}^N c_i^\uparrow - \frac{1}{N^+N^-} \sum_{i \in \mathcal{I}^+} c_i. \quad (3)$$

Like AUC risk,  $\tilde{L}$  is always nonnegative, as the second term, which is the sum of the prediction scores of all the positive examples, is less than or equal to the first term, which is the sum of the top- $N^+$  elements of  $(c_1, \dots, c_N)$ . Equality holds only when the predictions of all positive examples rank higher than any of the negative examples. Our next result shows that we can bound AUC risk using  $\tilde{L}$ .

**Theorem 1.** *When there is no ties in training data, i.e.,  $c_i \neq c_j$  for  $i \neq j$ , we have  $\tilde{L} \geq 0$ . Furthermore, there exist constants  $\bar{\alpha} \geq \underline{\alpha} > 0$ , such that  $\bar{\alpha}\tilde{L} \geq L_{AUC} \geq \underline{\alpha}\tilde{L}$ .*

*Proof.* Using Lemma 1, we have  $\sum_{i=1}^{N^+} (c_{N^++i}^\uparrow - c_i^{\uparrow+}) \geq 0$ , therefore  $\tilde{L} \geq 0$ , and it is zero when  $c_{N^++i}^\uparrow = c_i^{\uparrow+}$  for all  $i = 1, \dots, N^+$ , i.e., all positive examples outrank all negative examples.

We set  $\bar{\alpha}^{-1} = \min_i (c_{i+1}^\uparrow - c_i^\uparrow) > 0$ , and for  $i > j$ , we have  $c_i^\uparrow - c_j^\uparrow = (c_i^\uparrow - c_{i-1}^\uparrow) + (c_{i-1}^\uparrow - c_{i-2}^\uparrow) + \dots + (c_{j+1}^\uparrow - c_j^\uparrow) \geq \frac{i-j}{\bar{\alpha}}$ . Then we have

$$\begin{aligned} \bar{\alpha}\tilde{L} &= \frac{\bar{\alpha}}{N^+N^-} \sum_{i=N^++1}^N c_i^\uparrow - \frac{\bar{\alpha}}{N^+N^-} \sum_{i \in \mathcal{I}^+} c_i \\ &= \frac{\bar{\alpha}}{N^+N^-} \sum_{i=1}^{N^+} (c_{N^++i}^\uparrow - c_i^{\uparrow+}) \\ &= \frac{\bar{\alpha}}{N^+N^-} \sum_{i=1}^{N^+} (c_{N^++i}^\uparrow - c_{r_i}^{\uparrow+}) \\ &\geq \frac{1}{N^+N^-} \sum_{i=1}^{N^+} (N^- + i - r_i^{\uparrow+}) = L_{AUC}. \end{aligned}$$

Next, setting  $\underline{\alpha}^{-1} = \max_i (c_{i+1}^\uparrow - c_i^\uparrow)$ , and follow a similar derivation, we can obtain the other bound, i.e.,  $L_{AUC} \geq \underline{\alpha}\tilde{L}$ . The equalities in the bounds hold when  $c_{i+1}^\uparrow - c_i^\uparrow$  is a constant for  $i = 1, \dots, N$ , i.e., they are equally spaced.  $\square$

## 4.3 Computing $\tilde{L}$ without Explicit Ranking

However, the ranking operation in  $\tilde{L}$  is the main obstacle of using Eq.(3) as a learning objective. However, this can be solved based on the following result on the sum of the top  $k$  elements in a set [17, 18], from which we can derive an equivalent form of Eq.(3) that does not rely on ranking elements explicitly.

**Lemma 3** ([17, 18]). *For  $N$  real numbers  $z_1 < \dots < z_N$ , we have the equivalence of the sum-of-top- $k$  elements with an optimization problem as*

$$\sum_{i=N-k+1}^N z_i = \min_{\lambda} \left\{ k\lambda + \sum_{i=1}^N [z_i - \lambda]_+ \right\}, \quad (4)$$

with the optimal  $\lambda^*$  satisfying  $z_{N-k} \leq \lambda^* < z_{N-k+1}$ .

Proof of Lemma 3 is provided in the Appendix A. Using Lemma 3, we can rewrite  $\tilde{L}$  by as a minimization problem over the auxiliary variable  $\lambda$ , as

$$N^+N^-\tilde{L} = \min_{\lambda} \left\{ N^+\lambda + \sum_{i=1}^N [c_i - \lambda]_+ \right\} - \sum_{i \in \mathcal{I}^+} c_i,$$

which can be further converted to

$$\min_{\lambda} \left\{ \sum_{i \in \mathcal{I}^+} ([c_i - \lambda]_+ - (c_i - \lambda)) + \sum_{j \in \mathcal{I}^-} [c_j - \lambda]_+ \right\}.$$

Using the property of the hinge function that  $[a]_+ - a = [-a]_+$ , we can further simplify  $\tilde{L}$ , as

$$\begin{aligned} \tilde{L} &= \frac{1}{N^+N^-} \min_{\lambda} \left\{ \sum_{i \in \mathcal{I}^+} [\lambda - c_i]_+ + \sum_{j \in \mathcal{I}^-} [c_j - \lambda]_+ \right\} \\ &= \frac{1}{N^+N^-} \min_{\lambda} \sum_{i=1}^N [y_i(\lambda - c_i)]_+. \end{aligned}$$

Bringing back the parametric model to form a learning objective based on  $\tilde{L}$  as

$$\tilde{L}(\mathbf{w}) = \frac{1}{N^+N^-} \min_{\lambda} \sum_{i=1}^N [y_i(\lambda - f_{\mathbf{w}}(\mathbf{x}_i))]_+. \quad (5)$$

This reformulation of  $\tilde{L}$  is still a bound for AUC risk, but it does not require pairwise comparisons between predictions of positive and negative examples, and there is no need to explicitly ranking the predictions. Furthermore, in Eq.(5), the auxiliary variable  $\lambda$  can be understood as a threshold that separates the two classes, and  $\tilde{L}(\mathbf{w})$  becomes independent of the choice of threshold by taking the overall minimum over all possible values for the threshold, as in the case of the original definition of AUC risk.

The learning objective  $\tilde{L}(\mathbf{w})$  affords an intuitive interpretation in the context of binary classification. It only penalizes those positive examples with predictions less than the threshold, i.e.,  $[\lambda - f_{\mathbf{w}}(\mathbf{x}_i)]_+$  for  $i \in \mathcal{I}^+$ , and negative examples with predictions greater than the threshold, i.e.,  $[f_{\mathbf{w}}(\mathbf{x}_i) - \lambda]_+$  for  $i \in \mathcal{I}^-$ . All examples that are on the ‘‘correct’’ side of the threshold receive no penalty. According to Lemma 3, the optimal  $\lambda$  takes value in the range of  $[c_{N^+}^\uparrow, c_{N^++1}^\uparrow)$ .

#### 4.4 Relation with SVM Objective

There are some strong similarities between  $\tilde{L}(\mathbf{w})$  and the SVM objective, which is particularly striking in the case of linear prediction function  $f_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$ . This becomes clearer if we reformulate the SVM objective: if we regard the threshold  $\lambda$  as the bias term in the linear prediction function for SVM<sup>1</sup>,  $\mathbf{w}^\top \mathbf{x} - \lambda$ , we can formulate the linear SVM objective [19] as

$$\tilde{L}_{\text{SVM}}(\mathbf{w}, \lambda) = \sum_{i=1}^N [1 + y_i(\lambda - \mathbf{w}^\top \mathbf{x}_i)]_+.$$

Now comparing with Eq.(5), the two objectives has similar forms involving the hinge function. We can further show that  $\tilde{L}_{\text{SVM}}(\mathbf{w}, \lambda)$  is an upper-bound of  $\tilde{L}(\mathbf{w})$ . This is because we have  $[1 + y_i(\lambda - \mathbf{w}^\top \mathbf{x}_i)]_+ \geq [y_i(\lambda - \mathbf{w}^\top \mathbf{x}_i)]_+$ , so

$$\begin{aligned} \tilde{L}_{\text{SVM}}(\mathbf{w}, \lambda) &= \sum_{i=1}^N [1 + y_i(\lambda - \mathbf{w}^\top \mathbf{x}_i)]_+ \\ &\geq \sum_{i=1}^N [y_i(\lambda - \mathbf{w}^\top \mathbf{x}_i)]_+ \\ &\geq \min_{\lambda} \sum_{i=1}^N [y_i(\lambda - \mathbf{w}^\top \mathbf{x}_i)]_+ \\ &= \tilde{L}(\mathbf{w}). \end{aligned}$$

As we have shown in Theorem 1, an upper-bound of AUC risk can be established with  $\tilde{L}(\mathbf{w})$ , and this relation suggests the SVM objective  $\tilde{L}_{\text{SVM}}(\mathbf{w}, \lambda)$  is also an upper-bound (albeit looser than  $\tilde{L}$ ) of AUC risk.

This helps to explain some long standing experimental observations (e.g., [5, 6, 7, 8]) that when assessed with AUC, standard SVMs could not be consistently outperformed by other approaches tailored to directly maximize AUC, such as RankBoost [20], AUCsplit (local optimization of AUC) [21], or ROC-SVM [8].

The two learning objectives also differ in two important aspects. The first is the constant 1 in the SVM objective, which corresponds to the margin in constructing the binary classifier. The second difference is that the bias  $\lambda$  in  $\tilde{L}$  is eliminated through minimization, but it is still present in the SVM objective.

## 5 OPTIMIZATION

In this section, we discuss batch and online learning algorithms based on learning objectives formed from Eq.(5).

### 5.1 Resolving Ties in Prediction Scores

However, Eq.(5) cannot be used as a learning objective due to one important issue. Note that in Eq.(5), the scale

<sup>1</sup>Typically in SVM we define the linear prediction function as  $\mathbf{w}^\top \mathbf{x} + b$ , but here we flip the sign of the bias so to better compare with  $\tilde{L}(\mathbf{w})$ .

of the parameter  $\mathbf{w}$  is not fixed, so the learning objective can be reduced by shrinking the scale of  $\mathbf{w}$ , which leads to a trivial solution with  $\mathbf{w} = 0$ . The underlying reason is that the formulation of  $\tilde{L}$  is based on the assumption of no ties in the prediction scores, while the trivial solution corresponds to the extreme contrary, i.e., the prediction function always produce the same output (zero) regardless of the data.

To resolve this problem, we augment the objective function with two other terms

$$\min_{\mathbf{w}} \tilde{L}(\mathbf{w}) + \frac{\beta}{2} \sum_{i=1}^N (1 - y_i f_{\mathbf{w}}(\mathbf{x}_i))^2 + \gamma \Omega(\mathbf{w}), \quad (6)$$

where the second term corresponds to a least squares term to counteract the effect of concentrating  $\mathbf{w}$  to zero, the third term  $\Omega(\mathbf{w})$  is a regularizer on parameter  $\mathbf{w}$ , and  $(\beta, \gamma)$  are weights to the two extra terms.

### 5.2 Linear Predictor

In general, the learning objective of Eq.(6) is not a convex function of  $\mathbf{w}$ , but if we choose  $f_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$  and  $\Omega(\mathbf{w})$  is convex with respect to  $\mathbf{w}$  (i.e.,  $\Omega(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2$ ), then we can show it is a convex function of  $\mathbf{w}$ . We first show that  $[\mathbf{x}^\top \mathbf{w} - \lambda]_+$  is a convex function. For  $\alpha \in [0, 1]$ ,  $\mathbf{w}, \mathbf{w}', \lambda$ , and  $\lambda'$ , we have

$$\begin{aligned} &[\alpha \mathbf{x}^\top \mathbf{w} + (1 - \alpha) \mathbf{x}^\top \mathbf{w}' - (\alpha \lambda + (1 - \alpha) \lambda')]_+ = \\ &[\alpha (\mathbf{x}^\top \mathbf{w} - \lambda) + (1 - \alpha) (\mathbf{x}^\top \mathbf{w}' - \lambda')]_+ \leq \\ &\alpha [\mathbf{x}^\top \mathbf{w} - \lambda]_+ + (1 - \alpha) [\mathbf{x}^\top \mathbf{w}' - \lambda']_+. \end{aligned} \quad (7)$$

Therefore,  $\sum_{i=1}^N [\mathbf{x}_i^\top \mathbf{w} - \lambda]_+ + N^+ \lambda$  is a convex function jointly for  $(\mathbf{w}, \lambda)$ . As the minimization of one variable in a joint convex function,  $\min_{\lambda} \sum_{i=1}^N [c_i - \lambda]_+ + N^+ \lambda$  is also a convex function of  $\mathbf{w}$ .

In summary, for the linear case, we can obtain the following convex learning objective with regards to  $\mathbf{w}$  and  $\lambda$  jointly,

$$\begin{aligned} &(\mathbf{w}^*, \lambda^*) \leftarrow \operatorname{argmin}_{\mathbf{w}, \lambda} \frac{\gamma}{2} \|\mathbf{w}\|^2 + \\ &\sum_{i=1}^N \left\{ [y_i(\lambda - \mathbf{x}_i^\top \mathbf{w})]_+ + \frac{\beta}{2} (1 - y_i \mathbf{x}_i^\top \mathbf{w})^2 \right\} \end{aligned} \quad (8)$$

In the following, we discuss the batch and online optimization of Eq.(8), for which the convergence to global minimum is guaranteed.

#### 5.2.1 Batch Learning

In the batch setting, where we have access to all training examples, we can use block coordinate descent algorithm to optimize Eq.(8). We initialize  $\mathbf{w}$  and  $\lambda$ , then iterate between

$$\begin{aligned} \mathbf{w}^{(t+1)} &\leftarrow \operatorname{argmin}_{\mathbf{w}} \sum_{i=1}^N [y_i(\lambda^{(t)} - \mathbf{w}^\top \mathbf{x}_i)]_+ + \\ &\quad \frac{\beta}{2} \sum_{i=1}^N (1 - y_i \mathbf{x}_i^\top \mathbf{w})^2 + \frac{\gamma}{2} \|\mathbf{w}\|^2; \\ \lambda^{(t+1)} &\leftarrow \frac{1}{2} (c_{N+}^\uparrow + c_{N+1}^\uparrow), \end{aligned}$$

where  $c_i^\uparrow$  is the rerank of  $\{\mathbf{x}_i^\top \mathbf{w}^{(t+1)}\}_{i=1}^N$  in the ascending order. The  $\mathbf{w}$  sub-problem can be converted to a constrained optimization problem as

$$\begin{aligned} \min_{\mathbf{w}, \mathbf{t}} \quad & \sum_{i=1}^N t_i + \frac{\beta}{2} \sum_{i=1}^N (1 - y_i \mathbf{x}_i^\top \mathbf{w})^2 + \frac{\gamma}{2} \|\mathbf{w}\|^2; \\ \text{s.t.} \quad & \mathbf{y}_i(\lambda^{(t)} - \mathbf{w}^\top \mathbf{x}_i) \geq t_i, t_i \geq 0. \end{aligned}$$

This is a quadratic convex optimization problem and can be solved with interior point method when the dimensionality of  $\mathbf{w}$  is low to medium. For high dimensional  $\mathbf{w}$ , the online learning algorithm is more effective as it avoids building the Hessian matrix.

### 5.2.2 Online Learning

Because Eq.(8) does not involve pairwise comparison, we can also derive an online learning algorithm based on stochastic gradient descent [22, 23]. The runtime of the online algorithm does not depend on the number of training examples and thus this algorithm is especially suited for large datasets. Specifically, with initial choice for the value of  $\mathbf{w}^{(0)}$ , at the  $t^{\text{th}}$  iteration, a single training example  $(\mathbf{x}_{i_t}, y_{i_t})$  is chosen at random from the training set and used to estimate a sub-gradient of the objective, and a step with pre-determined step-size is taken in the opposite direction, as

$$\begin{aligned} \mathbf{w}^{(t+1)} &\leftarrow \mathbf{w}^{(t)} - \eta_t \left( (\gamma \mathbf{I} + \beta \mathbf{x}_{i_t} \mathbf{x}_{i_t}^\top) \mathbf{w}^{(t)} - \right. \\ &\quad \left. (\beta + \mathbf{1}_{y_i(\lambda^{(t)} - \mathbf{w}^\top \mathbf{x}_i) > 0}) y_{i_t} \mathbf{x}_{i_t} \right) \\ \lambda^{(t+1)} &\leftarrow \lambda^{(t)} - \eta_t y_{i_t} \mathbf{1}_{y_i(\lambda^{(t)} - \mathbf{w}^\top \mathbf{x}_i) > 0}, \end{aligned} \quad (9)$$

where we can choose the step-size  $\eta_t \sim \frac{1}{\sqrt{t}}$ , then the SGD algorithm will converge in  $O(1/\epsilon)$  steps to the  $\epsilon$ -accuracy of the global optimal value of Eq.(5) [22, 23]. Note that each step of our online iterative algorithm has space and time complexity of  $O(d)$  and  $O(1)$ , and obviates the need to store or buffer data in previous online AUC optimization methods [10, 9].

## 6 EXPERIMENTS

We perform several experiments of learning binary classifiers to evaluate the batch and online algorithms optimizing learning objectives given in Eq.(8) (subsequently denoted as `ba-UBAUC` and `ol-UBAUC`, respectively), and compare their performance with existing learning algorithms for AUC optimization.

As in previous works [10, 11], we perform experiments on 12 benchmark datasets that have been used in previous studies. A summary of the data in these datasets

	train	test	data dim.
diabetes	389	389	8
fourclass	431	431	2
german	500	500	24
splice	1,000	2,175	60
usps	7,291	2,007	256
a9a	32,561	16,281	123
w8a	49,749	14,951	300
mnist	60,000	10,000	780
acoustic	78,823	19,705	50
ijcnn1	49,990	91,701	22
sector	6,412	3,207	55,197
news20	15,935	3,993	62,061

Table 1: Summary of the 12 benchmark datasets used in our experiments. The training/testing splitting is from the original datasets.

is given in Table 1, with the training/testing split obtained from the original dataset. For datasets that are for data with more than 2 class labels (*i.e.*, `news20` and `sector`), following the convention of previous work [10, 11], we convert them to binary classification problems by randomly partitioning the data into two groups, each with equal number of classes. Then the binary class labels are determined from the group to which the original class label belongs. Following the evaluation protocol of [10, 11], the performance of reported is obtained by averaging the AUC scores on the test set for 25 models learned from subsets of the same training set, each is chosen as a random 80% of the original training data.

On these datasets, we evaluate and compare UBAUC-based algorithms with four state-of-the-art online and two batch learning algorithms for learning linear binary classifiers that minimizes various pairwise surrogates to the original AUC risk  $L_{\text{AUC}}$ . The hyper parameters  $(\beta, \gamma)$  for UBAUC are determined by a grid search on the validation set. The initial learning rate for the online learning algorithm is also set for different dataset by a grid search. We compare the following algorithms with UBAUC-based algorithms.

- SOLAM [11], an online AUC optimization algorithm based on a saddle point reformulation of the pairwise  $\ell_2$  surrogate loss of AUC risk;
- OPAUC [9], an online AUC optimization algorithm that uses the pairwise  $\ell_2$  loss surrogate of the AUC objective function;
- OAM [10], an online AUC optimization algorithm that uses the pairwise hinge loss surrogate of the AUC objective function with two variants, one with sequential update (OAMseq) and the other using gradient update (OAMgra);
- B-SVM-OR [6], a batch learning algorithm using the pairwise hinge loss surrogate of the AUC objective function;

	ol-UBAUC	ba-UBAUC	SOLAM	OPAUC	OAM <sub>seq</sub>	OAM <sub>gra</sub>	B-SVM-OR	SVM
diabetes	.8326±.0299	.8328±.0352	.8253±.0314	.8309±.0350	.8264±.0367	.8262±.0338	.8326±.0328	.7821±.0145
fourclass	.8301±.0318	.8310±.0296	.8226±.0240	.8310±.0251	.8306±.0247	.8295±.0251	.8305±.0311	.7717±.0294
german	.7928±.0371	.7933±.0324	.7882±.0243	.7978±.0347	.7747±.0411	.7723±.0358	.7935±.0348	.7641±.0283
splice	.9231±.0224	.9269±.0094	.9253±.0097	.9232±.0099	.8594±.0194	.8864±.0166	.9239±.0089	.8439±.0096
usps	.9728±.0051	.9730±.0066	.9766±.0032	.9620±.0040	.9310±.0159	.9348±.0122	.9630±.0047	.8930±.0075
a9a	.9005±.0019	.9009±.0041	.9001±.0042	.9002±.0047	.8420±.0174	.8571±.0173	.9009±.0036	.8213±.0064
w8a	.9673±.0993	.9695±.0079	.9114±.0075	.9633±.0035	.9304±.0074	.9418±.0070	.9495±.0082	.8964±.0029
mnist	.9327±.0239	.9340±.0024	.9324±.0020	.9242±.0021	.8615±.0087	.8643±.0112	.9340±.0020	.8406±.0072
acoustic	.8871±.0035	.8962±.0046	.8898±.0026	.8192±.0032	.7113±.0590	.7711±.0217	.8262±.0032	.7629±.0045
ijcnn1	.9264±.0039	.9337±.0038	.9215±.0045	.9269±.0021	.9209±.0079	.9100±.0092	.9337±.0024	.8793±.0094
sector	.9845±.0033	-	.9834±.0023	.9292±.0081	.9163±.0087	.9043±.0100	-	.8815±.0062
news20	.9468±.0045	-	.9467±.0039	.8871±.0083	.8543±.0099	.8346±.0094	-	.8431±.0127

Table 2: Comparison of the AUC scores (mean±std.) on test sets of the evaluated datasets.

- UNI-SVM, a linear SVM algorithm implemented using LIBSVM with SMO minimization [24].

Classification performances measured by the AUC score on the testing dataset of all compared methods for all 12 benchmark datasets are given in Table 2. For fair comparison, we implement all algorithms using MATLAB, and following the default parameter settings in the original papers. Note that the simple implementation of the two batch algorithms cannot handle datasets with high dimensional datasets, *i.e.*, `sector` and `news20`, due to the memory requirement. However, for those datasets that it is feasible to run, `ba-UBAUC`, the batch version optimizing the proposed learning objective, performs best. On the other hand, the results of `uUNI-SVM`, though optimizing a different objective, still achieves reasonable performance when evaluated with AUC. The online algorithm based on the proposed learning objective, `ol-UBAUC`, achieves comparable performance as other state-of-the-art online algorithms based on pairwise surrogate losses to AUC risk, although the improvements of performance on some of the datasets are not conspicuous due to the nature of the data.

On the other hand, the main advantage of `ol-UBAUC` in comparison with other online algorithms is the running efficiency – its per-iteration running time and space complexity is linear in data dimension and do not depend on the iteration number. Furthermore, each iteration of `ol-UBAUC` Eq. (9) corresponds to a simpler update step than the saddle point solve in SOLAM [11]. In Table 3, we show the per-iteration running time and the total running time for the learning objective function to converge to have smaller than  $10^{-7}$  relative changes<sup>2</sup> of the five online algorithms we compared. Note that the online version of the UBAUC-based algorithms has more efficient running time with comparable performances in

<sup>2</sup>Experiments were performed with running time reported based on a cluster with 12 nodes, each with an Intel Xeon E5-2620 2.0GHz CPU and 64GB RAM. All algorithms are implemented using MATLAB, with available code obtained from the authors of the corresponding publications.

	a9a	usps	sector
ol-UBAUC	0.48	0.15	11.24
SOLAM	0.50	0.19	19.90
OPAUC	6.24	4.62	120.30
OAMseq	34.31	13.98	1350.41
OAMgra	34.35	12.54	1350.50

	a9a	usps	sector
ol-UBAUC	0.83	0.15/0.58	276.41
SOLAM	0.99	0.19/0.81	721.52
OPAUC	14.21	4.62/11.23	5540.24
OAMseq	78.42	13.98/32.71	6730.75
OAMgra	69.23	12.54/39.54	6324.64

Table 3: **(top)** The average running time (in seconds) per pass over training data for each online algorithm, and **(bottom)** the average running time (in seconds) for the learning objective function to converge to have smaller than  $10^{-7}$  relative changes for each online algorithm.

comparison to existing AUC optimization methods.

## 7 POPULATION FORM

So far, we have described the proposed learning objective over a set of finite training data. In this section, we discuss the population form of the surrogate loss using probability distributions of data. This analysis will shed light on the formal connection of the new surrogate loss with existing methods and can lead to deeper theoretical studies.

We start with the population form of the equivalent definition of AUC risk in Eq.(2). We assume that the input data and label are from a joint model  $p(\mathbf{x}, y)$ , which induces density models for the predictions  $c = f(\mathbf{x})$ . As such, we denote  $\rho^+(c) = p(c|y = 1)$  and  $\rho^-(c) = p(c|y = -1)$  as the (conditional) probability density functions (PDFs) for positive and negative class, respectively. For simplicity, we assume both PDFs have infinite support, *i.e.*, is non-zero for the whole  $\mathcal{R}$ . Also, we denote  $p = Pr(y = 1)$  as the class prior probability.

The joint probability density function of the classification output  $c$  is then given by  $\rho(c) = p\rho^+(c) + (1 -$

$p)\rho^-(c)$ . We also denote  $F^+(c) = \int_{-\infty}^c \rho^+(c')dc'$ ,  $F^-(c) = \int_{-\infty}^c \rho^-(c')dc'$ , and  $F(c) = \int_{-\infty}^c \rho(c')dc'$  as the cumulative distribution functions (CDFs) for  $\rho^+$ ,  $\rho^-$  and  $\rho$ , respectively, with  $F(c) = pF^+(c) + (1-p)F^-(c)$ .  $F^+(c)$  is the false negative rate (FNR) and  $1 - F^-(c)$  is the false positive rate (FPR).

AUC risk is defined as the area under the whole curve of FNR vs. FPR, as  $L_{AUC} = \int_{-\infty}^{\infty} (1 - F^-(c))dF^+(c)$  [4]. Using relation  $F^-(c) = \frac{1}{1-p}(F(c) - pF^+(c))$  yields

$$L_{AUC} = \frac{1}{1-p} \int_{-\infty}^{\infty} (1 - p + pF^+(c) - F(c))dF^+(c).$$

Because  $F$  is a CDF is a continuous monotonic function and  $F(c) \leq 1 - p + pF^+(c) \leq 1$ , using the mean value theorem, there exists  $c' \geq c_0 = \max\{c|F(c) = 1 - p\}$ , such that  $1 - p = F(c_0) \leq F(c') = 1 - p + pF^+(c) \leq 1$ , and

$$L_{AUC} = \frac{1}{1-p} \int_{-\infty}^{\infty} (F(c') - F(c))dF^+(c).$$

Next, note that  $F(c)$  is Lipschitz with constant  $\alpha' \geq \max_c |\rho(c)|$ , i.e.,  $|F(c') - F(c)| \leq \alpha'|c' - c|$ , we have

$$L_{AUC} \leq \frac{\alpha'}{1-p} \int_{-\infty}^{\infty} (c' - c)dF^+(c). \quad (10)$$

Next, we use the following result

**Lemma 4.** For  $F(c') = 1 - p + pF^+(c)$ , we have

$$\int_{-\infty}^{\infty} c' dF^+(c) = \min_{\lambda} \int_{-\infty}^{\infty} \frac{(c - \lambda)_+}{p} dF(c) + \lambda.$$

Proof of Lemma 4 is provided in the Appendix A. Using Lemma 4, we can rewrite the integral of the right hand side of Eq.(10) as

$$\min_{\lambda} \int_{-\infty}^{\infty} \frac{(c - \lambda)_+}{p} dF(c) + \lambda - \int_{-\infty}^{\infty} cdF^+(c),$$

where the terms being minimized can be further simplified as

$$\int_{-\infty}^{\infty} \frac{(c - \lambda)_+}{p} dF(c) + (\lambda - c)dF^+(c).$$

This can be further expanded using the relation  $dF(c) = (1 - p)dF^-(c) + pdF^+(c)$  to have

$$\int_{-\infty}^{\infty} (c - \lambda)_+(1 - p)dF^-(c) + \int_{-\infty}^{\infty} [(\lambda - c) + (c - \lambda)_+]pdF^+(c).$$

Putting all terms together and using the relation  $(c - \lambda)_+ + (\lambda - c) = (\lambda - c)_+$  we have

$$L_{AUC} \leq \frac{\alpha'}{p(1-p)} \min_{\lambda} E_{c,y}[y(c - \lambda)_+], \quad (11)$$

where  $E_{c,y}$  represents the expectation over  $c$  and  $y$ .

## 8 CONCLUSION

In this work, we describe a new surrogate loss to the AUC metric based on a formulation of AUC, which does not require pairwise comparison but rankings of the prediction scores. We further show that the ranking operation can be avoided and the learning objective obtained based on this surrogate affords complexity in time and storage that is linear in the number of training data. We perform experiments to demonstrate the effectiveness of the online and batch algorithms for AUC optimization based on the proposed surrogate.

There are several directions we would like to further explore for this work. First, from the theoretical point of view, we would like to establish the consistency of the proposed learning objective with regards to AUC risk, i.e., the question if the surrogate loss will also lead to the convergence to the optimal AUC risk. The form of our surrogate loss (Eq.(5)) as an optimization problems makes it difficult to apply the techniques used in previous works [14, 15] to this case. We would also like to establish the generalization error between the data form of the loss Eq.(5) and its population form counterpart Eq.(11). From the algorithm perspective, we would like to extend this learning objective to substitute multi-class AUC [4], where multi-class AUC risk is computed as the average of binary class AUC between each pairs of classes. Last, we are interested in applying the online algorithm based on the proposed surrogate loss to non-convex learning objectives such as those used for training deep neural networks.

**Acknowledgement.** Siwei Lyu is supported by the National Science Foundation (NSF, Grant IIS-1537257) and Yiming Ying is supported by the Simons Foundation (#422504) and the 2016-2017 Presidential Innovation Fund for Research and Scholarship (PIFRS) program from SUNY Albany.

## A Appendix: Proofs

*Proof of Lemma 1.* Being the  $i^{\text{th}}$  positive example encountered in the ordered list  $(c_1^{\uparrow}, \dots, c_N^{\uparrow})$ ,  $c_i^{\uparrow+}$  can outrank no more than  $N^- + i$  elements in the list, i.e.,  $i$  positive examples and at most  $N^-$  negative examples. Therefore, we have  $r_i^+ \leq N^- + i$ . By the ranking order we also have  $c_{N^-+i}^{\uparrow} \geq c_{r_i^+}^{\uparrow} = c_i^{\uparrow+}$ .  $\square$

*Proof of Lemma 2.* Consider the  $i^{\text{th}}$  positive example encountered in  $(c_1^{\uparrow}, \dots, c_N^{\uparrow})$  starting from the beginning, which has rank  $r_i^+$ . The number of negative examples that rank lower than it is  $r_i^+ - i$ , i.e., there will be  $N^- - (r_i^+ - i) = N^- + i - r_i^+$  negative examples with

ranks higher than this positive example, *i.e.*, forming a reversed ordered pair with it. This corresponds to the sum over reversed ordered pairs in the definition of AUC risks of Eq.(1). Summing over all such reverse ordered pairs divided by the number of all such positive-negative pairs ( $N^+N^-$ ) proves the result.  $\square$

*Proof of Lemma 3.* First, we note that  $\sum_{i=N-k+1}^N z_i$  is the solution of the following linear programming problem

$$\max_{\mathbf{p} \in \mathbb{R}^{n \times 1}} \mathbf{p}^\top \mathbf{z}, \quad \text{s.t. } \mathbf{p}^\top \mathbf{1} = k, p_i \in [0, 1], \quad (12)$$

We form its Lagrangian as

$$L = -\mathbf{p}^\top \mathbf{z} - \mathbf{a}^\top \mathbf{p} + \mathbf{b}^\top (\mathbf{p} - \mathbf{1}) + \lambda (\mathbf{p}^\top \mathbf{1} - k), \quad (13)$$

where  $\mathbf{a} \geq \mathbf{0}$ ,  $\mathbf{b} \geq \mathbf{0}$  and  $\lambda$  are Lagrangian multipliers. Setting the derivative of  $L$  with respect to  $\mathbf{p}$  to be  $\mathbf{0}$ , we obtain  $\mathbf{a} = \mathbf{b} - \mathbf{z} + \lambda \mathbf{1}$ . Substituting this into Eq (13), we get the dual problem of (12) as

$$\min_{\mathbf{b}, \lambda} \mathbf{b}^\top \mathbf{1} + k\lambda, \quad \text{s.t. } \mathbf{b} \geq \mathbf{0}, \mathbf{b} + \lambda \mathbf{1} - \mathbf{z} \geq \mathbf{0}, \quad (14)$$

The constraints of Eq. (14) suggest that we should have  $\mathbf{b}^\top \mathbf{1} \geq \sum_{i=1}^n [z_i - \lambda]_+$ . As such, the objective function achieves its minimum when the equality holds. Reorganizing terms leads to Eq.(4). Further, when we choose  $\lambda^*$  satisfying  $z_{N-k} \leq \lambda^* < z_{N-k+1}$ , we have  $k\lambda^* + \sum_{i=1}^N [z_i - \lambda^*]_+ = k\lambda^* + \sum_{i=N-k+1}^N (z_i - \lambda^*) = \sum_{i=N-k+1}^N z_i$ . Thus proves the lemma.  $\square$

*Proof of Lemma 4.* First, we have  $dF(c') = p dF^+(c)$ , then  $\int_{-\infty}^{\infty} c' dF^+(c) = \frac{1}{p} \int_{c_0}^{\infty} c' dF(c')$ , where the lower limit of the integral,  $c_0 = \max\{c | F(c) = 1 - p\}$ , originates from the range of value  $c'$ . Next, we compute  $\min_{\lambda} \int_{-\infty}^{\infty} (c - \lambda)_+ dF(c) + p\lambda = \min_{\lambda} \int_{\lambda}^{\infty} c dF(c) - \lambda \int_{\lambda}^{\infty} dF(c) + p\lambda$ . Differentiating the inner terms with regards to  $\lambda$ , we obtain  $\int_{\lambda}^{\infty} dF(c) = p$ , or  $\int_{-\infty}^{\lambda} dF(c) = 1 - p$ , so we have at optimum,  $\lambda = c_0$ . Therefore we have  $\min_{\lambda} \int_{-\infty}^{\infty} (c - \lambda)_+ dF(c) + p\lambda = \int_{c_0}^{\infty} c' dF(c')$ . Further rearranging terms proves the result.  $\square$

## References

- [1] C. Cortes and M. Mohri, "AUC optimization vs. error rate minimization," in *Advances in Neural Information Processing Systems (NIPS)*, 2003.
- [2] W. Kotlowski, K. Dembczynski, and E. Hüllermeier., "Bipartite ranking through minimization of univariate loss," in *International Conference on Machine Learning (ICML)*, 2011.
- [3] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under of receiver operating characteristic (ROC) curve," *Radiology*, vol. 143, no. 1, pp. 29–36, 1982.
- [4] D. J. Hand and R. J. Till, "A simple generalisation of the area under the ROC curve for multiple class classification problems," *Machine learning*, vol. 45, pp. 171–186, 2001.
- [5] U. Brefeld and T. Scheffer, "AUC maximizing support vector learning.," in *Workshop ROC Analysis in AI in conjunction with European Conference on Artificial Intelligence*, 2005.
- [6] T. Joachims, "A support vector method for multivariate performance measures," in *International Conference on Machine Learning (ICML)*, 2005.
- [7] R. Caruana and A. Niculescu-Mizil, "Data mining in metric space: an empirical analysis of supervised learning performance criteria.," in *International Conference on Knowledge Discovery and Data Mining (KDD)*, 2004.
- [8] A. Rakotomamonjy, "Optimizing area ROC curve with SVMs," in *Workshop ROC Analysis in AI in conjunction with European Conference on Artificial Intelligence*, 2004.
- [9] P. Zhao, S. C. H. Hoi, R. Jin, and T. Yang, "Online AUC maximization," in *International Conference on Machine Learning (ICML)*, 2011.
- [10] W. Gao, R. Jin, S. Zhu, and Z. H. Zhou, "One-pass AUC optimization," in *International Conference on Machine Learning (ICML)*, 2013.
- [11] Y. Ying, L. Wen, and S. Lyu, "Stochastic online AUC maximization," in *Advances in Neural Information Processing Systems (NIPS)*, (Barcelona, Spain), December 2016.
- [12] C. Rudin, C. Cortes, M. Mohri, and R. E. Schapire, "Margin-based ranking meets boosting in the middle," in *Learning Theory* (P. Auer and R. Meir, eds.), (Berlin, Heidelberg), pp. 63–78, Springer Berlin Heidelberg, 2005.
- [13] H. Steck, "Hinge rank loss and the area under the ROC curve," in *European Conference on Machine Learning (ECML)*, 2007.
- [14] S. Clemencon and S. Robbiano, "Minimax learning rates for bipartite ranking and plug-in rules," in *International Conference on Machine Learning (ICML)*, 2011.

- [15] S. Agarwal, “Surrogate regret bounds for the area under the roc curve via strongly proper losses,” *Journal of Machine Learning Research*, 2013.
- [16] W. Gao and Z. Zhou, “On the consistency of auc pairwise optimization,” *Artificial Intelligence Journal*, 2014.
- [17] W. Ogryczak and A. Tamir, “Minimizing the sum of the k largest functions in linear time,” *Information Processing Letters*, vol. 85, no. 3, pp. 117–122, 2003.
- [18] Y. Fan, S. Lyu, Y. Ying, and B. Hu, “Learning with average top-k loss,” in *Advances in Neural Information Processing Systems (NIPS)*, (Long Beach, CA), 2017.
- [19] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.
- [20] Y. Freund, R. Iyer, R. Schapire, and Y. Singer, “An efficient boosting algorithm for combining preferences,” *Journal of Machine Learning Research*, vol. 4, pp. 933–969, 2003.
- [21] R. Herbrich, T. Graepel, and K. Obermayer, “Support vector learning for ordinal regression,” in *International Conference on Neural Networks*, 1999.
- [22] O. Bousquet and L. Bottou, “The tradeoffs of large scale learning,” in *NIPS*, pp. 161–168, 2008.
- [23] N. Srebro and A. Tewari, “Stochastic optimization for machine learning,” *ICML Tutorial*, 2010.
- [24] C.-C. Chang and C.-J. Lin, “LIBSVM: A library for support vector machines,” *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.



---

# Efficient Bayesian Inference for a Gaussian Process Density Model

---

**Christian Donner\***

Artificial Intelligence Group  
Technische Universität Berlin  
christian.donner@bccn-berlin.de

**Manfred Opper**

Artificial Intelligence Group  
Technische Universität Berlin  
manfred.opper@tu-berlin.de

## Abstract

We reconsider a nonparametric density model based on Gaussian processes. By augmenting the model with latent Pólya–Gamma random variables and a latent marked Poisson process we obtain a new likelihood which is conjugate to the model’s Gaussian process prior. The augmented posterior allows for efficient inference by Gibbs sampling and an approximate variational mean field approach. For the latter we utilise sparse GP approximations to tackle the infinite dimensionality of the problem. The performance of both algorithms and comparisons with other density estimators are demonstrated on artificial and real datasets with up to several thousand data points.

## 1 INTRODUCTION

Gaussian processes (GP) provide highly flexible non-parametric prior distributions over functions [1]. They have been successfully applied to various statistical problems such as e.g. regression [2], classification [3], point processes [4] or the modelling of dynamical systems [5, 6]. Hence, it would seem natural to apply Gaussian processes also to density estimation which is one of the most basic statistical problems. GP density estimation, however, is a nontrivial task: Typical realisations of a GP do not respect non-negativity and normalisation of a probability density. Hence, functions drawn from a GP prior have to be passed through a nonlinear squashing function and the results have to be normalised subsequently to model a density. These operations make the corresponding posterior distributions non-Gaussian. Moreover, likelihoods depend on all the infinitely many

---

\*Also affiliated with Bernstein Center for Computational Neuroscience.

GP function values in the domain rather than on the finite number of function values at observed data points. Since analytical inference is impossible, [7] introduced an interesting Markov chain Monte–Carlo sampler which allows for (asymptotically) exact inference for a Gaussian process density model, where the GP is passed through a sigmoid link function.<sup>1</sup> The approach is able to deal with the infinite dimensionality of the model, because the sampling of the GP variables is reduced to a finite dimensional problem by a point process representation. However, since the likelihood of the GP variables is not conjugate to the prior, the method has to resort to a time-consuming Metropolis–Hastings approach. In this paper we will use recent results on representing the sigmoidal squashing function as an infinite mixture of Gaussians involving Pólya–Gamma random variables [9] to augment the model in such a way that the model becomes tractable by a simpler Gibbs sampler. The new model structure allows also for a much faster variational Bayesian approximation.

The paper is organised as follows: Sec. 2 introduces the GP density model, followed by an augmentation scheme that makes its likelihood conjugate to the GP prior. With this model representation we derive two efficient Bayesian inference algorithms in Sec. 3, namely an exact Gibbs sampler and an approximate, fast variational Bayes algorithm. The performance of both algorithms is demonstrated in Sec. 4 on artificial and real data. Finally, Sec. 5 discusses potential extensions of the model.

## 2 GAUSSIAN PROCESS DENSITY MODEL

The generative model proposed by [7] constructs densities over some  $d$ -dimensional data space  $\mathcal{X}$  to be of the

---

<sup>1</sup>See [8] for an alternative model allowing, however, only for approximate inference schemes.

form

$$\rho(\mathbf{x}|g) = \frac{\sigma(g(\mathbf{x}))\pi(\mathbf{x})}{\int_{\mathcal{X}} \sigma(g(\mathbf{x}))\pi(\mathbf{x})d\mathbf{x}}. \quad (1)$$

$\pi(\mathbf{x})$  defines a (bounded) base probability measure over  $\mathcal{X}$ , which is usually taken from a fixed parametric family. The denominator ensures normalisation  $\int_{\mathcal{X}} \rho(\mathbf{x}|g)d\mathbf{x} = 1$ . The choice of  $\pi(\mathbf{x})$  is important as will be discussed Sec. 5. A prior distribution over densities is introduced by assuming a Gaussian process prior [1] over the function  $g(\mathbf{x}) : \mathcal{X} \rightarrow \mathbb{R}$ . The GP is defined by a mean function  $\mu(\mathbf{x})$  (in this paper, we consider only constant mean functions  $\mu(\mathbf{x}) = \mu_0$ ) and covariance kernel  $k(\mathbf{x}, \mathbf{x}')$ . Finally,  $\sigma(z) = \frac{1}{1+e^{-z}}$  is the sigmoid function, which guarantees that the density is non-negative and bounded.

In Bayesian inference, the posterior distribution of  $g$  given observed data  $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^N$  with  $\mathbf{x} \in \mathcal{X}$  is computed from the GP prior  $p(g)$  and the likelihood as

$$p(g|\mathcal{D}) \propto p(\mathcal{D}|g)p(g).$$

The likelihood is given by

$$p(\mathcal{D}|g) = \frac{\prod_{n=1}^N \sigma(g(\mathbf{x}_n))\pi(\mathbf{x}_n)}{\left(\int_{\mathcal{X}} \sigma(g(\mathbf{x}))\pi(\mathbf{x})d\mathbf{x}\right)^N}. \quad (2)$$

Practical inference for this problem, however, is non-trivial, because (i) the posterior is non-Gaussian and (ii) the likelihood involves an integral of  $g$  over the whole space. Thus, in contrast to simpler problems such as GP regression or classification, it is impossible to reduce inference to finite dimensional integrals. To circumvent the problem that the likelihood is not conjugate to the GP prior, [7] proposed a Metropolis-Hastings MCMC algorithm for this model. We will show in the next sections that one can augment the model with auxiliary latent random variables in such a way that the resulting likelihood is of a conjugate form allowing for a more efficient Gibbs sampler with explicit conditional probabilities.

## 2.1 LIKELIHOOD AUGMENTATION

To obtain a likelihood which is conjugate to the GP  $p(g)$  we require that it assumes a Gaussian form in  $g$ .

**Representing the denominator** As a starting point, we follow [10] and use the representation

$$\frac{1}{z^N} = \frac{\int_0^\infty \lambda^{N-1} e^{-\lambda z} d\lambda}{\Gamma(N)},$$

where  $\Gamma(\cdot)$  is the gamma function. Identifying  $z = \int_{\mathcal{X}} \sigma(g(\mathbf{x}))\pi(\mathbf{x})d\mathbf{x}$  in Eq. (2) we can rewrite the

likelihood as  $p(\mathcal{D}|g) = \int_0^\infty p(\mathcal{D}, \lambda|g)d\lambda$  where

$$p(\mathcal{D}, \lambda|g) \propto \exp\left(-\int_{\mathcal{X}} \lambda \sigma(g(\mathbf{x}))\pi(\mathbf{x})d\mathbf{x}\right) \times p(\lambda) \prod_{n=1}^N \lambda \sigma(g(\mathbf{x}_n))\pi(\mathbf{x}_n), \quad (3)$$

with the improper prior  $p(\lambda) = \lambda^{-1}$  over the auxiliary latent variable  $\lambda$ . To transform the likelihood further into a form which is Gaussian in  $g$ , we utilise a representation of the sigmoid function as a scale mixture of Gaussians.

### Pólya-Gamma representation of sigmoid function

As discovered by [9], the inverse hyperbolic cosine can be represented as an infinite mixture of scaled Gaussians

$$\cosh^{-b}(z/2) = \int_0^\infty e^{-\frac{z^2}{2}\omega} p_{\text{PG}}(\omega|b, 0)d\omega,$$

where  $p_{\text{PG}}(\omega|b, 0)$  is the *Pólya-Gamma density* of random variable  $\omega \in \mathbb{R}^+$ . Moments of those densities can be easily computed [9]. Later, we will also use the *tilted* Pólya-Gamma densities defined as

$$p_{\text{PG}}(\omega|b, c) \propto \exp\left(-\frac{c^2}{2}\omega\right) p_{\text{PG}}(\omega|b, 0). \quad (4)$$

These definitions allows for a Gaussian representation of the sigmoid function as

$$\sigma(z) = \frac{e^{z/2}}{2 \cosh(z/2)} = \int_0^\infty e^{f(\omega, z)} p_{\text{PG}}(\omega|1, 0)d\omega \quad (5)$$

with  $f(\omega, z) = \frac{z}{2} - \frac{z^2}{2}\omega - \ln 2$ . This result will be used to transform the products over observations  $\sigma(g(\mathbf{x}_n))$  in the likelihood (3) into a Gaussian form.

We will next deal with the first term in the likelihood (3) which contains the integral over  $\mathbf{x}$ . For this part of the model we will derive a point process representation which can be understood as a generalisation of the approach of [7].

**Marked-Poisson representation** Utilising the sigmoid property  $\sigma(z) = 1 - \sigma(-z)$  and the Pólya-Gamma representation (5) the integral in the exponent of Eq. (3) can be written as a double integral

$$\begin{aligned} -\int_{\mathcal{X}} \lambda \sigma(g(\mathbf{x}))\pi(\mathbf{x})d\mathbf{x} &= \\ \int_{\mathcal{X}} (\sigma(-g(\mathbf{x})) - 1)\lambda\pi(\mathbf{x})d\mathbf{x} &= \\ \int_{\mathcal{X}} \int_{\mathbb{R}^+} \left(e^{f(\omega, -g(\mathbf{x}))} - 1\right) \lambda\pi(\mathbf{x})p_{\text{PG}}(\omega|1, 0)d\omega d\mathbf{x} \end{aligned}$$

Next we will use a result for the characteristic function of a Poisson process. Following [11, chap. 3] one has

$$\mathbb{E}_\phi \left[ \prod_{z \in \Pi} h(z) \right] = \exp \left( \int_{\mathcal{Z}} (h(z) - 1) \phi(z) dz \right). \quad (6)$$

$h(\cdot)$  is a function on a space  $\mathcal{Z}$  and the expectation is over a Poisson process  $\Pi$  with rate function  $\phi(z)$ .  $\Pi = \{z_m\}_{m=1}^M$  denotes a random set of points on the space  $\mathcal{Z}$ . To apply this result to our problem, we identify  $\mathcal{Z} = \mathcal{X} \times \mathbb{R}^+$ ,  $z = (\mathbf{x}, \omega)$  and  $\phi_\lambda(\mathbf{x}, \omega) = \lambda \pi(\mathbf{x}) p_{\text{PG}}(\omega|1, 0)$  and finally  $h(z) = e^{f(\omega, -g(\mathbf{x}))}$  to rewrite the exponential in Eq. (3) as

$$e^{-\int_{\mathcal{X}} \lambda \sigma(g(\mathbf{x})) \pi(\mathbf{x}) d\mathbf{x}} = \mathbb{E}_{\phi_\lambda} \left[ \prod_{(\omega, \mathbf{x}) \in \Pi} e^{f(\omega, -g(\mathbf{x}))} \right]. \quad (7)$$

By substituting Eq. (5) and (7) into Eq.(3) we obtain the final augmented form of the likelihood of Eq. (2) which is one of the main results of our paper.

$$p(\mathcal{D}, \lambda, \Pi, \boldsymbol{\omega}_N | g) \propto \prod_{n=1}^N \phi_\lambda(\mathbf{x}_n, \omega_n) e^{f(\omega_n, g(\mathbf{x}_n))} \times p_{\phi_\lambda}(\Pi | \lambda) p(\lambda) \prod_{(\omega, \mathbf{x}) \in \Pi} e^{f(\omega, -g(\mathbf{x}))}, \quad (8)$$

with  $p_\phi(\Pi | \lambda)$  being the density over a Poisson process  $\Pi = \{(\mathbf{x}_m, \omega_m)\}_{m=1}^M$  in the augmented space  $\mathcal{X} \times \mathbb{R}^+$  with intensity  $\phi_\lambda(\mathbf{x}, \omega)$ .<sup>2</sup> This new process can be identified as a *marked Poisson process* [11, chap. 5], where the events  $\{\mathbf{x}_m\}_{m=1}^M$  in the original data space  $\mathcal{X}$  follow a Poisson process with rate  $\lambda \pi(\mathbf{x})$ . Then, on each event  $\mathbf{x}_m$  an independent *mark*  $\omega_m \sim p_{\text{PG}}(\omega_m | b, 0)$  is drawn at random from the Pólya–Gamma density. Finally,  $\boldsymbol{\omega}_N = \{\omega_n\}_{n=1}^N$  is the set of latent Pólya–Gamma variables which result from the sigmoid augmentation at the observations  $\mathbf{x}_n$ .

**Augmented posterior over GP density** With Eq. (8) we obtain the joint posterior over the GP  $g$ , the rate scaling  $\lambda$ , the marked Poisson process  $\Pi$ , and the Pólya–Gamma variables at the observations  $\boldsymbol{\omega}_N$  as

$$p(\boldsymbol{\omega}_N, \Pi, \lambda, g | \mathcal{D}) \propto p(\mathcal{D}, \boldsymbol{\omega}_N, \Pi, \lambda | g) p(g). \quad (9)$$

In the following, this new representation will be used to derive two inference algorithms.

<sup>2</sup>Densities such as  $p_{\phi_\lambda}(\Pi | \lambda)$  could be understood as the Radon–Nykodym derivative [12] of the corresponding probability measure with respect to some fixed dominating measure. However, we will not need an explicit form here.

### 3 INFERENCE

We will first derive an efficient Gibbs sampler which (asymptotically) solves the inference problem exactly, and then a variational mean-field algorithm, which only finds an approximate solution, but in a much faster time.

#### 3.1 GIBBS SAMPLER

Gibbs sampling [13] generates samples from the posterior by creating a Markov chain, where at each time, a block of variables is drawn from the conditional posterior given all the other variables. Hence, to perform Gibbs sampling, we have to derive these conditional distributions for each set of variables from Eq. (9). Most of the following results are easily obtained by direct inspection. The only non-trivial case is the conditional distribution over the latent point process  $\Pi$ .

**Pólya-Gamma variables at observations** The conditional posterior over the set of Pólya–Gamma variables  $\boldsymbol{\omega}_N$  depends only on the function  $g$  at the observations  $\{g(\mathbf{x}_n)\}_{n=1}^N$  and turns out to be

$$p(\boldsymbol{\omega}_N | g) = \prod_{n=1}^N p_{\text{PG}}(\omega_n | 1, g(\mathbf{x}_n)), \quad (10)$$

where we have used the definition of a tilted Pólya–Gamma density in Eq. (4). This density can be efficiently sampled by methods developed by [9]<sup>3</sup>.

**Rate scaling** The rate scaling  $\lambda$  has a conditional Gamma density given by

$$\text{Gamma}(\lambda | \alpha, 1) = \frac{(\lambda)^{\alpha-1} e^{-\lambda}}{\Gamma(\alpha)}. \quad (11)$$

with  $\alpha = |\Pi| + N = M + N$ . Hence, the posterior is dependent on the number of observations and the number on events of the marked Poisson process  $\Pi$ .

**Posterior Gaussian process** Due to the form of the augmented likelihood the conditional posterior for the GP  $g_{N+M}$  at the observations  $\{\mathbf{x}_n\}_{n=1}^N$  and the latent events  $\{\mathbf{x}_m\}_{m=1}^M$  is a multivariate Gaussian density

$$p(g_{N+M} | \Pi, \boldsymbol{\omega}_N) = \mathcal{N}(\boldsymbol{\mu}_{N+M}, \Sigma_{N+M}), \quad (12)$$

with covariance matrix  $\Sigma_{N+M} = [D + K_{N+M}^{-1}]^{-1}$ . The diagonal matrix  $D$  has its first  $N$  entries given by  $\boldsymbol{\omega}_N$  followed by  $M$  entries being  $\{\omega_m\}_{m=1}^M$ . The mean is  $\boldsymbol{\mu}_{N+M} = \Sigma_{N+M} \left[ \mathbf{u} + K_{N+M}^{-1} \boldsymbol{\mu}_0^{(N+M)} \right]$ , where the

<sup>3</sup>The sampler implemented by [14] is used for this work.

first  $N$  entries of  $N + M$  dimensional vector  $\mathbf{u}$  are  $1/2$  and the rest are  $-1/2$ .  $K_{N+M}$  is the prior covariance kernel matrix of the GP evaluated at the observed points  $\mathbf{x}_n$  and the latent events  $\mathbf{x}_m$ , and  $\boldsymbol{\mu}_0^{(N+M)}$  is an  $N + M$  dimensional vector with all entries being  $\mu_0$ .

The predictive conditional posterior for the GP for any set of points in  $\mathcal{X}$  is simply given via the conditional prior  $p(g|\mathbf{g}_{N+M})$ , which has a well known form and can be found in [1].

**Sampling the latent marked point process** We easily find that the conditional posterior of the marked point process is given by

$$p(\Pi|g, \lambda) = \frac{\prod_{\omega, \mathbf{x} \in \Pi} e^{f(\omega, -g(\mathbf{x}))} p_{\phi_\lambda}(\Pi|\lambda)}{\exp\left(\int_{\mathcal{X} \times \mathbb{R}^+} \left(e^{f(\omega, -g(\mathbf{x}))} - 1\right) \phi_\lambda(\mathbf{x}, \omega) d\omega d\mathbf{x}\right)}, \quad (13)$$

where the form of the normalising denominator is obtained using Eq. (6). By computing the characteristic function of this conditional point process (see App. A) we can show that it is again a marked Poisson process with intensity

$$\Lambda(\mathbf{x}, \omega) = \lambda \pi(\mathbf{x}) \sigma(-g(\mathbf{x})) p_{\text{PG}}(\omega|1, g(\mathbf{x})). \quad (14)$$

To sample from this process we first draw Poisson events  $\mathbf{x}_m$  in the original data space  $\mathcal{X}$  using the rate  $\int_{\mathbb{R}^+} \Lambda(\mathbf{x}, \omega) d\omega = \lambda \pi(\mathbf{x}) \sigma(-g(\mathbf{x}))$  [11, chap. 5]. Subsequently for each event  $\mathbf{x}_m$  a mark  $\omega_m$  is generated from the conditional density  $\omega_m \sim p_{\text{PG}}(\omega|1, g(\mathbf{x}_m))$ .

To sample the events  $\{\mathbf{x}_m\}_{m=1}^M$ , we use the well known approach of *thinning* [4]. We note, that the rate is upper bounded by the base measure  $\lambda \pi(\mathbf{x})$ . Hence, we first generate points  $\tilde{\mathbf{x}}_m$  from a Poisson process with intensity  $\lambda \pi(\mathbf{x})$ . This is easily achieved by noting that the required number  $M_{\text{max}}$  of such events is Poisson distributed with mean parameter  $\int_{\mathcal{X}} \lambda \pi(\mathbf{x}) d\mathbf{x} = \lambda$ . The position of the events can then be obtained by sampling  $\{\tilde{\mathbf{x}}_m\}_{m=1}^{M_{\text{max}}}$  independent points from the base density  $\tilde{\mathbf{x}}_m \sim \pi(\mathbf{x})$ . These events are *thinned* by keeping each point  $\tilde{\mathbf{x}}_m$  with probability  $\sigma(-g(\tilde{\mathbf{x}}_m))$ . The kept events constitute the final set  $\{\mathbf{x}_m\}_{m=1}^M$ .

**Sampling hyperparameters** In this work we will consider specific functional forms for the kernel  $k(\mathbf{x}, \mathbf{x}')$  and the base measure  $\pi(\mathbf{x})$  which are parametrised by hyperparameters  $\boldsymbol{\theta}_k$  and  $\boldsymbol{\theta}_\pi$ . These will be sampled by a Metropolis-Hastings method [15]. The GP prior mean  $\mu_0$  can be directly sampled from the conditional posterior given  $\mathbf{g}_{M+N}$ . In this work, the hyperparameters are sampled every  $v = 10$  step. Different choices of  $v$  might yield faster convergence of the Markov Chain. Pseudo code for the Gibbs sampler is provided in Alg. 1.

---

**Algorithm 1:** Gibbs sampler for GP density model.

---

**Init:**  $\{\mathbf{x}_m\}_{m=1}^M$ ,  $\mathbf{g}_{N+M}$ ,  $\lambda$ , and  $\boldsymbol{\theta}_k, \boldsymbol{\theta}_\pi, \mu_0$   
**1 for** *Length of Markov chain* **do**  
**2**     **Sample PG variables at**  $\{\mathbf{x}_m\}$ :  $\omega_N \sim \text{Eq. (10)}$   
**3**     **Sample latent Poisson process:**  $\Pi \sim \text{Eq. (13)}$   
**4**     **Sample rate scaling:**  $\lambda \sim \text{Eq. (11)}$   
**5**     **Sample GP:**  $\mathbf{g}_{N+M} \sim \text{Eq. (12)}$   
**6**     **Sample hyperparameters:** Every  $v^{\text{th}}$  sample with Metropolis–Hastings  
**7 end**

---

### 3.2 VARIATIONAL BAYES

While expected to be more efficient than a Metropolis-Hastings sampler based on the unaugmented likelihood [7], the Gibbs sampler is practically still limited. The main computational bottleneck comes from the sampling of the conditional Gaussian over function values of  $g$ . The computation of the covariances requires the inversion of matrices of dimensions  $N + M$ , with a complexity  $\mathcal{O}((N + M)^3)$ . Hence the algorithm does not only become infeasible, when we have many observations, i.e when  $N$  is large, but also if the sampler requires many thinned events, i.e. if  $M$  is large. This can happen in particular for bad choices of the base measure  $\pi(\mathbf{x})$ . In the following, we introduce a variational Bayes algorithm [16], which solves the inference problem approximately, but with a complexity which scales linearly in the data size and is independent of structure.

**Structured mean–field approach** The idea of variational inference [16] is to approximate an intractable posterior  $p(Z|\mathcal{D})$  by a simpler distribution  $q(Z)$  from a tractable family.  $q(Z)$  is optimised by minimising the Kullback-Leibler divergence between  $q(Z)$  and  $p(Z|\mathcal{D})$  which is equivalent to maximising the so called *variational lower bound* (sometimes also called ELBO for evidence lower bound) given by

$$\mathcal{L}(q(Z)) = \mathbb{E}_Q \left[ \ln \frac{p(Z, \mathcal{D})}{q(Z)} \right] \leq \ln p(\mathcal{D}), \quad (15)$$

where  $Q$  denotes the probability measure with density  $q(Z)$ . A common approach for variational inference is a structured mean–field method, where dependencies between sets of variables are neglected. For the problem at hand we assume that

$$q(\boldsymbol{\omega}_N, \Pi, g, \lambda) = q_1(\boldsymbol{\omega}_N, \Pi) q_2(g, \lambda). \quad (16)$$

A standard result for the variational mean–field approach shows that the optimal independent factors, which max-

imise the lower bound in Eq. (15) are given by

$$\ln q_1(\boldsymbol{\omega}_N, \Pi) = \mathbb{E}_{Q_2} [\ln p(\mathcal{D}, \boldsymbol{\omega}_N, \Pi, \lambda, g)] + \text{const.}, \quad (17)$$

$$\ln q_2(g, \lambda) = \mathbb{E}_{Q_1} [\ln p(\mathcal{D}, \boldsymbol{\omega}_N, \Pi, \lambda, g)] + \text{const.} \quad (18)$$

By inspecting Eq. (9), (17), and (18) it turns out that the densities of all four sets of variables factorise as

$$\begin{aligned} q_1(\boldsymbol{\omega}_N, \Pi) &= q_1(\boldsymbol{\omega}_N)q_1(\Pi), \\ q_2(g, \lambda) &= q_2(g)q_2(\lambda). \end{aligned}$$

We will optimise the factors by a straightforward iterative algorithm, where each factor is updated given expectations over the others based on the previous step. Hence, the lower bound in Eq. (15) is increased in each step. Again we will see that the augmented likelihood in Eq. (8) allows for analytic solutions of all required factors.

**Pólya–Gamma variables at the observations** Similar to the Gibbs sampler, the variational posterior of the Pólya–Gamma variables at the observations is a product of tilted Pólya–Gamma densities given by

$$q_1(\boldsymbol{\omega}_N) = \prod_{n=1}^N p_{\text{PG}}(\omega_n | 1, c_n), \quad (19)$$

with  $c_n = \sqrt{\mathbb{E}_{Q_2} [g(\mathbf{x}_n)^2]}$ . The only difference is, that the second argument of  $p_{\text{PG}}$  depends on the expectation of the square of  $g(\mathbf{x}_n)$ .

**Posterior marked Poisson process** Similar to the corresponding result for the Gibbs sampler we can show<sup>4</sup> that the optimal latent point process  $\Pi$  is a Poisson process with rate given by

$$\begin{aligned} \Lambda_1(\mathbf{x}, \omega) &= \lambda_1 \pi(\mathbf{x}) \sigma(-c(\mathbf{x})) p_{\text{PG}}(\omega | 1, c(\mathbf{x})) \\ &\times e^{(c(\mathbf{x}) - g_1(\mathbf{x}))/2} \end{aligned} \quad (20)$$

with  $\lambda_1 = e^{\mathbb{E}_{Q_2} [\ln \lambda]}$ ,  $c(\mathbf{x}) = \sqrt{\mathbb{E}_{Q_2} [f(\mathbf{x})^2]}$ , and  $g_1(\mathbf{x}) = \mathbb{E}_{Q_2} [g(\mathbf{x})]$ . Note also the similarity to the Gibbs sampler in Eq. (14).

**Optimal posterior for rate scaling** The posterior for the rate scaling  $\lambda$  is a Gamma distribution given by

$$q_2(\lambda) = \text{Gamma}(\lambda | \alpha_2, 1) = \frac{\lambda^{\alpha_2 - 1} e^{-\lambda}}{\Gamma(\alpha_2)}, \quad (21)$$

where  $\alpha_2 = N + \mathbb{E}_{Q_1} [\sum_{\mathbf{x}' \in \Pi} \delta(\mathbf{x} - \mathbf{x}')]$ , and  $\mathbb{E}_{Q_1} [\sum_{\mathbf{x}' \in \Pi} \delta(\mathbf{x} - \mathbf{x}')] = \int_{\mathcal{X}} \int_{\mathbb{R}^+} \Lambda_1(\mathbf{x}, \omega) d\omega d\mathbf{x}$ , and  $\delta(\cdot)$  is the Dirac delta function. The integral is solved by importance sampling as will be explained (see Eq. (25)).

<sup>4</sup>The proof is similar to the one from App. A.

**Approximation of GP via sparse GP** The optimal variational form for the posterior  $g$  is a GP given by

$$q_2(g) \propto e^{U(g)} p(g),$$

where  $U(g) = \mathbb{E}_{Q_1} [\ln p(\mathcal{D}, \boldsymbol{\omega}_N, \Pi, \lambda | g)]$  results in the Gaussian log-likelihood

$$U(g) = -\frac{1}{2} \int_{\mathcal{X}} A(\mathbf{x}) g(\mathbf{x})^2 d\mathbf{x} + \int_{\mathcal{X}} B(\mathbf{x}) g(\mathbf{x}) d\mathbf{x} + \text{const.}$$

with

$$\begin{aligned} A(\mathbf{x}) &= \sum_{n=1}^N \mathbb{E}_{Q_1} [\omega_n] \delta(\mathbf{x} - \mathbf{x}_n) + \int_{\mathbb{R}^+} \omega \Lambda_1(\mathbf{x}, \omega) d\omega, \\ B(\mathbf{x}) &= \frac{1}{2} \sum_{n=1}^N \delta(\mathbf{x} - \mathbf{x}_n) - \frac{1}{2} \int_{\mathbb{R}^+} \Lambda_1(\mathbf{x}, \omega) d\omega. \end{aligned}$$

For general GP priors, this free form optimum is intractable by the fact that the likelihood depends on  $g$  at infinitely many points. Hence, we resort to an additional approximation which makes the dimensionality of the problem again finite. The well known framework of *sparse* GPs [17, 18, 19] turns out to be useful in this case. This has been introduced for likelihoods with large, but finite dimensional likelihoods [19, 20] and later generalised to infinite dimensional problems [21, 22]. The sparse approximation assumes a variational posterior of the form

$$q_2(g) = p(g | \mathbf{g}_s) q_2(\mathbf{g}_s),$$

where  $\mathbf{g}_s$  is the GP evaluated at a finite set of *inducing points*  $\{\mathbf{x}_l\}_{l=1}^L$  and  $p(g | \mathbf{g}_s)$  is the conditional prior. A variational optimisation yields

$$q_2(\mathbf{g}_s) \propto e^{U^s(\mathbf{g}_s)} p(\mathbf{g}_s), \quad (22)$$

where the first term can be seen as a new ‘effective’ likelihood only depending on the inducing points. This new (log) likelihood is given by

$$\begin{aligned} U^s(\mathbf{g}_s) &= \mathbb{E}_P [U(g) | \mathbf{g}_s] = \\ &= -\frac{1}{2} \int_{\mathcal{X}} A(\mathbf{x}) \tilde{g}_s(\mathbf{x})^2 d\mathbf{x} + \int_{\mathcal{X}} B(\mathbf{x}) \tilde{g}_s(\mathbf{x}) d\mathbf{x} + \text{const.}, \end{aligned}$$

with  $\tilde{g}_s(\mathbf{x}) = \mu_0 + \mathbf{k}_s(\mathbf{x})^\top K_s^{-1}(\mathbf{g}_s - \boldsymbol{\mu}_0^{(L)})$ ,  $\mathbf{k}_s(\mathbf{x})$  being an  $L$  dimensional vector, where the  $l^{\text{th}}$  entry is  $k(\mathbf{x}, \mathbf{x}_l)$  and  $K_s$  being the prior covariance matrix for all inducing points. The expectation is computed with respect to the GP prior conditioned on the sparse GP  $\mathbf{g}_s$ . We identify Eq. (22) being a multivariate normal distribution with covariance matrix

$$\Sigma_2^s = \left[ K_s^{-1} \int_{\mathcal{X}} A(\mathbf{x}) \mathbf{k}_s(\mathbf{x})^\top \mathbf{k}_s(\mathbf{x}) d\mathbf{x} K_s^{-1} + K_s^{-1} \right]^{-1}, \quad (23)$$

---

**Algorithm 2:** Variational Bayes algorithm for GP density model

---

**Init:** Inducing points,  $q_2(\mathbf{g}_s)$ ,  $q_2(\lambda)$ , and  $\boldsymbol{\theta}_k, \boldsymbol{\theta}_\pi, \mu_0$

```

1 while  $\mathcal{L}$  not converged do
2   Update  $q_1$ 
3   PG distributions at observations:  $q_1^*(\omega_N)$ 
   with Eq. (19)
4   Rate of latent process:  $\Lambda_1(\mathbf{x}, \omega)$  with Eq. (20)
5   Update  $q_2$ 
6   Rate scaling:  $\alpha_2$  with Eq. (21)
7   Sparse GP:  $\Sigma_2^s, \mu_2^s$  with Eq. (23), (24)
8   Update  $\boldsymbol{\theta}_k, \boldsymbol{\theta}_\pi, \mu_0$  with gradient update
9 end

```

---

and mean

$$\boldsymbol{\mu}_2^s = \Sigma_2^s \left( K_s^{-1} \int_{\mathcal{X}} \mathbf{k}_s(\mathbf{x}) \tilde{B}(\mathbf{x}) d\mathbf{x} + K_s^{-1} \boldsymbol{\mu}_0^{(L)} \right), \quad (24)$$

with  $\tilde{B}(\mathbf{x}) = B(\mathbf{x}) - A(\mathbf{x})(\mu_0 - \mathbf{k}_s(\mathbf{x})^\top K_s^{-1} \boldsymbol{\mu}_0^{(L)})$ .

**Integrals over  $\mathbf{x}$**  The sparse GP approximation and the posterior over  $\lambda$  in Eq. (21) requires the computation of integrals of the form

$$I \doteq \int_{\mathcal{X}} \int_{\mathbb{R}^+} y(\mathbf{x}, \omega) \Lambda_1(\mathbf{x}, \omega) d\omega d\mathbf{x},$$

with specific functions  $y(\mathbf{x}, \omega)$ . For these functions, the inner integral over  $\omega$  can be computed analytically, but the outer one over the space  $\mathcal{X}$  has to be treated numerically. We approximate it via importance sampling

$$I \approx \frac{1}{R} \sum_{r=1}^R \int_{\mathbb{R}^+} y(\mathbf{x}_r, \omega_r) \frac{\Lambda_1(\mathbf{x}_r, \omega_r)}{\pi(\mathbf{x}_r)} d\omega_r, \quad (25)$$

where every sample point  $\mathbf{x}_r$  is independently drawn from the base measure  $\pi(\mathbf{x})$ .

**Updating hyperparameters** Having an analytic solution for every factor of the variational posterior in Eq. (16) we further require the optimisation of hyperparameters.  $\boldsymbol{\theta}_k, \boldsymbol{\theta}_\pi$  and  $\mu_0$  are optimised by maximising the lower bound in Eq. (15) (see App. B for explicit form) with a gradient ascent algorithm having an adaptive learning rate (Adam) [23]. Additional hyperparameters are the locations of inducing points  $\{\mathbf{x}_l\}_{l=1}^L$ . Half of them are drawn randomly from the initial base measure, while half of them are positioned on regions with a high density of observations found by a k-means algorithm. Pseudo code for the complete variational algorithm is provided in Alg. 2.

Python code for Alg. 1 and 2 is provided at [24].

## 4 RESULTS

To test our two inference algorithms, the Gibbs sampler and the variational Bayes algorithm (VB), we will first evaluate them on data drawn from the generative model. Then we compare both on an artificial dataset and several real datasets. We will only consider cases with  $\mathcal{X} = \mathbb{R}^d$ . To evaluate the quality of inference we consider always the logarithm of the expected test likelihood

$$\ell_{\text{test}}(\tilde{\mathcal{D}}) \doteq \ln \left( \mathbb{E} \left[ \prod_{\mathbf{x} \in \tilde{\mathcal{D}}} \rho(\mathbf{x}) \right] \right),$$

where  $\tilde{\mathcal{D}}$  is test data unknown to the inference algorithm and the expectation is over the inferred posterior measure. In practice we sample this expectation from the inferred posterior over  $g$ . Since this quantity involves an integral, that is again approximated by Eq. (25), we check that the standard deviation  $\text{std}(I)$  is less than 1% of the value of the estimated value  $I$ .

**Data from generative model.** We generate datasets according to Eq. (1), where  $g$  is drawn from the GP prior with  $\mu_0 = 0$ . As covariance kernel we assume a squared exponential throughout this work

$$k(\mathbf{x}, \mathbf{x}') = \theta_k^{(0)} \prod_{i=1}^d \exp \left( -\frac{(x_i - x'_i)^2}{2(\theta_k^{(i)})^2} \right).$$

The base measure  $\pi(\mathbf{x})$  is a standard normal density. We use the algorithm described in [7] to generate exact samples. In this section, the hyperparameters  $\boldsymbol{\theta}_k, \boldsymbol{\theta}_\pi$  and  $\mu_0$  are fixed to the true values for inference. Unless stated otherwise for the VB the number of inducing points is fixed to 200 and the number of integration points for importance sampling to  $5 \times 10^3$ . For the Gibbs sampler, we sample a Markov chain of  $5 \times 10^3$  samples after a burn-in period of  $2 \times 10^3$  samples.

In Fig. 1 we see a 1 dimensional example dataset, where both inference algorithms recover well the structure of the underlying density. The inferred posterior means are barely distinguishable. However, evaluating the inferred densities on an unseen test set, we note that the Gibbs sampler performs slightly better. Of course, this is expected since the sampler provides exact inference for the generative model and should (on average) not be outperformed by the approximate VB as long as the sampled Markov chain is long enough. In Fig. 1 (bottom left) we see that only 13 iterations of the VB are required to meet the convergence criterion. For Markov chain samplers to be efficient, correlations between samples should decay quickly. Fig. 1 (bottom middle) shows the autocorrelation of  $\ell_{\text{test}}$ , which was evaluated at each sample of the

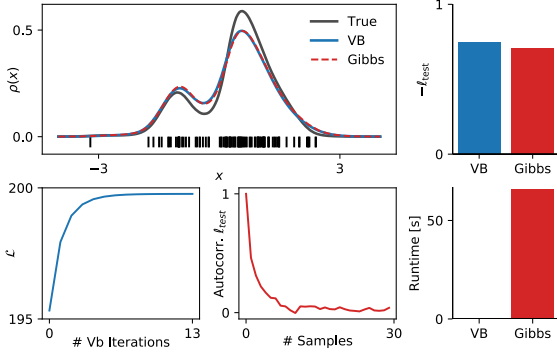


Figure 1: **1D data from the generative model.** Data consist of 100 samples from the underlying density sampled from the GP density model. **Upper left:** True density (black line), data (black vertical bars), mean posterior density inferred by Gibbs sampler (red dashed line) and VB algorithm (blue line). **Upper right:** Negative log expected test likelihood of Gibbs and VB inferred posterior. **Lower left:** Variational lower bound as function of iterations of the VB algorithm. **Lower middle:** Autocorrelation of test likelihood as function of Markov chain samples obtained from Gibbs sampler. **Lower right:** Runtime of the two algorithms (VB took 0.3 s).

Dim	# points	Gibbs		VB	
		$\ell_{\text{test}}$	$T$ [s]	$\ell_{\text{test}}$	$T$ [s]
1	50	-146.9	30.1	-149.2	1.13
2	100	-257.0	649.9	-260.2	2.03
2	200	-285.3	546.1	-289.6	1.41
6	400	-823.9	4667	-822.2	0.89

Table 1: **Performance of Gibbs sampler and VB** on different datasets sampled from generative model.  $\ell_{\text{test}}$  was evaluated on a unknown test set including 50 samples. In addition, runtime  $T$  is reported in seconds.

Markov chain. After about 10 samples the correlations reach a plateau close to 0, demonstrating excellent mixing properties of the sampler. Comparing the run time of both algorithms, VB (0.3 s) outperforms the sampler  $\sim 1$  min by more than 2 orders of magnitude.

To demonstrate the inference for more complicated problems, 2 dimensional data are generated with 200 samples (Fig. 2). The posterior mean densities inferred by both algorithms capture the structure well. As before, the log expected test likelihood is larger for the Gibbs sampler ( $\ell_{\text{test}} = -296.2$ ) compared to VB ( $\ell_{\text{test}} = -306.0$ ). However, the Gibbs sampler took  $> 20$  min while the VB required only 1.8 s to obtain the result.

In Tab. 1 we show results for datasets with different size and different dimensionality. The results confirm that the

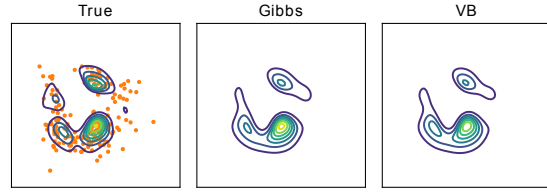


Figure 2: **2D data from generative model.** **Right:** 200 samples from the underlying two dimensional density. **Middle:** Posterior mean of Gibbs sampler inferred density. **Right:** Posterior mean of VB inferred density.

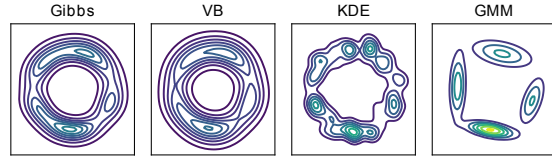


Figure 3: **Comparison to other density estimation methods on artificial 2D data.** Training data consist of 100 data points uniformly distributed on a circle (1.5 radius) and additional Gaussian noise (0.2 std.). **From left to right:** The posterior mean inferred by Gibbs sampler and VB algorithm, followed by density estimation using KDE and GMM.

run time for the Gibbs sampler scales strongly with size and dimensionality of a problem, while the VB algorithm seems relatively unaffected in this regard. However, the VB is in general outperformed by the sampler in terms of expected test likelihood or in the same range. Note, that the runtime of the Gibbs sampler does not solely depend on the number of observed data points  $N$  (compare data set 2 and 3 in Tab. 1). As discussed earlier this can happen, when the base measure  $\pi(\mathbf{x})$  is very different from the target density  $\rho(\mathbf{x})$  resulting in many latent Poisson events (i.e.  $M$  is large).

**Circle data** In the following, we compare the GP density model and its two inference algorithms with two alternative density estimation methods. These are given by a kernel density estimator (KDE) with a Gaussian kernel and a Gaussian mixture model (GMM) [25]. The free parameters of these models (kernel bandwidth for KDE and number of components for GMM) are optimised by 10-fold cross-validation. Furthermore, GMM is initialised 10 times and the best result is reported. For the GP density model a Gaussian density is assumed as base measure  $\pi(\mathbf{x})$ , and hyperparameters  $\theta_{\pi}$ ,  $\theta_k$ , and  $\mu_0$  are now optimised. Similar to [7] we consider 100 samples uniformly drawn from a circle with additional Gaussian

	Gibbs	VB	KDE	GMM
$\ell_{\text{test}}$	-220.31	-230.53	-228.43	-237.34

Table 2: Log expected test likelihood for circle data.

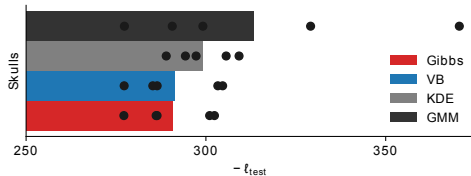


Figure 4: **Performance on ‘Egyptian Skulls’ dataset [26].** 100 training points and 4 dimensions. Bar height shows average negative log test likelihood obtained by five random permutations of training and test set and points mark single permutation results.

noise. The inferred densities (only the mean of the posterior for Gibbs and VB) are shown in Fig. 3. Both GP density methods recover well the structure of the data, but the VB seems to overestimate the width of the Gaussian noise compared to the Gibbs sampler. While the KDE also recovers relatively well the data structure the GMM fails in this case. This is also reflected on the log expected test likelihoods (Tab. 2).

**Real data sets** The ‘Egyptian Skulls’ dataset [26] contains 150 data points in 4 dimensions. 100 training points are randomly selected and performance is evaluated on the remaining ones. Before fitting data is whitened. Base measure and fitting procedure for all algorithms are the same as for the circular data. Furthermore, fitting is done for 5 random permutations of training and test set. The results in Fig. 4 show that both algorithms for the GP density model outperform the two other ones on this dataset.

Often practical problems may consist of many more data points and dimensions. As discussed, the Gibbs sampler is not practical for such kind of problems, while the VB could handle larger amounts of data. Unfortunately, the sparsity assumption and the integration via importance sampling is expected to become poorer with increasing number of dimensions. Noting, however, that the ‘effective’ dimensionality in our model is determined by the base measure  $\pi(\mathbf{x})$ , one can circumvent this problem by an educated choice of  $\pi(\mathbf{x})$  if data  $\mathcal{D}$  lie in a submanifold of the high dimensional space  $\mathcal{X}$ .

We employ this strategy by first fitting a GMM to the problem and then utilising the fit as base measure. In Fig. 5 we consider 3 different datasets<sup>5</sup> to test this pro-

<sup>5</sup>Only real valued dimensions are considered and for the

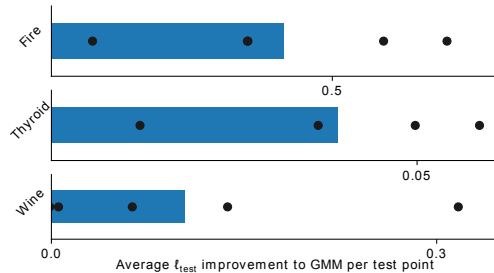


Figure 5: **Application on higher dimensional data with many data points.** The improvement on log expected test likelihood  $\ell_{\text{test}}$  per test point compared to GMM, when using same as base measure  $\pi(\mathbf{x})$  for the VB inference. **From top to bottom:** ‘Forest Fire’ dataset [27, 28] (400 training points, 117 test points, 5 dim.), ‘Thyroid’ dataset [29] ( $3 \times 10^3$ , 772, 6), ‘Wine’ dataset [27] ( $6 \times 10^3$ , 498, 9). Bars mark improvement on average of random permutations of training and test set while points mark single runs.

cedure. As in Fig. 4, fitting is repeated 5 times for random permutations if training and test set. For the ‘Thyroid’ dataset, one of the 5 fits is excluded, because the importance sampling yielded poor approximation  $\text{std}(I) > I \times 10^{-2}$ . The training sets contain 400 to 6000 data points with 5 to 9 dimensions. The results for KDE are not reported, since it is always outperformed by the GMM. Fig. 5 demonstrates combining the GMM and VB algorithm results in an improvement of the log test likelihood  $\ell_{\text{test}}$  compared to using only GMM. Average relative improvements of  $\ell_{\text{test}}$  are 8.9 % for ‘Forest Fire’, 4.1 % for ‘Thyroid’, and 1.1 % for ‘Wine’ dataset.

## 5 DISCUSSION

We have shown how inference for a nonparametric, GP based, density model can be made efficient. In the following we would like to discuss various possible extensions but also limitations of our approach.

**Choice of base measure** As we have shown for applications to real data, the choice of the base measure is quite important, especially for the sampler and for high dimensional problems. While many datasets might favour a normal distribution as base measure, problems with outliers might favour fat tailed densities. In general, any density which can be evaluated on the data space  $\mathcal{X}$  and which allows for efficient sampling, is a valid choice as base measure  $\pi(\mathbf{x})$  in our inference approach for the GP density model. Any powerful density estima-

‘forest fire’ dataset dimensions are excluded, where data have more than half 0 entries.



tor which fulfils this condition could provide a base measure which could then potentially be improved by the GP model. It would e.g. be interesting to apply this idea to neural networks [30, 31] based estimators. Other generalisations of our model could consider alternative data spaces  $\mathcal{X}$ . One might e.g. think of specific discrete and structured sets  $\mathcal{X}$  for which appropriate Gaussian processes could be defined by suitable Mercer kernels.

**Big data & high dimensionality** Our proposed Gibbs sampler suffers from cubic scaling in the number of data points and is found to be already impractical for problems with hundreds of observations. This could potentially be tackled by using sparse (approximate) GP methods for the sampler (see [32] for a potential approach). On the other hand, the proposed VB algorithm scales only linearly with the training set size and can be applied to problems with several thousands of observations. The integration of stochastic variational inference into our method could potentially increase this limit [33].

Potential limitations of the GP density model are given by high dimensional problems. If approached naively, the combination of the sparse GP approximation and the numerical integration using importance sampling is expected to yield bad approximations in such cases.<sup>6</sup> If the data is concentrated on a low dimensional submanifold of the high-dimensional space, one could still try to combine our method with other density estimators providing a base measure  $\pi(\mathbf{x})$  that is adapted to this submanifold, to allow for tractable GP inference.

## Acknowledgements

CD was supported by the Deutsche Forschungsgemeinschaft (GRK1589/2) and partially funded by Deutsche Forschungsgemeinschaft (DFG) through grant CRC 1294 “Data Assimilation”, Project (A06) “Approximative Bayesian inference and model selection for stochastic differential equations (SDEs)”.

## References

- [1] Carl Edward Rasmussen and Christopher KI Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.
- [2] Christopher KI Williams and Carl Edward Rasmussen. Gaussian processes for regression. In *Advances in neural information processing systems*, pages 514–520, 1996.
- [3] Hannes Nickisch and Carl Edward Rasmussen. Approximations for binary gaussian process classification. *Journal of Machine Learning Research*, 9(Oct):2035–2078, 2008.
- [4] Ryan Prescott Adams, Iain Murray, and David JC MacKay. Tractable nonparametric bayesian inference in poisson processes with gaussian process intensities. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 9–16. ACM, 2009.
- [5] Cedric Archambeau, Dan Cornford, Manfred Opper, and John Shawe-Taylor. Gaussian process approximations of stochastic differential equations. In *Gaussian Processes in Practice*, pages 1–16, 2007.
- [6] Andreas Damianou, Michalis K Titsias, and Neil D Lawrence. Variational gaussian process dynamical systems. In *Advances in Neural Information Processing Systems*, pages 2510–2518, 2011.
- [7] Iain Murray, David MacKay, and Ryan P Adams. The gaussian process density sampler. In *Advances in Neural Information Processing Systems*, pages 9–16, 2009.
- [8] Jaakko Riihimäki, Aki Vehtari, et al. Laplace approximation for logistic gaussian process density estimation and regression. *Bayesian analysis*, 9(2):425–448, 2014.
- [9] Nicholas G Polson, James G Scott, and Jesse Windle. Bayesian inference for logistic models using pólya-gamma latent variables. *Journal of the American statistical Association*, 108(504):1339–1349, 2013.
- [10] Stephen G Walker. Posterior sampling when the normalizing constant is unknown. *Communications in Statistics—Simulation and Computation*, 40(5):784–792, 2011.
- [11] John Frank Charles Kingman. *Poisson processes*. Wiley Online Library, 1993.
- [12] Takis Konstantopoulos, Zurab Zerakidze, and Grigol Sokhadze. *Radon–Nikodým Theorem*, pages 1161–1164. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.
- [13] Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. In *Readings in Computer Vision*, pages 564–584. Elsevier, 1987.
- [14] Scott Linderman. pypolyagamma. <https://github.com/slinderman/pypolyagamma>, 2017.

<sup>6</sup>Potentially in such cases other sparsity methods [34] might be more favourable.

- [15] W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.
- [16] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.
- [17] Lehel Csató. Gaussian Processes -Iterative Sparse Approximations. *PhD Thesis*, 2002.
- [18] Lehel Csató, Manfred Opper, and Ole Winther. Tap gibbs free energy, belief propagation and sparsity. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 657–663. MIT Press, 2002.
- [19] Michalis K Titsias. Variational learning of inducing variables in sparse gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, pages 567–574, 2009.
- [20] Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, pages 1257–1264, 2006.
- [21] Alexander G de G Matthews, James Hensman, Richard Turner, and Zoubin Ghahramani. On sparse variational methods and the kullback-leibler divergence between stochastic processes. In *Artificial Intelligence and Statistics*, pages 231–239, 2016.
- [22] Philipp Batz, Andreas Ruttner, and Manfred Opper. Approximate bayes learning of stochastic differential equations. *arXiv preprint arXiv:1702.05390*, 2017.
- [23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [24] Christian Donner. Sgpd\_inference. [https://github.com/christiando/SGPD\\_Inference](https://github.com/christiando/SGPD_Inference), 2018.
- [25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [26] David J Hand, Fergus Daly, K McConway, D Lunn, and E Ostrowski. *A handbook of small data sets*, volume 1. cRc Press, 1993.
- [27] Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository, 2017.
- [28] Paulo Cortez and Aníbal de Jesus Raimundo Morais. A data mining approach to predict forest fires using meteorological data. 2007.
- [29] Fabian Keller, Emmanuel Muller, and Klemens Bohm. Hics: high contrast subspaces for density-based outlier ranking. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*, pages 1037–1048. IEEE, 2012.
- [30] Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 29–37, 2011.
- [31] Benigno Uribe, Iain Murray, and Hugo Larochelle. A deep and tractable density estimator. In *International Conference on Machine Learning*, pages 467–475, 2014.
- [32] Yves-Laurent Kom Samo and Stephen Roberts. Scalable nonparametric bayesian inference on point processes with gaussian processes. In *International Conference on Machine Learning*, pages 2227–2236, 2015.
- [33] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.
- [34] Yarin Gal and Richard Turner. Improving the gaussian process sparse spectrum approximation by representing uncertainty in frequency inputs. In *International Conference on Machine Learning*, pages 655–664, 2015.

---

# Comparing Direct and Indirect Temporal-Difference Methods for Estimating the Variance of the Return

---

Craig Sherstan<sup>1</sup>, Dylan R. Ashley<sup>2</sup>, Brendan Bennett<sup>2</sup>, Kenny Young,  
Adam White, Martha White, Richard S. Sutton

Reinforcement Learning and Artificial Intelligence Laboratory  
Department of Computing Science, University of Alberta, Edmonton, Canada

1. Corresponding author: sherstan@ualberta.ca

2. Co-authors contributed equally.

## Abstract

Temporal-difference (TD) learning methods are widely used in reinforcement learning to estimate the expected return for each state, without a model, because of their significant advantages in computational and data efficiency. For many applications involving risk mitigation, it would also be useful to estimate the *variance* of the return by TD methods. In this paper, we describe a way of doing this that is substantially simpler than those proposed by Tamar, Di Castro, and Mannor in 2012, or those proposed by White and White in 2016. We show that two TD learners operating in series can learn expectation and variance estimates. The trick is to use the square of the TD error of the expectation learner as the reward of the variance learner, and the square of the expectation learner’s discount rate as the discount rate of the variance learner. With these two modifications, the variance learning problem becomes a conventional TD learning problem to which standard theoretical results can be applied. Our formal results are limited to the table lookup case, for which our method is still novel, but the extension to function approximation is immediate, and we provide some empirical results for the linear function approximation case. Our experimental results show that our direct method behaves just as well as a comparable indirect method, but is generally more robust.

## 1 INTRODUCTION

Conventionally, in reinforcement learning (RL) the agent estimates the expected value of the return—the discounted sum of future rewards—as an intermediate step

to finding an optimal policy. The agent estimates the value function by averaging the returns observed from each state in a trajectory of experiences. To estimate this value function online—while the trajectory is still unfolding—we update the agent’s value estimates towards the expected return. Algorithms that estimate the expected value of the return in this way are called temporal-difference (TD) learning methods. However, it is reasonable to consider estimating other functions of the return beyond the first moment. For example, Belle-mare et al. (2017) estimated the distribution of returns explicitly. In this paper, we focus on estimating the variance of the return using TD methods.

The variance of the return can be used to design algorithms which account for risk in decision making. The main approach is to formulate the agent’s objective as maximizing reward, while minimizing the variance of the return (Sato et al., 2001; Prashanth and Ghavamzadeh, 2013; Tamar et al., 2012).

An estimate of the variance of the return can also be useful for adapting the parameters of a learning system automatically, thus avoiding time-consuming, human-driven meta parameter tuning. Sakaguchi and Takano (2004) used the variance estimate explicitly in the decision making policy to set the temperature variable in the Boltzmann action selection rule. Using variance in this way can automatically adjust the amount of exploration, allowing the learning system to adapt to new circumstances online. Conventionally, this temperature would either be set to a constant or decayed according to a fixed schedule. In either circumstance, the performance can be quiet poor in non-stationary domains, and a human expert is required to select the constant value or fixed schedule. Similarly, White and White (2016) estimated the variance of the return to automatically adapt the trace-decay parameter,  $\lambda$ , used in learning updates of TD algorithms (see Section 2 for an explanation of the role of  $\lambda$ ). Not only does this approach avoid the need to tune  $\lambda$  by hand, but it can result in faster learning.

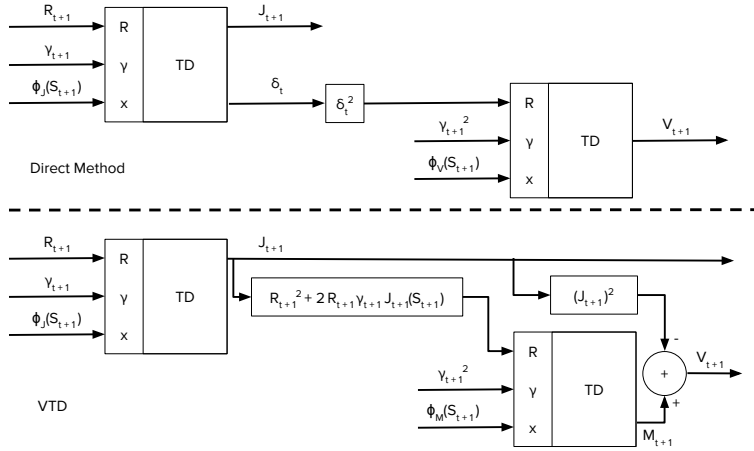


Figure 1: Each TD node takes as input a reward  $R$ , a discounting function  $\gamma$ , and features  $\phi$ . For the direct method (**top**) the squared TD error of the first-stage value estimator is used as the meta-reward for the second-stage  $V$  estimator. For VTD (**bottom**), a more complex computation is used for the meta-reward and an extra stage of computation is required.

The variance  $V$  of the return can be estimated either directly or indirectly. Indirect estimation involves computing an estimate of variance from estimates of the first and second moments. Sobel (1982) was the first to formulate Bellman operators for the second moment and showed how this could be used to compute variance indirectly. This is the approach used by Tamar et al. (2016), Tamar and Mannor (2013), and Prashanth and Ghavamzadeh (2013). White and White (2016) introduced several extensions to this indirect method including estimation of the  $\lambda$ -return (Sutton and Barto, 1998), support for off-policy learning (Sutton, Maei, et al., 2009; Maei, 2011), and state-dependent discounting (Sutton, Modayil, et al., 2011; White, 2017). Their method, which they refer to as VTD, serves as the indirect estimation algorithm used in this paper. We note that an alternative method, which we do not investigate here, could be to estimate the distribution of returns as done by Bellemare et al. (2017) and compute the variance from this estimated distribution.

Variance may also be estimated directly. Tamar et al. (2012) gave a direct algorithm but restricted it to estimating cost-to-go returns in a strictly episodic manner, i.e., estimates are only updated after an entire trajectory has been captured. We introduce a new algorithm for directly estimating the variance of the return incrementally using TD methods. Our algorithm uses two TD learners, one for estimating value and the other for estimating the variance of the return. These estimators operate in series with the squared TD error of the value learner serving as the reward of the variance learner and the squared discount rate of the value learner serving as the discount rate of the variance learner. Like VTD (White and White, 2016), our algorithm supports estimating the variance of the  $\lambda$ -return, state-dependent discounting, estimating the variance of the on-policy return from off-policy samples, and estimating the variance of the off-policy return from on-policy samples (Section 3.2 motivates these extensions). We call our new algorithm Direct Variance TD

(DVTD). We recognize that the algorithm of Sato et al. (2001) can be seen as the simplest instance of our algorithm, using the on-policy setting with fixed discounting and no traces<sup>1</sup>. Sakaguchi and Takano (2004) also used this simplified algorithm, but treated the discount of the variance estimator as a free parameter.

We introduce a Bellman operator for the variance of the return, and further prove that, even for a value function that does not satisfy the Bellman operator for the expected return, the error in this recursive formulation is proportional to the error in the value function estimate. Interestingly, the Bellman operator for the second moment requires an unbiased estimate of the return (White and White, 2016). Since our Bellman operator for the variance avoids this term, it has a simpler update. As shown in Figure 1, Both DVTD and VTD can be seen as a network of two TD estimators running sequentially. Note, that we restrict our formal derivations and subsequent analysis to the table lookup setting.

Our primary goal is to understand the empirical properties of the direct and indirect approaches. In general, we found that DVTD is just as good as VTD and in many cases better. We observe that DVTD behaves better in the early stages of learning before the value function has converged. Furthermore, we observe that the variance of the estimates of  $V$  can be higher for VTD under several circumstances: (1) when there is a mismatch in step-sizes between the value estimator and the  $V$  estimator, (2) when traces are used with the value estimator, (3) when estimating  $V$  of the off-policy return, and (4) when there is error in the value estimate. Finally, we observe significantly better performance of DVTD in a linear function approximation setting. Overall, we conclude that the direct approach to estimating  $V$ , DVTD, is both simpler and better behaved than VTD.

<sup>1</sup>Dimitrakakis (2006) used a related TD method, which estimates the squared TD error

## 2 THE MDP SETTING

We model the agent’s interaction with the environment as a finite Markov decision process (MDP) consisting of a finite set of states  $\mathcal{S}$ , a finite set of actions,  $\mathcal{A}$ , and a transition model  $p : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$  defining the probability  $p(s'|s, a)$  of transitioning from state  $s$  to  $s'$  when taking action  $a$ . In the policy evaluation setting considered in this paper, the agent follows a fixed policy  $\pi(a|s) \in [0, 1]$  that provides the probability of taking action  $a$  in state  $s$ . At each timestep the agent receives a random reward  $R_{t+1}$ , dependent only on  $S_t, A_t, S_{t+1}$ . The return is the discounted sum of future rewards

$$\begin{aligned} G_t &= R_{t+1} + \gamma_{t+1}R_{t+2} + \gamma_{t+1}\gamma_{t+2}R_{t+3} + \dots \\ &= R_{t+1} + \gamma_{t+1}G_{t+1}. \end{aligned} \quad (1)$$

where  $\gamma \in [0, 1]$  specifies the degree to which future rewards are discounted. Note that we define discounting as state-dependent such that  $\gamma_{t+1} \equiv \gamma(S_{t+1})$ . This allows us to combine the specification of continuing and episodic tasks. Further implications of this are discussed in Section 3.2.

The value of a state,  $j(s)$ , is defined as the expected return from state  $s$  under a particular policy  $\pi$ :

$$j(s) = \mathbb{E}_\pi[G_t | S_t = s]. \quad (2)$$

We use  $j$  to indicate the true value function and  $J$  the estimate. The TD-error is the difference between the one-step approximation and the current estimate:

$$\delta_t = R_{t+1} + \gamma_{t+1}J_t(S_{t+1}) - J_t(S_t). \quad (3)$$

This can then be used to update the value estimator using a TD method, such as TD(0) as follows:

$$J(s)_{t+1} = J(s)_t + \alpha\delta_t \quad (4)$$

## 3 ESTIMATING THE VARIANCE OF THE RETURN

For clarity of presentation, we first discuss the simplest version of both the direct and indirect methods and present the full algorithms in Section 3.2.

The direct TD method uses both a value estimator and a variance estimator. The *value estimator* provides an estimate of the expected return. The *variance estimator*, on the other hand, uses the value estimator to provide an estimate of the variance of the return. Since we use TD methods for both the value and variance estimators we need to adopt additional notation; variables with a bar are used by either the second moment or variance estimator. Otherwise, they are used by the value estimator.

The key to using both the indirect and direct methods as TD methods is to provide a discounting function,  $\bar{\gamma}$ , and a meta-reward,  $\bar{R}$ . In the following, we present a simplified TD(0) version of both algorithms.

### Simplified Direct Variance Algorithm

$$\begin{aligned} \bar{\gamma}_{t+1} &\leftarrow \gamma_{t+1}^2 \\ \bar{R}_{t+1} &\leftarrow \delta_t^2 \\ \bar{\delta}_t &\leftarrow \bar{R}_{t+1} + \bar{\gamma}_{t+1}V_t(s') - V_t(s) \\ V_{t+1}(s) &\leftarrow V_t(s) + \bar{\alpha}\bar{\delta}_t \end{aligned} \quad (5)$$

### Simplified Second Moment Algorithm

$$\begin{aligned} \bar{\gamma}_{t+1} &\leftarrow \gamma_{t+1}^2 \\ \bar{R}_{t+1} &\leftarrow R_{t+1}^2 + 2\gamma_{t+1}R_{t+1}J_{t+1}(s') \\ \bar{\delta}_t &\leftarrow \bar{R}_{t+1} + \bar{\gamma}_{t+1}M_t(s') - M_t(s) \\ M_{t+1}(s) &\leftarrow M_t(s) + \bar{\alpha}\bar{\delta}_t \\ V_{t+1}(s) &\leftarrow M_{t+1}(s) - J_{t+1}(s)^2 \end{aligned} \quad (6)$$

### 3.1 DERIVATION OF THE DIRECT METHOD

We now derive the direct method for estimating the variance of the return. Again, for clarity, we only consider the simple case described in Section 3 (See Appendix B for a derivation of the more general extended algorithm).

The derivation of the direct method follows from characterizing the Bellman operator for the variance of the return: Theorem 1 gives a Bellman equation for the variance  $v$ . It has the form of a TD target with meta-reward  $\bar{R}_t = \delta_t^2$  and discounting function  $\bar{\gamma}_{t+1} = \gamma_{t+1}^2$ . Therefore, we can estimate  $V$  using TD methods. The Bellman operators for the variance are general, in that they allow for either the episodic or continuing setting, by using variable  $\gamma$ . By directly estimating variance, we avoid a second term in the cumulant that is present in approaches that estimate the second moment (Tamar and Mannor, 2013; Tamar et al., 2016; White and White, 2016).

To have a well-defined solution to the fixed point, we need the discount to be less than one for some transition (White, 2017; Yu, 2015). This corresponds to assuming that the policy is proper, for the cost-to-go setting (Tamar et al., 2016).

**Assumption 1.** *The policy reaches a state  $s$  where  $\gamma(s) < 1$  in a finite number of steps.*

**Theorem 1.** *For any  $s \in \mathcal{S}$ ,*

$$\begin{aligned} j(s) &= \mathbb{E}[R_{t+1} + \gamma_{t+1}j(S_{t+1}) | S_t = s] \\ v(s) &= \mathbb{E}[\delta_t^2 + \gamma_{t+1}^2v(S_{t+1}) | S_t = s] \end{aligned} \quad (7)$$

*Proof.* First we expand  $G_t - j(S_t)$ , from which we re-

cover a series with the form of a return.

$$\begin{aligned} G_t - j(S_t) &= R_{t+1} + \gamma_{t+1}G_{t+1} - j(S_t) \\ &= R_{t+1} + \gamma_{t+1}j(S_{t+1}) - j(S_t) + \gamma_{t+1}(G_{t+1} - j(S_{t+1})) \\ &= \delta_t + \gamma_{t+1}(G_{t+1} - j(S_{t+1})) \end{aligned} \quad (8)$$

The variance of  $G_t$  is therefore

$$\begin{aligned} v(s) &= \mathbb{E} \left[ (G_t - \mathbb{E}[G_t | S_t = s])^2 | S_t = s \right] \\ &= \mathbb{E} \left[ (G_t - j(s))^2 | S_t = s \right] \\ &= \mathbb{E} \left[ (\delta_t + \gamma_{t+1}(G_{t+1} - j(S_{t+1})))^2 | S_t = s \right] \\ &= \mathbb{E} [\delta_t^2 | S_t = s] \\ &\quad + \mathbb{E} [\gamma_{t+1}^2 (G_{t+1} - j(S_{t+1}))^2 | S_t = s] \\ &\quad + 2\mathbb{E} [\gamma_{t+1}\delta_t(G_{t+1} - j(S_{t+1})) | S_t = s] \end{aligned} \quad (9)$$

Equation (7) follows from Lemma 1 in Appendix B which shows  $\mathbb{E}[\gamma_{t+1}\delta_t(G_{t+1} - j(S_{t+1})) | S_t = s] = 0$ . Similar to Lemma 1, using the law of total expectation,  $\mathbb{E}[\gamma_{t+1}^2(G_{t+1} - j(S_{t+1}))^2 | S_t = s] = \mathbb{E}[\gamma_{t+1}^2 v(S_{t+1}) | S_t = s]$ .  $\square$

We provide an initial characterization of error in the variance estimate obtained under this recursion, when an approximate value function rather than the true value function is used. As we show in the below theorem, the resulting error in the variance estimator is proportional to the squared error in the value estimate, and discounted accumulated errors into the future. If the approximation error is small, we expect this accumulated error to be small, particularly as the accumulation errors are signed and so can cancel, and because they are discounted. However, more needs to be done to understand the impact of this accumulated error.

**Theorem 2.** *For approximate value function  $J$  with variance estimate  $V(s) = \mathbb{E}[\delta_t^2 + \gamma_{t+1}^2 V(S_{t+1}) | S_t = s]$ , if there exists  $\epsilon : \mathcal{S} \rightarrow [0, \infty)$  bounding squared value estimation error  $(J(s) - j(s))^2 \leq \epsilon(s)$  and accumulation error  $|\mathbb{E}[\gamma_{t+1}\delta_t(j(S_{t+1}) - J(S_{t+1})) + \gamma_{t+1}^2\gamma_{t+2}\delta_{t+1}(j(S_{t+2}) - J(S_{t+2})) + \dots | S_t = s]| \leq \epsilon(s)$ , then*

$$|v(s) - \mathbb{E}[\delta_t^2 + \gamma_{t+1}^2 V(S_{t+1}) | S_t = s]| \leq 3\epsilon(s)$$

*Proof.* We can re-express the true variance in terms of the approximation  $J$ , as

$$\begin{aligned} v(s) &= \mathbb{E} [(G_t - j(s) + J(s) - J(s))^2 | S_t = s] \\ &= \mathbb{E} [(G_t - J(s))^2 | S_t = s] + (J(s) - j(s))^2 \\ &\quad + 2\mathbb{E}[G_t - J(s) | S_t = s] (J(s) - j(s)) \end{aligned} \quad (10)$$

This last term simplifies to

$$\begin{aligned} \mathbb{E}[G_t - J(s) | S_t = s] &= \mathbb{E}[G_t - j(s) | S_t = s] + j(s) - J(s) \\ &= j(s) - J(s) \end{aligned} \quad (11)$$

giving  $(J(s) - j(s))^2 + 2(j(s) - J(s))(J(s) - j(s)) = -(J(s) - j(s))^2$ . We can use the same recursive form as (9), but with  $J$ , giving

$$\begin{aligned} \mathbb{E}[(G_t - J(s))^2 | S_t = s] &= \mathbb{E}[\delta_t^2 + \gamma_{t+1}^2 V(S_{t+1}) | S_t = s] \\ &\quad + 2\mathbb{E}[\gamma_{t+1}\delta_t(G_{t+1} - J(S_{t+1})) | S_t = s] \\ &\quad + 2\mathbb{E}[\gamma_{t+1}^2\gamma_{t+2}\delta_{t+1}(G_{t+2} - J(S_{t+2})) | S_t = s] + \dots \end{aligned} \quad (12)$$

where the terms involving  $\delta_t(G_{t+1} - J(S_{t+1}))$  accumulate. Notice that

$$\begin{aligned} &|\mathbb{E}[\gamma_{t+1}\delta_t(G_{t+1} - J(S_{t+1})) | S_t = s]| \\ &= |\mathbb{E}[\gamma_{t+1}\delta_t(G_{t+1} - j(S_{t+1})) | S_t = s] \\ &\quad + \mathbb{E}[\gamma_{t+1}\delta_t(j(S_{t+1}) - J(S_{t+1})) | S_t = s]| \\ &= |\mathbb{E}[\gamma_{t+1}\delta_t(j(S_{t+1}) - J(S_{t+1})) | S_t = s]| \end{aligned}$$

where the second equality follows from Lemma 1. By the same argument as in Lemma 1, this will also hold true for all the other terms in (12). By assumption, the sum of all these covariance terms between  $j$  and  $J$  are bounded by  $\epsilon(s)$ . Putting this together, we get

$$\begin{aligned} |v(s) - V(s)| &= |v(s) - \mathbb{E}[\delta_t^2 + \gamma_{t+1}^2 V(S_{t+1}) | S_t = s]| \\ &\leq 2\epsilon(s) + (J(s) - j(s))^2 \leq 3\epsilon(s) \end{aligned} \quad \square$$

### 3.2 THE EXTENDED DIRECT METHOD

Here, we extend the direct method to support estimating the  $\lambda$ -return, state-dependent  $\gamma$ , eligibility traces and off-policy estimation, just as White and White, 2016 did with VTD (derivation provided in Appendix B). We first explain each of these extensions before providing our full direct algorithm and VTD.

The  $\lambda$ -return is defined as

$$G_t^\lambda = R_{t+1} + \gamma_{t+1}(1 - \lambda_{t+1})J_t(S_{t+1}) + \gamma_{t+1}\lambda_{t+1}G_{t+1}^\lambda$$

and provides a bias-variance trade-off by incorporating  $J$ , which is a potentially lower-variance but biased estimate of the return. This trade-off is determined by a state-dependent trace-decay parameter,  $\lambda_t \equiv \lambda(S_t) \in [0, 1]$ . When  $J_t(S_{t+1})$  is equal to the expected return from  $S_{t+1} = s$ , then  $\mathbb{E}_\pi[(1 - \lambda_{t+1})J_t(S_{t+1}) + \gamma_{t+1}\lambda_{t+1}G_{t+1}^\lambda | S_{t+1} = s] = \mathbb{E}_\pi[G_{t+1}^\lambda | S_{t+1} = s]$ , and so the  $\lambda$ -return is unbiased. Beneficially the expected value  $J_t(S_{t+1})$  is lower-variance than the sample  $G_{t+1}^\lambda$ . If  $J_t$  is inaccurate, however, some bias is introduced. Therefore, when  $\lambda = 0$ , the  $\lambda$ -return is lower-variance but can be biased. When  $\lambda = 1$ , the  $\lambda$ -return equals the Monte Carlo return (Equation (1)); in this case, the update target exhibits more variance, but no bias. In the tabular setting evaluated in this paper,  $\lambda$  does not affect

Table 1: Algorithm Notation

$J$	estimated value function of the target policy $\pi$ .
$M$	estimate of the second moment.
$V$	estimate of the variance.
$R, \bar{R}$	meta-reward used by the $J$ and $(M, V)$ estimators.
$\lambda$	bias-variance parameter of the target $\lambda$ -return.
$\kappa, \bar{\kappa}$	trace-decay parameter of the $J$ and $(M, V)$ estimators.
$\gamma, \bar{\gamma}$	discounting function used by $J$ and $(M, V)$ estimators.
$\delta_t, \bar{\delta}_t$	TD error of the $J$ and $(M, V)$ estimators at time $t$ .
$\bar{\rho}$	importance sampling ratio for estimating the variance of the target return from off-policy samples.
$\eta$	importance sampling ratio used to estimate the variance of the off-policy return.

the fixed point solution of the value estimate, only the rate at which learning occurs. It does, however, affect the observed variance of the return, which we estimate. The  $\lambda$ -return is implemented using traces as in the following TD( $\lambda$ ) algorithm, shown with accumulating traces:

$$e_t(s) \leftarrow \begin{cases} \gamma_t \lambda_t e_{t-1}(s) + 1 & s = S_t \\ \gamma_t \lambda_t e_{t-1}(s) & \forall s \in \mathcal{S}, s \neq S_t \end{cases}$$

$$J_{t+1}(S_t) \leftarrow J_t(S_t) + \alpha \delta_t e_t(S_t) \quad (13)$$

For notational purposes, we define the trace parameter for the value and secondary estimators as  $\kappa$  and  $\bar{\kappa}$  respectively. Both of these parameters are independent of the  $\lambda$ -return for which we estimate the variance. That is, we are entirely free to estimate the variance of the  $\lambda$ -return for any value of  $\lambda$  independently of the use of any traces in either the value or secondary estimator.

**State-Dependent  $\gamma$ .** While most RL methods focus on fixed discounting values, it is straightforward to use state-based discounting (Sutton, Modayil, et al., 2011), where  $\gamma_t \equiv \gamma(S_t)$  (White (2017) go further by defining transition based discounting). This generalization enables a wider variety of returns to be considered. First, it allows a convenient means of describing both episodic and continuing tasks and provides an algorithmic mechanism for terminating an episode without defining a recurrent terminal state explicitly. Further, it allows for event-based terminations (Sutton, Modayil, et al., 2011). It also enables soft terminations which may prove useful when training an agent with sub-goals (White, 2017). The use of state-dependent discounting functions is relatively new and remains to be extensively explored.

**Off-policy learning.** Value estimates are made with respect to a target policy,  $\pi$ . If the behavior policy,  $\mu = \pi$  then we say that samples are collected on-policy, otherwise, the samples are collected off-policy. An off-policy learning approach is to weight each update by the importance sampling ratio:  $\rho_t = \frac{\pi(S_t, A_t)}{\mu(S_t, A_t)}$ . There are two different scenarios to be considered when estimating the

variance of the return in the off-policy setting. The first is estimating the variance of the on-policy return of the target policy while following a different behavior policy. The second has the goal of estimating the variance of the off-policy return itself. The off-policy  $\lambda$ -return is:

$$G_t^{\lambda:\rho} = \rho_t (R_{t+1} + \gamma_{t+1} (1 - \lambda_{t+1}) j_t(S_{t+1}) + \gamma_{t+1} \lambda_{t+1} G_{t+1}^{\lambda:\rho}). \quad (14)$$

where the multiplication by the potentially large importance sampling ratios can significantly increase variance.

It is important to note you would only ever estimate one or the other of these settings with a given estimator. Let  $\eta$  be the weighting for the value estimator, and  $\bar{\rho}$  the weighting for the variance estimator. If estimating the variance of the target return from off-policy samples, the first scenario,  $\eta_t = 1 \forall t$  and  $\bar{\rho}_t = \rho_t$ . If estimating the variance of the off-policy return  $\bar{\rho}_t = 1 \forall t$  and  $\eta_t = \rho_t$ .

### 3.2.1 The Extended Algorithms

To estimate  $V$ , our method uses both value and variance estimators. The *value estimator* provides an estimate of the expected return. The *variance estimator*, on the other hand, uses the value estimator to provide an estimate of the variance of the return. Our method, DVTD, and the indirect method, VTD, can be seen as simply defining a meta-reward and a discounting function and can thus be learned with any known TD method, such as TD with accumulating traces as shown in Equation 13. Table 1 summarizes our notation.

#### Direct Variance Algorithm - DVTD

$$\begin{aligned} \bar{R}_{t+1} &\leftarrow (\eta_t \delta_t + (\eta_t - 1) J_{t+1}(s))^2 \\ \bar{\gamma}_{t+1} &\leftarrow \gamma_{t+1}^2 \lambda_{t+1}^2 \eta_t^2 \\ \bar{\delta}_t &\leftarrow \bar{R}_{t+1} + \bar{\gamma}_{t+1} V_t(s') - V_t(s) \\ \bar{e}_t(s) &\leftarrow \begin{cases} \bar{\rho}_t (\bar{\gamma}_t \bar{\kappa}_t \bar{e}_{t-1}(s) + 1) & s = S_t \\ \bar{\rho}_t (\bar{\gamma}_t \bar{\kappa}_t \bar{e}_{t-1}(s)) & \forall s \in \mathcal{S}, s \neq S_t \end{cases} \\ V_{t+1}(s) &\leftarrow V_t(s) + \bar{\alpha} \bar{\delta}_t \bar{e}_t(s) \end{aligned} \quad (15)$$

We also present the full VTD algorithm below (again, shown with accumulating traces). Note that this algorithm does not impose that the variance be non-negative.

### Second Moment Algorithm - VTD

$$\begin{aligned}
\bar{G}_t &\leftarrow R_{t+1} + \gamma_{t+1}(1 - \lambda_{t+1})J_{t+1}(s') \\
\bar{R}_{t+1} &\leftarrow \eta_t^2 \bar{G}_t^2 + 2\eta_t^2 \gamma_{t+1} \lambda_{t+1} \bar{G}_t J_{t+1}(s') \\
\bar{\gamma}_{t+1} &\leftarrow \eta_t^2 \gamma_{t+1}^2 \lambda_{t+1}^2 \\
\bar{\delta}_t &\leftarrow \bar{R}_{t+1} + \bar{\gamma}_{t+1} M_t(s') - M_t(s) \\
\bar{e}_t(s) &\leftarrow \begin{cases} \bar{\rho}_t(\bar{\gamma}_t \bar{\kappa}_t \bar{e}_{t-1}(s) + 1) & s = S_t \\ \bar{\rho}_t(\bar{\gamma}_t \bar{\kappa}_t \bar{e}_{t-1}(s)) & \forall s \in \mathcal{S}, s \neq S_t \end{cases} \\
M_{t+1}(s) &\leftarrow M_t(s) + \bar{\alpha} \bar{\delta}_t \bar{e}_t(s) \\
V_{t+1}(s) &\leftarrow M_{t+1}(s) - J_{t+1}(s)^2
\end{aligned} \tag{16}$$

## 4 EXPERIMENTS

The primary purpose of these experiments is to demonstrate that both algorithms can approximate the true expected  $V$  under various conditions in the tabular setting. We consider two domains. The first is a deterministic chain, which is useful for basic evaluation and gives results which are easy to interpret (Figure 2). The second is a randomly generated MDP, with different discount and trace-decay parameters in each state (Figure 3). For all experiments Algorithm 13 is used as the value estimator. Unless otherwise stated, traces are not used ( $\kappa = \bar{\kappa} = 0$ ) and estimates were initialized to zero. For each experimental setting the average of 30 separate experiments is presented with standard deviation shown as shaded regions. True values were determined by Monte Carlo estimation and are shown as dashed lines in the figures.

We look at the effects of relative step-size between the value estimator and the variance estimators in Section 4.1. Then, in Section 4.2 we use the random MDP to show that both algorithms can estimate the variance with state-dependent  $\gamma$  and  $\lambda$ . In Section 4.3 we evaluate the two algorithms' responses to errors in the value estimate. Section 4.4 looks at the effect of using traces in the estimation method. We then examine the off-policy setting in Section 4.5. Finally, Section 4.6 provides experimental results in a linear function approximation setting.

### 4.1 THE EFFECT OF STEP-SIZE

We use the chain MDP to investigate the impact of step-size choice. In Figure 4(a) all step-sizes are the same ( $\alpha = \bar{\alpha} = 0.001$ ) and here both algorithms behave similarly. For Figure 4(b) the step-size of the value estimate, ( $\alpha = 0.01$ ), is greater than that of the secondary estimators, ( $\bar{\alpha} = 0.001$ ). Now DVTD smoothly approaches the

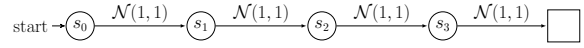


Figure 2: **Chain MDP** with 4 non-terminal states and 1 terminal state. From non-terminal states there is a single action with a deterministic transition to the right. On each transition, rewards are drawn from a normal distribution with mean and variance of 1.0. Evaluation was performed for  $\lambda = 0.9$ , which was chosen because it is not at either extreme and because 0.9 is a commonly used value for many RL experimental domains.

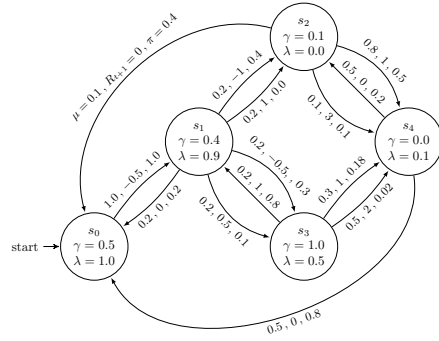


Figure 3: **Random MDP**, with a stochastic policy and state-based  $\gamma$  and  $\lambda$ . The state-dependent values of  $\gamma$  and  $\lambda$  are chosen to provide a range of values, with at least one state acting as a terminal state where  $\gamma = 0$ . On-policy action probabilities are indicated by  $\mu$  and off-policy ones by  $\pi$ .

correct value, while VTD first dips well below zero. This is expected as the estimates are initialized to zero and the variance is calculated as  $V(s) = M(s) - J(s)^2$ . If the second moment lags behind the value estimate, then the variance will be negative. In Figure 4(c) the step-size for the secondary estimators is larger than for the value estimator ( $0.001 = \alpha < \bar{\alpha} = 0.01$ ). While both methods overshoot the target in this example, VTD has greater overshoot. For both cases of unequal step-size, we see higher variance in the estimates for VTD.

Figure 5 explores this further. Here the value estimator is initialized to the true values and updates are turned off ( $\alpha = 0$ ). The secondary estimators are initialized to zero and learn with  $\bar{\alpha} = 0.001$ , chosen simply to match the step-sizes used in the previous experiments. Despite being given the true values the VTD algorithm produces higher variance in its estimates, suggesting that VTD is dependent on the value estimator tracking.

This sensitivity to step-size is shown in Figure 6. All estimates are initialized to their true values. For each ratio, we computed the average variance of the 30 runs of 2000



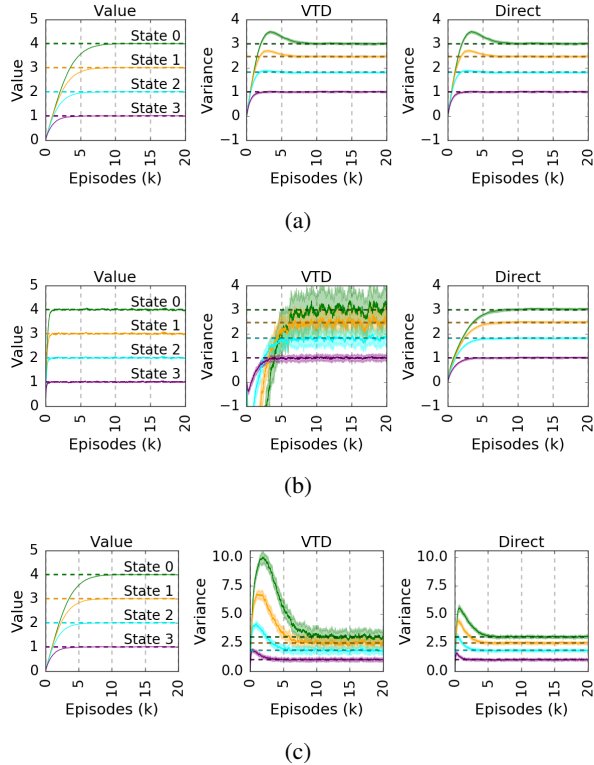


Figure 4: **Chain MDP** ( $\lambda = 0.9$ ). Varying the ratio of step-size between value and variance estimators. **a)** Step-sizes equal.  $\alpha = \bar{\alpha} = 0.001$ . **b)** Variance step-size smaller.  $\alpha = 0.01, \bar{\alpha} = 0.001$ . **c)** Variance step-size larger.  $\alpha = 0.001, \bar{\alpha} = 0.01$ . We see greater variance in the estimates and greater over/undershoot for VTD when step-sizes are not equal.

episodes. We can see that DVTD is largely insensitive to step-size ratio, but that VTD has higher mean squared error (MSE) except when the step-sizes are equal. This result holds for the other experimental settings of this paper, including the random MDP, but further results are omitted for brevity.

Would there ever be a situation where different step-sizes between value and secondary estimators is justified? The automatic tuning of parameters, such as step-size, is an important area of research, seeking to make learning algorithms more efficient, robust and easier to deploy. Methods which automatically set the step-sizes may produce different values specific to the performance of each estimator. One such algorithm is ADADELTA, which adapts the step-size based on the TD error of the estimator (Zeiler, 2012). Figure 7 shows that using a separate ADADELTA ( $\rho = 0.99, \epsilon = 1e-6$ ) step-size calculation for each estimator results in higher variance for VTD as expected, given that the value estimator and VTD produce different TD errors.

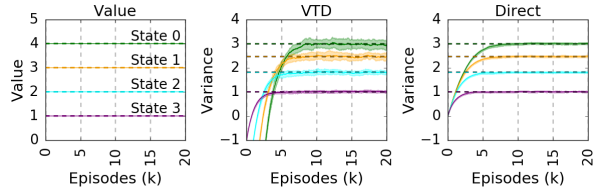


Figure 5: **Chain MDP** ( $\lambda = 0.9$ ). Value estimate held fixed at the true values ( $\alpha = 0, \bar{\alpha} = 0.001$ ). Notice the increased estimate variance for VTD, especially State 0.

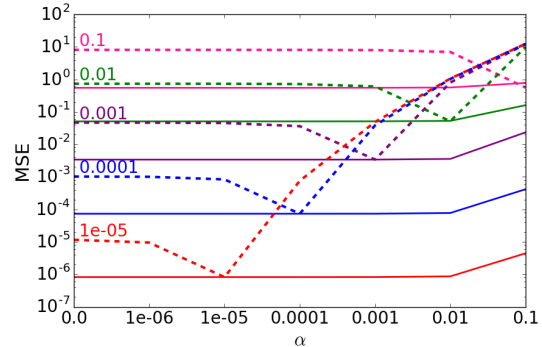


Figure 6: **Chain MDP** ( $\lambda = 0.9$ ). The MSE summed over all states as a function of ratios between the value step-size  $\alpha$  (shown along the x-axis) and the variance step-size  $\bar{\alpha}$  (shown in the 5 series). The direct algorithm is indicated by the solid lines, and VTD is indicated by the dashed. The MSE of the VTD algorithm is higher than the direct algorithm, except when the step-size is the same for all estimators,  $\alpha = \bar{\alpha}$  or for very small  $\bar{\alpha}$ .

## 4.2 STATE-DEPENDENT $\gamma$ AND $\lambda$ .

One of the contributions of VTD was the generalization to support state-based  $\gamma$  and  $\lambda$ . Here we evaluate the random MDP from Figure 3 (in the on-policy setting, using  $\mu$ ), which was designed for this scenario and which has a stochastic policy, is continuing, and has multiple possible actions from each state. Both algorithms achieved similar results (see Appendix A).

## 4.3 VARIABLE ERROR IN THE VALUE ESTIMATES

The derivation of our DVTD assumes access to the true value function. The experiments of the previous sections demonstrate that both methods are robust under this assumption, in the sense that the value function was estimated from data and used to estimate  $V$ . It remains unclear, however, how well these methods perform when the value estimates converge to biased solutions.

To examine this, we again use the random MDP shown

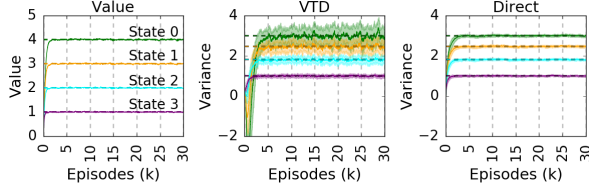


Figure 7: **Chain MDP** ( $\lambda = 0.9$ ). Results using ADADELTA algorithm to automatically and independently set the step-sizes  $\alpha$  and  $\bar{\alpha}$ . The step-sizes produced are given in Appendix D.

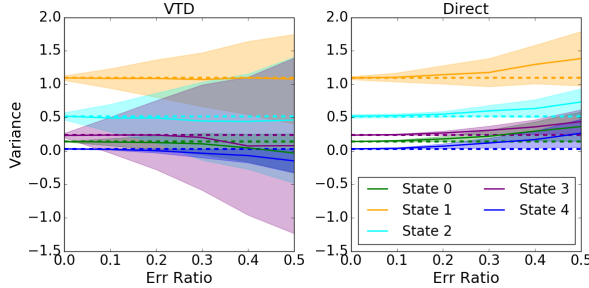


Figure 8: **Random MDP**. For each run, the value estimate of each state is offset by a random noise drawn from a uniform distribution whose size is a function of an error ratio and the maximum true value in the MDP. Standard deviation of the estimates is shown by shading.

by Figure 3. True values for the value functions and variance estimates are calculated from Monte Carlo simulation of 10,000,000 timesteps. For each run of the experiment each state of the value estimator was initialized to the true value plus an error ( $J(s)_0 = j(s) + \epsilon(s)$ ) drawn from a uniform distribution:  $\epsilon(s) \in [-\zeta, \zeta]$ , where  $\zeta = \max_s(|v(s)|) * \text{err ratio}$  (the maximum value in this domain is 1.55082409). The value estimate was held constant throughout the run ( $\alpha = 0.0$ ). The experiment consisted of 120 runs of 80,000 timesteps. To look at the steady-state response of the algorithms we use only the last 10,000 timesteps in our calculations. Figure 8 plots the average variance estimate for each state with the average standard deviation of the estimates as the shaded regions. Sweeps over step-size were conducted,  $\bar{\alpha} \in [0.05, 0.04, 0.03, 0.02, 0.01, 0.007, 0.005, 0.003, 0.001]$ , and the MSE evaluated for each state. Each data point is for the step-size with the lowest MSE for that error ratio and state. While the average estimate is closer to the true values for VTD, the variance of the estimates is much larger. Further, the average estimates for VTD are either unchanged or move negative, while those of the direct algorithm tend toward positive bias.

For Figure 9 the MSE is summed over all states. Again, for each error ratio the MSE was compared over the same

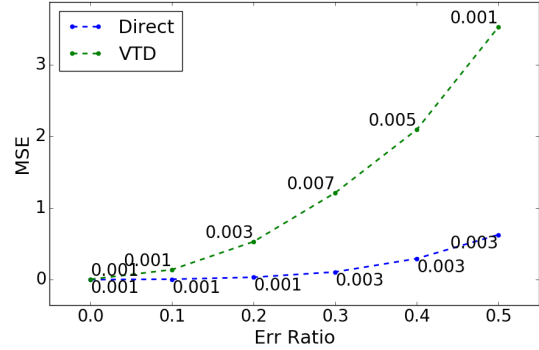


Figure 9: **Random MDP**. The MSE computed for the last 10,000 timesteps of 120 runs summed over all states using the step-size with the lowest overall MSE at each error ratio. For each point the step-size used ( $\alpha = \bar{\alpha}$ ) is displayed.

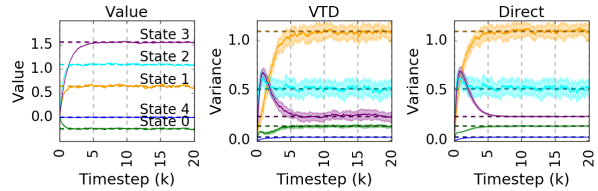


Figure 10: **Random MDP**. Using traces (TD( $\lambda$ ),  $\alpha = \bar{\alpha} = 0.01$ ). Traces only used in value estimator ( $\kappa = 1.0, \bar{\kappa} = 0.0$ ). Notice the slight increase in the variance of the VTD estimates for State 0 and 3.

step-sizes as before and, for each point, the smallest MSE is plotted. These results suggest the direct algorithm is less affected by error in  $J$ .

#### 4.4 USING TRACES

We briefly look at the behavior of the random MDP when traces are used. We found no difference when traces are only used in the secondary estimator and not in the value estimator ( $\kappa = 0.0, \bar{\kappa} = 1.0$ . See Appendix A, Figure 14). Figure 10 considers the opposite scenario, where traces are only used in the value estimator ( $\kappa = 1.0, \bar{\kappa} = 0.0$ ). Here we do see a difference. Particularly the VTD method shows more variance in its estimates for State 0 and 3.

#### 4.5 OFF-POLICY LEARNING

We evaluate two different off-policy scenarios on the random MDP. First, we estimate  $V$  under the target policy from off-policy samples. That is, we estimate the  $V$  that would be observed if we followed the target policy, i.e.,  $\eta = 1, \bar{\rho} = \rho$ . Both methods achieved similar results in

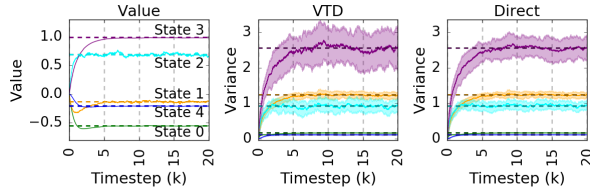


Figure 11: **Random MDP** estimating the variance of the off-policy return ( $\alpha = \bar{\alpha} = 0.01, \bar{\rho} = 1, \eta = \rho$ ).

this setting (Figure 15). In the second off-policy setting, we estimate the variance of the off-policy return (Equation 14). Here  $\bar{\rho} = 1$  and  $\eta = \rho$ . Figure 11 shows that both algorithms successfully estimate the return in this setting. However, despite having the same step-size as the value estimator, VTD produces higher variance in its estimates, as is most clearly seen in State 3.

#### 4.6 FUNCTION APPROXIMATION

While this paper has focused on the tabular case, where each state is represented uniquely, here we include a first empirical result in the function approximation setting. We evaluate both methods on the random walk shown in Figure 12(a). This domain was previously used by Tamar et al. (2016) for indirectly estimating the variance of the return with LSTD( $\lambda$ ). We use transition based  $\gamma$  (White, 2017) to remove the terminal state and translate the task into a continuing task. Further, we alter the state representation to make it more amenable to TD( $\lambda$ ). For a state  $s_i$  we used  $\phi_J(i) = [1, (i + 1)/30]^T$  as features for the value learner and  $\phi_M(i) = \phi_V(i) = [1, (i + 1)/30, (i + 1)^2/30^2]^T$  as features for the secondary learner. We set  $\kappa = \bar{\kappa} = 0.95$  and performed sweeps over step-sizes of  $\{2^i, i \in \{-15, -12, \dots, -1, 0\}\}$ . We first found the best step-size for the value learner and then found the best step-size for VTD. Using the same step-size for VTD and DVTD, we obtain the results shown in Figure 12(b). Here we see DVTD drastically outperforms VTD. Further details are available in Appendix C.

## 5 DISCUSSION

Both DVTD and VTD effectively estimate the variance across a range of settings, but DVTD is simpler and more robust. This simplicity alone makes DVTD preferable. The higher variance in estimates produced by VTD is likely due to the larger target which VTD uses in its learning updates:  $\mathbb{E}[X^2] \geq \mathbb{E}[(X - \mathbb{E}[X])^2]$ ; we show more explicitly how this affects the updates of VTD in Appendix E. We expect the differences between the two approaches to be most pronounced for domains with larger returns than those demonstrated here. Consider

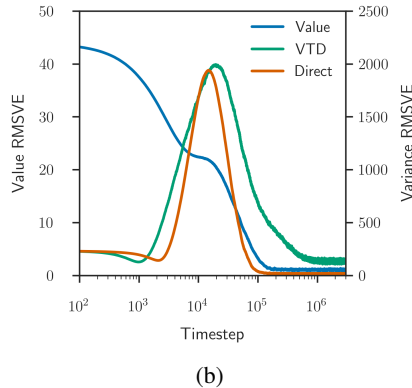
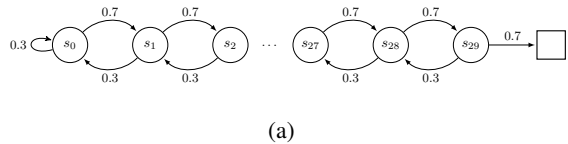


Figure 12: **Random Walk.** **a)** Random walk with rewards of  $-1$  for every transition to a non-terminal state. Note that there is no discounting in this domain. **b)** Results under linear function approximation averaged over 100 runs. Shading indicates standard error (negligible).

the task of a helicopter hovering formalized as a reinforcement learning task (Kim et al., 2004). In the most well-known variants of this problem the agent receives a massive negative reward for crashing the helicopter (e.g., minus one million). In such problems the magnitude and variance of the return is large. Here, estimating the second moment may not be feasible from a statistical point of view, whereas the target of our direct variance estimate should be better behaved. By focusing on simple MDPs we were able to carefully evaluate the properties of these algorithms while keeping them isolated from additional effects like state-aliasing due to function approximation. Further studies in more complex settings, such as function approximation, are left to future work.

#### Acknowledgements

Funding was provided by the Natural Sciences and Engineering Research Council of Canada, Alberta Innovates, and Google DeepMind. Thanks to Hossein Aboutaleb who notified us of an error in the original proof of Theorem 2.

## References

- Bellemare, M. G., Dabney, W., & Munos, R. (2017). A distributional perspective on reinforcement learning. In *International Conference on Machine Learning (ICML)*. Sydney, Australia.
- Dimitrakakis, C. (2006). *Ensembles for sequence learning* (Doctoral dissertation, EPFL).
- Kim, H. J., Jordan, M. I., Sastry, S., & Ng, A. Y. (2004). Autonomous helicopter flight via reinforcement learning. In *Advances in Neural Information Processing Systems (NIPS)* (pp. 799–806).
- Maei, H. R. (2011). *Gradient temporal-difference learning algorithms* (Doctoral dissertation, University of Alberta).
- Prashanth, L. & Ghavamzadeh, M. (2013). Actor-critic algorithms for risk-sensitive mdps. In *Advances in Neural Information Processing Systems (NIPS)* (pp. 252–260).
- Sakaguchi, Y. & Takano, M. (2004). Reliability of internal prediction/estimation and its application. I. Adaptive action selection reflecting reliability of value function. *Neural Networks*, 17(7), 935–952.
- Sato, M., Kimura, H., & Kobayashi, S. (2001). TD algorithm for the variance of return and mean-variance reinforcement learning. *Transactions of the Japanese Society for Artificial Intelligence*, 16(3), 353–362.
- Sobel, M. J. (1982). The variance of discounted markov decision processes. *Journal of Applied Probability*, 19(4), 794–802.
- Sutton, R. S. & Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.
- Sutton, R. S., Maei, H. R., Precup, D., Bhatnagar, S., Silver, D., Szepesvári, C., & Wiewiora, E. (2009). Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *International Conference on Machine Learning (ICML)* (pp. 993–1000).
- Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., & Precup, D. (2011). Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)* (Vol. 2, pp. 761–768).
- Tamar, A., Di Castro, D., & Mannor, S. (2012). Policy gradients with variance related risk criteria. In *International Conference on Machine Learning (ICML)* (pp. 387–396).
- Tamar, A., Di Castro, D., & Mannor, S. (2016). Learning the variance of the reward-to-go. *Journal of Machine Learning Research*, 17(13), 1–36.
- Tamar, A. & Mannor, S. (2013). Variance adjusted actor critic algorithms. arXiv: 1310.3697
- White, M. (2017). Unifying task specification in reinforcement learning. In *International Conference on Machine Learning (ICML)* (pp. 3742–3750).
- White, M. & White, A. (2016). A greedy approach to adapting the trace parameter for temporal difference learning. In *International Conference on Autonomous Agents & Multiagent Systems (AAMAS)* (pp. 557–565).
- Yu, H. (2015). On convergence of emphatic temporal-difference learning. In *Conference on Learning Theory (COLT)* (pp. 1724–1751).
- Zeiler, M. D. (2012). ADADELTA: An adaptive learning rate method. arXiv: 1212.5701

---

# How well does your sampler really work?

---

**Ryan Turner**  
Uber AI Labs

**Brady Neal**  
MILA, Université de Montréal

## Abstract

We present a data-driven benchmark system to evaluate the performance of new MCMC samplers. Taking inspiration from the COCO benchmark in optimization, we view this benchmark as having critical importance to machine learning and statistics given the rate at which new samplers are proposed. The common hand-crafted examples to test new samplers are unsatisfactory; we take a meta-learning-like approach to generate realistic benchmark examples from a large corpus of data sets and models. Surrogates of posteriors found in real problems are created using highly flexible density models including modern neural network models. We provide new insights into the real effective sample size of various samplers per unit time and the estimation efficiency of the samplers per sample. Additionally, we provide a meta-analysis to assess the predictive utility of various MCMC diagnostics and perform a nonparametric regression to combine them.

## 1 INTRODUCTION

Markov chain Monte Carlo (MCMC) methods have seen a huge increase in use over the last few decades. The goal in MCMC methods is to take samples from a complex probability distribution  $p^*$  given access only to its unnormalized density  $\tilde{p}$ . The primary use case for MCMC methods is sampling from Bayesian posteriors for the purpose of Monte Carlo integration, which includes building posterior predictive distributions and posterior summaries. These posteriors are generally intractable to normalize and sample from in modern models, including models as simple as logistic regression.

Approaches such as rejection sampling provide exact independent samples, and importance sampling provides exact independent (but weighted) samples. These approaches are generally computationally inefficient (rejection sampling) or are statistically unsound (importance sampling) except in very low dimensional problems [MacKay, 2003, Ch. 29]. MCMC methods produce a Markov chain that marginally samples from the target distribution  $p^*$  exactly and have a low per sample computation cost. The downside is that they provide a sequence of *correlated* samples, albeit marginally from the target distribution. Therefore, any estimates derived from an MCMC chain of length  $N$  will have far less accuracy than  $N$  iid samples. Despite there being numerous MCMC diagnostics, there is no practical way to guarantee the accuracy of derived estimates in practice.

Each machine learning conference contains a publication proposing a new variation on MCMC methods. The community lacks a method to determine if these new methods actually sample from posteriors found in real problems with improved accuracy over existing samplers. New methods are benchmarked via either 1) hand-crafted toy problems (where a ground-truth is known) or 2) test set performance on real problems. The issue with hand-crafted examples is obvious: Performance on these problems may have little relation to performance on real problems and it is at odds with accepted practice in modern machine learning.

Benchmarking via test set performance on real problems is laudable. However, it confounds the specification of the model and priors with the performance of the sampler. In a misspecified model it is possible that a sampler stuck in an unrepresentative part of the posterior could actually have higher test set performance [Sharp and Rattray, 2010]. Conversely, a better sampler may improve test set performance by having good local mixing; however, it is still nowhere near exact iid samples. There is no way to quantify the distance to exact iid samples from test set performance alone.

Whether current samplers are providing samples from anything close to the true posterior on difficult problems is of critical importance for determining future research directions. Are samplers with higher test set performance actually sampling from real posteriors more faithfully? Can we sample with any fidelity from complex high dimensional distributions? Is that merely a “fool’s errand”? The answers to these questions will determine if it is a worthwhile endeavor to continue to hone MCMC methods for application in successful modern models such as deep neural networks.

Practitioners in Bayesian statistics have long faced the dilemma of whether they can trust the output of their sampler, in particular, because statisticians are not traditionally concerned only with test set error rates. As a result, there is decades of work in developing MCMC diagnostics that aim to *alert* a practitioner to a *poorly mixing* chain [Cowles and Carlin, 1996]. That is, if a chain has a long autocorrelation time, the entire chain may be of equivalent accuracy to just a few iid samples. The diagnostics, by construction, have a low type I error: If a chain closely resembles iid samples, they will not alert that it is mixing poorly. However, there are no guarantees on type II error: If a chain is mixing poorly, the diagnostic might not alert. Indeed, there are many ways to construct examples where an MCMC procedure undetectably fails: distant modes, Neal’s funnel [Thompson, 2011], extreme ill-conditioning, etc. However, are these realistic stress tests for MCMC methods or merely pathological cases? We do not know.

We propose a new data-driven approach to create a benchmark that estimates how well various MCMC procedures work on real problems. Arguably, algorithms in machine learning and statistics rely on the “workhorses” of either optimization or sampling methods. The world of (non-convex) optimization has already tackled this challenge with the COCO benchmark [Hansen et al., 2016], which contains a test battery of difficult optimization problems. Various approaches are tested to validate if they can optimize the objective function to a target level within a fixed number of function evaluations. Our approach is an analogous system for sampling methods. However, we further improve upon this using flexible (including neural net based) benchmark examples that have been trained to match posteriors found in practice.

In our approach we use a large “data set of data sets” and a diverse “model zoo” to create a representative set of examples. Long MCMC chains are drawn (using NUTS [Hoffman and Gelman, 2014]) from each of these posteriors. Flexible unsupervised models that serve as a *ground-truth* in the benchmarking phase are fit to the chains to construct the benchmark examples.

More concretely, each combination of real data set (e.g., MNIST) and real model (e.g., logistic regression) results in a Markov chain from NUTS. We then fit an unsupervised model (e.g., mixture of Gaussians) to this chain to serve as a *benchmark example distribution*. Once trained, these benchmark example distributions are functionally equivalent to hand-crafted examples such as the toy posterior distributions usually used to benchmark samplers (or such as those in COCO). However, these examples are not hand-crafted but rather are much more representative of real problems. Because it is possible to draw exact (iid) samples from the benchmark example distributions, we now have a ground-truth set of samples to validate the accuracy of the sampling methods.

We derive a variety of metrics that summarize the performance of a sampler for comparing its output to ground-truth iid samples. The ground-truth samples also allow us to assess how well the MCMC diagnostics actually predict estimation performance. In particular, we look at the effective sample size (ESS) because it provides a concrete statement on sample quality [Kass et al., 1998].

The outline of this paper is as follows: In Section 2 we provide some background on MCMC and its diagnostics/performance measures. In Sections 3 and 4 we explain the methodology of the benchmark and its pipeline of five sequential phases. Finally, in Section 5, we present results illustrating the advantages of various samplers and the utility of various MCMC diagnostics.

**Contributions** We summarize the contributions of this work as follows: 1) We provide a new and novel benchmark to describe how well various samplers work on realistic problems. This involves design of fair and sensible metrics to score samplers across problems. This work creates a software system that will serve as a practical tool in algorithm development analogous to ML-comp/CodaLab or COCO. 2) We shed light on how well the common MCMC diagnostics predict the real estimation performance of MCMC methods. We further create a data-driven meta-diagnostic by combining MCMC diagnostics to predict real sampler performance. The code for the system is available at [github.com/bradyneal/sampling-benchmark](https://github.com/bradyneal/sampling-benchmark).

**Related work** The closest existing system is SamplerCompare of Thompson [2011], which tests samplers on a handful of hand-crafted stress-test cases such as Neal’s funnel. However, SamplerCompare is more an R package to aid evaluation than a complete benchmark. A recent piece of work from systems biology [Ballnus et al., 2017] compares various samplers for dynamical systems (i.e., filtering) on a set of hand-crafted ODE systems inspired by biological models.

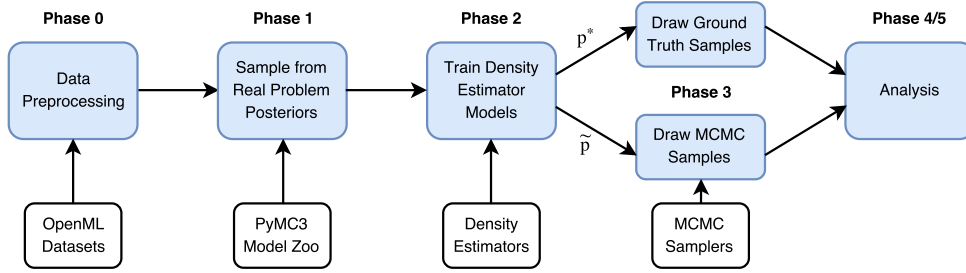


Figure 1: Flowchart illustrating the six phases in our methodology. Phases 0–2 are for creating benchmark examples and are not re-run when new samplers are tested. Phase 2 includes mixture models and modern neural net methods.

## 2 BACKGROUND

The notion of a black box is highly relevant to conceptual understanding of this work. Fundamentally, an MCMC sampler is a system that inputs a black box that computes an unnormalized density  $\tilde{p} \propto p^*$  (and possibly its gradient  $\nabla \log \tilde{p}$ ) and a previous sample  $\mathbf{x}_{t-1} \in \mathbb{R}^D$  in the Markov chain; and outputs another sample  $\mathbf{x}_t \in \mathbb{R}^D$ . Once the Markov chain has converged, these samples are theoretically guaranteed to marginally come from the density  $p^*$ , albeit with temporal correlation. If the previous sample was drawn exactly,  $\mathbf{x}_{t-1} \sim p^*$ , then  $\mathbf{x}_t \sim p^*$  exactly as well. This is a result of *detailed balance*.

By analogy, optimization algorithms take an objective function  $f \in \mathbb{R}^D \rightarrow \mathbb{R}$  (and possibly its gradient  $\nabla f$ ) as a black box and produce points  $\mathbf{x}_t \in \mathbb{R}^D$  that successively minimize  $f$  as much as possible. Just as COCO provides its benchmark objective functions  $f$  as a black box to the optimizers and keeps hidden the true optimum, our benchmark provides the unnormalized density  $\tilde{p}$  as a black box to the samplers. Our benchmark keeps hidden the parameterization of  $\tilde{p}$  needed to efficiently take iid samples from  $p^*$ .

### 2.1 TRADITIONAL MCMC DIAGNOSTICS

Given that we have a ground-truth to evaluate the performance of the various samplers, we can also benchmark the diagnostics by seeing how predictive they are of actual performance. In particular, we consider three diagnostics in this paper: ESS, Gelman-Rubin (GR), and Geweke. ESS aims to estimate how many iid samples have the same estimation performance as the correlated samples found in the MCMC chain. Gelman-Rubin [Gelman and Rubin, 1992] and Geweke [Geweke, 1992] more closely follow a test statistic paradigm than an estimation one. Gelman-Rubin compares the variance within a single chain to variance between chains (independent restarts). This quantity should be close to one

for well-mixing chains and can be very large for poorly performing chains. The Geweke diagnostic uses a single chain and compares the variance between chunks.

The ESS diagnostic is basically a rescaling of the expected square error (i.e., MSE) on estimating the mean in a *single dimension* (marginal) of  $\mathbf{x}$ . ESS is based on the notion that for the marginal  $x_d$ :

$$\begin{aligned} \mathbb{E}_{p^*}[(\hat{\mu}_d - \mu_d)^2] &= \text{Var}_{p^*}[\hat{\mu}_d - \mu_d] + \mathbb{E}_{p^*}[\hat{\mu}_d - \mu_d]^2 \\ &= \text{Var}_{p^*}[x_d]/N, \quad d \in 1:D, \end{aligned} \quad (1)$$

$$\hat{\mu} := \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i, \quad \boldsymbol{\mu} := \mathbb{E}_{p^*}[\mathbf{x}], \quad (2)$$

which utilizes that  $\hat{\mu}_d$  is an unbiased estimate of  $\mu_d$ . We are careful to distinguish expectations and variances with respect to  $p^*$ , where  $\mathbf{x}$  is iid, from  $q$ , where the samples are correlated and drawn from an MCMC method. Naturally, by re-arranging (1), the effective sample size for non-iid samples is:

$$\text{ESS} := \frac{\text{Var}_q[x_d]}{\mathbb{E}_q[(\hat{\mu}_d - \mu_d)^2]} \in \mathbb{R}^+. \quad (3)$$

Unlike (1), this can be estimated without ground-truth samples from  $p^*$ . However, the difficult denominator term is typically estimated using the empirical linear auto-correlation of the Markov chain. This linearity assumption is obviously a potential source of error in the ESS. The fixation in estimating the accuracy of the mean  $\hat{\mu}$  is also a weakness. In Section 4.6, we look at the *real* effective sample size by comparing estimates with the ground-truth samples. It also allows us to look at measures other than simply the fidelity in matching the means ( $\hat{\mu} - \mu$ ), such as variance or shape of the marginals.

## 3 METHODOLOGY

Our benchmark system follows a six phase approach, which we explain at a high level in this section. In Section 4, we provide low-level specifics. A graphical summary of this section is provided in Figure 1.

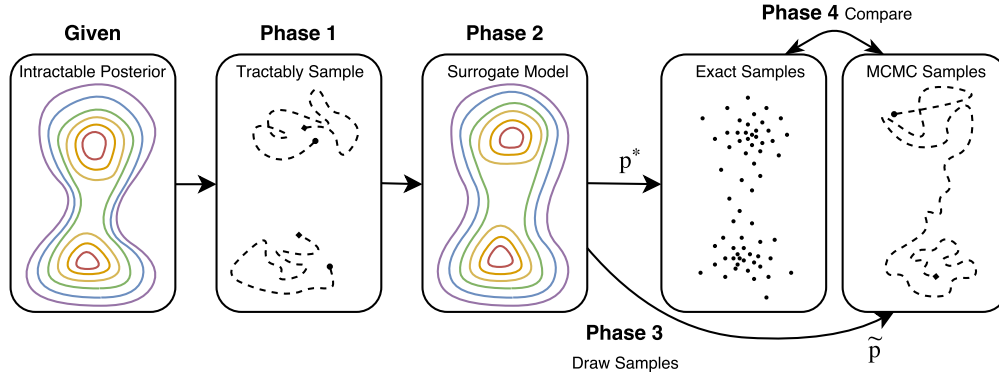


Figure 2: Graphical depiction of Figure 1 on a particular example. We begin with a real posterior from a real problem on the left, which is sampled via a Markov Chain to get samples in phase 1. These are fit to get a similar surrogate posterior in phase 2. MCMC samplers are run on this phase 2 density, but exact samples can also be taken for comparison. Note that the exact posterior, and the real data that produced it, *are not used* for comparing the exact samples and the MCMC samples in phase 4. The phase 2 models are used where toy examples are often used; although it is not the original posterior, it is a far-reaching improvement.

In phase 0, we create a “corpus” of data sets that we refer to as a “data set of data sets.” This is meant to create a realistic sample of problems that a practitioner may encounter “in the wild.” Such an approach was also taken in the AutoML competition [Guyon et al., 2015] and the automated statistician project [Lloyd et al., 2014]. Our approach can be thought of as a form of *meta-learning*.

In phase 1, we use a model zoo to simulate a variety of (Bayesian) models that a practitioner might attempt to apply to a real problem. There are models for regression and classification. Each model/data set pair results in a posterior over a parameter space, which varies in dimensionality depending on the problem. Except in very simple cases (e.g., linear regression), we are not able to obtain samples from these posteriors exactly. We use NUTS, the default sampler in probabilistic programming languages (PyMC3 [Salvatier et al., 2016] and Stan [Carpenter et al., 2016]), because it is generally considered to be a good off-the-shelf sampler—especially when paired with the intelligent initialization and automatic tuning found in these systems. Therefore, by running multiple long chains of NUTS (3–5 chains for 30 minutes each) on the posteriors, we obtain a sufficient approximation and representation for phase 2.

In phase 2, we run various density estimation models to generate benchmark example distributions on the Markov chains from phase 1. We run a separate training procedure on each model/data set pair. These benchmark example distributions serve as surrogates for the real posteriors found in phase 1. Note that the goal is not to replicate the posteriors from phase 1 exactly, but to generate example distributions that are *qualitatively similar* to the real posteriors in phase 1.

This gives us example distributions that are more realistic than the usual hand-crafted toy problems. Nonetheless, we train multiple models and take the one with the highest held-out likelihood on the last 20% of the Markov chain found in phase 1. We use held-out likelihood because it is the most widely accepted generic method of verifying model fidelity. Model checking diagnostics are also run to verify the similarity between the benchmark example distributions (surrogates) and their corresponding Markov chains from the real posteriors (originals).

When selecting models for benchmark example distributions in phase 2, we have the following requirements: 1) The models are flexible enough to closely fit the posteriors found in phase 1, 2) They can serve as a black box, providing an unnormalized density  $\tilde{p}$  (and its gradient) when queried at an arbitrary point  $\mathbf{x}$ , and 3) We can efficiently sample (ground-truth) from them given their parameters (which are hidden from the samplers).

In phase 3, we benchmark a collection of samplers. If someone invents and provides a new sampling algorithm, it is added in phase 3. Phases 0–2 remain fixed as new samplers are submitted to be benchmarked. Each sampler to be benchmarked is run on each of the benchmark example distributions for multiple chains. Each chain is allowed to run for a fixed period of time. The samples from the Markov chains are saved as the phase 3 output.

In phase 4, we take a large number (e.g.,  $\sim 10^5$ ) of exact iid samples from the benchmark example distributions as a ground-truth. The square loss between point estimates (e.g.,  $\hat{\mu}_d$  or  $\hat{\sigma}_d^2$ ) taken from the Markov chains from phase 3 and the point estimates from the exact chains are aggregated. We also compute and store the MCMC diagnostics for each chain.



In phase 5, we aggregate the performance results by looking at the real effective sample size as derived from the square errors in point estimation. We also define transformations of the real effective sample size, which we will refer to as efficiency, normalized effective sample size, and effective sample size deviation. In addition, we perform a meta-analysis using Gaussian process (GP) [Rasmussen and Williams, 2006] regression to predict the real effective sample size given the MCMC diagnostics. This will be useful to practitioners aiming to quantify their confidence in an MCMC-based estimate using the diagnostics available.

We present an example posterior following this pipeline in Figure 2. Note that after the explicit model is fit in phase 2, the data that produced the original posterior is completely *irrelevant* for the rest of the process. Only the surrogate model is used for benchmarking.

## 4 ADDITIONAL DETAILS

In this section we present additional details for the construction of each phase.

### 4.1 PHASE 0: COLLECT DATA SETS

Phase 0 involved downloading 2,200 data sets from `openml.org` to form our data set of data sets. We considered other sources, such as the classic UCI repository, `mldata.org`, and Kaggle, but settled on OpenML because it had the most standardized format and meta-data. Such systems are necessary for automated processing.

The data sets were diverse in that they varied in dimension from 1 to 61,359, sample size from 5 to 7,619,400, and the number of output classes (for classification) from binary to 100.

After downloading, we subjected each data set to some preprocessing to simulate the diverse set of practices a practitioner might follow. Each data set was randomly preprocessed in one of three ways: standardization, robust standardization (using medians and interquartile ranges), or whitening. Categorical variables were represented with one-hot encodings.

### 4.2 PHASE 1: SAMPLE THE MODEL ZOO

For the model zoo, we used all of the standard models (regression and classification) typically used with PyMC3. This includes generalized linear models (GLMs) such as logistic regression, but also atypical GLMs such as robust linear regression (linear regression with Student’s- $t$  noise). In addition to models that are linear in the feature space, we included models that

are linear in a second order transformation of the feature space. We included Gaussian processes with unknown hyper-parameters (e.g., MCMC sampling was done on the unknown hyper-parameters). Bayesian neural networks were also included.

To keep computation time reasonable, we limited the sample size for expensive models (e.g., GPs), and placed some limits on input dimensionality. Where dimensionality needed to be reduced we used PCA [Jolliffe, 1986], as that is the most frequently used method in practice to reduce dimensionality.

### 4.3 PHASE 2: FIT FLEXIBLE SURROGATES

There are three varieties of models that satisfy the three requirements (flexibility, tractable density, and fast exact sampling) for benchmark example densities: mixture models, RNADE [Uria et al., 2013], and Real NVP [Dinh et al., 2016]. In each example, we pick the model with the best held-out likelihood on the last 20% of the chain.

For mixture models, we considered mixture of Gaussians (MoG) with expectation-maximization (EM) [Dempster et al., 1977] and variational MoG. Note that, for simplicity, these models are *not* themselves fit using MCMC. The Bayesian Occam’s razor effect [Jefferys and Berger, 1992] allowed us to simply fix the number of mixture components to 25 in variational MoG. We used five-fold cross-validation to select the number of components in EM MoG. There is no consistent winner between these models; the chosen model is example dependent.

We also tuned the RNADE learning rate and hyper-parameters based on pilot runs. Surprisingly, the mixture models often, but not always, outperformed RNADE on the held-out likelihood. Real NVP based models struggled to achieve competitive test set scores.

These models behave better numerically when trained on standardized data. Care is taken to reverse this standardization in phase 3, so the samplers are forced to attempt to sample from the posterior in its original scale.

### 4.4 PHASE 3: RUN THE SAMPLERS

Phase 3 forms the real “meat” of the benchmark. This is where candidate sampling algorithms are actually run on the benchmark example densities. The list of sampling algorithms is not intended to be exhaustive but rather demonstrate the utility of the benchmark system.

Whether originally designed this way or not, nearly all respected MCMC procedures proceed by proposing a new point using a *proposal distribution*. The new point is then accepted or rejected using a Metropolis-Hastings step. Therefore, the difference between samplers is based

upon their proposal distributions. We provide a preview of the proposals used in Section 5.

Until recently, the most widely used MCMC procedure was *random walk Metropolis*, which uses a Gaussian random walk proposal  $p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t|\mathbf{x}_{t-1}, \Sigma)$ , where  $\Sigma$  is typically diagonal. Modern packages such as PyMC3 allow for automatic tuning of the proposal width  $\Sigma$ , which is critical to achieve good performance. We also consider Cauchy and Laplace distributed proposals.

We include Hamiltonian Monte Carlo (HMC) [Duane et al., 1987] methods, which also utilize gradient information to more efficiently “explore” the space. Recently, the No-U-Turn-Sampler (NUTS) [Hoffman and Gelman, 2014] was introduced as an extension of HMC that automatically adapts some of its tuning parameters in order to attempt high off-the-shelf performance. We include an alternate auxiliary variable method known as *slice sampling* [Neal, 2003], which we apply in a coordinate Gibbs-like fashion.

We also alternate different proposals to form compound proposals. For instance, we consider mixing expensive efficient proposals like NUTS with cheap inefficient proposals like random walk Metropolis.

Finally, we consider an unconventional sampler known as *emcee* [Foreman-Mackey et al., 2013], which is popular in fields such as astrophysics. However, it has not gained much use in machine learning. It works by running multiple “walkers” to explore the space in parallel. Emcee is very fast and can be parallelized, but its efficacy in higher dimensions is somewhat controversial.

**Initialization** The accuracy of MCMC based estimates are a function of two factors: the *burn-in time* and the *mixing time*. Burn-in time, or time until convergence, is how many steps  $k$  are required before  $p(\mathbf{x}_k) \approx p^*$  if  $\mathbf{x}_0 \sim p_0$ , where  $p_0$  is some distribution to initialize the chain. The mixing time, or memory length, is how long it takes to get an independent sample once a chain has converged: how many steps  $k$  are required before  $\text{MI}(\mathbf{x}_k; \mathbf{x}_0) \approx 0$  if  $\mathbf{x}_0 \sim p^*$ . The burn-in time is crucially dependent on the initialization while the mixing time is purely a function of the proposal.

In order to evaluate these two effects separately, we offer two options for initialization: 1) initialize the chain from an exact sample (because we can do that with the benchmark density examples), or 2) initialize from an ADVI [Kucukelbir et al., 2017] fit to the example density. Additionally, most methods benefit from a prior guess at the relative scale of the variables before tuning. We can use the resulting scales from ADVI for this purpose as well. This allows us to *separate the effects* of initializa-

tion and mixing. We use the PyMC3 defaults for these tuning parameters as that is what a practitioner is most likely to use in practice. However, alternate schemes can certainly be used within the benchmark.

#### 4.5 PHASE 4: LOG PERFORMANCE

Each sampler is run for a fixed time limit of 15 minutes of CPU time. We log the performance of the chain along a uniform grid of 100 points in time (i.e., every 9s) to monitor *real* convergence over time. Fair evaluation requires evaluating each sampler with a fixed time budget rather than a fixed number of samples. We expect samplers such as NUTS to be very efficient and high performing on a per-sample basis. However, they require significantly more computation (including gradients) per sample than simpler methods. Therefore, their comparison is not as obvious a-priori. We also log the traditional MCMC diagnostics of each chain.

#### 4.6 PHASE 5: ANALYZE

To summarize the performance of a Markov chain in comparison with ground-truth samples we need to define some evaluation quantities. First, recall that we have  $K$  Markov chains  $\{\mathbf{x}_{1:N_k}\}_{k=1}^K$  for each example  $p^* \in \mathcal{M}$  and sampler  $S \in \mathcal{S}$ .

Each sampler is evaluated on each example separately and can be scored relative to a variety of *estimators*  $\hat{\theta}(\mathbf{x}_{1:N})$ . Analogous to (3), we can score the samples of a Markov chain by the closeness of its mean on a dimension  $d$  to the ground-truth samples:  $\theta = \mathbb{E}[x_d]$  and  $\hat{\theta}(\mathbf{x}_{1:N}) = \frac{1}{N} \sum_{i=1}^N [\mathbf{x}_i]_d$ . We can also consider how close the variance of the Markov chain samples match the ground-truth samples:  $\theta = \text{Var}[x_d]$ . This flexibility is a generalization of ESS. As in (3), we assume the estimators  $\hat{\theta}$  are unbiased, and just as with the sample mean  $\hat{\mu}$ :  $\text{Var}_{p^*}[\hat{\theta}] \propto N^{-1}$ . Furthermore, we assume here that each dimension of the samples  $\mathbf{x}$  has been standardized using the variance of the ground-truth samples, which makes the estimation errors on each dimension  $d$  comparable even when their units differ.

**Real ESS** In analogy to the ESS diagnostic we define the *real ESS* (RESS) based on the estimation error relative to the ground-truth:

$$\text{RESS} := \frac{R}{\text{mean sq. error}} = \frac{RK}{\sum_{k=1}^K (\hat{\theta}_k - \theta)^2} \in \mathbb{R}^+,$$

$$R := \mathbb{E}_{p^*}[(\hat{\theta} - \theta)^2] = N \text{Var}_{p^*}[\hat{\theta}] \in \mathbb{R}^+, \quad (4)$$

where  $K$  is the number of independent MCMC chains and  $R$  is a constant to make RESS comparable across different types of estimators  $\hat{\theta}$ . It also ensures that RESS

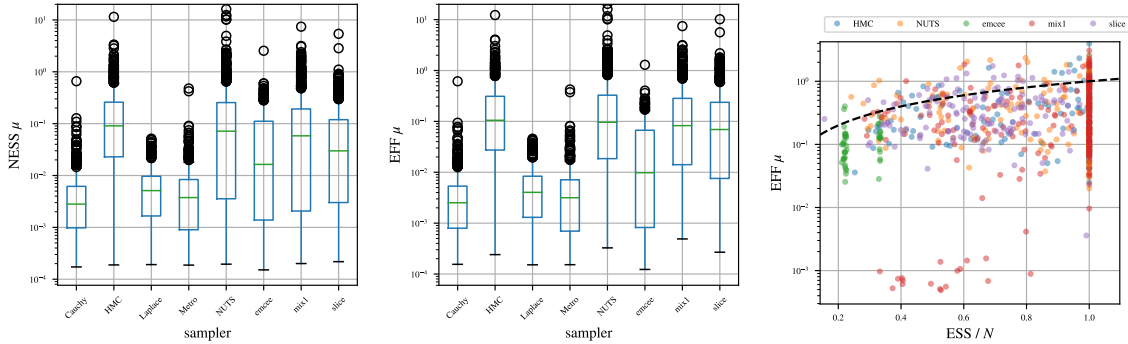


Figure 3: Performance summaries: The box plots demonstrate the distribution on NESS (left) and efficiency (center) conditional on the sampler achieving an RESS of at least 12 to only show the mode where the samplers don’t completely fail. We also show a calibration plot to assess if ESS is a good predictor of efficiency with the diagonal in dashed black. Cauchy and Laplace refer to random walk Metropolis with these corresponding proposals.

tends towards  $N$  when the samples are iid. We do not need the  $\text{Var}[x]$  term from (3) because the samples have been standardized using the ground-truth samples’ scale. If the estimator  $\theta$  in (4) is the mean  $\mu_d$ , then  $R = 1$ . In this case, the RESS measures the exact same expectation (expected square loss) as ESS attempts to estimate. Therefore, if the chain is sufficiently long for accurate estimation of ESS, the two metrics should converge. For variance  $\sigma_d^2$  estimation,  $R = 2$  in large  $N$ .

We also consider the Kolmogorov-Smirnov (KS) distance between the samples and the ground-truth samples as a metric.<sup>1</sup> This also results in a separate metric on each marginal. To match the  $N^{-1}$  convergence assumption of (4) we use  $\sum_{k=1}^K \text{KS}_d(\mathbf{x}_{1:N}^k, p^*)^2$  as the denominator in (4), where  $\text{KS}_d$  signifies the KS distance on the marginal  $x_d$ . By numerically integrating (4) with the Kolmogorov distribution, one finds that  $R = 0.822$ .

RESS is also general in that we can sensibly combine the errors across dimensions by evaluating multivariate estimators  $\hat{\theta} \in \mathbb{R}^D$ :

$$\text{RESS} = \frac{RKD}{\sum_{k=1}^K \|\hat{\theta}_k - \theta\|_2^2} \in \mathbb{R}^+. \quad (5)$$

This assumes that  $\hat{\theta}$  is an unbiased estimator of  $\theta$ . This, like (4), tends towards  $N$  for iid samples.

Because  $p^*$  may be complex, yet cheap to take many (e.g.,  $10^4$ ) iid samples from, we use the ground-truth samples from  $p^*$  to estimate  $\theta$  for use in (4). The error in estimating  $\theta$  is negligible compared to  $\hat{\theta}_k - \theta$ . Likewise, for the KS metric we use a two-sample KS distance between the MCMC samples and ground-truth from  $p^*$ .

<sup>1</sup>Recall that the KS distance between samples  $x_{1:N}$  and a CDF  $F$  is given by  $\max_a |\hat{F}(a) - F(a)|$  where  $\hat{F}$  is the empirical CDF on  $x$ .

**Efficiency** Likewise, it is useful for practitioners to get a ballpark estimate of the *efficiency* of a sampler:

$$\text{EFF} := \frac{\text{RESS}}{N} \in \mathbb{R}^+. \quad (6)$$

If the number of samples per chain  $N$  differs across chains, it is more appropriate to use the harmonic mean of  $N$  than the mean; this ensures that EFF tends towards unity when samples are drawn iid from  $p^*$ . Although EFF is useful, RESS is more appropriate for comparisons between samplers. Thinning can increase EFF without increasing estimation accuracy.

**Normalization** When looking at the distribution of sampler performance across examples it is more appropriate to look at normalized ESS (NESS):

$$\text{NESS} := \frac{\text{RESS}}{\text{median}_{S \in \mathcal{S}} N_S} \in \mathbb{R}^+, \quad (7)$$

where the median is taken across different samplers on the same example. The RESS, when evaluating with a fixed time limit, varies widely across examples. The computational cost of each sample varies greatly between benchmark examples.

**ESS Deviation** In order to evaluate the diagnostics in a meta-analysis, we define the ESS deviation (ESSD) metric, which gives a sense on whether the ESS is biased or a generally poor predictor of estimation accuracy. The ESSD is defined as:

$$\text{ESSD} := \Phi^{-1} \left( \chi_K^2 \text{CDF} \left( \frac{\text{ESS}}{\text{RESS}} K \right) \right) \in \mathbb{R}, \quad (8)$$

where  $\Phi^{-1}(\cdot)$  is the inverse CDF of the standard normal. ESSD has a standard normal distribution (under CLT assumptions) if the estimates are derived ESS iid samples;

Table 1: Quantitative summary on sampler performance. We show the NESS on various estimation tasks (e.g.,  $\mu$  vs  $\sigma^2$ ) averaged over all examples on the left. The right shows the probability of success, i.e., how often  $\text{RESS} \geq 12$ . The first three rows are different proposals for random walk Metropolis. Mix is a compound proposal of NUTS and Gauss. For both NESS and prob. success, higher is better.

sampler	NESS			prob. success		
	KS	$\mu$	$\sigma^2$	KS	$\mu$	$\sigma^2$
Cauchy	.004	.004	.003	.604	.582	.441
Laplace	.007	.004	.006	.566	.547	.439
Gauss	.007	.005	.007	.585	.565	.436
HMC	.061	.151	.106	.580	.604	.531
NUTS	<b>.068</b>	<b>.375</b>	<b>.115</b>	.875	.783	.711
emcee	.016	.038	.025	.389	.489	.379
mix	.067	.164	.113	<b>.911</b>	<b>.825</b>	<b>.715</b>
slice	.044	.078	.070	.745	.703	.643

$\text{ESSD} > 0$  indicates the estimation is higher error than expected from ESS. More precisely, if  $\hat{\theta}$  is derived from  $m$  iid samples then,

$$\hat{\theta} \stackrel{d}{\rightarrow} \mathcal{N}(\theta, \sqrt{R/m}) \implies \sqrt{m/R}(\hat{\theta} - \theta) \sim \mathcal{N}(0, 1)$$

$$\implies \sum_{k=1}^K \frac{m}{R} (\hat{\theta} - \theta)^2 = \frac{m}{\text{RESS}} K \sim \chi_K^2, \quad (9)$$

which implies that  $\text{ESSD} \sim \mathcal{N}(0, 1)$ . Note that (8) is merely a transformation to put the RESS-vs-ESS performance ratio on a standardized scale, which does not cause issues if the central limit theorem (CLT) assumption in (9) does not hold exactly.

**Meta-analysis** In our meta-analysis, we perform a Gaussian process regression to predict ESSD from  $\text{ESS} \in \mathbb{R}^+$ , Gelman-Rubin  $\text{GR} \in [1, \infty)$ , and Geweke  $G \in \mathbb{R}$ . We also include the dimension  $D$  of the sample space  $\mathbf{x}$ . Recall that if ESS is a perfect predictor of MCMC performance, then ESSD will resemble white-noise (i.e., iid standard normal). Given that the scales of diagnostics vary widely, we use  $\log \text{ESS}$ ,  $\log |\text{GR} - 1|$ , and  $\log |G|$  to put them all on a sensible scale.

To assess the regression, we test on a held-out 20% test set of unseen examples (i.e., we do random split on a per example basis) to see if we can predict the ESSD on new unseen benchmark examples from the MCMC diagnostics. We compare performance of the regression to linear regression and an iid normal to see if the features provide any predictive gain. Furthermore, we assess the predictive value of each feature by performing the regression after removing each feature and studying the performance delta.

Table 2: Results of meta-analysis. We show the MSE and log-loss of different models attempting to predict the ESSD for mean estimation on a held-out 20% of unseen examples. The log-loss has the advantage that it is parameterization invariant and provides the same results in ESSD or ESS space. The GP- rows show the results of GP regression without the feature named. GP shows the performance of the GP using all features. We assess the statistical significance of the delta to GP using a pairwise t-test in p.

method	MSE	p	NLL (nats)	p
GP	2.8588	–	0	–
GP-D	<b>2.779(70)</b>	0.0252	<b>-0.0096(97)</b>	0.0504
GP-ESS	3.16(23)	0.0097	0.045(31)	0.0034
GP-G	2.858(1)	0.0198	-0.0001(1)	0.0016
GP-GR	3.17(20)	0.0017	0.045(25)	0.0005
iid	3.30(28)	0.0016	0.067(36)	0.0003
linear	3.03(19)	0.0726	0.027(25)	0.0350

## 5 RESULTS

We first show an overall summary of final performance using NESS at the end of 15 minutes per chain, with  $K = 8$  chains in Table 1. The box plots in Figure 3 provide a sense of the variation. We found the NESS of the samplers to generally be bimodal: either the samples achieve an efficiency above 1%, or they completely fail with an  $\text{RESS} < 1$ . Therefore, in Figure 3 we show the box plots after excluding the complete failures. Inspired by the rule of  $N = 12$  from MacKay [2003], we use an RESS of 12 to threshold failure-vs-success.

Table 1 also provides an overall success probability for each method. Emcee shows the most bimodal performance: while sometimes achieving a high NESS competitive with other advanced methods, it has the lowest success probability. Emcee also has the lowest efficiency of any methods except random walk Metropolis, but emcee makes up for its lack of efficiency with higher per sample speed.

Other results from Figure 3 are unsurprising: NUTS and HMC are the highest performers, despite their higher per sample cost. Slice sampling also makes a “strong showing” with its performance more competitive in the lower dimensional examples. Random walk Metropolis methods generally have an efficiency in the 0.1% to 1% range, while slice sampling and HMC based methods have efficiencies in the ballpark of 2% to 40%, with NUTS showing the highest performance. Emcee seems to vary widely. Note that although the compound proposal (mix) does not substantially increase NESS (over NUTS), when the methods succeed Figure 3, mix increases the chance of success (Table 1).

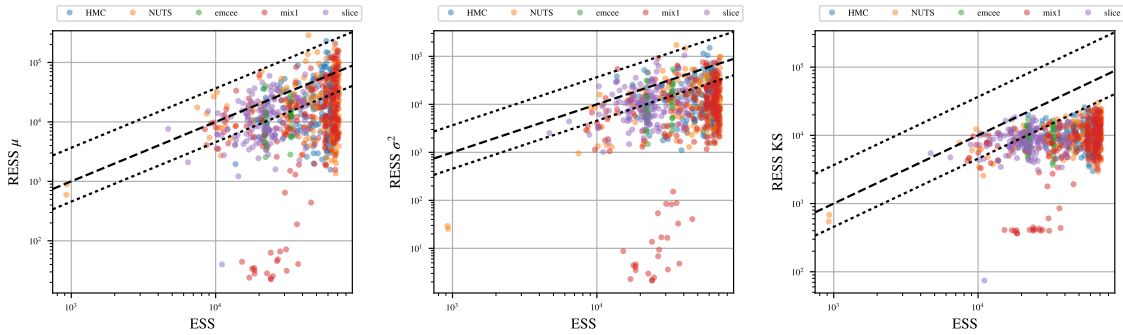


Figure 4: Calibration plots of the ESS diagnostic against real ESS with  $\hat{\theta}$  being the mean (left), variance (center), or KS (right). We show the diagonal for a perfect match in dashed black. In dotted black we show the 95% region for what the observed real ESS would be if the estimates  $\hat{\theta}$  were derived from ESS iid samples. The RESS is below the lower error bar 55% of the time for mean estimation, 68% for variance, and 83% for KS; these would be 2.5% if a chain with  $\text{ESS} = m$  were functionally equivalent to  $m$  iid samples.

We show calibration plots of ESS in Figure 4 and efficiency in Figure 3. The ESS diagnostic is clearly best calibrated for mean estimation, which is not surprising given it was derived for that purpose. However, the ESS diagnostic clearly has an optimistic bias. These results provide caution of ESS.

Finally, we present the results of the meta-analysis to predict ESS deviation. We report the predictive value provided by various features in Table 2 by showing how much performance changes when they are removed. ESS appears very predictive in Figure 4, but the relationship has already largely been accounted for with ESSD (8). In log-loss, the remaining predictive utility of ESS equals that of Gelman-Rubin. Geweke and the dimension  $D$  show no predictive utility. Predictive performance of ESSD goes up when they are removed, which indicates they are of little utility when assessing the validity of a Markov chain. One expects sampling to be more difficult in higher dimensions  $D$ , however this slower mixing may already be evident from ESS and Gelman-Rubin.

## 6 CONCLUSIONS

We have presented a general system to benchmark the real performance of MCMC samplers on realistic problems. The data-driven nature of the benchmark makes it a highly novel development. This benchmark is intended to become a general service that will become as widespread as COCO or MLcomp. Careful attention has been paid to fairly and sensibly derive metrics that compare samplers. This benchmark will evolve with time by including ever more models in phase 1 and more advanced example densities in phase 2. New and more sophisticated samplers can easily be added in phase 3.

## References

- B. Ballnus, S. Hug, K. Hatz, L. Görlitz, J. Hasenauer, and F. J. Theis. Comprehensive benchmarking of Markov chain Monte Carlo methods for dynamical systems. *BMC Systems Biology*, 11(1):63, 2017.
- B. Carpenter, A. Gelman, M. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. A. Brubaker, J. Guo, P. Li, and A. Riddell. Stan: A probabilistic programming language. *Journal of Statistical Software*, 20: 1–37, 2016.
- M. K. Cowles and B. P. Carlin. Markov chain Monte Carlo convergence diagnostics: A comparative review. *Journal of the American Statistical Association*, 91 (434):883–904, 1996.
- A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society (Series B)*, pages 1–38, 1977.
- L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using Real NVP. *arXiv:1605.08803*, 2016.
- S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222, 1987.
- D. Foreman-Mackey, D. W. Hogg, D. Lang, and J. Goodman. Emcee: The MCMC hammer. *Publications of the Astronomical Society of the Pacific*, 125(925):306, 2013.
- A. Gelman and D. B. Rubin. Inference from iterative simulation using multiple sequences. *Statistical Science*, pages 457–472, 1992.
- J. Geweke. Evaluating the accuracy of sampling-

- based approaches to calculating posterior moments. *Bayesian Statistics 4*, 1992.
- I. Guyon, K. Bennett, G. Cawley, H. J. Escalante, S. Escalera, T. K. Ho, N. Macia, B. Ray, M. Saeed, A. Statnikov, et al. Design of the 2015 ChaLearn AutoML challenge. In *Neural Networks (IJCNN), 2015 International Joint Conference on*, pages 1–8. IEEE, 2015.
- N. Hansen, A. Auger, O. Mersmann, T. Tusar, and D. Brockhoff. COCO: A platform for comparing continuous optimizers in a black-box setting. *arXiv:1603.08785*, 2016.
- M. D. Hoffman and A. Gelman. The No-U-Turn Sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(1):1593–1623, 2014.
- W. H. Jefferys and J. O. Berger. Ockham’s razor and Bayesian analysis. *American Scientist*, 80(1):64–72, 1992.
- I. T. Jolliffe. Principal component analysis and factor analysis. In *Principal Component Analysis*, pages 115–128. Springer, 1986.
- R. E. Kass, B. P. Carlin, A. Gelman, and R. M. Neal. Markov chain Monte Carlo in practice: A roundtable discussion. *The American Statistician*, 52(2):93–100, 1998.
- A. Kucukelbir, D. Tran, R. Ranganath, A. Gelman, and D. M. Blei. Automatic differentiation variational inference. *Journal of Machine Learning Research*, 18(14):1–45, 2017.
- J. R. Lloyd, D. K. Duvenaud, R. B. Grosse, J. B. Tenenbaum, and Z. Ghahramani. Automatic construction and natural-language description of nonparametric regression models. In *Association for the Advancement of Artificial Intelligence*, pages 1242–1250, 2014.
- D. J. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.
- R. M. Neal. Slice sampling. *Annals of Statistics*, pages 705–741, 2003.
- C. E. Rasmussen and C. K. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.
- J. Salvatier, T. V. Wiecki, and C. Fonnesbeck. Probabilistic programming in Python using PyMC3. *PeerJ Computer Science*, 2:e55, 2016.
- K. Sharp and M. Rattray. Dense message passing for sparse principal component analysis. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 725–732, 2010.
- M. B. Thompson. Introduction to SamplerCompare. *Journal of Statistical Software*, 43(12):1–10, 2011.
- B. Uria, I. Murray, and H. Larochelle. RNADE: The real-valued neural autoregressive density-estimator. In *Advances in Neural Information Processing Systems*, pages 2175–2183, 2013.

---

# Learning Deep Hidden Nonlinear Dynamics from Aggregate Data

---

Yisen Wang<sup>1,2,3</sup> Bo Dai<sup>1,4</sup> Lingkai Kong<sup>1</sup> Sarah Monazam Erfani<sup>5</sup> James Bailey<sup>5</sup> Hongyuan Zha<sup>1</sup>  
<sup>1</sup>Georgia Tech <sup>2</sup>Tsinghua University <sup>3</sup>Ant Financial Services Group <sup>4</sup>Google Brain <sup>5</sup>University of Melbourne  
wangys14@mails.tsinghua.edu.cn, bohr.dai@gmail.com, zha@gatech.edu

## Abstract

Learning nonlinear dynamics from diffusion data is a challenging problem since the individuals observed may be different at different time points, generally following an aggregate behaviour. Existing work cannot handle the tasks well since they model such dynamics either directly on observations or enforce the availability of complete longitudinal individual-level trajectories. However, in most of the practical applications, these requirements are unrealistic: the evolving dynamics may be too complex to be modeled directly on observations, and individual-level trajectories may not be available due to technical limitations, experimental costs and/or privacy issues. To address these challenges, we formulate a model of diffusion dynamics as the *hidden stochastic process* via the introduction of hidden variables for flexibility, and learn the hidden dynamics directly on *aggregate observations* without any requirement for individual-level trajectories. We propose a dynamic generative model with Wasserstein distance for LEarninG dEep hidden Nonlinear Dynamics (LEGEND) and prove its theoretical guarantees as well. Experiments on a range of synthetic and real-world datasets illustrate that LEGEND has very strong performance compared to state-of-the-art baselines.

## 1 INTRODUCTION

*Diffusion data* is a widespread form of data that involves spatial or status transitions over time, *e.g.*, Brownian movement in physics, cell differentiation or gene expression in biology, molecular motion in chemistry, bird migration in ecology, traffic flows in transportation, population trends in social sciences and so on. Learning

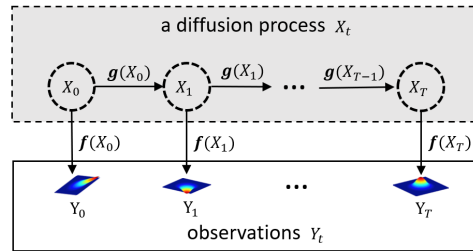


Figure 1: An illustration of the framework which builds dynamics on an auxiliary hidden variable  $X_t$  with a observation function. Observation  $Y_t$  is the aggregate formatted data.

the underlying dynamics which governs the evolution of such data is a fundamental problem. It reveals the nature of the dynamical phenomenon, based on which we can make better future predictions. However, in these areas, complete longitudinal individual-level trajectories (*i.e.*, the tracking of one individual over the entire diffusion process) may often not be available due to technical limitations, experimental costs and/or privacy issues. Rather, one often instead observes a random group of independently sampled individuals from the population, and the observations can contain different individuals at different time points. This is common for catch and release experiments in ecology (*e.g.*, bird migration) where it is difficult to observe a single bird twice (Bartholomew & Bohnsack, 2005), and in biological research where a cell may need to be sacrificed in order for an observation on it to be made (Banks & Potter, 2004). We refer to observations made in these scenarios as *aggregate observations* to differentiate them from the case of individual-level trajectories which provide full information.

Modeling the dynamics on aggregate observations have been investigated recently in (Hashimoto et al., 2016), where a stochastic differential equation (SDE) has been used to capture the transition directly on observations  $Y_t$ . However, its performance degrades when the dynamics

become complex due to their limited expressive ability, as illustrated later in our experiments. Instead of modeling dynamics directly on observations, a hidden variable  $X_t$  can be introduced for modeling complicated dynamics, which can be decomposed into a relatively simple hidden dynamic on  $X_t$  with a complicated observation function. As illustrated in Figure 1,  $Y_t(t \in [0, T])$  is a series of aggregated observations of a diffusion process. We formulate that  $Y_t$  is determined by the hidden dynamic on  $X_t$  and the observation function  $\mathbf{f}(X_t)$ . Existing models such as Hidden Markov Model (HMM) (Eddy, 1996), Kalman Filter (KF) (Harvey, 1990) and Particle Filter (PF) (Djuric et al., 2003) are popular methods with hidden variables. However, these models and their variants (Langford et al., 2009; Hefny et al., 2015) require individual-level trajectories, which may not be available, as was mentioned earlier. It consequently still remains an open issue as to how one can learn the underlying dynamics directly from aggregate observations with a hidden stochastic process, for complicated real-world scenarios.

To address these challenges, we propose a novel framework to incorporate the use of hidden variables into the modeling of diffusion dynamics from evolving distributions (as those approximated from aggregate observations). We bypass the need for likelihood-based estimation of model parameters and posterior estimation of hidden variables by using Wasserstein distance learning. The model we propose is named LEGEND (LEarninG dEep hidden Nonlinear Dynamics) and the main contributions are:

- We propose a framework for learning complicated nonlinear dynamics from aggregate data via a hidden continuous stochastic process.
- We extend Wasserstein learning to likelihood-free and posterior-free estimations of dynamic parameter learning and hidden state inference.
- We theoretically provide a generalization bound and convergence analysis of our framework, which is the first theoretical result as far as we know.
- We empirically demonstrate the effectiveness of our framework for learning nonlinear dynamics from aggregate observations on both synthetic and real-world datasets.

## 2 PROBLEM DEFINITION

We first introduce a continuous model of diffusion dynamics using a stochastic differential equation (SDE), then formally define the tasks of filtering and smoothing based inference, then review the Wasserstein distance objective as one of the distribution metric.

**Hidden Continuous Nonlinear Dynamics.** To characterize the dynamics of observations, we introduce a hidden continuous nonlinear dynamical system as shown in Figure 1, together with a measurement of hidden states. In particular, the hidden state  $X_t \in \mathbb{R}^n$  is the underlying auxiliary variable that cannot be accessed directly, and it follows a SDE:

$$dX_t = \mathbf{g}(X_t)dt + \Sigma^{1/2}d\omega_t, \quad (1)$$

where  $\mathbf{g} : \mathbb{R}^n \rightarrow \mathbb{R}^n$  is a nonlinear deterministic drift function, and  $\omega_t \in \mathbb{R}^n$  is a Brownian motion process with noise covariance  $\Sigma \in \mathbb{R}^n \times \mathbb{R}^n$ . At each time point, the observation  $Y_t$  is written as a measurement of the hidden state  $X_t$ :

$$Y_t = \mathbf{f}(X_t), \quad (2)$$

where  $\mathbf{f} : \mathbb{R}^n \rightarrow \mathbb{R}^m$  is a nonlinear observation function. Together, Eqs. (1) and (2) define the nonlinear dynamics with continuous hidden states.

**Aggregate Observations.** We obtain a collection of independent and identically distributed (i.i.d) samples  $\{y_t^i\}_{i=1}^N$  of  $Y_t$  at some time point  $t$ , that we term aggregate or distributional observations. The observed individuals in previous time observations  $\{y_{t-1}^i\}_{i=1}^N$  are often not identical to those for the current time observations  $\{y_t^i\}_{i=1}^N$ , implying it is not possible to construct the full trajectory of a single individual. However, we can approximate the probability distribution in terms of a finite number of samples as

$$p(Y_t) \approx \frac{1}{N} \sum_{i=1}^N \delta(Y_t - y_t^i). \quad (3)$$

Thus, we can treat these aggregate samples from the same time point together as a distribution which evolves in the dynamic system.

**Problems.** Under this aggregate setting for observations, it is difficult to obtain individual-level trajectories due to technical limitations, experimental costs and/or privacy issues. Therefore, we propose a new framework to learn the nonlinear dynamics for the distributions (as approximated from aggregate observations) without the need for individual-level trajectories. That is, we treat  $\{y_t^i\}_{i=1}^N$  as an empirical approximation to the distribution at time  $t$  and its dynamics is learned via an auxiliary hidden variable  $X_t$ . Once the dynamics are learned, we are faced with two inference tasks:

- 1) *Filtering based inference*: the task is to infer the next future observation  $Y_{T+1}$ , given observations  $\{Y_0, Y_1, \dots, Y_T\}$ .
- 2) *Smoothing based inference*: the task is to infer a missing intermediate observation  $Y_k (0 < k < T)$ , given  $\{Y_0, \dots, Y_{k-1}, Y_{k+1}, \dots, Y_T\}$ .



**Wasserstein Distance Objective.** Following our distribution-based problem definition, the metric on distributions is a key criterion for our objection function, just like the mean squared error (MSE) criterion for regression problems. Among the well-known distribution-based measures, such as Total Variation (TV) distance, Kullback-Leibler (KL) divergence, Jensen-Shannon (JS) divergence, Wasserstein distance has recently been shown to possess more appealing properties for distance measurement of distributions (Arjovsky et al., 2017). We therefore choose to adopt the Wasserstein distance for measuring the discrepancy between the learned distributions and their ground truth.

The definition of Wasserstein-1 distance (also named the Earth-Mover distance (EM)) is:

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y) \sim \gamma} [\|x - y\|], \quad (4)$$

where  $\Pi(\mathbb{P}_r, \mathbb{P}_g)$  denotes the set of all joint distributions  $\gamma(x, y)$  whose marginals are  $\mathbb{P}_r$  and  $\mathbb{P}_g$  respectively. The infimum in (4) is highly intractable. However, Kantorovich-Rubinstein duality (Villani, 2008) shows that

$$W(\mathbb{P}_r, \mathbb{P}_g) = \sup_{\|D\|_L \leq 1} \mathbb{E}_{x \sim \mathbb{P}_r} [D(x)] - \mathbb{E}_{x \sim \mathbb{P}_g} [D(x)], \quad (5)$$

where the supremum is over all 1-Lipschitz functions  $D$ . We can assume a parameterized family of functions  $\{D_w\}_{w \in \mathcal{W}}$  lying in the 1-Lipschitz function space. Therefore, Eq. (5) could be solved by

$$\max_{w \in \mathcal{W}} \mathbb{E}_{x \sim \mathbb{P}_r} [D_w(x)] - \mathbb{E}_{x \sim \mathbb{P}_g} [D_w(x)]. \quad (6)$$

With Wasserstein distance, our objectives for dynamic learning and inference tasks can be unified to minimize the Wasserstein distance between the generated distribution and the target distribution, which will be instantiated in the following Section 3.

### 3 PROPOSED FRAMEWORK

In this section, we first discuss our methodology for parameter learning of dynamics within the LEGEND framework, and then introduce in detail how the framework addresses the filtering and smoothing based inference problems.

#### 3.1 Parameter Learning of Dynamics

In order to efficiently solve the SDE of hidden state  $X_t$ , we adopt an approximate numerical solution called the Euler-Maruyama method (Talay, 1994). Suppose the SDE is defined on  $[0, T]$ , then the Euler-Maruyama approximation to the true solution of SDE is a Markov chain defined

as follows:

$$X_{t+\Delta t} = X_t + \mathbf{g}(X_t)\Delta t + \Sigma^{1/2}\Delta\omega_t, \quad (7)$$

where the interval  $[0, T]$  is partitioned into  $M$  equal sub-intervals of width  $\Delta t = T/M > 0$  and  $\Delta\omega_t$  are independent and identically distributed normal random variables with expected value zero and variance  $\Delta t$ . Correspondingly, observations  $Y_t$  are functions of  $X_t$ :

$$Y_t = \mathbf{f}(X_t). \quad (8)$$

Given a sequence of distributional observations, we need to minimize the Wasserstein distance between the generated distribution and the observed distribution at each time point to learn functions  $\mathbf{f}$  and  $\mathbf{g}$ . The objective function for parameter learning of dynamics is

$$\min_{\mathbf{f}, \mathbf{g}} \sum_t W(\mathbb{P}(Y_t), \mathbb{P}(\mathbf{f}(X_t))), \quad (9)$$

where  $X_t \sim \mathbb{P}(X_t|X_{t-1})$ . The evolving process<sup>1</sup> from  $X_{t-1}$  to  $X_t$  is controlled by function  $\mathbf{g}$  following the SDE in Eq. (7). One common approach for learning  $\mathbf{f}$  and  $\mathbf{g}$  is to calculate the likelihood of  $Y_t$  under distributions, however in many cases, this is intractable. Here, we propose to use generative models to directly generate samples which satisfy the target distribution  $Y_t$ . Following the minimization of the Wasserstein distance between generations and observations, the generative model can eventually learn the dynamics of  $Y_t$ . The specific form of parameterization will be described in Section 4.

#### 3.2 Filtering based Inference

The inference of  $Y_{T+1}$  given observations  $\mathcal{Y}_T = \{Y_0, Y_1, \dots, Y_T\}$  can be solved by:

$$Y_{T+1} = (\mathbf{f} \circ \mathbf{g})(X_T). \quad (10)$$

To achieve this, we need to obtain the hidden state  $X_T$  first, that is, find the posterior probability  $p(X_T|\mathcal{Y}_T)$  of the hidden state conditioned on the entire sequence of observations  $\mathcal{Y}_T$ , which is a filtering problem.

To obtain the posterior of the hidden state, the classical forward algorithm needs to solve one dynamic programming problem per sample, which requires individual-level trajectory for posterior inference. However, for our aggregate observation setting, we alternatively treat the Bayesian inference problem from an optimization perspective following (Dai et al., 2016).

We first briefly introduce the idea of the optimization method, then generalize it to solve our problem. Dai

<sup>1</sup>There may be several  $\Delta t$  intervals between time  $t - 1$  and  $t$

et al. (2016) use a probability  $q(U)$  to approximate the posterior probability  $p(U|V)$  by minimizing

$$\min_{q(U) \in \mathcal{P}} -\langle q(U), \log p(V|U) \rangle + KL(q(U) \parallel p(U)) \quad (11)$$

over the space of all valid densities  $\mathcal{P}$ .  $\langle \cdot \rangle$  is the inner product,  $KL$  is the Kullback-Leibler divergence,  $U$  is the hidden variable and  $V$  is the observation variable. Thus,  $p(V|U)$  is the likelihood of observation and  $p(U)$  is the prior of the hidden variable. Assuming we have the trajectory for a single individual ( $x_t \sim X_t, y_t \sim Y_t$ ), then the posterior probability of filtering is

$$p(x_t|y_{1:t}) = \frac{p(y_t|x_t)p(x_t|y_{1:t-1})}{\int p(y_t|x_t)p(x_t|y_{1:t-1})dx_t}, \quad (12)$$

where  $p(x_t|y_{1:t-1})$  is the propagation probability and  $p(x_t|y_{1:t})$  is the updated probability. Generally,  $p(x_t|y_{1:t-1})$  could be regarded as the prior of  $x_t$  for the updated probability  $p(x_t|y_{1:t})$ . Following the idea of Eq. (11), we can use a probability  $\pi(x_t)$  to approximate the posterior probability  $p(x_t|y_{1:t})$  by recursively optimizing

$$\min_{\pi(x_t) \in \mathcal{P}} -\langle \pi(x_t), \log p(y_t|x_t) \rangle + KL(\pi(x_t) \parallel p(x_t|y_{1:t-1})), \quad (13)$$

where

$$\begin{aligned} p(x_t|y_{1:t-1}) &= \int p(x_t, x_{t-1}|y_{1:t-1})dx_{t-1} \\ &= \int p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1})dx_{t-1} \\ &= \int p(x_t|x_{t-1})\pi(x_{t-1})dx_{t-1}. \end{aligned} \quad (14)$$

In the following, we generalize Eqs. (13) and (14) which were defined on an individual trajectory, to the case for aggregate/distributional data. In Eq. (13), to obtain the optimal solution, we need to maximize the inner product (the first term) and minimize the KL divergence (the second term). We redefine the two terms using Wasserstein distance. For the first term, since maximizing the inner product is equivalent to minimizing the distance between distributions, we replace the inner product with the Wasserstein distance between the distributions of  $\mathbf{f}(\pi_t)$  (generated) and  $Y_t$  (ground truth). For the second term, we replace KL divergence with Wasserstein distance which is a better measurement for distributions and is thus more suitable for aggregate data (Arjovsky et al., 2017). In Eq. (14), the relationship between two consecutive time of hidden variables is replaced by function  $\mathbf{g}$ . We then can generalize Eqs. (13) and (14) to our filtering objective function under aggregate observations:

$$\min \sum_t W(\mathbb{P}(\mathbf{f}(\pi_t)), \mathbb{P}(Y_t)) + W(\mathbb{P}(\pi_t), \mathbb{P}(\mathbf{g}(\pi_{t-1}))), \quad (15)$$

where  $\pi_t \sim \mathbb{P}(X_t|Y_t)$  is our target filtering distribution.

### 3.3 Smoothing based Inference

Smoothing based inference is for predicting the missing intermediate observation  $Y_k (0 < k < T)$  given observations  $\mathcal{Y}_{T \setminus k} = \{Y_0, \dots, Y_{k-1}, Y_{k+1}, \dots, Y_T\}$ . One method (Desbouvieres et al., 2011) to solve this is

$$Y_k = (\mathbf{f} \circ \mathbf{g})(X_{k-1}). \quad (16)$$

To achieve this, we need to obtain the hidden state  $X_{k-1}$  first. This is a smoothing problem which focuses on a hidden state somewhere in the middle of a sequence conditioned on the whole sequence of observations.

Different from the filtering task where the current state is estimated recursively from all past observations, smoothing computes the best state estimates given all available observations from both the past and the future. One well-known and simple approach for smoothing is the forward-backward smoother. During a forward pass the standard filtering algorithm is applied to the observations. Afterwards, on the backward pass, inverse filtering is applied to the same time series of observations. Finally the filtering estimates of both the forward and backward pass are combined into the smoothed estimates (Briers et al., 2010). Since the information from the observation should be incorporated only once into the smoothed estimate, we need to combine the posterior estimate of the forward pass with the prior estimate of the backward pass and vice versa.

Thus, following the idea in the above filtering problem (treating the posterior estimation from a optimization perspective), we first learn a forward estimate of the hidden state  $\pi_t^f$  and also a backward estimate  $\pi_t^b$  using Eq. (15). These then form a weighted Wasserstein barycenter problem (Agueh & Carlier, 2011) whose solution is the posterior of smoothing (Kitagawa, 1994). That is, we can obtain the optimal smoothing result  $\pi_t^s$  by optimizing the Wasserstein distance to the observations and the weighted Wasserstein barycenter:

$$\begin{aligned} \min \sum_t W(\mathbb{P}(\mathbf{f}(\pi_t^s)), \mathbb{P}(Y_t)) \\ + \lambda_1 W(\mathbb{P}(\pi_t^s), \mathbb{P}(\pi_t^f)) \\ + \lambda_2 W(\mathbb{P}(\pi_t^s), \mathbb{P}(\pi_t^b)), \end{aligned} \quad (17)$$

where  $\pi_t^s \sim \mathbb{P}(X_t|Y_T)$  is our target smoothing distribution. And the weights  $\lambda_1$  and  $\lambda_2$  are hyperparameters, which are given intuitively with  $\lambda_1 = t/T$  and  $\lambda_2 = 1 - \lambda_1$  such that smoothing problem becomes filtering problem when  $t = T \rightarrow \lambda_1 = 1$ . Actually, there are other alternatives one could use for the weights, but these basic settings already work well in our experiments.

## 4 MODEL PARAMETERIZATION

As stated above, we adopt Wasserstein distance to measure the difference between distributions, and have defined our objectives accordingly. In this section, we establish a dynamic generative model via neural network parameterization based on Wasserstein distance.

According to the dual formulation of Wasserstein distance Eq. (6), our distribution-based objective of parameter learning of dynamics in Eq. (9) becomes

$$\min_{\mathbf{f}, \mathbf{g}} \sum_t \left( \max_{D_t^1} (\mathbb{E}_{y_t \sim \mathbb{P}(Y_t)} [D_t^1(y_t)]) - \mathbb{E}_{x_t \sim \mathbb{P}(x_t | \mathbf{g}(x_{t-1}))} [D_t^1(\mathbf{f}(x_t))] \right). \quad (18)$$

For the filtering and smoothing tasks, we introduce a new function  $\mathbf{h}$  to characterize the target filtering or smoothing distributions. The filtering objective in Eq. (15) becomes

$$\begin{aligned} \min_{\mathbf{h}} \sum_t \left( \max_{D_t^1} (\mathbb{E}_{y_t \sim \mathbb{P}(Y_t)} [D_t^1(y_t)]) \right. \\ \left. - \mathbb{E}_{\pi_t \sim \mathbb{P}(\mathbf{h}(Y_t))} [D_t^1(\mathbf{f}(\pi_t))] \right) \\ + \max_{D_t^2} (\mathbb{E}_{\pi_t \sim \mathbb{P}(\mathbf{h}(Y_t))} [D_t^2(\pi_t)] \\ - \mathbb{E}_{\pi_{t-1} \sim \mathbb{P}(\mathbf{h}(Y_{t-1}))} [D_t^2(\mathbf{g}(\pi_{t-1}))]). \end{aligned} \quad (19)$$

And the smoothing objective in Eq. (17) becomes

$$\begin{aligned} \min_{\mathbf{h}^s} \sum_t \left( \max_{D_t^1} (\mathbb{E}_{y_t \sim \mathbb{P}(Y_t)} [D_t^1(y_t)]) \right. \\ \left. - \mathbb{E}_{\pi_t^s \sim \mathbb{P}(\mathbf{h}^s(Y_t))} [D_t^1(\mathbf{f}(\pi_t^s))] \right) \\ + \lambda_1 \max_{D_t^2} (\mathbb{E}_{\pi_t^s \sim \mathbb{P}(\mathbf{h}^s(Y_t))} [D_t^2(\pi_t^s)] \\ - \mathbb{E}_{\pi_t^f \sim \mathbb{P}(\mathbf{h}^f(Y_t))} [D_t^2(\pi_t^f)]) \\ + \lambda_2 \max_{D_t^3} (\mathbb{E}_{\pi_t^s \sim \mathbb{P}(\mathbf{h}^s(Y_t))} [D_t^3(\pi_t^s)] \\ - \mathbb{E}_{\pi_t^b \sim \mathbb{P}(\mathbf{h}^b(Y_t))} [D_t^3(\pi_t^b)]). \end{aligned} \quad (20)$$

In traditional implicit generative models, given a random variable  $z$  with a fixed distribution  $p(z)$ , we can pass it through a parametric generator  $G_\theta$  (typically a neural network) which directly generates samples following a certain distribution  $\mathbb{P}_\theta$ . Such design is of high flexibility, as by varying the parameters  $\theta$  of the neural networks, we can change this distribution to any distribution of interest. While in our framework, we need a dynamic generative model to match distributions at each time step which can be regarded as a combination of several Generative Adversarial Networks (GANs) (Goodfellow et al., 2014). Specifically, functions  $\mathbf{f}$ ,  $\mathbf{g}$ ,  $\mathbf{h}$  are all generators (sharing parameters over time) and  $D_t$  is a discriminator at time

$t$ . We formulate functions  $\mathbf{f}$ ,  $\mathbf{g}$  and  $D_t$  as normal feed-forward neural networks<sup>2</sup>:

$$f^l = \sum_k \sigma(w_k^f f^{l-1} + b_k^f), \quad (21)$$

$$g^l = \sum_k \sigma(w_k^g g^{l-1} + b_k^g), \quad (22)$$

$$D_t^l = \sum_k \sigma(w_k^{D_t} D_t^{l-1} + b_k^{D_t}), \quad (23)$$

where  $f^l$ ,  $g^l$ ,  $D_t^l$  are the  $l$ -th layers of the neural networks,  $\sigma$  is the activation function, and  $w_k$ ,  $b_k$  are the neural network parameters. Note the output layer of the discriminator  $D_t$  only has one single neuron to output a scalar value.

As for the function  $\mathbf{h}$  (in both filtering and smoothing), we use a recurrent neural network (RNN) to model it, similar to (Mogren, 2016). For the purposes of simplicity and clarity of exposition, we illustrate the computational process here using a vanilla RNN, whereas the actual recursive unit used in our experiments is LSTM unit (Hochreiter & Schmidhuber, 1997). Given inputs as sequences of observations  $\{Y_0, Y_1, \dots, Y_t, \dots, Y_T\}$  and outputs as filtering or smoothing hidden states  $\{\pi_0, \pi_1, \dots, \pi_t, \dots, \pi_T\}$ , the parameterization function  $\mathbf{h}$  works as follows:

$$s_t = \sigma(AY_t + Bs_{t-1} + b), \quad (24)$$

$$\pi_t = \sigma(Cs_t + b), \quad (25)$$

where  $s_t$  is the memorized history information and  $A, B, C$  are parameter matrices of RNN.

Note that we need to enforce the Lipschitz constraints when solving Wasserstein distance from duality in Eq. (5). To achieve this, we adopt the strategy of gradient penalty in (Gulrajani et al., 2017) to regularize the Wasserstein distance<sup>3</sup>.

For parameter learning of dynamics in Eq. (18), we can obtain the optimal discriminator and gradients by

$$D_t^* = \arg \max_{D_t} (E_{y_t^i} [D_t(y_t^i)] - E_{x_{t-1}^i} [D_t(\mathbf{f} \circ \mathbf{g})(x_{t-1}^i)]) \quad (26)$$

$$g_{\mathbf{f}, \mathbf{g}} = - \sum_t \nabla_{\mathbf{f}, \mathbf{g}} E_{x_{t-1}^i} [D_t^*((\mathbf{f} \circ \mathbf{g})(x_{t-1}^i))], \quad (27)$$

where the gradient of  $\mathbf{g}$  needs to back propagate through the entire chain. In practice, we use gradient decent to update the discriminator. The parameter learning procedure of our model is presented in Algorithm 1. Similarly, we can derive the results for filtering and smoothing from Eq. (19) and (20), respectively.

<sup>2</sup> $\mathbf{g}$  could be several nested  $\mathbf{g}$  due to the  $\Delta t$  in SDE.

<sup>3</sup>We omit this term in our equations for simplicity.

---

**Algorithm 1** Parameter learning of dynamics

---

```

for # training iterations do
  for  $k$  steps do
    Sample  $\{\varepsilon^i\}_{i=1}^N \sim \mathbb{P}(\varepsilon)$ 
    for time  $t$  in  $[0:T]$  do
      Sample  $\{y_t^i\}_{i=1}^N \sim \mathbb{P}(Y_t)$ 
      for  $i=1$  to  $N$  do
         $x_0^i = \varepsilon^i$ ,
         $x_{t+\Delta t}^i = x_t^i + g(x_t^i)\Delta t + \Sigma^{1/2}\mathcal{N}(0, \Delta t)$ 
      end for
    end for
    Update the discriminator  $D_t$  by
     $\nabla_{D_t} \frac{1}{N} \sum_{i=1}^N D_t(y_t^i) - \nabla_{D_t} \frac{1}{N} \sum_{i=1}^N D_t((\mathbf{f} \circ \mathbf{g})(x_{t-1}^i))$ 
    end for
    Update  $\mathbf{f}, \mathbf{g}$  by ascending its stochastic gradient
     $-\sum_t \nabla_{\mathbf{f}, \mathbf{g}} \frac{1}{N} \sum_{i=1}^N D_t((\mathbf{f} \circ \mathbf{g})(x_{t-1}^i))$ 
  end for

```

---

## 5 THEORETICAL ANALYSIS

In this section, we provide a generalization error analysis and a convergence guarantee for our learning framework. Our analysis mainly focuses on the parameter learning component of our method, however, similar results can also be derived for filtering and smoothing based inference. For the purpose of simplicity, we briefly present our main results here and leave detailed theorems and proofs to Appendix A.

**Generalization Error.** We denote  $\mathcal{F}$  and  $\mathcal{G}$  as the function spaces of  $\mathbf{f}$  and  $\mathbf{g}$ , respectively, and the  $\mathcal{D}$  as the function space of the  $\{D_t\}_{t=0}^T$ , and  $\mathbf{g}^{ot}(x, \xi_t) = \underbrace{((I + \mathbf{g}) \circ (I + \mathbf{g}) \circ \dots \circ (I + \mathbf{g}))}_t(x) + \xi_t$  with  $\xi_t \sim \mathcal{N}(0, \Delta t)$ . We define

$$\ell(\mathbf{f}, \mathbf{g}) = \mathbb{E}_{y_{0:T}, x_0 \sim p(x), \xi_{0:T}} \left[ \sum_{t=0}^T \max_{D_t \in \mathcal{D}} \left[ D_t(y_t) - D_t((\mathbf{f} \circ \mathbf{g}^{ot}(x_0, \xi_t))) \right] \right]. \quad (28)$$

**Theorem 1.** *Without loss of generality, we assume in each timestamp the number of the observations is  $N$ . Given the samples  $\mathcal{Y} = \{(y_t^i)_{t=0}^T\}_{i=1}^N$  ( $|\mathcal{Y}|_\infty = C_Y$ ) where  $y_{0:T} = (y_t^i)_{t=0}^T$  are sampled i.i.d. from the underline stochastic processes, and  $\mathcal{X} = \{x_0^i\}_{i=1}^N$ ,  $\Xi = \{\xi_{0:T}^i\}_{i=1}^N$  are also i.i.d. sampled. Assume  $\mathcal{D}$  is a subset of  $k$ -Lipschitz functions and denote the  $\mathfrak{R}(\mathcal{F} \circ \mathcal{G}^{ot})$  as the Rademacher complexity of the function space  $\mathcal{F} \circ \mathcal{G}^{ot}$ . We*

have

$$\frac{1}{T} \ell(\mathbf{f}, \mathbf{g}) \leq \frac{1}{T} \hat{\ell}(\mathbf{f}, \mathbf{g}) + \frac{4kC}{\sqrt{N}} + 4k \frac{\sum_{i=1}^T \mathfrak{R}(\mathcal{F} \circ \mathcal{G}^{ot})}{T}. \quad (29)$$

For the different parametrizations, *i.e.*, different function spaces  $\mathcal{F}$  and  $\mathcal{G}$ , the Rademacher complexity of  $\mathfrak{R}(\mathcal{F} \circ \mathcal{G}^{ot})$  will be different. For example, if we parametrize the  $\mathbf{f}(z) = \sigma(W_f z)$  and  $\mathbf{g}(z, \xi) = I^\top[z, \xi] + \sigma(W_g z)$  as single layer neural networks, where  $\sigma$  satisfies some mild condition (Bartlett et al., 2017), following (Bartlett et al., 2017), we have the  $\mathfrak{R}(\mathcal{F} \circ \mathcal{G}^{ot}) = \tilde{\mathcal{O}}\left(\frac{\sqrt{C_1(W_f)C_2^t(I, W_g)(\sum_{i=1}^t C_3(I, W_g))}}{N}\right)$ , where  $C_1, C_2$  and  $C_3$  are some constants related to the parameters. For the details of the conditions on  $\sigma$  and the exact formulation of the constants, please refer to (Bartlett et al., 2017).

**Convergence Analysis.** Convexity-concavity no longer holds for the objectives in the learning and inference parts, therefore, convergence analysis for convex-concave saddle point problem in (Nemirovski et al., 2009) cannot be directly applied. Inspired by (Dai et al., 2017), we can see that once we obtain  $D_t^*$ , Algorithm 1 can be understood as a special case of stochastic gradient descent for a non-convex problem. Thus, we have the following finite-step convergence guarantee for our framework.

**Theorem 2.** *Assume that the parametrized empirical loss function  $\hat{\ell}(\mathbf{f}, \mathbf{g})$  is  $K$ -Lipschitz and variance of its stochastic gradient is bounded by  $\varsigma^2$ . Let the algorithm run  $M$  iterations with stepsize  $\zeta = \min\{\frac{1}{K}, \frac{C'}{\varsigma\sqrt{M}}\}$  for some  $C' > 0$  and output  $(w_f^1, w_g^1), \dots, (w_f^M, w_g^M)$ . Setting the candidate solution to be  $w = (\hat{w}_f^M, \hat{w}_g^M)$  randomly chosen from  $(w_f^1, w_g^1), \dots, (w_f^M, w_g^M)$  such that  $P(w = w^j) = \frac{2\zeta - K\zeta^2}{\sum_{j=1}^M (2\zeta - K\zeta^2)}$ , then it holds that*

$$\mathbb{E} \left[ \left\| \nabla \hat{\ell}(\hat{\mathbf{f}}_w^M, \hat{\mathbf{g}}_w^M) \right\| \right] \leq \frac{KC^2}{M} + (C' + \frac{C}{C'}) \frac{\varsigma}{\sqrt{M}}, \quad (30)$$

where  $C := \sqrt{2(\hat{\ell}(w_f^1, w_g^1) - \min \hat{\ell}(w_f, w_g))}/K$  represents the distance of the initial solution to the optimal solution.

## 6 EXPERIMENTS

In this section, we evaluate our LEGEND framework on various types of synthetic and real-world datasets.

**Baselines:** We compare our model with two recently proposed methods that learn dynamics directly from aggregate observations — modeling directly on  $Y_t$  using a SDE. The baselines differ from one other in their characterization of the drift term  $\mathbf{g}(X_t)$  of the SDE. The two

baselines considered in our experiments are two representatives from parametric and non-parametric categories: 1) OU (Orstein-Uhlenbeck (Huang et al., 2016)): modeling the drift term using an Orstein-Uhlenbeck process (Gillespie, 1996), which is a stationary Gauss-Markov process with the drift term  $\theta(\mu - x_t)$  ( $\theta, \mu$  are parameters); and 2) NN (Hashimoto et al., 2016): modeling the drift term using a neural network (NN) which is a sum of ramps.

## 6.1 Synthetic Data

We first assess our model on three synthetic datasets generated using the following three diffusion dynamics:

### Synthetic-1:

$$\begin{aligned} x_0 &\sim \mathcal{N}(0, \Sigma_1), \\ x_{t+\Delta t} &= x_t + \frac{1}{4}x_t\Delta t + \mathcal{N}(0, \Sigma_0), \\ y_t &= 2x_t. \end{aligned} \quad (31)$$

### Synthetic-2:

$$\begin{aligned} x_0 &\sim \mathcal{N}(0, \Sigma_2), \\ x_{t+\Delta t} &= x_t + (0.1x_t^2 + 0.5x_t)\Delta t + \mathcal{N}(0, \Sigma_0), \\ y_t &= \exp(x_t). \end{aligned} \quad (32)$$

### Synthetic-3:

$$\begin{aligned} x_0 &\sim \mathcal{U}([-2, 2]), \\ x_{t+\Delta t} &= x_t + (0.5x_t + |x_t|)\Delta t + \mathcal{N}(0, \Sigma_0), \\ y_t &= \log|x_t|. \end{aligned} \quad (33)$$

*Synthetic-1* is a linear dynamic on  $x_t$  with linear measurement  $y_t$  where  $x_0$  are sampled from multivariate normal distributions with covariance matrix  $\Sigma_1$  (diagonal elements are 0.04 and others are 0.032). A nonlinear dependency between  $x_t$  and  $y_t$  is formulated in *Synthetic-2*: a quadratic dynamic on  $x_t$  and an exponential dependency of  $y_t$  on  $x_t$  where  $x_0$  are sampled from multivariate normal distributions with covariance matrix  $\Sigma_2$  (diagonal elements are 0.01 and others are 0.008). In *Synthetic-3*, we test a more complex scenario: highly nonlinear dynamics on  $x_t$  with highly nonlinear measurement  $y_t$  where  $x_0$  are sampled from a uniform distribution  $\mathcal{U}([-2, 2])$ . For each synthesized dynamic, we obtain  $x_t$  like  $\{x_0, x_1, x_2, x_3\}$  every  $5\Delta t$  time following a discretized SDE in Eq. (7), and generate 1000 samples at each time step from  $x_t$  out of which only 500 samples are chosen as observations  $y_t$  like  $\{y_0, y_1, y_2, y_3\}$ . We consider population evolution  $\mathbb{R}^d$  with three different dimensions:  $d = 2$ ,  $d = 5$  and  $d = 10$ . Note that  $\Delta t$  is set to 0.2 for all datasets. The stochastic terms are all sampled from multivariate

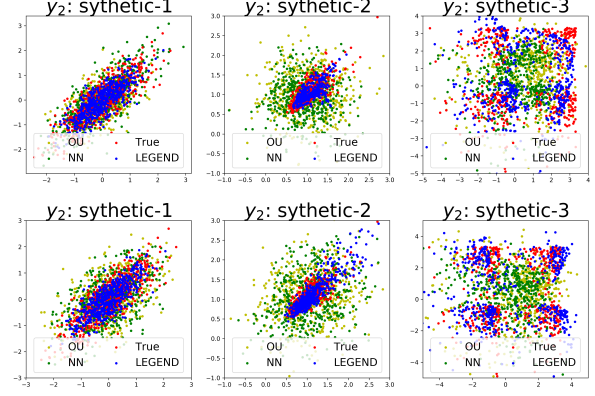


Figure 2: The true and predicted distributions for different models in filtering (top row) and smoothing (bottom row) based inference tasks on 2-dimensional synthetic-1 (left column), synthetic-2 (middle column) and synthetic-3 (right column) datasets.

normal distributions with covariance matrix  $\Sigma_0$  (diagonal elements are 0.0025 and others are 0.002).

Our proposed model along with the baselines are evaluated on two tasks: 1) filtering based inference: given observations  $y_0$  and  $y_1$ , the task is to predict  $y_2$ ; and 2) smoothing based inference: given observations  $y_0, y_1$  and  $y_3$ , the task is to predict  $y_2$ .

**Experimental Setup:** For our LEGEND model, we set  $D, \mathbf{f}, \mathbf{g}$  as a four-layer, two-layer and four-layer feed-forward neural network respectively with ReLU (Glorot et al., 2011) activation function, and set  $\mathbf{h}$  as a one-layer RNN with LSTM unit. In terms of training, we use the Adam optimizer (Kingma & Ba, 2014) with learning rate  $10^{-4}$ ,  $\beta_1 = 0.5$  and  $\beta_2 = 0.9$ . The baselines OU and NN are configured with respect to their settings in the original papers without using pre-training (Hashimoto et al., 2016).

**Results:** We first show the capability of our model for learning low-dimensional ( $d = 2$ ) diffusion dynamics. As visualized in Figure 2, given  $\{y_0\}$  and  $\{y_1\}$ , our model can precisely learn the dynamics and correctly predict  $y_2$  (top row) where a better match was observed between the predictions (blue points) and the ground truth (red points). Note that the dynamics on  $y_t$  become more and more complicated from synthetic-1 to synthetic-3. It can be seen from Figure 2 that our model works well on both simple and complex dynamics, while baselines OU and NN only work well on simple dynamics. Similar results are also observed in the smoothing based inference task, as shown in the bottom row of Figure 2.

We then evaluate our model using Wasserstein error for both low-dimensional ( $d = 2$ ) and high-dimensional ( $d =$

Table 1: The Wasserstein error of different models on synthetic-1/2/3 (Syn-1/2/3), RNA-seq (RNA) and bird migration (Bird) datasets. The best results are highlighted in **bold**.

Data	Target	Task	NN	OU	LEG- END
Syn-1	filtering $y_2$	$d = 2$	0.30	0.29	<b>0.06</b>
		$d = 5$	3.09	2.52	<b>0.06</b>
		$d = 10$	11.19	9.61	<b>0.18</b>
	smoothing $y_2$	$d = 2$	0.70	0.80	<b>0.04</b>
		$d = 5$	3.40	2.92	<b>0.08</b>
		$d = 10$	9.58	8.96	<b>0.12</b>
Syn-2	filtering $y_2$	$d = 2$	0.87	1.36	<b>0.17</b>
		$d = 5$	3.49	4.38	<b>0.47</b>
		$d = 10$	8.55	10.42	<b>1.37</b>
	smoothing $y_2$	$d = 2$	1.62	1.75	<b>0.22</b>
		$d = 5$	5.28	4.17	<b>0.57</b>
		$d = 10$	11.14	9.91	<b>2.84</b>
Syn-3	filtering $y_2$	$d = 2$	8.55	10.79	<b>3.79</b>
		$d = 5$	31.95	35.17	<b>13.21</b>
		$d = 10$	113.21	116.42	<b>42.52</b>
	smoothing $y_2$	$d = 2$	8.43	9.08	<b>2.22</b>
		$d = 5$	28.37	31.26	<b>11.26</b>
		$d = 10$	102.65	109.80	<b>38.73</b>
RNA	Krt8	D7	6.16	9.82	<b>2.31</b>
		D4	27.98	24.54	<b>4.89</b>
	Krt18	D7	6.86	9.80	<b>3.16</b>
		D4	24.75	25.88	<b>4.21</b>
Bird	GrayJay	June	1.9e3	2.5e3	<b>1.2e3</b>
		April	1.5e3	1.1e3	<b>0.3e3</b>

5, 10) diffusion dynamics. Wasserstein error measures the difference between predicted distribution and the true distribution. As reported in Table 1, our model achieves much lower Wasserstein error than the two baselines on all the 3 datasets for 2/5/10-dimensional dynamics. The poor performance of OU and NN may due to the fact that  $y_t$  becomes more and more complicated as dimension increases on all three datasets. The superior performance of our model verifies the importance of hidden variables — they are necessary for the modeling of complex nonlinear dynamics and complex measurements of hidden states.

## 6.2 Real Data: Single-cell RNA-seq

In this section, we evaluate our model on a typical application of distribution based continuous diffusion dynamics in biology: learning the diffusion process where embryonic stem cells differentiate into mature cells (Klein et al., 2015). The cell population begins to differentiate from embryonic stem cells after the removal of LIF (leukemia inhibitory factor) at day 0 (D0). Single-cell RNA-seq measurements (or observations) are sampled at day 0 (D0), day 2 (D2), day 4 (D4), and day 7 (D7). At each time

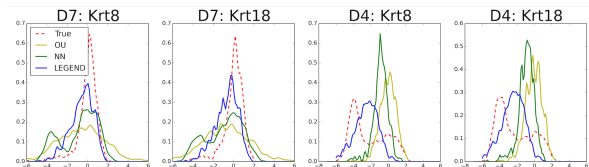


Figure 3: The true and predicted marginal distributions of the differentiating genes at D7 (filtering based inference task) and D4 (smoothing based inference task) for different models.

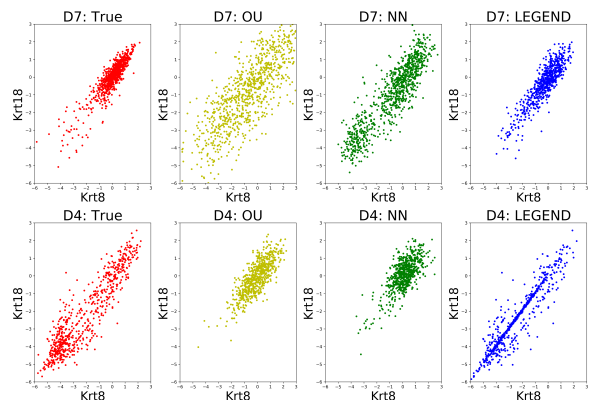


Figure 4: The true (left column) and predicted (right 3 columns) correlations between Krt8 and Krt18 at D7 (top row) and D4 (bottom row). The closer to the true correlation the better the performance.

point, the expression of 24,175 genes for several hundreds cells are measured (933, 303, 683 and 798 cells at D0, D2, D4, and D7 respectively). We focus on the dynamics of cell differentiation for the two main epithelial makers studied in (Klein et al., 2015), *i.e.*, Keratin 8 (Krt8) and Keratin 18 (Krt18). We evaluate two tasks on this data: 1) filtering based inference: predicting the gene expression level at D7 given only the observations at D0 and D4; and 2) smoothing based inference: predicting the gene expression level at D4 given D0, D2 and D7.

**Experimental Setup:** We set  $f$  as a one-layer feed-forward neural network,  $g$  as a three-layer feed-forward neural network and  $h$  as a one-layer RNN with LSTM neurons. For preprocessing, we apply standard normalization procedures (Hicks et al., 2015) to correct for batch effects, and impute missing expression levels using non-negative matrix factorization, similarly as it did in (Hashimoto et al., 2016). The stochastic term  $\Sigma$  in SDE are sampled from multivariate normal distributions with diagonal covariance matrix (diagonal elements are 1). Other configurations and baselines are the same as those in Section 6.1.

**Results:** We first show in Table 1 that compared to other baselines our model achieves the lowest Wasserstein error in both filtering (D7) and smoothing (D4) tasks on both Krt8 and Krt18. This proves that our model is capable of learning the precise differentiation dynamics and the distributions of the two studied gene expressions. We further provide a closer look into the learned distributions of the two genes in Figure 3. As can be seen, the distributions of Krt8 and Krt18 predicted by our model (curves in blue) are much closer to their true distributions (curves in red) at both D4 and D7, as compared to the baseline models. Moreover, our model can effectively identify the correlations between Krt8 and Krt18, as shown in Figure 4. This implies that our model can accurately learn the dynamics even considering the correlational structure of the true dynamics.

### 6.3 Real Data: Bird Migration

We also evaluate our model on another typical application of distribution based diffusion dynamics: bird migration research in ecology. We use the eBird basic dataset (EBD), which gathers large volumes of information on where and when birds occur in the world (Sullivan et al., 2009). We down-sampled EBD to only include the tracking records for the species GrayJay between January, 2017 and June, 2017 (monthly data) at United States where 400 samples are randomly selected as observations for each month. There are again two tasks evaluated here: 1) filtering based inference: we apply our model on the months February and April so as to predict the population at June; 2) smoothing based inference: we apply our model on months February, March and June so as to predict the population at April. The experimental setups are the same as those in Section 6.2.

**Results:** We plot the true and predicted locations (longitude and latitude) of the species GrayJay in Figure 5, and report the Wasserstein error<sup>4</sup> in Table 1. Again, our model achieves the lowest Wasserstein error in both filtering (June) and smoothing (April) based inference tasks. The evolving dynamics of bird migration can be very complicated and extremely difficult to learn, mostly because bird migration could be affected by many irregular factors related to the specific time. Even so, with the introduction of the hidden state variable, our model can predict locations which are close to the ground truth, and with better performance than OU and NN which directly build models on observations. This result demonstrates the advantages of our framework in solving real-world problems involving complicated diffusion dynamics.

<sup>4</sup>Our model could be further improved if considering more complex hidden diffusion process, *e.g.*, jump diffusion process, but the framework is the same to this paper.

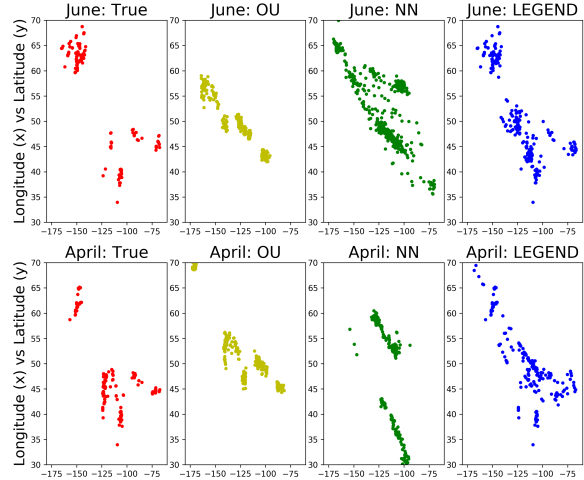


Figure 5: The true (left column) and predicted (right 3 columns) distributions of GrayJay species at month June (filtering based inference task) and April (smoothing based inference task) for different models.

## 7 CONCLUSIONS

In this paper, we formulated a novel technique to learn nonlinear continuous diffusion dynamics from *aggregate observations*. In particular, we showed how one can model dynamics as a *hidden continuous stochastic process*, and proposed a framework that employs a dynamic generative model with Wasserstein distance to learn the evolving dynamics. In addition to deriving solutions for both filtering and smoothing based inference tasks, we also established theoretical guarantees on the generalization and convergence properties of our framework. Through comprehensive experimental evaluation on synthetic and real-world datasets, we demonstrated that our approach has very strong performance compared to state-of-the-art techniques on both filtering and smoothing based inference tasks.

## Acknowledgements

This work is partially supported by NSF IIS-1717916, CIMM-1745382 and China Scholarship Council (CSC).

## References

- Agueh, Martial and Carlier, Guillaume. Barycenters in the wasserstein space. *SIAM Journal on Mathematical Analysis*, 43(2):904–924, 2011.
- Arjovsky, Martin, Chintala, Soumith, and Bottou, Léon. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

- Banks, HT and Potter, Laura K. Probabilistic methods for addressing uncertainty and variability in biological models: Application to a toxicokinetic model. *Mathematical Biosciences*, 192(2):193–225, 2004.
- Bartholomew, Aaron and Bohnsack, James A. A review of catch-and-release angling mortality with implications for no-take reserves. *Reviews in Fish Biology and Fisheries*, 15(1-2):129–154, 2005.
- Bartlett, Peter L, Foster, Dylan J, and Telgarsky, Matus J. Spectrally-normalized margin bounds for neural networks. In *NIPS*, 2017.
- Briers, Mark, Doucet, Arnaud, and Maskell, Simon. Smoothing algorithms for state–space models. *Annals of the Institute of Statistical Mathematics*, 62(1):61, 2010.
- Dai, Bo, He, Niao, Dai, Hanjun, and Song, Le. Provable bayesian inference via particle mirror descent. In *AISTATS*, 2016.
- Dai, Bo, Shaw, Albert, Li, Lihong, Xiao, Lin, He, Niao, Chen, Jianshu, and Song, Le. Smoothed dual embedding control. *arXiv preprint arXiv:1712.10285*, 2017.
- Desbouvries, François, Petetin, Yohan, and Ait-El-Fquih, Boujemaa. Direct, prediction- and smoothing-based kalman and particle filter algorithms. *Signal Processing*, 91(8):2064–2077, 2011.
- Djuric, Petar M, Kotecha, Jayesh H, Zhang, Jianqui, Huang, Yufei, Ghirmai, Tadesse, Bugallo, Mónica F, and Miguez, Joaquin. Particle filtering. *IEEE Signal Processing Magazine*, 20(5):19–38, 2003.
- Eddy, Sean R. Hidden markov models. *Current Opinion in Structural Biology*, 6(3):361–365, 1996.
- Ghadimi, Saeed and Lan, Guanghui. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.
- Gillespie, Daniel T. Exact numerical simulation of the ornstein-uhlenbeck process and its integral. *Physical Review E*, 54(2):2084, 1996.
- Glorot, Xavier, Bordes, Antoine, and Bengio, Yoshua. Deep sparse rectifier neural networks. In *AISTATS*, 2011.
- Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and Bengio, Yoshua. Generative adversarial nets. In *NIPS*, 2014.
- Gulrajani, Ishaan, Ahmed, Faruk, Arjovsky, Martin, Dumoulin, Vincent, and Courville, Aaron C. Improved training of wasserstein gans. In *NIPS*, 2017.
- Harvey, Andrew C. *Forecasting, structural time series models and the Kalman filter*. Cambridge University Press, 1990.
- Hashimoto, Tatsunori, Gifford, David, and Jaakkola, Tommi. Learning population-level diffusions with generative rnns. In *ICML*, 2016.
- Hefny, Ahmed, Downey, Carlton, and Gordon, Geoffrey J. Supervised learning for dynamical system learning. In *NIPS*, 2015.
- Hicks, Stephanie C, Teng, Mingxiang, and Irizarry, Rafael A. On the widespread and critical impact of systematic bias and batch effects in single-cell rna-seq data. *BioRxiv*, 2015.
- Hochreiter, Sepp and Schmidhuber, Jürgen. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.
- Huang, Gang, Jansen, HM, Mandjes, Michel, Spreij, Peter, and De Turck, Koen. Markov-modulated ornstein-uhlenbeck processes. *Advances in Applied Probability*, 48(1):235–254, 2016.
- Kingma, Diederik P and Ba, Jimmy. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- Kitagawa, Genshiro. The two-filter formula for smoothing and an implementation of the gaussian-sum smoother. *Annals of the Institute of Statistical Mathematics*, 46(4):605–623, 1994.
- Klein, Allon M, Mazutis, Linas, Akartuna, Ilke, Tallapragada, Naren, Veres, Adrian, Li, Victor, Peshkin, Leonid, Weitz, David A, and Kirschner, Marc W. Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. *Cell*, 161(5):1187–1201, 2015.
- Langford, John, Salakhutdinov, Ruslan, and Zhang, Tong. Learning nonlinear dynamic models. In *ICML*, 2009.
- Mogren, Olof. C-rnn-gan: Continuous recurrent neural networks with adversarial training. *arXiv preprint arXiv:1611.09904*, 2016.
- Nemirovski, Arkadi, Juditsky, Anatoli, Lan, Guanghui, and Shapiro, Alexander. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.
- Sullivan, Brian L, Wood, Christopher L, Iliff, Marshall J, Bonney, Rick E, Fink, Daniel, and Kelling, Steve. ebird: A citizen-based bird observation network in the biological sciences. *Biological Conservation*, 142(10):2282–2292, 2009.
- Talay, Denis. Numerical solution of stochastic differential equations. 1994.
- Villani, Cédric. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.



---

# Revisiting differentially private linear regression: optimal and adaptive prediction & estimation in unbounded domain

---

Yu-Xiang Wang \*

University of California, Santa Barbara, CA  
Amazon AWS AI, Palo Alto, CA

## Abstract

We revisit the problem of linear regression under a differential privacy constraint. By consolidating existing pieces in the literature, we clarify the correct dependence of the feature, label and coefficient domains in the optimization error and estimation error, hence revealing the delicate price of differential privacy in statistical estimation and statistical learning. Moreover, we propose simple modifications of two existing DP algorithms: (a) posterior sampling, (b) sufficient statistics perturbation, and show that they can be *upgraded* into adaptive algorithms that are able to exploit data-dependent quantities and behave nearly optimally *for every instance*. Extensive experiments are conducted on both simulated data and real data, which conclude that both ADAOPS and ADASSP outperform the existing techniques on nearly all 36 data sets that we test on.

## 1 INTRODUCTION

Linear regression is one of the oldest tools for data analysis (Galton, 1886) and it remains one of the most commonly-used as of today (Draper & Smith, 2014), especially in social sciences (Agresti & Finlay, 1997), econometrics (Greene, 2003) and medical research (Armitage et al., 2008). Moreover, many nonlinear models are either intrinsically linear in certain function spaces, e.g., kernels methods, dynamical systems, or can be reduced to solving a sequence of linear regressions, e.g., iterative reweighted least square for generalized Linear models, gradient boosting for additive models and so on (see Friedman et al., 2001, for a detailed review).

In order to apply linear regression to sensitive data such as those in social sciences and medical studies, it is of-

ten needed to do so such that the privacy of individuals in the data set is protected. Differential privacy (Dwork et al., 2006b) is a commonly-accepted criterion that provides provable protection against identification and is resilient to arbitrary auxiliary information that might be available to attackers. In this paper, we focus on linear regression with  $(\epsilon, \delta)$ -differentially privacy (Dwork et al., 2006a).

**Isn't it a solved problem?** It might be a bit surprising why this is still a problem, since several general frameworks of differential privacy have been proposed that cover linear regression. Specifically, in the agnostic setting (without a data model), linear regression is a special case of differentially private empirical risk minimization (ERM), and its theoretical properties have been quite well-understood in a sense that the minimax lower bounds are known (Bassily et al., 2014) and a number of algorithms (Chaudhuri et al., 2011; Kifer et al., 2012) have been shown to match the lower bounds under various assumptions. In the statistical estimation setting where we assume the data is generated from a linear Gaussian model, linear regression is covered by the sufficient statistics perturbation approach for exponential family models (Dwork & Smith, 2010; Foulds et al., 2016), propose-test-release framework (Dwork & Lei, 2009) as well as the the subsample-and-aggregate framework (Smith, 2008), with all three approaches achieving the asymptotic efficiency in the fixed dimension ( $d = O(1)$ ), large sample ( $n \rightarrow \infty$ ) regime.

Despite these theoretical advances, very few empirical evaluations of these algorithms were conducted and we are not aware of a commonly-accepted best practice. Practitioners are often left puzzled about which algorithm to use for the specific data set they have. The nature of differential privacy often requires them to set parameters of the algorithm (e.g., how much noise to add) according to the diameter of the parameter domain, as well as properties of a hypothetical worst-case data set, which often leads to

---

Corresponding email: yuxiangw@cs.ucsb.edu

an inefficient use of their valuable data.

The main contribution of this paper is threefold:

1. We consolidated many bits and pieces from the literature and clarified the price of differential privacy in statistical estimation and statistical learning.
2. We carefully analyzed One Posterior Sample (OPS) and Sufficient Statistics Perturbation (SSP) for linear regression and proposed simple modifications of them into adaptive versions: ADAOPS and ADASSP. Both work near optimally for every problem instance without any hyperparameter tuning.
3. We conducted extensive real data experiments to benchmark existing techniques and concluded that the proposed techniques give rise to the more favorable privacy-utility tradeoff relative to existing methods.

**Outline of this paper.** In Section 2 we will describe the problem setup and explain differential privacy. In Section 3, we will survey the literature and discuss existing algorithms. Then we will propose and analyze our new method ADASSP and ADAOPS in Section 4 and conclude the paper with experiments in Section 5.

## 2 NOTATIONS AND SETUP

Throughout the paper we will use  $X \in \mathbb{R}^{n \times d}$  and  $\mathbf{y} \in \mathbb{R}^n$  to denote the design matrix and response vector. These are collections of data points  $(x_1, y_1), \dots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$ . We use  $\|\cdot\|$  to denote Euclidean norm for vector inputs,  $\ell_2$ -operator norm for matrix inputs. In addition, for set inputs,  $\|\cdot\|$  denotes the radius of the smallest Euclidean ball that contains the set. For example,  $\|\mathcal{Y}\| = \sup_{y \in \mathcal{Y}} |y|$  and  $\|\mathcal{X}\| = \sup_{x \in \mathcal{X}} \|x\|$ . Let  $\Theta$  be the domain of coefficients. Our results do not require  $\Theta$  to be compact but existing approaches often depend on  $\|\Theta\|$ .  $\lesssim$  and  $\gtrsim$  denote greater than or smaller to up to a universal multiplicative constant, which is the same as the big  $O(\cdot)$  and the big  $\Omega(\cdot)$ .  $\tilde{O}(\cdot)$  hides at most a logarithmic term.  $\prec$  and  $\succ$  denote the standard semidefinite ordering of positive semi-definite (psd) matrices.  $\cdot \vee \cdot$  and  $\cdot \wedge \cdot$  denote the bigger or smaller of the two inputs.

We now define a few data dependent quantities. We use  $\lambda_{\min}(X^T X)$  (abbrv.  $\lambda_{\min}$ ) to denote the smallest eigenvalue of  $X^T X$ , and to make the implicit dependence in  $d$  and  $n$  clear from this quantity, we define  $\alpha := \lambda_{\min} \frac{d}{n \|\mathcal{X}\|^2}$ . One can think of  $\alpha$  as a normalized smallest eigenvalue of  $X^T X$  such that  $0 \leq \alpha \leq 1$ . Also,  $1/\alpha$  is closely related to the condition number of  $X^T X$ .

Define the least square solution  $\theta^* = (X^T X)^\dagger X^T \mathbf{y}$ . It is the optimal solution to  $\min_{\theta} \frac{1}{2} \|\mathbf{y} - X\theta\|^2 =: F(\theta)$ . Similarly, we use  $\theta_\lambda^* = (X^T X + \lambda I)^{-1} X^T \mathbf{y}$  denotes the optimal solution to the ridge regression objective  $F_\lambda(\theta) = F(\theta) + \lambda \|\theta\|^2$ .

In addition, we denote the global Lipschitz constant of  $F$  as  $L^* := \|\mathcal{X}\|^2 \|\Theta\| + \|\mathcal{X}\| \|\mathcal{Y}\|$  and data-dependent local Lipschitz constant at  $\theta^*$  as  $L := \|\mathcal{X}\|^2 \|\theta^*\| + \|\mathcal{X}\| \|\mathcal{Y}\|$ . Note that when  $\Theta = \mathbb{R}^d$ ,  $L^* = \infty$ , but  $L$  will remain finite for every given data set.

**Metric of success.** We measure the performance of an estimator  $\hat{\theta}$  in two ways.

First, we consider the optimization error  $F(\hat{\theta}) - F(\theta^*)$  in expectation or with probability  $1 - \rho$ . This is related to the prediction accuracy in the distribution-free statistical learning setting.

Second, we consider how well the coefficients can be estimated under the linear Gaussian model:

$$\mathbf{y} = X\theta_0 + \mathcal{N}(0, \sigma^2 I_n)$$

in terms of  $\mathbb{E}[\|\hat{\theta} - \theta_0\|^2]$  or in some cases  $\mathbb{E}[\|\hat{\theta} - \theta_0\|^2 | E]$  where  $E$  is a high probability event.

The optimal error in either case will depend on the specific design matrix  $X$ , optimal solution  $\theta^*$ , the data domain  $\mathcal{X}, \mathcal{Y}$ , the parameter domain  $\Theta$  as well as  $\theta_0, \sigma^2$  in the statistical estimation setting.

**Differential privacy.** We will focus on estimators that are differential private, as defined below.

**Definition 1** (Differential privacy (Dwork et al., 2006b)). *We say a randomized algorithm  $\mathcal{A}$  satisfies  $(\epsilon, \delta)$ -DP if for all fixed data set  $(X, \mathbf{y})$  and data set  $(X', \mathbf{y}')$  that can be constructed by adding or removing one row  $(x, y)$  from  $(X, \mathbf{y})$ , and for any measurable set  $\mathcal{S}$  over the probability of the algorithm*

$$\mathbb{P}(\mathcal{A}((X, \mathbf{y})) \in \mathcal{S}) \leq e^\epsilon \mathbb{P}(\mathcal{A}((X', \mathbf{y}')) \in \mathcal{S}) + \delta,$$

Parameter  $\epsilon$  represents the amount of privacy loss from running the algorithm and  $\delta$  denotes a small probability of failure. These are user-specified targets to achieve and the differential privacy guarantee is considered meaningful if  $\epsilon \leq 1$  and  $\delta \ll 1/n$  (see, e.g., Section 2.3.3 of Dwork et al., 2014a, for a comprehensive review).

**The pursuit for adaptive estimators.** Another important design feature that we will mention repeatedly in this paper is *adaptivity*. We call an estimator  $\hat{\theta}$  *adaptive* if it behaves optimally simultaneously for a wide range of parameter choices. Being adaptive is of great practical

relevance because we do not need to specify the class of problems or worry about whether our specification is wrong (see examples of adaptive estimators in e.g., [Donoho, 1995](#); [Birgé & Massart, 2001](#)). *Adaptivity* is particularly important for differentially private data analysis because often we need to decide the amount of noise to add by the size of the domain. For example, an adaptive algorithm will not rely on conservative upper bounds of  $\theta_0$ , or a worst case  $\lambda_{\min}$  (which would be 0 on any  $\mathcal{X}$ ), and it can take advantage of favorable properties when they exist in the data set. We want to design an estimator that does not take these parameters as inputs and behave nearly optimally for every fixed data set  $X \in \mathcal{X}^n, \mathbf{y} \in \mathcal{Y}$  under a variety of configuration of  $\|\mathcal{X}\|, \|\mathcal{Y}\|, \|\Theta\|$ .

### 3 A SURVEY OF PRIOR WORK

In this section, we summarize existing theoretical results in linear regression with and without differential privacy constraints. We will start with lower bounds.

#### 3.1 Information-theoretic lower bounds

**Lower bounds under linear Gaussian model.** Under the statistical assumption of linear Gaussian model  $\mathbf{y} = X\theta_0 + \mathcal{N}(0, \sigma^2)$ , the minimax risk for both estimation and prediction are crisply characterized for each fixed design matrix  $X$ :

$$\inf_{\hat{\theta}} \sup_{\theta_0 \in \mathbb{R}^d} \mathbb{E}[F(\hat{\theta}) - F(\theta_0)|X] = \frac{d\sigma^2}{2}, \quad (1)$$

and if we further assume that  $n \geq d$  and  $X^T X$  is invertible (for identifiability), then

$$\inf_{\hat{\theta}} \sup_{\theta_0 \in \mathbb{R}^d} \mathbb{E}[\|\hat{\theta} - \theta_0\|_2^2|X] = \sigma^2 \text{tr}[(X^T X)^{-1}]. \quad (2)$$

In the above setup,  $\hat{\theta}$  is any measurable function of  $\hat{\mathbf{y}}$  (note that  $X$  is fixed). These are classic results that can be found in standard statistical decision theory textbooks (See, e.g., [Wasserman, 2013](#), Chapter 13).

Under the same assumptions, the Cramer-Rao lower bound mandates that the covariance matrix of any unbiased estimator  $\hat{\theta}$  of  $\theta_0$  to obey that

$$\text{Cov}(\hat{\theta}) \succ \sigma^2 (X^T X)^{-1}. \quad (3)$$

This bound applies to every problem instance separately and also implies a sharp lower bound on the prediction variance on every data point  $x$ . More precisely,  $\text{Var}(\hat{\theta}^T x) \geq \sigma^2 x^T (X^T X)^{-1} x$  for any  $x$ .

Minimax risk (1), (2) and the Cramer-Rao lower bound (3) are simultaneously attained by  $\theta^*$ .

**Statistical learning lower bounds.** Perhaps much less well-known, linear regression is also thoroughly studied in the distribution-free statistical learning setting, where the only assumption is that the data are drawn iid from some unknown distribution  $\mathcal{P}$  defined on some compact domain  $\mathcal{X} \times \mathcal{Y}$ . Specifically, let the risk ( $\mathbb{E}[\text{loss}]$ ) be

$$R(\theta) = \mathbb{E}_{(x,y) \sim \mathcal{P}} \left[ \frac{1}{2} (x^T \theta - y)^2 \right] = \frac{1}{n} \mathbb{E}_{(X,\mathbf{y}) \sim \mathcal{P}^n} [F(\theta)].$$

[Shamir \(2015\)](#) showed that when  $\Theta, \mathcal{X}$  are  $\mathcal{Y}$  are Euclidean balls,

$$\begin{aligned} & \inf_{\hat{\theta}} \sup_{\mathcal{P}} \left[ \mathbb{E}[n \cdot R(\hat{\theta})] - \inf_{\theta \in \Theta} [n \cdot R(\theta)] \right] \\ & \gtrsim \min\{n\|\mathcal{Y}\|^2, \|\Theta\|^2\|\mathcal{X}\|^2 + d\|\mathcal{Y}\|^2, \sqrt{n}\|\Theta\|\|\mathcal{X}\|\|\mathcal{Y}\|\}. \end{aligned} \quad (4)$$

where  $\hat{\theta}$  be any measurable function of the data set  $X, \mathbf{y}$  to  $\Theta$  and the expectation is taken over the data generating distribution  $X, \mathbf{y} \sim \mathcal{P}^n$ . Note that to be compatible to other bounds that appear in this paper, we multiplied the  $R(\cdot)$  by a factor of  $n$ . Informally, one can think of  $\|\mathcal{Y}\|$  as  $\sigma$  in (1) so both terms depend on  $d\sigma^2$  (or  $d\|\mathcal{Y}\|^2$ ), but the dependence on  $\|\Theta\|\|\mathcal{X}\|$  is new for the distribution-free setting.

[Koren & Levy \(2015\)](#) later showed that this lower bound is matched up to a constant by Ridge Regression with  $\lambda = 1$  and both [Koren & Levy \(2015\)](#) and [Shamir \(2015\)](#) conjecture that ERM without additional regularization should attain the lower bound (4). If the conjecture is true, then the unconstrained OLS is simultaneously optimal for all distributions supported on the smallest ball that contains all data points in  $X, \mathbf{y}$  for any  $\Theta$  being an  $\ell_2$  ball with radius larger than  $\|\theta^*\|$ .

**Lower bounds with  $(\epsilon, \delta)$ -privacy constraints.** Suppose that we further require  $\hat{\theta}$  to be  $(\epsilon, \delta)$ -differentially private, then there is an additional price to pay in terms of how accurately we can approximate the ERM solution. Specifically, the lower bounds for the *empirical* excess risk for differentially private ERM problem in ([Bassily et al., 2014](#)) implies that for  $\delta < 1/n$  and sufficiently large  $n$ :

1. There exists a triplet of  $(\mathcal{X}, \mathcal{Y}, \Theta) \subset \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}^d$ , such that

$$\begin{aligned} & \inf_{\hat{\theta} \text{ is } (\epsilon, \delta)\text{-DP}} \sup_{X \in \mathcal{X}^n, \mathbf{y} \in \mathcal{Y}^n} \left[ F(\hat{\theta}) - \inf_{\theta \in \Theta} F(\theta) \right] \\ & \gtrsim \min\{n\|\mathcal{Y}\|^2, \frac{\sqrt{d}(\|\mathcal{X}\|^2\|\Theta\|^2 + \|\mathcal{X}\|\|\Theta\|\|\mathcal{Y}\|)}{\epsilon}\}. \end{aligned} \quad (5)$$

2. Consider the class of data set  $\mathcal{S}$  where all data sets  $X \in \mathcal{S} \subset \mathcal{X}^n$  obeys that the inverse condition number  $\alpha \geq \alpha^* \geq \frac{d^{1.5}(\|\mathcal{X}\|\|\Theta\| + \|\mathcal{Y}\|)}{n\|\mathcal{X}\|\|\Theta\|\epsilon}$ . There exists a

<sup>1</sup>This requires  $\lambda_{\min} \geq \sqrt{dL}/\epsilon$  for all data sets  $X$ .

triplet of  $(\mathcal{X}, \mathcal{Y}, \Theta) \subset \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}^d$  such that

$$\hat{\theta} \text{ is } (\epsilon, \delta)\text{-DP } X \in \mathcal{S}, \mathbf{y} \in \mathcal{Y}^n \left[ F(\hat{\theta}) - \inf_{\theta \in \Theta} F(\theta) \right] \quad (6)$$

$$\gtrsim \min \left\{ n \|\mathcal{Y}\|^2, \frac{d^2 (\|\mathcal{X}\| \|\Theta\| + \|\mathcal{Y}\|)^2}{n \alpha^* \epsilon^2} \right\}.$$

These bounds are attained by a number of algorithms, which we will go over in Section 3.2.

Comparing to the non-private minimax rates on prediction accuracy, the bounds look different in several aspects. First, neither rate for prediction error in (1) or (4) depends on whether the design matrix  $X$  is well-conditioned or not, while  $\alpha^*$  appears explicitly in (6). Secondly, the dependence on  $\|\Theta\|, \|\mathcal{X}\|, \|\mathcal{Y}\|, d, n$  are different, which makes it hard to tell whether the optimization error lower bound due to privacy requirement is limiting. One may ask the following question:

When is privacy *for free* in statistical learning?

Specifically, what is the smallest  $\epsilon$  such that an  $(\epsilon, \delta)$ -DP algorithm matches the minimax rate in (4)? The answer really depends on the relative scale of  $\|\mathcal{X}\| \|\Theta\|$  and  $\|\mathcal{Y}\|$  and that of  $n, d$ . When  $\|\mathcal{X}\| \|\Theta\| \asymp \|\mathcal{Y}\|$ , (5) says that  $(\epsilon, \delta)$ -DP algorithms can achieve the nonconvex minimax rate provided that  $\epsilon \gtrsim \min \left\{ \frac{1}{\sqrt{d}} \sqrt{\frac{d}{n}}, \sqrt{\frac{d^2}{n^{1.5} \alpha^*}} \sqrt{\frac{d}{n \alpha^*}} \right\}$ . On the other hand, if  $\|\mathcal{X}\| \|\Theta\| \asymp \sqrt{d} \|\mathcal{Y}\|^2$  and  $n > d$ , then we need  $\epsilon \gtrsim \min \left\{ \sqrt{d} \sqrt{\frac{d^{3/2}}{n}}, \frac{d}{\sqrt{n \alpha^*}} \sqrt{\frac{d^{3/2}}{n \alpha^*}} \right\}$ .

The regions are illustrated graphically in Figure 1. In the first case, there is a large region upon  $n \gtrsim d$ , where meaningful differential privacy (with  $\epsilon \leq 1$  and  $\delta = o(1/n)$ ) can be achieved without incurring a significant toll relative to (4). In the second case, we need at least  $n \gtrsim d^2$  to achieve ‘‘privacy-for-free’’ in the most favorable case where  $\alpha^* = 1$ . In the case when  $X$  could be rank-deficient, then it is infeasible to achieve ‘‘privacy for free’’ no matter how large  $n$  is.

It might be tempting to conclude that one should always prefer Case 1 over Case 2. This is unfortunately not true because the artificial restriction of the model class via a bounded  $\|\Theta\|$  also weakens our non-private baseline. In other word, the best solution within a small  $\Theta$  might be significantly worse than the best solution in  $\mathbb{R}^d$ .

In practice, it is hard to find a  $\Theta$  with a small radius that fits all purposes<sup>3</sup> and it is unreasonable to assume  $\alpha^* > 0$ .

<sup>2</sup>This is arguably the more relevant setting. Note that if  $x \sim \mathcal{N}(0, I_d)$  and  $\theta$  is fixed, then  $x^T \theta = O_P(d^{-1/2} \|x\| \|\theta\|)$ .

<sup>3</sup>If  $\|\Theta\| \gg \|\theta^*\|$  then the constraint becomes limiting. If  $\|\theta^*\| \ll \|\Theta\|$  instead, then calibrating the noise according to  $\|\Theta\|$  will inject more noise than necessary.

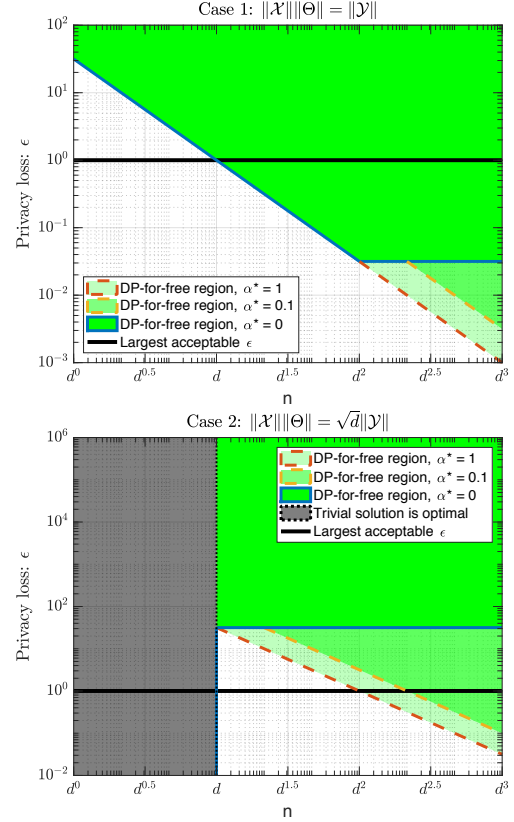


Figure 1: Illustration of the region of  $\epsilon$  where DP can be obtained without losing minimax rate (4). [Zoom to see!]

This motivates us to go beyond the worst-case and come up with *adaptive* algorithms that work without knowing  $\|\theta^*\|$  and  $\alpha$  while achieving the minimax rate for the class with  $\|\Theta\| = \|\theta^*\|$  and  $\alpha^* = \alpha$  (in hindsight).

In Appendix B, we provide an alternative illustration of the lower bounds and highlight the price of differential privacy for different configuration of  $n, d, \alpha, \epsilon$ .

### 3.2 Existing algorithms and our contribution

We now survey the following list of five popular algorithms in differentially private learning and highlight the novelty in our proposals<sup>4</sup>.

1. Sufficient statistics perturbation (SSP) (Vu & Slavkovic, 2009; Foulds et al., 2016): Release  $X^T X$  and  $X \mathbf{y}$  differentially privately and then output  $\hat{\theta} = (\widehat{X^T X})^{-1} \widehat{X \mathbf{y}}$ .
2. Objective perturbation (OBJPERT) (Kifer et al.,

<sup>4</sup>While we try to be as comprehensive as possible, the literature has grown massively and the choice of this list is limited by our knowledge and opinions.

2012):  $\hat{\theta} = \operatorname{argmin} F(\theta) + 0.5\lambda\|\theta\|^2 + Z^T\theta$  with an appropriate  $\lambda$  and  $Z$  is an appropriately chosen iid Gaussian random vector.

3. Subsample and Aggregate (Sub-Agg) (Smith, 2008; Dwork & Smith, 2010): Subsample many times, apply debiased MLE to each subset and then randomize the way we aggregate the results.
4. Posterior sampling (OPS) (Mir, 2013; Dimitrakakis et al., 2014; Wang et al., 2015; Minami et al., 2016): Output  $\hat{\theta} \sim P(\theta) \propto e^{-\gamma(F(\theta)+0.5\lambda\|\theta\|^2)}$  with parameters  $\gamma, \lambda$ .
5. NOISYSGD (Bassily et al., 2014): Run SGD for a fixed number of iterations with additional Gaussian noise added to the stochastic gradient evaluated on one randomly-chosen data point.

We omit detailed operational aspects of these algorithms and focus our discussion on their theoretical guarantees. Interested readers are encouraged to check out each paper separately. These algorithms are proven under different scalings and assumptions. To ensure fair comparison, we make sure that all results are converted to our setting under a subset of the following assumptions.

A.1  $\|\mathcal{X}\|$  is bounded,  $\|\mathcal{Y}\|$  is bounded.

A.2  $\|\Theta\|$  is bounded.

A.3 All possible data sets  $X$  obey that the smallest eigenvalue  $\lambda_{\min}(X^T X)$  is greater than  $\frac{n\|\mathcal{X}\|^2}{d}\alpha^*$ .

Note that A.3 is a restriction on the domain of the data set, rather than the domain of individual data points in the data set of size  $n$ . While it is a little unconventional, it is valid to define differential privacy within such a restricted space of data sets. It is the same assumption that we needed to assume for the lower bound in (6) to be meaningful. As in Koren & Levy (2015), we simplify the expressions of the bound by assuming  $\|\mathcal{Y}\| \leq \|\mathcal{X}\|\|\Theta\|$ , and in addition, we assume that  $\|\mathcal{Y}\| \lesssim \|\mathcal{X}\|\|\theta^*\|$ .

Table 1 summarizes the upper bounds of optimization error the aforementioned algorithms in comparison to our two proposals: ADAOPS and ADASSP. Comparing the rates to the lower bounds in the previous section, it is clear that NoisySGD, OBJPERT both achieve the minimax rate in optimization error but their hyperparameter choice depends on the unknown  $\|\Theta\|$  and  $\alpha^*$ . SSP is adaptive to  $\alpha$  and  $\|\theta^*\|$  but has a completely different type of issue — it can fail arbitrarily badly for regime covered under (5), and even for well-conditioned problems, its theoretical guarantees only kick in as  $n$  gets very large. Our proposed algorithms ADAOPS and ADASSP are able to simultaneously switch between the two regimes and get the best of both worlds.

Table 2 summarizes the upper bounds for estimation. The second row compares the approximation of  $\theta^*$  in MSE and the third column summarizes the statistical efficiency of the DP estimators relative to the MLE:  $\theta^*$  under the linear Gaussian model. All algorithms except OPS are asymptotically efficient. For the interest of  $(\epsilon, \delta)$ -DP, SSP has the fastest convergence rate and does not explicitly depend on the smallest eigenvalue, but again it behaves differently when  $n$  is small, while ADAOPS and ADASSP work optimally (up to a constant) for all  $n$ .

### 3.3 Other related work

The problem of adaptive estimation is closely related to model selection (see, e.g., Birgé & Massart, 2001) and an approach using Bayesian Information Criteria was carefully studied in the differential private setting for the problem of  $\ell_1$  constrained ridge regression by Lei et al. (2017). Their focus is different to ours in that they care about inferring the correct model, while we take the distribution-free view. Linear regression is also studied in many more specialized setups, e.g., high dimensional linear regression (Kifer et al., 2012; Talwar et al., 2014, 2015), statistical inference (Sheffet, 2017) and so on. For the interest of this paper, we focus on the standard regime of linear regression where  $d < n$  and do not use sparsity or  $\ell_1$  constraint set to achieve the  $\log(d)$  dependence. That said, we acknowledge that Sheffet (2017) analyzed SSP under the linear Gaussian model (the third row in Table 2) and their techniques of adaptively adding regularization have inspired ADASSP.

## 4 MAIN RESULTS

In this section, we present and analyze ADAOPS and ADASSP that achieve the aforementioned adaptive rate. The pseudo-code of these two algorithms are given in Algorithm 1 and Algorithm 2.

The idea of both algorithms is to release key data-dependent quantities differentially privately and then use a high probability confidence interval of these quantities to calibrate the noise to privacy budget as well as to choose the ridge regression’s hyperparameter  $\lambda$  for achieving the smallest prediction error. Specifically, ADAOPS requires us to release both the smallest eigenvalue  $\lambda_{\min}$  of  $X^T X$  and the local Lipschitz constant  $L := \|\mathcal{X}\|(\|\mathcal{X}\|\|\theta_x^*\| + \|\mathcal{Y}\|)$ , while ADASSP only needs the smallest eigenvalue  $\lambda_{\min}$ .

In both ADASSP and ADAOPS, we choose  $\lambda$  by minimizing an upper bound of  $F(\hat{\theta}) - F(\theta^*)$  in the form of “variance” and “bias”

$$\tilde{O}\left(\frac{d\|\mathcal{X}\|^4\|\theta^*\|^2}{\lambda + \lambda_{\min}}\right) + \lambda\|\theta^*\|^2.$$

Table 1: Summary of optimization error bounds. This table compares the (expected or high probability ) additive suboptimality of different differentially private linear regression procedures relative to the (non-private) empirical risk minimizer  $\theta^*$ . In particular, the results for NoisySGD holds in expectation and everything else with probability  $1 - \rho$  (hiding at most a logarithmic factor in  $\sqrt{1/\rho}$ ). Constant factors are dropped for readability.

	$F(\hat{\theta}) - F(\theta^*)$	Assumptions	Remarks
NoisySGD	$\frac{\sqrt{d \log(\frac{n}{\delta})} \ \mathcal{X}\ ^2 \ \Theta\ ^2}{\epsilon}$	A.1, A.2	Theorem 2.4 (Part 1) of (Bassily et al., 2014).
	$\frac{d^2 \log(\frac{n}{\delta}) \ \Theta\ ^2}{\alpha^* n \epsilon^2}$	A.1, A.2, A.3	Theorem 2.4 (Part 2) of (Bassily et al., 2014)
OBJPERT	$\frac{\sqrt{d \log(\frac{1}{\delta})} \ \mathcal{X}\ ^2 \ \Theta\  \ \theta^*\ }{\epsilon}$	A.1, A.2	Theorem 4 (Part 2) of (Kifer et al., 2012).
	$\frac{d^2 \log(\frac{1}{\delta}) \ \Theta\ ^2}{\alpha^* n \epsilon^2}$	A.1, A.2, A.3	Theorem 5 & Appendix E.2 of (Kifer et al., 2012).
OPS	$\frac{d \ \mathcal{X}\ ^2 \ \Theta\ ^2}{\epsilon}$	A.1, A.2	Results for $\epsilon$ -DP (Wang et al., 2015)
SSP	$\frac{d^2 \log(\frac{1}{\delta}) \ \mathcal{X}\ ^2 \ \theta^*\ ^2}{\alpha n \epsilon^2}$	A.1	Adaptive to $\ \theta^*\ , X, \alpha$ , but requires $n = \Omega(\frac{d^{1.5} \log(4/\delta)}{\alpha \epsilon})^5$ .
ADAOPS & ADASSP	$\frac{\sqrt{d \log(\frac{1}{\delta})} \ \mathcal{X}\ ^2 \ \theta^*\ ^2}{\epsilon} \wedge \frac{d^2 \log(\frac{1}{\delta}) \ \theta^*\ ^2}{\alpha n \epsilon^2}$	A.1	Adaptive in $\ \theta^*\ , X, \alpha$ .

Table 2: Summary of estimation error bounds under the linear Gaussian model. On the second column we compare the approximation of MLE  $\theta^*$  in mean square error up to a universal constant. On the third column, we compare the relative efficiency. The relative efficiency bounds are simplified with the assumption of  $\alpha = \Omega(1)$ , which implies that  $\text{tr}[(X^T X)^{-1}] = O(d^2 n^{-1} \|\mathcal{X}\|^{-2})$  and  $\text{tr}[(X^T X)^{-2}] = O(d n^{-1} \|\mathcal{X}\|^{-2} \text{tr}[(X^T X)^{-1}])$ .  $\tilde{O}(\cdot)$  hides  $\text{polylog}(1/\delta)$  terms.

	Approx. MLE: $\mathbb{E} \ \hat{\theta} - \theta^*\ ^2$	Rel. efficiency: $\frac{\mathbb{E} \ \hat{\theta} - \theta_0\ ^2}{\mathbb{E} \ \theta^* - \theta_0\ ^2}$	Remarks
Sub-Agg	$O\left(\frac{\text{poly}(d, \ \Theta\ , \ \mathcal{X}\ , \alpha^{-1})}{\epsilon^{6/5} n^{6/5}}\right)$	$1 + \tilde{O}\left(\frac{\text{poly}(d, \ \Theta\ , \ \mathcal{X}\ )}{n^{1/5} \epsilon^{6/5}}\right)$	$\epsilon$ -DP, suboptimal in $n$ , possibly also in $d$ (Dwork & Smith, 2010).
OPS	$O\left(\frac{\ \mathcal{X}\ ^2 \ \Theta\ ^2}{\epsilon} \text{tr}[(X^T X)^{-1}]\right)$	$\tilde{O}\left(\frac{\ \mathcal{X}\ ^2 \ \Theta\ ^2}{\epsilon \sigma^2}\right)$	$\epsilon$ -DP, adaptive in $X$ , but not asymptotically efficient (Wang et al., 2015).
SSP	$O\left(\frac{\log(\frac{1}{\delta}) \ \mathcal{X}\ ^4 \ \theta^*\ ^2}{\epsilon^2} \text{tr}[(X^T X)^{-2}]\right)$	$1 + \tilde{O}\left(\frac{d \ \mathcal{X}\ ^2 \ \theta_0\ ^2}{n \epsilon^2 \sigma^2} + \frac{d^3}{n^2 \epsilon^2}\right)$	Adaptive in $\ \theta^*\ , X$ , no explicit dependence on $\alpha$ , but requires large $n$ . (Sheffet, 2017, Theorem 5.1)
ADAOPS & ADASSP	$O\left(\frac{d \log(\frac{1}{\delta}) \ \mathcal{X}\ ^2 \ \theta^*\ ^2}{\alpha n \epsilon^2} \text{tr}[(X^T X)^{-1}]\right)$	$1 + \tilde{O}\left(\frac{d \ \mathcal{X}\ ^2 \ \theta_0\ ^2}{n \epsilon^2 \sigma^2} + \frac{d^3}{n^2 \epsilon^2}\right)$	Adaptive in $\ \theta^*\ , X, \alpha$ .

---

**Algorithm 1** ADAOPS: One-Posterior Sample estimator with adaptive regularization
 

---

**input** Data  $X, \mathbf{y}$ . Privacy budget:  $\epsilon, \delta$ , Bounds:  $\|\mathcal{X}\|, \|\mathcal{Y}\|$ .

1. Calculate the minimum eigenvalue  $\lambda_{\min}(X^T X)$ .
2. Sample  $Z \sim \mathcal{N}(0, 1)$  and privately release

$$\tilde{\lambda}_{\min} = \max \left\{ \lambda_{\min} + \frac{\sqrt{\log(6/\delta)}}{\epsilon/4} Z - \frac{\log(6/\delta)}{\epsilon/4}, 0 \right\}.$$

3. Set  $\bar{\epsilon}$  as the positive solution of the quadratic equation

$$\bar{\epsilon}^2 / (2 \log(6/\delta)) + \bar{\epsilon} - \epsilon/4 = 0.$$

4. Set  $\varrho = 0.05$ ,  $C_1 = (d/2 + \sqrt{d \log(1/\varrho)} + \log(1/\varrho)) \log(6/\delta) / \bar{\epsilon}^2$ ,  $C_2 = \log(6/\delta) / (\epsilon/4)$ ,  $t_{\min} = \max \left\{ \frac{\|\mathcal{X}\|^2 (1 + \log(6/\delta))}{2\epsilon} - \tilde{\lambda}_{\min}, 0 \right\}$  and solve

$$\lambda = \underset{t \geq t_{\min}}{\operatorname{argmin}} \frac{\|\mathcal{X}\|^4 C_1 [1 + \|\mathcal{X}\|^2 / (t + \tilde{\lambda}_{\min})]^{2C_2}}{t + \tilde{\lambda}_{\min}} + t. \quad (7)$$

which has a unique solution.

5. Calculate  $\hat{\theta} = (X^T X + \lambda I)^{-1} X^T \mathbf{y}$ .

6. Sample  $Z \sim \mathcal{N}(0, 1)$  and privately release

$$\Delta = \log(\|\mathcal{Y}\| + \|\mathcal{X}\| \|\hat{\theta}\|) + \frac{\log(1 + \|\mathcal{X}\|^2 / (\lambda + \tilde{\lambda}_{\min}))}{\epsilon / (4 \sqrt{\log(6/\delta)})} Z +$$

$$\frac{\log(1 + \|\mathcal{X}\|^2 / (\lambda + \tilde{\lambda}_{\min}))}{\epsilon / (4 \log(6/\delta))}. \text{ Set } \tilde{L} := \|\mathcal{X}\| e^{\Delta}.$$

7. Calibrate noise by choosing  $\tilde{\epsilon}$  as the positive solution of the quadratic equation

$$\frac{\tilde{\epsilon}^2}{2} \left[ \frac{1}{\log(6/\delta)} \frac{1 + \log(6/\delta)}{\log(6/\delta)} \right] + \tilde{\epsilon} - \epsilon/2 = 0. \quad (8)$$

and then set  $\gamma = \frac{(\tilde{\lambda}_{\min} + \lambda) \tilde{\epsilon}^2}{\log(6/\delta) \tilde{L}^2}$ .

**output**  $\tilde{\theta} \sim p(\theta | X, \mathbf{y}) \propto e^{-\frac{\gamma}{2} (\|\mathbf{y} - X\theta\|^2 + \lambda \|\theta\|^2)}$ .

---

Note that while  $\|\theta^*\|^2$  cannot be privately released in general due to unbounded sensitivity, it appears in both terms and do not enter the decision process of finding the optimal  $\lambda$  that minimizes the bound. This convenient feature follows from our assumption that  $\|\mathcal{Y}\| \lesssim \|\mathcal{X}\| \|\theta^*\|$ . Dealing with the general case involving an arbitrary  $\|\mathcal{Y}\|$  is an intriguing open problem.

A tricky situation for ADAOPS is that the choice of  $\gamma$  depends on  $\lambda$  through  $\tilde{L}$ , which is the local Lipschitz constant at the ridge regression solution  $\theta_\lambda^*$ . But the choice of  $\lambda$  also depends on  $\gamma$  since the ‘‘variance’’ term above is inversely proportional to  $\gamma$ . Our solution is to express  $\tilde{L}$  (hence  $\gamma$ ) as a function of  $\lambda$  and solve the nonlinear univariate optimization problem (7).

We are now ready to state the main results.

**Theorem 2.** *Algorithm 1 outputs  $\tilde{\theta}$  which obeys that*

- (i) *It satisfies  $(\epsilon, \delta)$ -DP.*

---

**Algorithm 2** ADASSP: Sufficient statistics perturbation with adaptive damping
 

---

**input** Data  $X, \mathbf{y}$ . Privacy budget:  $\epsilon, \delta$ , Bounds:  $\|\mathcal{X}\|, \|\mathcal{Y}\|$ .

1. Calculate the minimum eigenvalue  $\lambda_{\min}(X^T X)$ .

2. Privately release  $\tilde{\lambda}_{\min} = \max \left\{ \lambda_{\min} + \frac{\sqrt{\log(6/\delta)}}{\epsilon/3} \|\mathcal{X}\|^2 Z - \frac{\log(6/\delta)}{\epsilon/3} \|\mathcal{X}\|^2, 0 \right\}$ , where  $Z \sim \mathcal{N}(0, 1)$ .

3. Set  $\lambda = \max \left\{ 0, \frac{\sqrt{d \log(6/\delta) \log(2d^2/\rho)} \|\mathcal{X}\|^2}{\epsilon/3} - \tilde{\lambda}_{\min} \right\}$

4. Privately release  $\widehat{X^T X} = X^T X + \frac{\sqrt{\log(6/\delta)} \|\mathcal{X}\|^2}{\epsilon/3} Z$  for  $Z \in \mathbb{R}^{d \times d}$  is a symmetric matrix and every element from the upper triangular matrix is sampled from  $\mathcal{N}(0, 1)$ .

5. Privately release  $\widehat{X \mathbf{y}} = X \mathbf{y} + \frac{\sqrt{\log(6/\delta)} \|\mathcal{X}\| \|\mathcal{Y}\|}{\epsilon/3} Z$  for  $Z \sim \mathcal{N}(0, I_d)$ .

**output**  $\tilde{\theta} = (\widehat{X^T X} + \lambda I)^{-1} \widehat{X \mathbf{y}}$

---

- (ii) *Assume  $\|\mathcal{Y}\| \lesssim \|\mathcal{X}\| \|\theta^*\|$ . With probability  $1 - \varrho$ ,  $F(\tilde{\theta}) - F(\theta^*) \leq$*

$$O \left( \frac{\sqrt{d + \log(\frac{1}{\varrho})} \|\mathcal{X}\|^2 \|\theta^*\|^2}{\epsilon / \sqrt{\log(\frac{1}{\varrho})}} \wedge \frac{d [d + \log(\frac{1}{\varrho})] \|\theta^*\|^2}{\alpha n \epsilon^2 / \log(\frac{1}{\varrho})} \right).$$

- (iii) *Assume that  $\mathbf{y} | X$  obeys a linear Gaussian model and  $X$  is full-rank. Then there is an event  $E$  satisfying  $\mathbb{P}(E) \geq 1 - \delta/3$  and  $E \perp \mathbf{y} | X$ , such that  $\mathbb{E}[\tilde{\theta} | X, E] = \theta_0$  and*

$$\operatorname{Cov}[\tilde{\theta} | X, E] \prec \left( 1 + O \left( \frac{\tilde{C} d \log(6/\delta)}{\sigma^2 \alpha n \epsilon^2} \right) \right) \sigma^2 (X^T X)^{-1}$$

where constant

$$\tilde{C} := \|\mathcal{Y}\|^2 + \|\mathcal{X}\|^2 (\|\theta_0\|^2 + \sigma^2 \operatorname{tr}[(X^T X)^{-1}]).$$

The proof, deferred to Appendix D, makes use of a fine-grained DP-analysis through the recent per instance DP techniques (Wang, 2017) and then convert the results to DP by releasing data dependent bounds of  $\alpha$  and the magnitude of a ridge-regression output  $\theta_\lambda^*$  with an adaptively chosen  $\lambda$ . Note that  $\|\theta_\lambda^*\|$  does not have a bounded global sensitivity. The method to release it differentially privately (described in Lemma 12) is part of our technical contribution.

The ADASSP algorithm is simpler and enjoys slightly stronger theoretical guarantees.

**Theorem 3.** *Algorithm 2 outputs  $\tilde{\theta}$  which obeys that*

- (i) *It satisfies  $(\epsilon, \delta)$ -DP.*

- (ii) *Assume  $\|\mathcal{Y}\| \lesssim \|\mathcal{X}\| \|\theta^*\|$ . With probability  $1 - \varrho$ ,  $F(\tilde{\theta}) - F(\theta^*) \leq$*

$$O \left( \frac{\sqrt{d \log(\frac{d^2}{\varrho})} \|\mathcal{X}\|^2 \|\theta^*\|^2}{\epsilon / \sqrt{\log(\frac{6}{\varrho})}} \wedge \frac{\|\mathcal{X}\|^4 \|\theta^*\|^2 \operatorname{tr}[(X^T X)^{-1}]}{\epsilon^2 / [\log(\frac{6}{\varrho}) \log(\frac{d^2}{\varrho})]} \right)$$

(iii) Assume that  $\mathbf{y}|X$  obeys a linear Gaussian model and  $X$  has a sufficiently large  $\alpha$ . Then there is an event  $E$  satisfying  $\mathbb{P}(E) \geq 1 - \delta/3$  and  $E \perp \mathbf{y}|X$ , such that  $\mathbb{E}[\tilde{\theta}|X, E] = \theta_0$  and

$$\begin{aligned} & \mathbb{E}[\|\tilde{\theta} - \theta_0\|^2|X, E] \\ &= \sigma^2 \text{tr}[(X^T X)^{-1}] + O\left(\frac{\tilde{C}\|\mathcal{X}\|^2 \text{tr}[(X^T X)^{-2}]}{\epsilon^2 / \log(\frac{6}{\delta})}\right), \end{aligned}$$

with the same constant  $\tilde{C}$  in Theorem 2 (iii).

The proof of Statement (1) is straightforward. Note that we release the eigenvalue  $\lambda_{\min}(X^T X)$ ,  $X\mathbf{y}$  and  $X^T X$  differentially privately each with parameter  $(\epsilon/3, \delta/3)$ . For the first two, we use Gaussian mechanism and for  $X^T X$ , we use the Analyze-Gauss algorithm (Dwork et al., 2014b) with a symmetric Gaussian random matrix. The result then follows from the composition theorem of differential privacy. The proof of the second and third statements is provided in Appendix C. The main technical challenge is to prove the concentration on the spectrum and the Johnson-Lindenstrauss-like distance preserving properties for symmetric Gaussian random matrices (Lemma 6). We note that while SSP is an old algorithm the analysis of its theoretical properties is new to this paper.

**Remarks.** Both ADAOPS and ADASSP match the smaller of the two lower bounds (5) and (6) for each problem instance. They are slightly different in that ADAOPS preserves the shape of the intrinsic geometry while ADASSP’s bounds are slightly stronger as they do not explicitly depend on the smallest eigenvalue.

## 5 EXPERIMENTS

In this section, we conduct synthetic and real data experiments to benchmark the performance of ADAOPS and ADASSP relative to existing algorithms we discussed in Section 3. NOISYSGD and Sub-Agg are excluded because they are dominated by OBJPERT and an  $(\epsilon, \delta)$ -DP version of OPS (see Appendix F for details)<sup>6</sup>.

### Prediction accuracy in UCI data sets experiments.

The first set of experiments is on training linear regression on a number of UCI regression data sets. Standard z-scoring are performed and all data points are normalized to having an Euclidean norm of 1 as a preprocessing step. The results on four of the data sets are presented in Figure 2. As we can see, SSP is unstable for small data. OBJPERT suffers from a pre-defined bound  $\|\Theta\|$  and

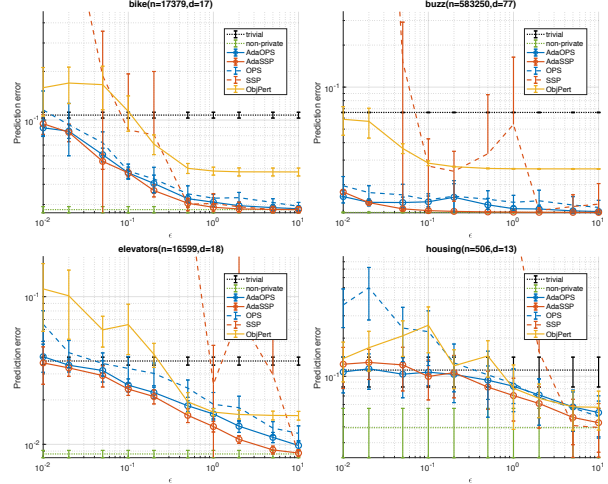


Figure 2: Example of results of differentially private linear regression algorithms on UCI data sets for a sequence of  $\epsilon$ . Reported on the y-axis is the cross-validation prediction error in MSE and their confidence intervals.

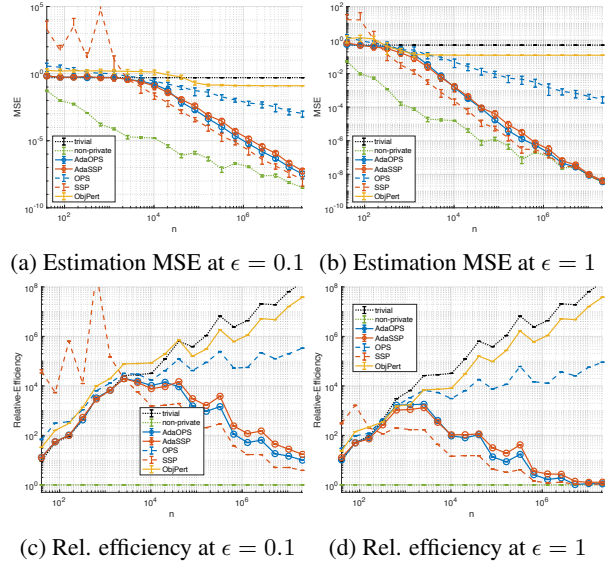


Figure 3: Example of differentially private linear regression under linear Gaussian model with an increasing data size  $n$ . We simulate the data from  $d = 10$ ,  $\theta_0$  drawn from a uniform distribution defined on  $[0, 1]^d$ . We generate  $X \in \mathbb{R}^{n \times d}$  as a Gaussian random matrix and then generate  $y \sim \mathcal{N}(X\theta_0, I_d)$ . We used  $\epsilon = 1$  and  $\epsilon = 0.1$ , both with  $\delta = 1/n^2$ . The results clearly illustrate the asymptotic efficiency of the proposed approaches.

does not converge to nonprivate solution even with a large  $\epsilon$ . OPS performs well but still does not take advantage of the strong convexity that is intrinsic to the data set. ADAOPS and ADASSP on the other hand are able to nicely interpolate between the trivial solution and the non-private baseline and performed as well as or better than baselines for all  $\epsilon$ . More detailed quantitative results on all the 36 UCI data sets are presented in Table 3.

<sup>6</sup>The code to reproduce all experimental results are available at [https://github.com/yuxiangw/optimal\\_dp\\_linear\\_regression](https://github.com/yuxiangw/optimal_dp_linear_regression).



Table 3: Summary of UCI data experiments at  $\epsilon = 0.1$ ,  $\delta = \min\{1e-6, 1/n^2\}$ . The boldface denotes the DP algorithm where the standard deviation is smaller than the error (a positive quantity), and the 95% confidence interval covers the observed best performance among benchmarked DP algorithms.

	Trivial	non-private	OBJPERT	OPS	SSP	ADAOPS	ADASSP
3droad	0.0275±0.00014	0.0265±0.00012	0.0267±0.00013	0.027±0.00026	<b>0.0265±0.00019</b>	<b>0.0265±0.00019</b>	<b>0.0265±0.00019</b>
airfoil	0.103±0.0069	0.0533±0.0074	0.356±0.064	<b>0.138±0.086</b>	0.232±0.28	<b>0.0914±0.015</b>	<b>0.0878±0.014</b>
autompng	0.113±0.011	0.0221±0.0032	<b>0.143±0.096</b>	0.242±0.11	5.44±6.1	<b>0.098±0.03</b>	<b>0.115±0.047</b>
autos	0.13±0.042	0.0274±0.011	<b>0.17±0.13</b>	0.308±0.13	1.7e+03±2.5e+03	<b>0.136±0.066</b>	<b>0.132±0.064</b>
bike	0.107±0.0028	0.0279±0.00078	0.113±0.018	<b>0.0484±0.005</b>	0.0869±0.067	<b>0.0471±0.004</b>	<b>0.0471±0.0026</b>
breastcancer	0.194±0.027	0.139±0.025	<b>0.212±0.078</b>	<b>0.269±0.13</b>	9.54e+03±1.9e+04	<b>0.204±0.037</b>	<b>0.196±0.051</b>
buzz	0.0658±0.00015	0.0127±4.6e-05	0.0285±0.00071	0.0156±0.001	0.0272±0.0097	0.0151±0.00095	<b>0.013±9.7e-05</b>
challenger	0.141±0.084	0.138±0.088	0.323±0.28	0.338±0.13	3.07±3.9	<b>0.159±0.13</b>	<b>0.146±0.093</b>
concrete	0.127±0.0043	0.0445±0.0033	0.237±0.076	0.181±0.042	1.94±1.8	<b>0.12±0.011</b>	<b>0.119±0.016</b>
concreteslump	0.149±0.039	0.0245±0.0071	0.349±0.094	0.549±0.24	3.14±2.5	<b>0.151±0.064</b>	<b>0.165±0.065</b>
elevators	0.0367±0.0014	0.00861±0.00031	0.0647±0.015	0.0327±0.0042	0.645±0.98	<b>0.0252±0.0026</b>	<b>0.0237±0.0022</b>
energy	0.235±0.012	0.0232±0.0023	0.332±0.09	<b>0.161±0.083</b>	1.7e+03±3.4e+03	<b>0.167±0.034</b>	<b>0.15±0.032</b>
fertility	0.0977±0.024	0.0863±0.024	0.203±0.04	0.639±0.16	439±8.6e+02	<b>0.108±0.048</b>	<b>0.115±0.032</b>
forest	0.0564±0.0081	0.0571±0.0086	0.12±0.022	0.177±0.036	41.9±77	<b>0.0622±0.017</b>	<b>0.0675±0.013</b>
gas	0.112±0.0062	0.0214±0.0028	0.109±0.015	<b>0.0546±0.012</b>	0.923±0.63	0.0801±0.0078	0.0875±0.0073
houseelectric	0.122±0.00017	0.0136±1.4e-05	0.0409±0.00027	0.0144±0.00017	<b>0.0136±2.2e-05</b>	<b>0.0136±2.2e-05</b>	<b>0.0136±2.2e-05</b>
housing	0.112±0.019	0.0394±0.01	0.253±0.063	0.225±0.065	2.24±2.3	<b>0.108±0.023</b>	<b>0.0997±0.035</b>
keggdirected	0.117±0.00095	0.0188±0.0011	0.0637±0.0042	0.0266±0.0019	0.23±0.33	0.0227±0.0015	<b>0.0212±0.0011</b>
keggundirected	0.0694±0.00074	0.00475±8.9e-05	0.0365±0.0028	0.0166±0.0033	0.353±0.4	0.0107±0.0012	<b>0.00912±0.00046</b>
kin40k	0.0634±0.0012	0.0632±0.0013	0.0871±0.0092	0.0717±0.0026	<b>0.0633±0.002</b>	<b>0.0639±0.0021</b>	<b>0.064±0.0021</b>
machine	0.121±0.013	0.0395±0.0051	0.282±0.14	0.347±0.14	2.27e+03±4.5e+03	<b>0.105±0.025</b>	<b>0.141±0.068</b>
parkinsons	0.17±0.0026	0.128±0.0024	0.211±0.014	<b>0.157±0.011</b>	132±2.6e+02	<b>0.159±0.0065</b>	<b>0.156±0.0064</b>
pendulum	0.0226±0.0061	0.0181±0.0049	0.118±0.027	0.122±0.041	24.8±45	<b>0.0276±0.011</b>	0.0346±0.0069
pol	0.345±0.0028	0.135±0.0023	0.302±0.032	<b>0.196±0.02</b>	281±5.3e+02	0.214±0.0056	0.214±0.0061
protein	0.167±0.0011	0.119±0.0014	0.158±0.01	0.137±0.0044	<b>0.149±0.06</b>	0.129±0.0015	<b>0.125±0.0026</b>
pumadyn32nm	0.0935±0.0039	0.0941±0.0039	0.124±0.0046	0.111±0.005	8.92e+03±1.8e+04	<b>0.0968±0.0065</b>	<b>0.0966±0.0063</b>
servo	0.184±0.039	0.0752±0.022	0.366±0.077	0.574±0.26	2.03±1.5	<b>0.195±0.065</b>	<b>0.198±0.081</b>
skillcraft	0.0439±0.0021	0.0203±0.0017	0.0817±0.013	0.0519±0.0099	4.72±4.3	<b>0.037±0.008</b>	<b>0.039±0.0056</b>
slice	0.196±0.0021	0.0283±0.00051	0.174±0.0053	<b>0.0924±0.0035</b>	11.2±9.4	0.0992±0.0021	0.132±0.0015
sml	0.211±0.0089	0.0143±0.00066	0.23±0.03	<b>0.0955±0.029</b>	59.9±80	0.134±0.0075	0.147±0.013
solar	0.0118±0.0042	0.0106±0.0038	0.0994±0.023	0.0667±0.017	5.95±9.6	<b>0.0165±0.0062</b>	<b>0.0204±0.0073</b>
song	0.0917±0.0003	0.0636±0.00033	0.0838±0.0014	0.072±0.00035	<b>0.0644±0.0005</b>	0.0685±0.00045	0.0697±0.00029
stock	0.0583±0.0095	0.013±0.0023	0.122±0.026	0.157±0.055	46.8±66	<b>0.0582±0.023</b>	<b>0.0651±0.024</b>
tamielectric	0.334±0.002	0.334±0.0021	0.341±0.0021	0.343±0.0065	<b>0.335±0.0033</b>	<b>0.337±0.0047</b>	<b>0.335±0.0033</b>
wine	0.0566±0.0028	0.0202±0.00099	0.153±0.028	0.0911±0.016	11.7±17	<b>0.058±0.011</b>	<b>0.0599±0.01</b>
yacht	0.105±0.017	0.0176±0.0055	0.273±0.076	0.371±0.14	4.92±6.8	<b>0.0967±0.035</b>	<b>0.109±0.03</b>

### Parameter estimation under linear Gaussian model.

To illustrate the performance of the algorithms under standard statistical assumptions, we also benchmarked the algorithms on synthetic data generated by a linear Gaussian model. The results, shown in Figure 3 illustrates that as  $n$  gets large, ADAOPS and ADASSP with  $\epsilon = 0.1$  and  $\epsilon = 1$  converge to the maximum likelihood estimator at a rate faster than the optimal statistical rate that MLE estimates  $\theta^*$ , therefore at least for large  $n$ , differential privacy comes for free. Note that there is a gap in SSP and ADASSP for large  $n$ , this can be thought of as a cost of adaptivity as ADASSP needs to spend some portion of its privacy budget to release  $\lambda_{\min}$ , which SSP does not, this can be fixed by using more careful splitting of the privacy budget.

## 6 CONCLUSION

In this paper, we presented a detailed case-study of the problem of differentially private linear regression. We clarified the relationships between various quantities of the problems as they appear in the private and non-private

information-theoretic lower bounds. We also surveyed the existing algorithms and highlighted that the main drawback using these algorithms relative to their non-private counterpart is that they cannot adapt to data-dependent quantities. This is particularly true for linear regression where the ordinary least square algorithm is able to work optimally for a large class of different settings.

We proposed ADAOPS and ADASSP to address the issue and showed that they both work in unbounded domain. Moreover, they smoothly interpolate the two regimes studied in Bassily et al. (2014) and behave nearly optimally for every instance. We tested the two algorithms on 36 real-life data sets from the UCI machine learning repository and we see significant improvement over popular algorithms for almost all configurations of  $\epsilon$ .

### Acknowledgements

The author thanks the anonymous reviewers for helpful feedbacks and Zichao Yang for sharing the 36 UCI regression data sets as was used in (Yang et al., 2015).

## References

- Agresti, A., & Finlay, B. (1997). Statistical methods for the social sciences.
- Armitage, P., Berry, G., & Matthews, J. N. S. (2008). *Statistical methods in medical research*. John Wiley & Sons.
- Bassily, R., Smith, A., & Thakurta, A. (2014). Private empirical risk minimization: Efficient algorithms and tight error bounds. In *Foundations of Computer Science (FOCS-14)*, (pp. 464–473). IEEE.
- Birgé, L., & Massart, P. (2001). Gaussian model selection. *Journal of the European Mathematical Society*, 3(3), 203–268.
- Chaudhuri, K., Monteleoni, C., & Sarwate, A. D. (2011). Differentially private empirical risk minimization. *The Journal of Machine Learning Research*, 12, 1069–1109.
- Dimitrakakis, C., Nelson, B., Mitrokotsa, A., & Rubinstein, B. I. (2014). Robust and private Bayesian inference. In *Algorithmic Learning Theory*, (pp. 291–305). Springer.
- Donoho, D. L. (1995). De-noising by soft-thresholding. *IEEE transactions on information theory*, 41(3), 613–627.
- Draper, N. R., & Smith, H. (2014). *Applied regression analysis*, vol. 326. John Wiley & Sons.
- Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., & Naor, M. (2006a). Our data, ourselves: Privacy via distributed noise generation. In *International Conference on the Theory and Applications of Cryptographic Techniques*, (pp. 486–503). Springer.
- Dwork, C., & Lei, J. (2009). Differential privacy and robust statistics. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, (pp. 371–380). ACM.
- Dwork, C., McSherry, F., Nissim, K., & Smith, A. (2006b). Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography*, (pp. 265–284). Springer.
- Dwork, C., Roth, A., et al. (2014a). The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4), 211–407.
- Dwork, C., & Smith, A. (2010). Differential privacy for statistics: What we know and what we want to learn. *Journal of Privacy and Confidentiality*, 1(2), 2.
- Dwork, C., Talwar, K., Thakurta, A., & Zhang, L. (2014b). Analyze gauss: optimal bounds for privacy-preserving principal component analysis. In *ACM symposium on Theory of computing (STOC-14)*, (pp. 11–20). ACM.
- Foulds, J., Geumlek, J., Welling, M., & Chaudhuri, K. (2016). On the theory and practice of privacy-preserving Bayesian data analysis. In *Conference on Uncertainty in Artificial Intelligence (UAI-16)*, (pp. 192–201). AUAI Press.
- Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning*, vol. 1. Springer series in statistics Springer, Berlin.
- Galton, F. (1886). Regression towards mediocrity in hereditary stature. *The Journal of the Anthropological Institute of Great Britain and Ireland*, 15, 246–263.
- Greene, W. H. (2003). *Econometric analysis*. Pearson Education India.
- Kifer, D., Smith, A., & Thakurta, A. (2012). Private convex empirical risk minimization and high-dimensional regression. *Journal of Machine Learning Research*, 1, 41.
- Koren, T., & Levy, K. (2015). Fast rates for exp-concave empirical risk minimization. In *Advances in Neural Information Processing Systems*, (pp. 1477–1485).
- Laurent, B., & Massart, P. (2000). Adaptive estimation of a quadratic functional by model selection. *Annals of Statistics*, (pp. 1302–1338).
- Lei, J., Charest, A.-S., Slavkovic, A., Smith, A., & Fienberg, S. (2017). Differentially private model selection with penalized and constrained likelihood. *Journal of the Royal Statistical Society*.
- Minami, K., Arai, H., Sato, I., & Nakagawa, H. (2016). Differential privacy without sensitivity. In *Advances in Neural Information Processing Systems*, (pp. 956–964).
- Mir, D. J. (2013). *Differential privacy: an exploration of the privacy-utility landscape*. Ph.D. thesis, Rutgers University.
- Shamir, O. (2015). The sample complexity of learning linear predictors with the squared loss. *Journal of Machine Learning Research*, 16, 3475–3486.
- Sheffet, O. (2017). Differentially private ordinary least squares. In *International Conference on Machine Learning (ICML-17)*, (pp. 3105–3114).
- Smith, A. (2008). Efficient, differentially private point estimators. *arXiv preprint arXiv:0809.4794*.
- Stewart, G. W. (1998). Perturbation theory for the singular value decomposition. Tech. rep.
- Talwar, K., Thakurta, A., & Zhang, L. (2014). Private empirical risk minimization beyond the worst case: The effect of the constraint set geometry. *arXiv preprint arXiv:1411.5417*.
- Talwar, K., Thakurta, A. G., & Zhang, L. (2015). Nearly optimal private lasso. In *Advances in Neural Information Processing Systems*, (pp. 3025–3033).

- Vu, D., & Slavkovic, A. (2009). Differential privacy for clinical trial data: Preliminary evaluations. In *Data Mining Workshops, 2009. ICDMW'09. IEEE International Conference on*, (pp. 138–143). IEEE.
- Wang, Y.-X. (2017). Per-instance differential privacy and the adaptivity of posterior sampling in linear and ridge regression. *arXiv preprint arXiv:1707.07708*.
- Wang, Y.-X., Fienberg, S., & Smola, A. (2015). Privacy for free: Posterior sampling and stochastic gradient monte carlo. In *International Conference on Machine Learning (ICML-15)*, (pp. 2493–2502).
- Wasserman, L. (2013). *All of statistics: a concise course in statistical inference*. Springer Science & Business Media.
- Yang, Z., Wilson, A., Smola, A., & Song, L. (2015). A la carte–learning fast kernels. In *Artificial Intelligence and Statistics (AISTATS-15)*, (pp. 1098–1106).

---

# IMAGINARY KINEMATICS

---

**Sabina Marchetti**

Sapienza University of Rome  
Rome (Italy)

sabina.marchetti@uniroma1.it

**Alessandro Antonucci**

IDSIA  
Manno-Lugano (Switzerland)  
alessandro@idsia.ch

## Abstract

We introduce a novel class of adjustment rules for a collection of beliefs. This is an extension of Lewis' imaging to absorb probabilistic evidence in generalized settings. Unlike standard tools for belief revision, our proposal may be used when information is inconsistent with an agent's belief base. We show that the functionals we introduce are based on the *imaginary* counterpart of *probability kinematics* for standard belief revision, and prove that, under certain conditions, all standard postulates for belief revision are satisfied.

## 1 INTRODUCTION

The theory of belief revision, originated in the work of Alchourrón, Gärdenfors and Makinson [Alchourrón et al., 1985], is aimed to maintain consistency of a knowledge base when updated information is gathered to a rational agent, or *You*. In the present work we will focus on the probabilistic framework, where Your knowledge base is represented by a (closed and convex) collection of probability mass functions, and some observational process is expected to induce an adjustment in the model.<sup>1</sup> With probabilities, evidence on some variables is called *inconsistent* when it contradicts certainty (or impossibility) in Your knowledge base. We provide an example to motivate our contribution.

**Example 1.** *While swimming in a lake, Celeste sees some black birds from the distance. She knows black birds living around that lake are rather tame, while swans might be very aggressive. She is also sure that only*

---

<sup>1</sup>Here we intend an *adjustment* as a generalized *updating*. We avoid this latter term as in the literature it is often intended as equivalent to conditioning.

*white or grey swans exist, although the birds she sees actually look like swans. While reasoning about that, a sailor informs her that a small group of black swans has been spotted around the area. Should Celeste be worried about the birds she sees?*

Classic belief revision operators, introduced in Section 2, fail to absorb information from an observational process when inconsistencies arise such as in Example 1. This feature was motivated in the literature by a *partiality* principle [Cozic, 2011], discussed below. Still, a rule for the adjustment of a model to any piece of evidence ought to be required by a rational agent, to avoid building a new model from scratch when unexpected information shows up. Such an operator ought to update the knowledge base to be consistent with new evidence, while leaving previous beliefs on related events as unchanged as possible. We will characterize optimality requirements for such adjustment operators as an *imaginary kinematics* in Section 3, and extend them to deal with generalized forms of evidence. Particularly, we consider probabilistic evidence, and extend it to i) conditional assessments, and ii) *imprecise* assessments, that may be intended as originating from a qualitative judgment. Section 4 will introduce adjustment functionals based on Lewis' imaging, and study their features and properties. We will refer throughout to partial operators as *revision* rules, as opposed to general *adjustment* ones.

## 2 BACKGROUND

Let  $\Omega$  be any space of atoms - atomic (Boolean) propositional variables - and let a *world*  $\omega$  be any assignment of truth to each element from  $\Omega$ , such that there exist up to  $2^{|\Omega|}$  *conceivable* worlds.

Any propositional formula  $\phi \in \mathcal{L}$ , countable set of all formulae on  $\Omega$ , is satisfied by worlds in  $[\phi] \subseteq \Omega$ . Formally, when  $\omega$  satisfies  $\phi$  we write  $\omega \models \phi$ ; that is,  $\omega \in [\phi]$  if and only if  $\omega \models \phi$ . Logical connectives  $\{\wedge, \vee, \neg\}$  - conjunction, disjunction and negation, re-

spectively - may be used to concatenate several formulae. Also,  $\top$  and  $\perp$  denote, respectively, tautology and contradiction.

A rational agent (or You) is equipped with a collection of belief states over some  $A \subseteq \Omega$ , whose elements may be equivalently defined by closed sets of formulas in a propositional logic language. Formally, a belief state over the set of all *conceivable* worlds  $A \subseteq \Omega$ , is represented by a *probability mass function* (PMF)  $P_A$ , defined as follows:

$$P_A(A) = \left\{ (\omega, P(\omega)) : \begin{array}{l} P(\omega) \geq 0, \omega \in A, \\ \sum_{\omega \in A} P(\omega) = 1 \end{array} \right\}.$$

Granular belief  $P_\Omega$  is similarly defined with respect to every  $\omega \in \Omega$ . We just write  $P$ , when the domain is clear from the context.

Let  $\mathbf{X}$  be a collection of  $n$  discrete variables,  $n \geq 1$ ,  $\omega$  corresponds to  $\mathbf{x}$ , configuration of  $\mathbf{X}$  in its joint possibility space, and  $\Omega \equiv \Omega_{\mathbf{X}}$ , while  $\mathcal{L}$  reduces to a collection of statements  $\{\phi \bowtie c : \phi \in \mathcal{L}, \bowtie \in \{=, \geq, \leq\}, c \in [0, 1]\}$ . Also,  $A$  represents any arbitrary tautology, such that any  $P_A$  is strictly positive on  $A$  (and contains zero elements only otherwise). For a given formula  $\phi$ ,

$$P(\phi) = \sum_{\mathbf{x} \in \Omega: \mathbf{x} \sim A} P(\mathbf{X} = \mathbf{x}) \mathbb{I}_{\mathbf{x} \models \phi},$$

with  $\sim$  denoting consistency among events. E.g., let  $n = 3$ ,  $\phi = \{x \wedge \neg y\}$ ,  $(x, \neg y, z) \sim [\phi]$ , whatever  $z$  in  $\Omega_Z$ , coarse partition of  $\Omega$  induced by variable  $Z$ . For the sake of brevity, in the following, we write  $P(\mathbf{x})$ , rather than  $P(\mathbf{X} = \mathbf{x})$ .

In the general case, a collection of deductively closed set of propositions, i.e., belief states, may be used to specify a *credal set* (CS)  $K(\mathbf{X})$ . Any CS  $K$  is defined by a set of linear constraints, and may be equivalently characterized as the convexification of its extreme points, denoted as  $\text{ext}[K]$ . Let  $K_1$  and  $K_2$  be any two CSs over  $\mathbf{X}$ , they are *equivalent*,  $K_1 \equiv K_2$ , if and only if  $\text{ext}[K_1] = \text{ext}[K_2]$ . For each  $x \in \Omega_X$ ,  $\underline{P}(x) = \min_{P(x) \in \text{ext}[K(X)]} P(x)$  (and  $\overline{P}(x) = \max_{P(x) \in \text{ext}[K(X)]} P(x)$ ) corresponds to the lower (and upper) envelope of CS  $K(X)$ , for any  $X \in \mathbf{X}$ . See [Walley, 1991] for details on CSs. We refer to *sharp* or *imprecise* probabilities to distinguish between  $|\text{ext}[K]| = 1$  and  $|\text{ext}[K]| > 1$ , respectively.

$K^\Phi$  denotes the subset of belief states in  $K$  that satisfy a collection of formulae  $\Phi$ . Any belief state satisfies  $\Phi$ , i.e.,  $P \models \Phi$ , whenever it holds  $P \models \phi$ , for each  $\phi \in \Phi$ . Any set  $\Phi$  is *accepted* whenever it is consistent with each  $P \in K$ , it is *rejected* if its negation only,  $\neg\Phi$ , is, or it is *neutral* if both are consistent. Let  $c \in [0, 1]$ , for a given formula  $\phi$ ,  $P \models (\phi \bowtie x)$  whenever  $P(\phi) \left( = \sum_{\mathbf{x} \sim [\phi]} P_A(\mathbf{x}) \right) \bowtie c$ ,  $\bowtie \in \{=, \leq, \geq\}$ .

For a given belief set, three main operations relevant to adjust it to satisfy any given  $\phi$ . These are

contraction, expansion and revision from AGM theory [Alchourrón et al., 1985], whose consistency postulates are mostly known from the KM reformulation in [Katzuno and Mendelzon, 1992]. Suppose an agent's knowledge base is represented by a CS  $K$  over  $\mathbf{X}$ , and let  $\phi$  be any upcoming formula, such that adjustment of  $K$  by  $\phi$  is operated by  $\circ$ . Katzuno and Mendelzon's postulates translate as follows:

**KM1**  $(K \circ \phi) \models \phi$ ,

**KM2** Let  $K \models \phi$ ,  $(K \circ \phi) \equiv (K \cup \phi)$ ,

**KM3** If  $\phi \neq \perp$ , then  $(K \circ \phi) \neq \perp$ ,

**KM4** If  $K_1 \equiv K_2$  and  $\phi_1 \equiv \phi_2$ , then  $(K_1 \circ \phi_1) \equiv (K_2 \circ \phi_2)$ ,

**KM5** If  $(K \circ \phi) \models \psi$ , then  $(K \circ (\phi \wedge \psi))$ , for any further formula  $\psi$ ,

**KM6** If  $(K \circ \phi) \models \psi$ , then  $(K \circ (\phi \wedge \psi))$  implies  $((K \circ \phi) \models \psi)$ .

Any operator  $\circ$  that satisfies all KM postulates is equivalent to a revision process based on total pre-orders [Katzuno and Mendelzon, 1992].

AGM postulates, and their KM formulation, have been followed by a massive literature on their limitations and possible extensions. Two major shortcomings of AGM theory arise when revision involves conditional formulae [Dovven and Romeijn, 2011], and in the iterated setting [Goldszmidt, 1992]. See also [Darwiche and Pearl, 1997] on additional postulates for iterated belief revision.

In the classical probabilistic framework,  $K(\mathbf{X})$  is made by a single PMF, that is  $\text{ext}[K(\mathbf{X})] = \{P(\mathbf{X})\}$ . When one or more elements from  $\mathbf{X}$  are observed,  $P$  is adjusted, i.e., updated, accordingly by standard *conditioning*. Let  $\alpha$  be any event from  $\Sigma$ , the  $\sigma$ -algebra induced by  $\Omega$ , and suppose  $(X = x)$  with  $x \in \Omega_X$  and  $X \in \mathbf{X}$ , is observed and such that  $P(x) > 0$ , it holds:

$$P(\alpha|x) = P(\alpha, x)/P(x). \quad (1)$$

A (marginal) probabilistic observation corresponds to a PMF over the countable possibility space of variable  $X \in \mathbf{X}$ . Such evidence bears an impression of the degree of reliability that is associated to each (forecasted) event, i.e., on the *evidence of uncertainty* [Peng et al., 2010]. We define probabilistic evidence as some PMF  $P'_X$  over  $\Omega_X$ , such that  $P(x) \neq P'_X(x)$  for some  $x \in \Omega_X$ . It corresponds to the collection of formulae  $\Phi_X$ , whose generic element is  $\phi_x = (\{x\} = c_x)$ ,  $c_x \in [0, 1]$ ,  $x \in \Omega_X$ , with  $\sum_{x \in \Omega_X} c_x = 1$ .  $P'_X$  may be intended

as a set of probabilistic constraints on the system modeled by  $P$  [da Rocha et al., 2008]. A general adjustment operator is the functional  $\circ$ , mapping any  $P$  to  $P^\circ$ , such that  $P^\circ \models P'_X$ . By the partiality principle mentioned above, standard revision of  $P$  by  $P'_X$  requires preservation of zero-probability events. Rationality of partiality has been advocated by several authors (e.g., [Dietrich, 2016]). The intuition is the following: Your beliefs ought to be calibrated with available evidence, if any. This way, certainty on the occurrence of event ( $X = x'$ ) requires  $P(x) = 0$ , for each  $x \neq x'$  in  $\Omega_X$ . If You accepted to change Your mind on ( $X = x$ ), then You would rather be reasonably sure about its non-occurrence, rather than certain; but then  $P(x) \neq 0$ . As a consequence, certainty on the occurrence of an event, say  $x$ , implies certainty to  $P'_X$ , since  $P'_X(x')$  is floored to zero by every  $x' \neq x$  in  $\Omega_X$ .

Kinematical mechanics for the adjustment of a belief set are intended as consistency principles, that we are willing to choose over a purely *minimal distance* based approach [Boutilier, 1996]. We introduce *probability kinematics* following Wagner's characterization [Wagner, 2002].

**Definition 1** (Probability kinematics [Jeffrey, 1965, Wagner, 2002]). *Let  $P$  and  $P^\circ$  be any two PMFs over  $(\Omega, \Sigma)$ , and let  $\Omega_X$  be a countable collection of pairwise disjoint events in  $\Sigma$ , i.e., a coarse partition of  $\Omega$  ( $\equiv \Omega_X$ ).  $P^\circ$  comes from  $P$  on  $\Omega_X$  based on probability kinematics (PK) if there exists a sequence  $P'_X(X) = \{P'_X(x) : x \in \Omega_X, \sum_{x \in \Omega_X} P'_X(x) = 1\}$  such that it holds:*

**PK1**  $P^\circ(\alpha|x) = P(\alpha|x)$ , for each  $x \in \Omega_X$ ,

**PK2**  $P^\circ(X) = P'_X(X)$ ,

for any event  $\alpha \in \Sigma$ .

In words,  $P$  is changed to agree with  $P'_X$  (PK2), while preserving relevance of each  $x \in \Omega_X$  to any event  $\alpha \in \Sigma$  (PK1).

An equivalent characterization of PK yields the well-known Jeffrey's rule:

**Definition 2** (Jeffrey's Rule [Jeffrey, 1965]). *Let  $P$ ,  $P^\circ$  and  $P'_X$  as above. Jeffrey's rule ( $\circ_J$ ) adjusts  $P$  to satisfy  $P'_X$ :*

$$(P \circ_J P'_X)(\alpha) = \sum_{x \in \Omega_X} P(\alpha, x) \frac{P'_X(x)}{P(x)}$$

We denote the Jeffrey's revision of  $P$  on  $\Omega_X$  as  $P_X^{\circ_J}$ .

Deterministic knowledge on event ( $X = x$ ) may be specified by  $P'_X(X)$  such that  $P'_X(x) = 1$  at  $x$  and zero

otherwise.<sup>2</sup> It holds:

$$(P \circ_J P'_X)(\alpha) \equiv P(\alpha|x), \quad (2)$$

where the right hand-side is just conditioning from Eq. (1). Such *hard* evidence [Valtorta et al., 2002] trivially corresponds to  $\phi = \{x\}$ ,  $x \in \Omega_X$ .

Suppose evidence is gathered conditional on some variable  $Y$  taking value  $y \in \Omega_Y$ . We define conditional (probabilistic) evidence as the collection of probabilistic statements  $P'_{X|y}(X|y)$ , such that  $P'_{X|y}(x|y) \geq 0$ , for each  $x \in \Omega_X$ , and  $\sum_{x \in \Omega_X} P'_{X|y}(x|y) = 1$ , provided  $P(y) > 0$ . Equivalently,  $\Phi_{X|y}$ , with generic element  $\phi_{x|y} = (\{y \rightarrow x\} = c_x)$ , with  $\sum_{x \in \Omega_X} c_x = 1$ . A kinematical revision rule would require the following conditions to hold:

**Definition 3** (Conditional PK [Bradley, 2005]). *Let  $P$  and  $P^\circ$  be any two PMFs on  $(\Omega, \Sigma)$ . Let  $P(y) > 0$ ,  $P^\circ$  comes from  $P$  on  $\Omega_X \times \{Y = y\}$  based on conditional probability kinematics (CPK) if there exists a sequence  $P'_{X|y}(X|y)$  as above such that it holds:*

**CPK1**  $P^\circ(\alpha|x, y) = P(\alpha|x, y)$ , for each  $x \in \Omega_X$ ,

**CPK2**  $P^\circ(\alpha|y') = P(\alpha|y')$ , for each  $y' \in \Omega_Y \setminus \{y\}$ ,

**CPK3**  $P^\circ(Y) = P(Y)$ ,

**CPK4**  $P^\circ(X|y) = P'_{X|y}(X|y)$ .

The following operator may be used to revise  $P$ , extending Jeffrey's rule to the conditional setting:

**Definition 4** (Adams' Conditioning [Bradley, 2005, Douven and Romeijn, 2011]). *Let  $P$ ,  $P^\circ$  and  $P'_{X|y}$  as above, with  $P(y) > 0$ . Operator  $\circ_A$  yields the Adams' revision ( $P_X^{\circ_A}$ ) of  $P$  that is consistent with  $P'_{X|y}$  if it is obtained as:*

$$(P \circ_A P'_{X|y})(\alpha) = P(\alpha, \neg y) + \sum_{x \in \Omega_X} P(\alpha, x, y) \frac{P'_{X|y}(x|y)}{P(x|y)}.$$

By [Bradley, 2005, Th.5], Adams' conditioning yields the unique PMF that satisfies CPK1-CPK4. Let us consider that in the running example.

<sup>2</sup>While probabilistic findings extend standard evidence, they do not necessarily result from an observation process. E.g., they may be gathered as forecasts produced by external sourced whose system of knowledge is not disclosed (e.g., betting odds), or qualitative evaluations from experts. Thorough characterization of uncertain evidence has been provided in the survey of [Mrad et al., 2015], and related works. There, probabilistic evidence is further distinguished into fixed and not-fixed. Such distinction is critical to iterated belief revision.

**Example 2** (Ex. 1 continued). *Celeste's beliefs are formalized as follows: let  $\Omega_Y = \{y \equiv \text{Swan}, \neg y \equiv \neg \text{Swan}\}$ ,  $\Omega_X = \{x_W \equiv \text{White}, x_G \equiv \text{Grey}, x_B \equiv \text{Black}\}$  and  $\Omega_Z = \{z \equiv \text{Aggressive}, \neg z \equiv \text{Tame}\}$ .*

*It holds:*

$$P(Y) = \{(y, 0.7), (\neg y, 0.3)\},$$

$$P(X|Y) = \left\{ \begin{array}{l} (x_W|y, 0.8), (x_G|y, 0.2), \\ (x_B|y, 0), (x_W|\neg y, 0.5), \\ (x_G|\neg y, 0.3), (x_B|\neg y, 0.2) \end{array} \right\},$$

$$P(Z|Y) = \left\{ \begin{array}{l} (z|y, 0.95), (\neg z|y, 0.05), \\ (z|\neg y, 0.2), (\neg z|\neg y, 0.8) \end{array} \right\}.$$

*According to Celeste's beliefs,  $P(z|x_B) = 0.2$ . Based on the sailor's words, Celeste is willing to adjust her beliefs to be consistent with  $P'_{X|y}(X|y) = \{(x_W, 0.8), (x_G, 0.1), (x_B, 0.1)\}$ . Straightforward application of Adams' conditioning is undefined, since  $P(x_B|y) = 0$ , while  $P'_{X|y}(x_B|y) \neq 0$ . The same would occur with simple Jeffrey's rule, if any  $P'_X(x) \neq 0$  was provided, given  $P(x) = 0$ , for some  $x \in \Omega_X$ . How could Celeste incorporate such reliable knowledge in her beliefs?*

Imaging was introduced by [Lewis, 1976] as a non-trivial alternative to conditioning on inconsistent events. Roughly, it represents the “*thought experiment by a minimal action*” [Fusaoka and Hiratsuka, 2003] that makes a formula consistent.

Going back to the propositional language, if some world  $\omega$  is inconsistent with formula  $\phi$ , according to a knowledge base, imaging shifts beliefs towards those that are closest to  $\phi$ , called  $\phi$ -worlds.  $\gamma(\omega, \phi)$  is called a *closest world function*, mapping  $\omega$  to its closest  $\phi$ -world; see [Lewis, 1986] for a detailed discussion. In our formalism,  $(\phi = \{x\})$  requires  $\gamma(\mathbf{x}, \phi) = (\mathbf{x} \setminus \{X\}, x) \in \Omega$ , for any  $\mathbf{x} \in \Omega$ .

**Definition 5** (Imaging [Lewis, 1976]). *Let  $P$  be any PMF over  $(\Omega, \Sigma)$ . For a given  $\phi$  and closest world function  $\gamma(\cdot, \phi)$ .  $P_\phi^{\circ_I}$  is the image of  $P$  on  $\phi$  if it is obtained by  $\circ_I$  as:*

$$(P \circ_I \{\phi\})(\alpha) = \sum_{\omega' \in \alpha} \sum_{\omega \in \Omega} P(\omega) \mathbb{I}_{\gamma(\omega, \phi) = \omega'}.$$

In Lewis' words, by imaging on event  $\phi$ , “*probability is moved around, but not created or destroyed*”, while “*every share stays as close to it as it can to the world it was originally created*” [Lewis, 1976, p. 310-311]. To summarize: i) inconsistent evidence is accounted for in the image of  $P$ , whereas conditioning is left undefined;

ii) imaging changes the whole belief set to comply with reliable knowledge  $\phi$ , while conditioning redefines the domain of  $P$ , focusing on worlds in  $\Omega$  consistent with  $\phi$ .

**Example 3.** *Let  $\mathbf{X} = \{X, Y\}$ , with  $P(x, y) = P(x, \neg y) = 0$ ,  $P(\neg x, y) = 0.6$  and  $P(\neg x, \neg y) = 0.4$ . Given  $(\phi = \{x\})$ , imaging on it yields  $(P \circ_I \{X = x\})(y) = 0.6$ , which corresponds to  $P(y)$ . If conditioning was applied,  $P(Y|x)$  would not be defined.*

*Consider  $\alpha = \{x\}$ ,  $(P \circ_I \{X = x\})(x) = 1$ :  $\circ_I$  adjusts  $P$  to always be consistent with  $\phi = \{x\}$ .*

Generalized forms of imaging were introduced in the literature, see, e.g., [Gärdenfors, 1988, Rens et al., 2016]. See also [Zhuang et al., 2017] on a unifying approach to belief adjustment.

Günther [Günther, 2017] introduced Jeffrey's imaging, that we denote as  $\circ_{jI}$ , for the generalized case of probabilistic formula  $(\phi = c)$ , with  $c \in [0, 1]$ .<sup>3</sup> Adjustment operator  $\circ_{jI}$  trivially extends *partial* imaging [Ramachandran et al., 2010].

**Definition 6** (Jeffrey's Imaging [Günther, 2017, Ramachandran et al., 2010]). *Let  $P$  be any PMF over  $(\Omega, \Sigma)$ . For a given formula  $\{\phi = c\}$ , with  $c \in [0, 1]$ ,  $P_X^{\circ_{jI}}$  comes from  $P$  by Jeffrey's imaging  $\circ_{jI}$  on  $\{\phi = c\}$  if it holds:*

$$(P \circ_{jI} \{\phi = c\})(\alpha) = P_\phi^{\circ_I}(\alpha)c + P_{\neg\phi}^{\circ_I}(\alpha)(1 - c)$$

*We denote the Jeffrey's image of  $P$  on  $\{\phi = c\}$  as  $P_\phi^{\circ_{jI}}$ .*

Both standard and Jeffrey's imaging are *homomorphic* change functions (see [Gärdenfors, 1988] and [Ramachandran et al., 2010, Obs.1], respectively), i.e., they define a structure-preserving map. A generalized characterization of Jeffrey's imaging will be provided below, within the multi-valued imprecise-probabilistic framework (see Definition 9).

Just like Your beliefs may be encoded by a CS  $K$  on  $\Omega$ , probabilistic evidence may come as a (closed and convex) collection of PMFs  $K'_X$  on  $\Omega_X$ , i.e., a CS that we call *credal* (or imprecise) evidence. This latter generalizes sharp probabilistic evidence to the case  $|\text{ext}[K'_X(X)]| \geq 1$ :

$$K'_X(X) = \{P(x) : \underline{P}(x) \leq P(x) \leq \overline{P}(x), x \in \Omega_X\}.$$

$K'_X$  may be equivalently specified by the collection of formulae  $\phi_x = (\{x\} \bowtie c_x)$ ,  $c_x \in [0, 1]$ , for each  $x \in \Omega_X$ , provided  $\sum_{x \in \Omega_X} c_x \bowtie 1, \bowtie \in \{=, \leq, \geq\}$ .<sup>4</sup>

<sup>3</sup>Günther's definition assumes  $c \in (0, 1)$ .

<sup>4</sup>To guarantee  $P'_X(x) \in [0, 1]$ , we also require  $P'_X(x) \leq 0$  and  $P'_X(x) \geq 1$  always reduce to equalities.

Our contributions will tackle probabilistic belief adjustment by (possibly inconsistent) sharp or imprecise probabilities, following an approach based on the imaginary counterparts of PK. This is analogous to what has been done in [Ma et al., 2011, Zhou et al., 2014] within the framework of evidence theory.

Following [Zhou et al., 2014], we are willing to check a further consistency requirement, that would reproduce Eq. (2). In this way, any adjustment *kinematical* operator reduces to some form of conditioning when probabilistic evidence strengthens to full observation.

### 3 IMAGINARY KINEMATICS

We lay bare the kinematical conditions that ought to be satisfied by any belief adjustment operator, when (possibly) inconsistent probabilistic evidence is gathered.<sup>5</sup>

Let us start with simple probabilistic evidence:  $P'_X$  on  $\Omega_X$ , such that  $|\Omega_X| \geq 2$ . Imaginary kinematics can be introduced as a counterpart of PK for imaging.

**Definition 7** (Imaginary Kinematics). *Any joint CS  $K^\circ$  on  $\mathbf{X}$  comes from  $K$  by imaginary kinematics (IK) on a (possibly inconsistent) credal evidence  $K'_X$  on variable  $X$  whenever it holds:*

$$\mathbf{IK1} \quad K^\circ(\alpha|x) \supseteq K_x^{\circ I}(\alpha), \text{ for any } \alpha \in \Sigma \text{ and each } x \in \Omega_X,$$

$$\mathbf{IK2} \quad K^\circ(X) \models \Phi_X,$$

$$\mathbf{IK3} \quad K^\circ(X) \equiv K_x^{\circ I}(X) \text{ whenever } c_x = 1 \text{ for some } x \in \Omega_X.$$

Analogously, based on Definition 3, we provide an imaginary characterization of CPK defined as follows.

**Definition 8** (Imaginary Conditional Kinematics). *Let  $K$ ,  $K^\circ$  as above, such that  $\underline{P}(y) > 0$  for each  $y \in \Omega_Y$ .  $K^\circ$  comes from  $K$  on  $\Omega_X \times \{Y = y\}$  based on imaginary conditional kinematics (ICK) if there exists a (possibly inconsistent) sequence  $P'_{X|y}$  such that it holds:*

$$\mathbf{ICK1} \quad K^\circ(\alpha|x, y) \supseteq K_x^{\circ I}(\alpha|y), \text{ for each } x \in \Omega_X,$$

$$\mathbf{ICK2} \quad K^\circ(\alpha|y') \equiv K(\alpha|y'), \text{ for each } y' \in \Omega_Y \setminus \{y\},$$

$$\mathbf{ICK3} \quad K^\circ(Y) \equiv K(Y),$$

$$\mathbf{ICK4} \quad K^\circ(x|y) \models \Phi_{X|y},$$

$$\mathbf{ICK5} \quad K^\circ(X|y) \equiv K_x^{\circ I}(X), \text{ whenever } c_x = 1, \text{ for some } x \in \Omega_X.$$

<sup>5</sup>With imprecise probabilities, inconsistency occurs when  $\overline{P}(x) = 0$  and positive evidence is provided for some  $x \in \Omega_X$ .

## 4 KINEMATICAL IMAGINARY ADJUSTMENT RULES

For any  $\alpha \in \Sigma$ , if a CS  $K$  over  $\mathbf{X}$  is used to represent Your beliefs, imaging on  $(\phi = \{x\})$  extends to:

$$(K \circ_I \{x\})(\alpha) = \{P_x^{\circ I}(\alpha) = (P \circ_I \{x\})(\alpha), P \in K\},$$

so that the lower envelope of  $K$ 's image on  $\{x\}$ , denoted as  $K_x^{\circ I}$ , at  $\alpha$ , writes:

$$\underline{P}_x^{\circ I}(\alpha) = \min_{P(\mathbf{x}) \in K(\mathbf{X})} \sum_{\mathbf{x}' \sim \alpha} \sum_{\mathbf{x} \in \Omega_{\mathbf{X}}} P(\mathbf{x}) \mathbb{1}_{\gamma(\mathbf{x}, \mathbf{x}) = \mathbf{x}'}$$

By [Rens et al., 2016, Th.1],  $K_x^{\circ I}$  may be efficiently obtained by taking the convex hull (CH) of the images on  $\{x\}$  of each  $P \in \text{ext}[K]$ . Since the image of each  $P \in \text{ext}[K]$  at  $\alpha = \{x'\}$  trivially corresponds to  $P_x^{\circ I}(x') = 0$ ,<sup>6</sup> whenever  $x' \neq x$ , refinement of  $K_x^{\circ I}(X)$  degenerates to a single PMF such that  $P'_X(x) = 1$ , and zero otherwise. With a small abuse of notation, this yields the following:

$$K_x^{\circ I}(\mathbf{X}) \equiv \begin{cases} 1 \cdot K(\mathbf{X} \setminus \{X\}) & \mathbf{x} \sim x, \\ 0 & \text{otherwise.} \end{cases}$$

**Example 4.** *Let  $K$  be a CS over  $\mathbf{X} = \{X, Y\}$  specified by probability intervals as follows:*

$$K \begin{pmatrix} x_1, y_1 \\ x_1, y_2 \\ x_2, y_1 \\ x_2, y_2 \\ x_3, y_1 \\ x_3, y_2 \end{pmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0.15 - 0.35 \\ 0.25 - 0.49 \\ 0 - 0.45 \\ 0.03 - 0.5 \end{bmatrix}.$$

*It is easy to see  $\underline{P}_{x_1}^{\circ I}(y_j) = \underline{P}(y_j)$ ,  $j = 1, 2$ , while  $\overline{P}_{x_1}^{\circ I}(x_k) = 0$ ,  $k = 2, 3$ .*

### 4.1 STANDARD PROBABILISTIC EVIDENCE

We start from the case of sharp probabilistic evidence on  $\Omega_X$ , i.e.,  $K'_X(X) = \{P'_X(X)\}$ . The following adjustment operator extends Definition 6. As we did before for imaging, notation that is used with sharp beliefs applies to the generalized case of belief sets, when  $|\text{ext}[K]| \geq 1$ .

**Definition 9** ((Probabilistic) Jeffrey's Imaging). *Let  $K$  be any joint CS over  $\mathbf{X}$  as above. Suppose probabilistic evidence  $P'_X$  is provided over a (possibly) inconsistent collection of events, i.e.,  $\overline{P}(x) = 0$ , whereas  $P'_X(x) > 0$ , for some  $x \in \Omega_X$ ,  $X \in \mathbf{X}$ . For any event  $\alpha$ ,  $K_X^{\circ j I}$  is the probabilistic Jeffrey's image of  $K$  if it holds:*

$$K_X^{\circ j I}(\alpha) = \{P_X^{\circ j I}(\alpha) = \sum_{x \in \Omega_X} P_x^{\circ I}(\alpha) P'_X(x), \\ P_x^{\circ I} \in K_x^{\circ I}, x \in \Omega_X\}.$$

<sup>6</sup>By definition,  $P_x^{\circ I}(x) = \sum_{\mathbf{x} \in \Omega} P(\mathbf{x}) = 1$ .



That is,  $K_X^{\circ_{jI}}(\alpha) = (K \circ_{jI} P'_X)(\alpha)$ , for any  $\alpha \in \Sigma$ .

The following result holds (the proofs of all the theorems are in the appendix).

**Theorem 1.** *Jeffrey's imaging is based on IK, and IK1 is strongly satisfied, i.e.,  $\models$  may be replaced by  $\equiv$ .*

**Corollary 1.** *Given sharp probabilistic knowledge on  $\Omega_X$ , the Jeffrey's image of any CS may be equivalently specified by the convexification of all PMFs  $P^\circ$ , each defined as follows:*

$$P^\circ(\alpha) = \sum_{x \sim \alpha} P'_X(x) P_i(\alpha) \quad \forall P \in \text{ext}[K].$$

It is easy to see that standard imaging is also trivially based on IK.

**Example 5.** *Consider the same setup as in Example 4, and suppose  $P'_X(X) = \{(x_1, 0.3), (x_2, 0), (x_3, 0.7)\}$ . By Jeffrey's imaging on  $P'_X$ , we obtain  $K_X^{\circ_{jI}}(Y) \equiv K(Y)$ , while  $P_X^{\circ_{jI}}(y_j|x_i) \equiv K_{x_i}^{\circ_I}(y_j)$ ,  $i = 1, 2, 3$ ,  $j = 1, 2$ . Also,  $K_X^{\circ_{jI}}(X) \models P'_X(X)$ , and  $K_X^{\circ_{jI}}$  is equivalent to the convex hull of PMFs  $P^\circ$ , defined as:*

$$P^\circ(x, y) = P'_X(x)P(y),$$

for each  $x \in \Omega_X, y \in \Omega_Y$  and  $P \in \text{ext}[K]$ .

## 4.2 SHARP CONDITIONAL EVIDENCE

We now introduce Adams' imaging as an adjustment operator  $\circ_{aI}$ , that extends  $\circ_{jI}$  to the conditional case, just like revision rule  $\circ_A$  extends  $\circ_J$ .

**Definition 10** (Adams' Imaging). *Let  $K$  be any joint CS on  $(\Omega, \Sigma)$  such that  $\underline{P}(y) > 0$ ,  $Y \in \mathbf{X}$ , and let conditional probabilistic knowledge  $P'_{X|y}$  on  $\Omega_X \times \{Y = y\}$ .  $K_X^{\circ_{aI}}$ , the Adams' image of  $K$  on  $P'_{X|y}$ , comes from  $K$  by Adams' imaging  $\circ_{aI}$ , if it holds:*

$$\begin{aligned} K_X^{\circ_{aI}}(\alpha) &= \\ \{P_X^{\circ_{aI}}(\alpha) &= P(\alpha, \neg y) + \sum_{x \in \Omega_X} P_x^{\circ_I}(\alpha, y) P'_{X|y}(x|y), \\ P &\in K, P_x^{\circ_I} \in K_x^{\circ_I}, x \in \Omega_X\}. \end{aligned}$$

*I.e.,  $K_X^{\circ_{aI}}(\alpha) = (K \circ_{aI} P'_{X|y})(\alpha)$ , for any  $\alpha \in \Sigma$ .*

When  $|\text{ext}[K]| = 1$ , from previous considerations, Adams' imaging reduces to the following:

$$P_X^{\circ_{aI}}(\alpha) = P(\alpha, \neg y) + \sum_{x \in \Omega_X} P_x^{\circ_I}(\alpha, y) P'_{X|y}(x|y). \quad (3)$$

**Example 6** (Ex. 1 continued). *The Adams' image on  $P'_{X|y}$  of Celeste's beliefs on  $\Omega_X \times \Omega_Z$  is the following:*

$$P_X^{\circ_{aI}} \begin{pmatrix} x_W z \\ x_W \neg z \\ x_G z \\ x_G \neg z \\ x_B z \\ x_B \neg z \end{pmatrix} = \begin{bmatrix} 0.5620 \\ 0.1480 \\ 0.0845 \\ 0.0755 \\ 0.0785 \\ 0.0515 \end{bmatrix}.$$

*It holds  $P_X^{\circ_{aI}}(X|y) = P'_{X|y}(X|y)$  and  $P_X^{\circ_{aI}}(Y, Z) = P(Y, Z)$ . Adjustment of her beliefs by  $P'_{X|y}$  yields  $P_X^{\circ_{aI}}(z|x_B) \approx 0.6$ , whereas  $P(z|x_B) = 0.2$ . Thus, Celeste rapidly swims back to shore.*

As a remark, inconsistency of  $P'_{X|y}(x|y)$ , for some  $x \in \Omega_X$ , with respect to any PMF  $P$ , may refer to either i)  $P(x|y) = 0$ , while  $P(y) > 0$ , (this is just the case of Adams' imaging above), or ii)  $P(y) = 0$  in the first place, and possibly  $P(x|y) = 0$ . We argue case ii) deserves some caution, since full inconsistency of event  $(Y = y)$  is likely not to yield any further conjecturing on related events, from a modeler's perspective. E.g., You are certain that no alien lives on Mars. Is it worth include Your belief on the alien having long hair in Your belief base, provided that You are not admitting the alien's existence upstream? On the other hand, we reckon arguments may be easily raised against our position, starting from our proposed running example. Still, if no evidence is provided on  $\Omega_Y$ , a cautious approach would require application of an iterated procedure. We leave this point for future work.

It is now straightforward to note that Adams' imaging generalizes Jeffrey's imaging to the conditional setting.

**Theorem 2.** *Adams' imaging is based on ICK, and ICK1 is strongly satisfied. Eq. (3) strongly satisfies all conditions.*

Analogously to Corollary 1, it might be easily shown that  $K_X^{\circ_{aI}}$  at any  $x \sim y$  is equivalent to the CS obtained taking the product of sharp assessment  $P'_{X|y}$  and the marginalization over variable  $X$  of the original belief set  $K$ . We also provide the following additional result, which extends [Rens et al., 2016].

**Theorem 3.** *Both Jeffrey's and Adams' imaging satisfy consistency axioms KM1, KM3 and KM4. KM2, KM5 and KM6 are satisfied only if  $K$  is degenerate at  $(X|y)$ , i.e.,  $|K(X|y)| = 1$  (and at  $(Z|w)$ , for KM5 and KM6).*

## 4.3 CREDAL JEFFREY'S IMAGING

When beliefs are expressed as a joint CS over  $\mathbf{X}$ , adjustment by a single reliable PMF requires simultaneous

computation of all bounds spanned by the updating of each  $P \in K$ . Also in this case, adjustment may be restricted to the PMFs in  $\text{ext}[K]$  only, and their convex hull consequently considered.

**Definition 11** (Credal Jeffrey’s Imaging). *Given CS  $K$  over  $\mathbf{X}$  and credal probabilistic evidence  $K'_X(X)$ , we define credal Jeffrey’s imaging  $\circ_{cjI}$  as the functional mapping  $K$  to CS  $K_X^{\circ_{cjI}}$ , consistent with  $K'_X(X)$  as follows:*

$$K_X^{\circ_{cjI}}(\alpha) = \left\{ P^\circ(\alpha) = (P \circ_{jI} P'_X)(\alpha), \begin{array}{l} P(\mathbf{X}) \in K(\mathbf{X}), \\ P'_X \in K'_X(X) \end{array} \right\}$$

The following result generalizes Theorem 1.

**Theorem 4.** *Given (possibly) inconsistent credal probabilistic evidence, credal Jeffrey’s imaging yields the unique joint CS based on IK.*

Table 1: Summary of belief adjustment rules/properties.

RULE	$\Phi_*$	KINEMATICS
$\circ_J$	$\phi_x = c_x, \forall x$	PK
$\circ_A$	$\{y \rightarrow x\} = c_x, \forall x$	CPK
$\circ_{jI}$	$\{x\} = c_x, \forall x$	IK (Th. 1)
$\circ_{aI}$	$\{y \rightarrow x\} = c_x, \forall x$	ICK (Th. 2)
$\circ_{cjI}$	$\{x\} \bowtie c_x, \forall x$	IK (Th. 4)

## 5 CONCLUSIONS AND FUTURE WORK

We introduced adjustment operators based on Lewis’ imaging functional, to deal with probabilistic inconsistent evidence, in a generalized setting of imprecise probabilities, specified by credal sets. These are summarized in Table 1. We point out that the revision rules (conditioning, Jeffrey’s rule and Adams’ conditioning) are not fully general due to partiality, whereas the remaining succeed in adjusting a given belief set following inconsistent observations.

Further generalization to the case of credal conditional probabilistic evidence is not straightforward as the adjustment process would likely incur in dilating mechanics, resulting in detrimental loose inclusion relationships. This reasoning also applies to the iterated framework, where additional considerations must be formulated on the role evidence plays on the adjustment process. As a future work we will tackle this sort of scenarios. Besides that, we also intend to compare our approach against methods based on lexicographic probabilities (e.g., [Benavoli et al., 2017]) as well as applying

these ideas to probabilistic graphical models by extending what have been already done for Jeffrey’s rule in [Marchetti and Antonucci, 2018].

## A PROOFS

This appendix provides proofs to the results stated in the paper.

**Proof of Th. 1.** *To prove  $\circ_{jI}$  is based on IK, we must check it produces a CS that satisfies IK1-IK3. Motivated by [Rens et al., 2016, Th.1], we restrict our attention toward the extreme points of  $K$ . Without loss of generality, let  $\mathbf{X} = \{X, Y\}$ . Each extreme point of  $K(\mathbf{X})$ , say  $P_{j,k} \in \text{ext}[K]$ , may be equivalently specified as:*

$$P_{j,k}(x, y) = P(x|x'_j)P(y|x, y'_k), \quad (4)$$

with  $P(y|x, y'_k)$  is set equal to zero whenever it is undefined and  $P(x|x'_k) = 0$ .<sup>7</sup>  $X'$  and  $Y'$  are uniformly distributed auxiliary random variables, used to index  $K$ ’s extreme points at  $X$  and at  $Y|X$ , respectively. This way, for a given ordering,

$$\begin{aligned} P(x|x'_1) &= \sum_{y'_k, y} P(x|x'_1)P(y|x, y'_k)P(y'_k) \\ &= \underline{P}(x), \end{aligned}$$

and  $\underline{P}(x, y) = P(x|x'_1)P(y|x, y'_1)$ .

It holds:

$$\begin{aligned} \underline{P}_x^{\circ_I}(x) &= P_x^{\circ_I}(x|x'_1) \\ &= \sum_{y'_k, y, x} P(x|x'_1)P(y|x, y'_k)P(y'_k) \\ &\leq 1. \end{aligned}$$

Since  $P_x^{\circ_I}(x'|x'_1) = 0$ , for any  $x' \neq x$  in  $\Omega_X$ , refinement of  $K_x^{\circ_I}(x)$  degenerates at 1. If  $P'_X \models (\phi = \{x\})$ , IK3 is satisfied.

When a non-trivial PMF is provided, i.e.,  $P(x) > 0$  for at least two elements in  $\Omega_X$ , it holds:

$$\begin{aligned} P_X^{\circ_{jI}}(x|x'_1) &= \left[ \sum_{y'_k, y, x} P(x|x'_1)P(y|x, y'_k)P(y'_k) \right] P'_X(x) \\ &\leq P'_X(x), \end{aligned}$$

and similarly  $P_X^{\circ_{jI}}(x|x'_{|\text{ext}[K]}) \geq P'_X(x)$ . This proves IK2 since  $K_X^{\circ_{jI}}(X) \ni P'_X(X)$ .

<sup>7</sup>As a remark,  $P(x) = 0$  does not necessarily imply  $P(y|x) = 0$ , in De Finetti’s view.

Proof of IK1 is also straightforward:

$$\begin{aligned} P_X^{\circ_{aI}}(y|x, y'_1) &= \frac{\left[ \sum_{x, x'_j} P(x|x'_j)P(y|x, y'_1) \right] P'_X(x)}{P'_X(x)} \\ &= \sum_{x, x'_j} P(x|x'_j)P(y|x, y'_1) \\ &= P_x^{\circ_I}(y|y'_1) \end{aligned}$$

Analogous reasoning applies to the upper envelope, and thus  $K_X^{\circ_{aI}}(Y|x) \equiv K_x^{\circ_I}(Y)$ . This ends the proof.  $\square$

**Proof of Th. 2.** To prove  $\circ_{aI}$  is based on ICK we need to check ICK1-ICK5 are satisfied by  $K^\circ = (K \circ_{aI} P'_{X|y})$ . When  $|\text{ext}[K]| = 1$ , ICK1-ICK5 reduce to the following:

- ICK1'**  $P^\circ(\alpha|x, y) = P_x^{\circ_I}(\alpha|y)$ , for each  $x \in \Omega_X$ ,
- ICK2'**  $P^\circ(\alpha|y') = P(\alpha|y')$ ,
- ICK3'**  $P^\circ(Y) = P(Y)$ ,
- ICK4'**  $P^\circ(X|y) = P'_{X|y}(X|y)$ ,
- ICK5'**  $P^\circ(X|y) = P_x^{\circ_I}(X|y)$ , whenever  $P'_{X|y}(x|y) = 1$  for some  $x \in \Omega_X$ .

We first prove consistency points ICK4' and ICK5'. Let  $P'_{X|y}$  be any PMF on  $\Omega_X \times \{Y = y\}$ , it holds:

$$\begin{aligned} P_X^{\circ_{aI}}(x|y) &= \frac{P_x^{\circ_I}(y)P'_{X|y}(x|y)}{\sum_x P_x^{\circ_I}(y)P'_{X|y}(x|y)} \\ &= P'_{X|y}(x|y) \end{aligned}$$

since  $P_x^{\circ_I}(x, y) = P_x^{\circ_I}(y) = P(y)$ , whatever  $x \in \Omega_X$ . Also,  $\sum_x P'_{X|y}(x|y) = 1$  by definition. If  $P'_{X|y}(x|y) = 1$  for some  $x$ ,  $P_X^{\circ_{aI}}(x|y) = 1, 0$  otherwise. The following holds:

$$\begin{aligned} \underline{P}^\circ(x|y) &= \min_{P^\circ \in \text{ext}[K^\circ]} P^\circ(x|y) \\ &= P'_{X|y}(x|y) \frac{\overline{P}_x^{\circ_I}(y)}{\underline{P}_x^{\circ_I}(y)} \\ &\leq P'_{X|y}(x|y). \end{aligned}$$

Similarly,  $\overline{P}^\circ(X|y) \geq P'_{X|y}(X|y)$ , for each  $P^\circ \in \text{ext}[K^\circ]$ .

We now prove condition ICK1 (and thus ICK1') is satisfied by  $\circ_{aI}$ . Without loss of generality, let  $\mathbf{X} = \{X, Y, Z\}$ . It holds:

$$\begin{aligned} \underline{P}^\circ(z|x, y) &= \frac{P'_{X|y}(x|y)\overline{P}_x^{\circ_I}(z, y)}{P'_{X|y}(x|y)\overline{P}_x^{\circ_I}(y)} \\ &= \underline{P}_x^{\circ_I}(z|y). \end{aligned}$$

As for point ICK2 (and ICK2'), it trivially holds by Definition 10:

$$\underline{P}^\circ(z|y') = \underline{P}(z|y').$$

for any  $y' \neq y$ . ICK3' is proved analogously, since  $P_X^{\circ_{aI}}(y) = 1 - P(\neg y) = 1 - \sum_{y' \neq y} P_X^{\circ_{aI}}(y')$ . Similarly, fulfillment of ICK3 may be derived by the conjugacy relation [Walley, 1991].  $\square$

**Proof of Th. 3.** Consider CS  $K$  and conditional probabilistic evidence  $P'_{X|y}(X|y)$ . To avoid cumbersome notation, we write  $\circ$  to denote  $\circ_{aI}$  throughout the proof. Also, we refer to general formula  $\phi = c$  to denote both  $\phi_x$  and  $\phi_{x|y}$ .

KM1 and KM3 follow from IK2 and ICK4 (cfr Th.1 and Th.2, respectively).

We prove KM2 is not satisfied under general conditions. Consider the lower envelope of  $K$  at  $(x|y)$ . If  $K \models P'_{X|y}$ , it holds:

$$\underline{P}(x|y) \leq P'_{X|y}(x|y)$$

by definition, and  $(K \cup P'_{X|y}) = K$ . From previous discussion, we expect  $(K \circ P'_{X|y}) \supseteq K$ , equality holding if and only if  $K(\mathbf{X})$  may be equivalently specified as the product of sharp conditional assessment on  $\Omega_X \times \{Y = y\}$  and CS over  $(\mathbf{X} \setminus \{Y\}, y)$ . Same reasoning applies to KM5 and KM6. These three postulates are satisfied if and only if  $K$  is already degenerate at the domain of probabilistic evidence, and consistent with it already.

Postulate KM4 holds by [Rens et al., 2016, Th.1].  $\square$

The following preliminary result holds:

**Lemma 1.** Let  $K$  be a joint CS over  $\mathbf{X}$ , and let  $K'_X$  denote a credal probabilistic finding, gathered on  $\Omega_X$ . For any event  $\alpha$ , the Jeffrey's image  $K_X^{\circ_{c_j I}}(\alpha)$  of  $K(\alpha)$  on  $K'_X(X)$  satisfies the following:

$$K_X^{\circ_{c_j I}}(\alpha) \supseteq K_X^{\circ_{c_j I}}(\alpha|x) \supseteq K_x^{\circ_I}(\alpha)$$

for any  $\alpha \in \Sigma$ . Equality holds when  $|K'(X)| = 1$ .

**Proof of Lemma 1.** Let  $\mathbf{X} = \{X, Y\}$  and  $K$  be any CS over  $\Omega$ .  $K'_X$  is gathered on  $\Omega_X$ , to adjust  $K$  accordingly. By definition of credal Jeffrey's imaging, it holds:

$$\begin{aligned} \min_{P_X^{\circ_{c_j I}} \in K_X^{\circ_{c_j I}}} P_X^{\circ_{c_j I}}(y|x) &= \min_{P(y) \in K(Y)} P(y) \frac{P'_X(x)}{\overline{P}'_X(x)} \\ &\leq \min_{P(y) \in K(Y)} P(y), \end{aligned}$$

and analogously for the upper envelope, with  $\geq$ . This proves the rightest inclusion relationship:  $K_X^{\circ_{cjI}}(Y|x) \supseteq K_x^{\circ_I}(Y) (\equiv K(Y))$ . We now prove inclusion of  $K_X^{\circ_{cjI}}(y|x)$  by  $K_X^{\circ_{cjI}}(y)$ :

$$\begin{aligned} \frac{P_X^{\circ_{cjI}}(y)}{P_X^{\circ_{cjI}}(y|x)} &= \frac{P(y) \sum_x P'_X(x)}{P(y) \frac{P'_X(x)}{P_X(x)}} \\ &= \overline{P}'_X(x) \sum_{x' \neq x} P'_X(x') \\ &\leq 1. \end{aligned}$$

Hence  $P_X^{\circ_{cjI}}(y) \leq P_X^{\circ_{cjI}}(y|x)$ , for any  $x \in \Omega_X$ ,  $y \in \Omega_Y$ .  $\overline{P}_X^{\circ_{cjI}}(y) \geq \overline{P}_X^{\circ_{cjI}}(y|x)$  is derived analogously. Equality holds when  $K'_X(X) = \{P'_X(X)\}$  as  $P'_X(x) = \overline{P}'_X(x)$ , for each  $x \in \Omega_X$ , summing to one. □

**Proof of Th. 4.** Given a joint CS  $K$  over  $\mathbf{X}$  and  $K'_X$ , let  $\circ$  denote credal Jeffrey's imaging. IK1 is satisfied by Lemma 1. IK2 is also satisfied as it holds:

$$P_X^{\circ_{cjI}}(x) = 1 \cdot P'_X(x),$$

for each  $x \in \Omega_X$ . And analogously for  $\overline{P}_X^{\circ_{cjI}}(X)$ . When  $K'_X(X) = \{P'_X(X)\}$  such that  $P'_X(x) = 1$ , IK3 is satisfied since  $\circ_{cjI}$  reduces to  $\circ_{jI}$ . □

## References

- [Alchourrón et al., 1985] Alchourrón, C. E., Gärdenfors, P., and Makinson, D. (1985). On the logic of theory change: partial meet contraction and revision functions. *Journal of Symbolic Logic*, 50(2):510–530.
- [Benavoli et al., 2017] Benavoli, A., Facchini, A., Vicente-Perez, J., and Zaffalon, M. (2017). A polarity theory for sets of desirable gambles. In *Proc. Isipta'17 Int. Symposium on Imprecise Probability: Theories and Applications*, pages 1–12. PJMLR.
- [Boutilier, 1996] Boutilier, C. (1996). Iterated revision and minimal change of conditional beliefs. *Journal of Philosophical Logic*, 25(3):263–305.
- [Bradley, 2005] Bradley, R. (2005). Radical probabilism and Bayesian conditioning. *Philosophy of Science*, 72(2):342–364.
- [Cozic, 2011] Cozic, M. (2011). Imaging and sleeping beauty: a case for double-halfers. *International Journal of Approximate Reasoning*, 52(2):137–143.
- [da Rocha et al., 2008] da Rocha, J., Guimaraes, A., and de Campos, C. (2008). Dealing with soft evidence in credal networks. In *Proceedings of Conferencia Latino-Americana de Informatica*.
- [Darwiche and Pearl, 1997] Darwiche, A. and Pearl, J. (1997). On the logic of iterated belief revision. *Artificial intelligence*, 89(1-2):1–29.
- [Dietrich, 2016] Dietrich, F. (2016). Judgment aggregation and agenda manipulation. *Games and Economic Behavior*, 95:113–136.
- [Douven and Romeijn, 2011] Douven, I. and Romeijn, J.-W. (2011). A new resolution of the Judy Benjamin problem. *Mind*, 120(479):637–670.
- [Fusaoka and Hiratsuka, 2003] Fusaoka, A. and Hiratsuka, S. (2003). On a linear representation for quantitative belief revision. *Transactions of the Japanese Society for Artificial Intelligence*, 18:75–85.
- [Gärdenfors, 1988] Gärdenfors, P. (1988). *Knowledge in flux: Modeling the dynamics of epistemic states*. The MIT press.
- [Goldszmidt, 1992] Goldszmidt, M. (1992). Qualitative probabilities: a normative framework for common-sense reasoning. Technical report, California University of Los Angeles, Dept. of Computer Science.
- [Günther, 2017] Günther, M. (2017). Learning conditional information by Jeffrey imaging on Stalnaker conditionals. *Journal of Philosophical Logic*, pages 1–26.
- [Jeffrey, 1965] Jeffrey, R. C. (1965). Ethics and the logic of decision. *The Journal of Philosophy*, 62(19):528–539.
- [Katzuno and Mendelzon, 1992] Katzuno, A. and Mendelzon, A. (1992). Propositional knowledge base revision and nonmonotonicity. *P. Gärdenfors ed.: Belief revision*.
- [Lewis, 1976] Lewis, D. (1976). Probabilities of conditionals and conditional probabilities. In *Iffs*, pages 129–147. Springer.
- [Lewis, 1986] Lewis, D. (1986). On the plurality of worlds. *London*, 5:221–36.
- [Ma et al., 2011] Ma, J., Liu, W., Dubois, D., and Prade, H. (2011). Bridging Jeffrey's rule, AGM revision and Dempster conditioning in the theory of evidence. *International Journal on Artificial Intelligence Tools*, 20(04):691–720.

- [Marchetti and Antonucci, 2018] Marchetti, S. and Antonucci, A. (2018). Reliable uncertain evidence modeling in Bayesian networks by credal networks. In *Proceedings of the 31st International FLAIRS Conference*, pages 513–518. AAAI.
- [Mrad et al., 2015] Mrad, A. B., Delcroix, V., Piechowiak, S., Leicester, P., and Abid, M. (2015). An explication of uncertain evidence in Bayesian networks: likelihood evidence and probabilistic evidence. *Applied Intelligence*, 43(4):802–824.
- [Peng et al., 2010] Peng, Y., Zhang, S., and Pan, R. (2010). Bayesian network reasoning with uncertain evidences. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 18(05):539–564.
- [Ramachandran et al., 2010] Ramachandran, R., Nayak, A. C., and Orgun, M. A. (2010). Belief erasure using partial imaging. In *Australasian Joint Conference on Artificial Intelligence*, pages 52–61. Springer.
- [Rens et al., 2016] Rens, G., Meyer, T., and Casini, G. (2016). On revision of partially specified convex probabilistic belief bases. In *Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI-16)*.
- [Valtorta et al., 2002] Valtorta, M., Kim, Y.-G., and Vomlel, J. (2002). Soft evidential update for probabilistic multiagent systems. *International Journal of Approximate Reasoning*, 29(1):71–106.
- [Wagner, 2002] Wagner, C. G. (2002). Probability kinematics and commutativity. *Philosophy of Science*, 69(2):266–278.
- [Walley, 1991] Walley, P. (1991). *Statistical reasoning with imprecise probabilities*. Chapman & Hall.
- [Zhou et al., 2014] Zhou, C., Wang, M., and Qin, B. (2014). Belief-kinematics Jeffrey’s rules in the theory of evidence. In *Proceedings of UAI 2014*, pages 917–926.
- [Zhuang et al., 2017] Zhuang, Z., Delgrande, J., Nayak, A., and Sattar, A. (2017). A unifying framework for probabilistic belief revision. In *Proceedings of IJCAI-17*.

---

# From Deterministic ODEs to Dynamic Structural Causal Models

---

**Paul K. Rubenstein\***  
Department of Engineering  
University of Cambridge  
pkr23@cam.ac.uk

**Stephan Bongers**  
Informatics Institute  
University of Amsterdam  
S.R.Bongers@uva.nl

**Bernhard Schölkopf**  
Max-Planck Institute for  
Intelligent Systems, Tübingen  
bs@tue.mpg.de

**Joris M. Mooij**  
Informatics Institute  
University of Amsterdam  
J.M.Mooij@uva.nl

## Abstract

Structural Causal Models are widely used in causal modelling, but how they relate to other modelling tools is poorly understood. In this paper we provide a novel perspective on the relationship between Ordinary Differential Equations and Structural Causal Models. We show how, under certain conditions, the asymptotic behaviour of an Ordinary Differential Equation under non-constant interventions can be modelled using Dynamic Structural Causal Models. In contrast to earlier work, we study not only the effect of interventions on equilibrium states; rather, we model asymptotic behaviour that is *dynamic* under interventions that vary in time, and include as a special case the study of static equilibria.

## 1 INTRODUCTION

Ordinary Differential Equations (ODEs) provide a universal language to describe deterministic systems via equations that determine how variables change in time as a function of other variables. They provide an immensely popular and highly successful modelling framework, with applications in many diverse disciplines, such as physics, chemistry, biology, and economy. They are *causal* in the sense that at least in principle they allow us to reason about interventions: any external intervention in a system—e.g., moving an object by applying a force—can be modelled using modified differential equations by, for instance, including suitable forcing terms. In practice, of course, this may be arbitrarily difficult.

Structural Causal Models (SCMs, also known as Structural Equation Models) are another language capable of

---

\*Also affiliated with Max Planck Institute for Intelligent Systems, Tübingen.

describing causal relations and interventions and have been widely applied in the social sciences, economics, genetics and neuroscience (Pearl, 2009; Bollen, 2014). One of the successes of SCMs over other causal frameworks such as causal Bayesian networks, for instance, has been their ability to express cyclic causal models (Spirtes, 1995; Mooij et al., 2011; Hyttinen et al., 2012; Voortman et al., 2010; Lacerda et al., 2008; Bongers et al., 2018).

We view SCMs as an intermediate level of description between the highly expressive differential equation models and the probabilistic, non-causal models typically used in machine learning and statistics. This intermediate level of description ideally retains the benefits of a data-driven statistical approach while still allowing a limited set of causal statements about the effect of interventions. While it is well understood how an SCM induces a statistical model (Bongers et al., 2018), much less is known about how a differential equation model—our most fundamental level of modelling—can imply an SCM in the first place. This is an important question because if we are to have models of a system on different levels of complexity, we should understand how they relate and the conditions under which they are consistent with one another.

Indeed, recent work has begun to address the question of how SCMs arise naturally from more fundamental models by showing how, under strong assumptions, SCMs can be derived from an underlying discrete time difference equation or continuous time ODE (Iwasaki and Simon, 1994; Dash, 2005; Lacerda et al., 2008; Voortman et al., 2010; Mooij et al., 2013; Sokol and Hansen, 2014). With the exception of (Voortman et al., 2010) and (Sokol and Hansen, 2014), each of these methods assume that the dynamical system comes to a static equilibrium that is independent of initial conditions, with the derived SCM describing how this equilibrium changes under intervention. More recently, the more general case in which the equilibrium state may depend on the initial conditions has been addressed (Bongers and Mooij, 2018; Blom and Mooij, 2018).

If the assumption that the system reaches a static equilibrium is reasonable for a particular system under study, the SCM framework can be useful. Although the derived SCM then lacks information about the (possibly rich) transient dynamics of the system, if the system equilibrates quickly then the description of the system as an SCM may be a more convenient and compact representation of the causal structure of interest. By making assumptions on the dynamical system and the interventions being made, the SCM effectively allows us to reason about a ‘higher level’ qualitative description of the dynamics—in this case, the equilibrium states.

There are, however, two major limitations that stem from the equilibrium assumption. First, for many dynamical systems the assumption that the system settles to a unique equilibrium, either in its observational state or under intervention, may be a bad approximation of the actual system dynamics. Second, this framework is only capable of modelling interventions in which a subset of variables are clamped to fixed values (*constant* interventions). Even for rather simple physical systems such as a forced damped simple harmonic oscillator, these assumptions are violated.

Motivated by these observations, the work presented in this paper tries to answer the following questions: (i) Can the SCM framework be extended to model systems that do not converge to an equilibrium? (ii) If so, what assumptions need to be made on the ODE and interventions so that this is possible? Since SCMs are used in a variety of situations in which the equilibrium assumption does not necessarily hold, we view these questions as important in order to understand when they are indeed theoretically grounded as modelling tools. The main contribution of this paper is to show that the answer to the first question is ‘Yes’ and to provide sufficient conditions for the second. We do this by extending the SCM framework to encompass time-dependent dynamics and interventions and studying how such objects can arise from ODEs. We refer to this as a *Dynamic SCM (DSCM)* to distinguish it from the static equilibrium case for the purpose of exposition, but note that this is conceptually the same as an SCM on a fundamental level. Our construction draws inspiration from the approach of Mooij et al. (2013), that was recently generalized to also incorporate the stochastic setting (Bongers and Mooij, 2018). Here, we adapt the approach by replacing the static equilibrium states by continuous-time *trajectories*, considering two trajectories as equivalent if they do not differ asymptotically.

Note that whilst this paper applies a causal perspective to the study of dynamical systems, the goal of this paper is not to derive a learning algorithm which can be applied to time series data. In this sense, we view our main re-

sults as ‘orthogonal’ to methods such as Granger causality (Granger, 1969) and difference-in-differences (Card and Krueger, 1993) which aim to infer causal effects given time-series observations of a system. We envision that DSCMs may be used for causal analysis of dynamical systems that undergo periodic motion. Although these systems have been mostly ignored so far in the field of causal discovery, they have been studied extensively in the field of control theory. Some examples of systems that naturally exhibit oscillatory stationary states and where our framework may be applicable are EEG signals, circadian signals, seasonal influences, chemical oscillations, electric circuits, aerospace vehicles, and satellite control. We refer the reader to (Bittanti and Colaneri, 2009) for more details on these application areas from the perspective of periodic control theory.

Since the DSCM derived for a simple harmonic oscillator (see Example 4) is already quite complex, we leave the task of deriving methods that estimate the parameters from data for future work. Rather, our current work presents a first necessary theoretical step that needs to be done before applications of this theory can be developed, enabling the development of data-driven causal discovery and prediction methods for oscillatory systems, and possibly even more general systems, down the road.

The remainder of this paper is organised as follows. In Section 2, we introduce notation to describe ODEs. In Section 3, we describe how to apply the notion of an intervention on an ODE to the dynamic case. In Section 4, we define regularity conditions on the asymptotic behaviour of an ODE under a set of interventions. In Section 5, we present our main result: subject to conditions on the dynamical system and interventions being modelled, a *Dynamic SCM* can be derived that allows one to reason about how the asymptotic dynamics change under interventions on variables in the system. We conclude in Section 6.

## 2 ORDINARY DIFFERENTIAL EQUATIONS

Let  $\mathcal{I} = \{1, \dots, D\}$  be a set of variable labels. Consider time-indexed variables  $X_i(t) \in \mathcal{R}_i$  for  $i \in \mathcal{I}$ , where  $\mathcal{R}_i \subseteq \mathbb{R}$  and  $t \in \mathbb{R}_{\geq 0} = [0, \infty)$ . For  $I \subseteq \mathcal{I}$ , we write  $\mathbf{X}_I(t) \in \prod_{i \in I} \mathcal{R}_i$  for the tuple of variables  $(X_i(t))_{i \in I}$ . By an ODE  $\mathcal{D}$ , we mean a collection of  $D$  coupled ordinary differential equations with initial conditions  $\mathbf{X}_0^{(k)}$ :

$$\mathcal{D} : \begin{cases} f_i(X_i, \mathbf{X}_{\text{pa}(i)})(t) = 0, & X_i^{(k)}(0) = (\mathbf{X}_0^{(k)})_i, \\ & 0 \leq k \leq n_i - 1, \quad i \in \mathcal{I}, \end{cases}$$

where the  $i$ th differential equation determines the evolution of the variable  $X_i$  in terms of  $\mathbf{X}_{\text{pa}(i)}$ , where  $\text{pa}(i) \subseteq \mathcal{I}$  are the *parents of  $i$* , and  $X_i$  itself, and where

$n_i$  is the order of the highest derivative  $X_i^{(k)}$  of  $X_i$  that appears in equation  $i$ . Here,  $f_i$  is a functional that can include time-derivatives of its arguments. We think of the  $i$ th differential equation as modelling the *causal mechanism* that determines the dynamics of the effect  $X_i$  in terms of its direct causes  $\mathbf{X}_{\text{pa}(i)}$ .

One possible way to write down an ODE is to canonically decompose it into a collection of first order differential equations, such as is done in Mooij et al. (2013). We choose to present our ODEs as “one equation per variable” rather than splitting up the equations due to complications that would otherwise occur when considering time-dependent interventions (cf. Section 3.3).

**Example 1.** Consider a one-dimensional system of  $D$  particles of mass  $m_i$  ( $i = 1, \dots, D$ ) with positions  $X_i$  coupled by springs with natural lengths  $l_i$  and spring constants  $k_i$ , where the  $i$ th spring connects the  $i$ th and  $(i + 1)$ th masses and the outermost springs have fixed ends (see Figure 1a). Assume further that the  $i$ th mass undergoes linear damping with coefficient  $b_i$ .

Denoting by  $\dot{X}_i$  and  $\ddot{X}_i$  the first and second time derivatives of  $X_i$  respectively, the equation of motion for the  $i$ th variable is given by

$$m_i \ddot{X}_i(t) = k_i [X_{i+1}(t) - X_i(t) - l_i] - k_{i-1} [X_i(t) - X_{i-1}(t) - l_{i-1}] - b_i \dot{X}_i(t)$$

where we take  $X_0 = 0$  and  $X_D = L$  to be the fixed positions of the end springs. For the case that  $D = 2$ , we can write the system of equations as:

$$D : \begin{cases} 0 = m_1 \ddot{X}_1(t) + b_1 \dot{X}_1(t) + (k_1 + k_0)X_1(t) \\ \quad - k_1 X_2(t) - k_0 l_0 + k_1 l_1, \\ 0 = m_2 \ddot{X}_2(t) + b_2 \dot{X}_2(t) + (k_2 + k_1)X_2(t) \\ \quad - k_2 L - k_1 X_1(t) - k_2 l_1 + k_2 l_2, \\ X_i^{(k)}(0) = (\mathbf{X}_0^{(k)})_i \quad k \in \{0, 1\}, i \in \{1, 2\}. \end{cases}$$

We can represent the functional dependence structure between variables implied by the functions  $f_i$  with a graph, in which variables are nodes and arrows point  $X_j \rightarrow X_i$  if  $j \in \text{pa}(i)$ . Self loops  $X_i \rightarrow X_i$  exist if  $X_i^{(k)}$  appears in the expression of  $f_i$  for more than one value of  $k$ . This is illustrated for the system described in Example 1 in Figure 1b.

### 3 INTERVENTIONS ON ODES

We interpret ODEs as *causal* models. In particular, we consider the graph expressing the functional dependence structure to be the causal graph of the system, with an

edge between  $X_i$  and  $X_j$  iff  $X_i$  is a direct cause of  $X_j$  (in the context of all variables  $\mathbf{X}_{\mathcal{I}}$ ). In this section, we will formalize this causal interpretation by studying interventions on the system.

#### 3.1 TIME-DEPENDENT PERFECT INTERVENTIONS

Usually in the causality literature, by a *perfect intervention* it is meant that a variable is clamped to take a specific given value. The natural analogue of this in the time-dependent case is a perfect intervention that forces a variable to take a particular *trajectory*. That is, given a subset  $I \subseteq \mathcal{I}$  and a function  $\zeta_I : \mathbb{R}_{\geq 0} \rightarrow \prod_{i \in I} \mathcal{R}_i$ , we can intervene on the subset of variables  $\mathbf{X}_I$  by forcing  $\mathbf{X}_I(t) = \zeta_I(t) \forall t \in \mathbb{R}_{\geq 0}$ . Using Pearl’s do-calculus notation (Pearl, 2009) and for brevity omitting the  $t$ , we write  $\text{do}(\mathbf{X}_I = \zeta_I)$  for this intervention. Such interventions are more general objects than those of the equilibrium or time-independent case, but in the specific case that we restrict ourselves to constant trajectories the two notions coincide.

#### 3.2 SETS OF INTERVENTIONS

Recall that when modelling equilibrating dynamical systems under constant interventions, the set of interventions modelled coincides with the asymptotic behaviour of the system. We will generalise this relation to non-equilibrating behaviour.

The Dynamic SCMs that we will derive will describe the asymptotic dynamics of the ODE and how they change under different interventions. If we want to model ‘all possible interventions’, then the resulting asymptotic dynamics that can occur are arbitrarily complicated. The idea is to fix a simpler set of interventions and derive an SCM that models only these interventions, resulting in a model that is simpler than the original ODE but still allows us to reason about interventions we are interested in. In the examples in this paper, we restrict ourselves to periodic or quasi-periodic interventions, but the results hold for more general sets of interventions that satisfy the stability definitions presented later.

We need to define some notation to express the sets of interventions and the set of system responses to these interventions that we will model. Since interventions correspond to forcing variables to take some trajectory, we describe notation for defining sets of trajectories: For  $I \subseteq \mathcal{I}$ , let  $\text{Dyn}_I$  be a set of trajectories in  $\prod_{i \in I} \mathcal{R}_i$ . Let  $\text{Dyn} = \cup_{I \in \mathcal{P}(\mathcal{I})} \text{Dyn}_I$  (where  $\mathcal{P}(\mathcal{I})$  is the power set of  $\mathcal{I}$  i.e., the set of all subsets of  $\mathcal{I}$ ). Thus, an element  $\zeta_I \in \text{Dyn}_I$  is a function  $\mathbb{R}_{\geq 0} \rightarrow \prod_{i \in I} \mathcal{R}_i$ , and  $\text{Dyn}$  consists of such functions for different  $I \subseteq \mathcal{I}$ . The main idea is that we want both the interventions and the system



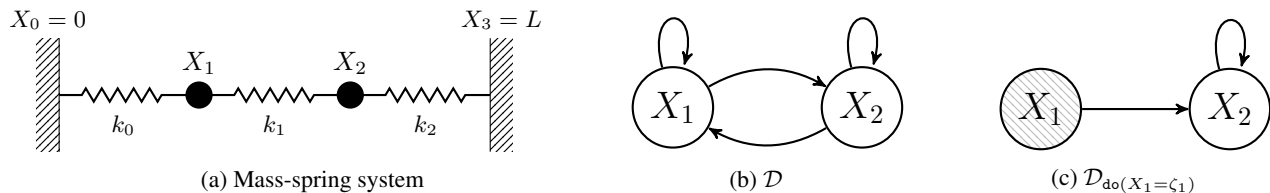


Figure 1: (a) The mass-spring system of Example 1 with  $D = 2$ ; (b–c) graphs representing the causal structure of the mass-spring system for (b) the observational system, (c) after the intervention on variable  $X_1$  described in Example 2. As a result of the intervention,  $X_1$  is not causally influenced by any variable, while the causal mechanism of  $X_2$  remains unchanged.

responses to be elements of  $\text{Dyn}$ ; in other words, the set of possible system responses should be large enough to contain all interventions that we would like to model, and in addition, all responses of the system to those interventions. The reader might wonder why we do not simply take the set of *all* possible trajectories, but that set would be so large that it would not be practical for modeling purposes.<sup>1</sup>

Since our goal will be to derive a causal model that describes the relations between components (variables) of the system, we will need the following definition in Section 5.

**Definition 1.** A set of trajectories  $\text{Dyn}$  is **modular** if, for any  $\{i_1, \dots, i_n\} = I \subseteq \mathcal{I}$ ,

$$\zeta_I \in \text{Dyn} \iff \zeta_{i_k} \in \text{Dyn} \quad \forall k \in \{1, \dots, n\}.$$

This should be interpreted as saying that admitted trajectories of single variables can be combined arbitrarily into admitted trajectories of the whole system (and *vice versa*, admitted system trajectories can be decomposed into trajectories of individual variables), and in addition, that interventions on each variable can be made independently and combined in any way.<sup>2</sup> This is not to say that all such interventions must be physically possible to implement in practice. Rather, this means that the mathematical model we derive should allow one to *reason* about all such interventions. Not all sets of trajectories  $\text{Dyn}$  are modular; in the following sections we will assume that the

<sup>1</sup>For example, one might want to parameterize the set of trajectories in order to learn the model from data. Without any restriction on the smoothness of the trajectories, the problem of estimating a trajectory from data becomes ill-posed. Secondly, since we would like to identify trajectories that are asymptotically identical in order to focus the modeling efforts on the *asymptotic* behaviour of the system, we will only put a single trajectory into  $\text{Dyn}$  to represent all trajectories that are asymptotically identical to that trajectory, but whose transient dynamics may differ.

<sup>2</sup>This is related to notions that have been discussed in the literature under various headings, for instance autonomy and invariance (Pearl, 2009).

sets of trajectories we are considering *are* for the purposes of constructing the Dynamic SCMs. Some examples of trivially modular sets of trajectories are: (i) all static (i.e., time-independent) trajectories, corresponding to (Mooij et al., 2013); (ii) all continuously-differentiable trajectories that differ asymptotically; (iii) all periodic motions. The latter is the running example in this paper.

### 3.3 DESCRIBING INTERVENTIONS ON ODES

We can realise a perfect intervention by replacing the equations of the intervened variables with new equations that fix them to take the specified trajectories:<sup>3</sup>

$$\mathcal{D}_{\text{do}(\mathbf{X}_I = \zeta_I)} : \begin{cases} f_i(X_i, \mathbf{X}_{\text{pa}(i)})(t) = 0, & X_i^{(k)}(0) = (\mathbf{X}_0^{(k)})_i, \\ 0 \leq k \leq n_i - 1, & i \in \mathcal{I} \setminus I, \\ X_i(t) - \zeta_i(t) = 0, & i \in I. \end{cases}$$

This procedure is analogous to the notion of intervention in an SCM. In reality, this corresponds to decoupling the intervened variables from their usual causal mechanism by forcing them to take a particular value, while leaving the non-intervened variables' causal mechanisms unaffected.

Perfect interventions will not generally be realisable in the real world. In practice, an intervention on a variable would correspond to altering the differential equation governing its evolution by adding extra forcing terms; perfect interventions could be realised by adding forcing terms that push the variable towards its target value at each instant in time, and considering the limit as these forcing terms become infinitely strong so as to dominate the usual causal mechanism determining the evolution of the variable.

**Example 2 (continued).** Consider the mass-spring system described in Example 1. If we were to intervene on

<sup>3</sup>Note that in the intervened ODE, the initial conditions of the intervened variables do not need to be specified explicitly as for the other variables, since they are implied by considering  $t = 0$ .

the system to force the mass  $X_1$  to undergo simple harmonic motion, we could express this as a change to the system of differential equations as:

$$\mathcal{D}_{\text{do}(X_1(t)=l_1+A \cos(\omega t))} : \begin{cases} 0 = X_1(t) - l_1 - A \cos(\omega t), \\ 0 = m_2 \ddot{X}_2(t) + b_2 \dot{X}_2(t) + (k_2 + k_1)X_2(t) \\ \quad - k_2 L - k_1 X_1(t) - k_2 l_1 + k_2 l_2, \\ X_2^{(k)}(0) = (\mathbf{X}_0^{(k)})_2 \quad k \in \{0, 1\}. \end{cases}$$

This induces a change to the graphical description of the causal relationships between the variables. We break any incoming arrows to any intervened variable, including self loops, as the intervened variables are no longer causally influenced by any other variable in the system. See Figure 1c for the graph corresponding to the intervened ODE in Example 2.

## 4 DYNAMIC STABILITY

A crucial assumption of Mooij et al. (2013) was that the systems considered were *stable* in the sense that they would converge to unique stable equilibria (if necessary, also after performing a constant intervention). This made them amenable to study by considering the  $t \rightarrow \infty$  limit in which any complex but transient dynamical behaviour would have decayed. The SCMs derived would allow one to reason about the asymptotic equilibrium states of the systems after interventions. Since we want to consider non-constant asymptotic dynamics, this is not a notion of stability that is fit for our purposes.

Instead, we define our stability with reference to a set of trajectories. We will use  $\text{Dyn}_{\mathcal{I}}$  for this purpose. Recall that elements of  $\text{Dyn}_{\mathcal{I}}$  are trajectories for all variables in the system. To be totally explicit, we can think of an element  $\boldsymbol{\eta} \in \text{Dyn}_{\mathcal{I}}$  as a function

$$\boldsymbol{\eta} : \mathbb{R}_{\geq 0} \rightarrow \mathcal{R}_{\mathcal{I}} \\ t \mapsto (\eta_1(t), \eta_2(t), \dots, \eta_D(t))$$

where  $\eta_i(t) \in \mathcal{R}_i$  is the state of the  $i$ th variable  $X_i$  at time  $t$ . Note that  $\text{Dyn}_{\mathcal{I}}$  is not a single fixed set, independent of the situation we are considering. We can choose  $\text{Dyn}_{\mathcal{I}}$  depending on the ODE  $\mathcal{D}$  under consideration, and the interventions that we may wish to make on it.

Informally, stability in this paper means that the asymptotic dynamics of the dynamical system converge to a unique element of  $\text{Dyn}_{\mathcal{I}}$ , independent of initial condition. If  $\text{Dyn}_{\mathcal{I}}$  is in some sense simple, we can simply characterise the asymptotic dynamics of the system under study. The following definitions of stability extend those

of Mooij et al. (2013) to allow for non-constant trajectories in  $\text{Dyn}_{\mathcal{I}}$ , and coincide with them in the case that  $\text{Dyn}_{\mathcal{I}}$  consists of all constant trajectories in  $\mathcal{R}_{\mathcal{I}}$ .

**Definition 2.** The ODE  $\mathcal{D}$  is **dynamically stable with reference to**  $\text{Dyn}_{\mathcal{I}}$  if there exists a unique  $\boldsymbol{\eta}_0 \in \text{Dyn}_{\mathcal{I}}$  such that  $\mathbf{X}_{\mathcal{I}}(t) = \boldsymbol{\eta}_0(t) \forall t$  is a solution to  $\mathcal{D}$  and that for any initial condition, the solution  $\mathbf{X}_{\mathcal{I}}(t) \rightarrow \boldsymbol{\eta}_0(t)$  as  $t \rightarrow \infty$ .<sup>4</sup>

We use a subscript  $\emptyset$  to emphasise that  $\boldsymbol{\eta}_0$  describes the asymptotic dynamics of  $\mathcal{D}$  without any intervention. Observe that  $\text{Dyn}_{\mathcal{I}}$  could consist of the single element  $\boldsymbol{\eta}_0$  in this case. The requirement that this hold for all initial conditions can be relaxed to hold for all initial conditions except on a set of measure zero, but that would mean that the proofs later on require some more technical details. For the purpose of exposition, we stick to this simpler case.

**Example 3.** Consider a single mass on a spring that is undergoing simple periodic forcing and is underdamped. Such a system could be expressed as a single (parent-less) variable with ODE description:

$$\mathcal{D} : \begin{cases} m\ddot{X}_1(t) + b\dot{X}_1(t) + k(X_1(t) - l) \\ \quad = F \cos(\omega t + \phi), \\ X_1^{(k)}(0) = (X_0^{(k)}) \quad k \in \{0, 1\}. \end{cases}$$

The solution to this differential equation is

$$X_1(t) = r(t) + l + A \cos(\omega t + \phi') \quad (1)$$

where  $r(t)$  decays exponentially quickly (and is dependent on the initial conditions) and  $A$  and  $\phi'$  depend on the parameters of the equation of motion (but not on the initial conditions).

Therefore such a system would be dynamically stable with reference to (for example)

$$\text{Dyn}_{\mathcal{I}} = \{l + A \cos(\omega t + \phi') : A \in \mathbb{R}, \phi' \in [0, 2\pi)\}.$$

**Remark 1.** We use a subscript  $\zeta_I$  to emphasise that  $\boldsymbol{\eta}_{\zeta_I}$  describes the asymptotic dynamics of  $\mathcal{D}$  after performing the intervention  $\text{do}(\mathbf{X}_I = \zeta_I)$ . Observe that  $\text{Dyn}_{\mathcal{I}}$  could consist only of the single element  $\boldsymbol{\eta}_{\zeta_I}$  and the above definition would be satisfied. But then the original ODE wouldn't be dynamically stable with reference to  $\text{Dyn}_{\mathcal{I}}$ , nor would other intervened versions of  $\mathcal{D}$ . This motivates the following definition, extending dynamic stability to sets of intervened systems.

<sup>4</sup>The convergence we refer to here is the usual asymptotic convergence of real-valued functions, i.e., for  $f : [0, \infty) \rightarrow \mathbb{R}^d$ ,  $g : [0, \infty) \rightarrow \mathbb{R}^d$  we have that  $f \rightarrow g$  iff for every  $\epsilon > 0$  there is a  $T \in [0, \infty)$  such that  $|f(t) - g(t)| < \epsilon$  for all  $t \in [T, \infty)$ .

**Definition 3.** Let  $\text{Traj}$  be a set of trajectories. We say that the pair  $(\mathcal{D}, \text{Traj})$  is **dynamically stable with reference to  $\text{Dyn}_{\mathcal{I}}$**  if, for any  $\zeta_{\mathcal{I}} \in \text{Traj}$ ,  $\mathcal{D}_{\text{do}(\mathbf{X}_{\mathcal{I}}=\zeta_{\mathcal{I}})}$  is dynamically stable with reference to  $\text{Dyn}_{\mathcal{I}}$ .

**Example 3** (continued). Suppose we are interested in modelling the effect of changing the forcing term, either in amplitude, phase or frequency. We introduce a second variable  $X_2$  to model the forcing term:

$$\mathcal{D} : \begin{cases} 0 = f_1(X_1, X_2)(t) \\ \quad = m\ddot{X}_1(t) + b\dot{X}_1(t) + k(X_1(t) - l) - X_2(t), \\ 0 = f_2(X_2)(t) \\ \quad = X_2(t) - F_0 \cos(\omega_0 t + \phi_0), \\ X_1^{(k)}(0) = (\mathbf{X}_0^{(k)})_1, \quad k \in \{0, 1\}. \end{cases}$$

If we want to change the forcing term that we apply to the mass, we can interpret this as performing an intervention on  $X_2$ . We could represent this using the notation we have developed as

$$\text{Dyn}_{\{2\}} = \{\zeta_2(t) = F_2 \cos(\omega t + \phi_2) : F_2, \omega \in \mathbb{R}, \phi_2 \in [0, 2\pi]\}.$$

For any intervention  $\zeta_2 \in \text{Dyn}_{\{2\}}$ , the dynamics of  $X_1$  in  $\mathcal{D}_{\text{do}(X_2=\zeta_2)}$  will be of the form (1). Therefore  $(\mathcal{D}, \text{Dyn}_{\{2\}})$  will be dynamically stable with reference to

$$\text{Dyn}_{\mathcal{I}} = \left\{ \zeta(t) = (l + F_1 \cos(\omega t + \phi_1), F_2 \cos(\omega t + \phi_2)) : F_1, F_2, \omega \in \mathbb{R}, \phi_1, \phi_2 \in [0, 2\pi] \right\}.$$

The independence of initial conditions for Example 3 is illustrated in Figure 2.

Note that if  $(\mathcal{D}, \text{Traj})$  is dynamically stable with reference to  $\text{Dyn}_{\mathcal{I}}$ , and  $\text{Dyn}'_{\mathcal{I}} \supseteq \text{Dyn}_{\mathcal{I}}$  is a larger set of trajectories that still satisfies the uniqueness condition in the definition of dynamic stability,<sup>5</sup> then  $(\mathcal{D}, \text{Traj})$  is dynamically stable with reference to  $\text{Dyn}'_{\mathcal{I}}$ .

## 5 DYNAMIC STRUCTURAL CAUSAL MODELS

A deterministic SCM  $\mathcal{M}$  is a collection of structural equations, the  $i$ th of which defines the value of variable  $X_i$

<sup>5</sup>Namely:  $\forall \zeta_{\mathcal{I}} \in \text{Traj}, \exists! \eta_{\zeta_{\mathcal{I}}} \in \text{Dyn}'_{\mathcal{I}}$  such that under  $\mathcal{D}_{\text{do}(\mathbf{X}_{\mathcal{I}}=\zeta_{\mathcal{I}})}$  and for any initial condition,  $X_{\mathcal{I}}(t) \rightarrow \eta_{\zeta_{\mathcal{I}}}(t)$  as  $t \rightarrow \infty$ . Assuming that  $(\mathcal{D}, \text{Traj})$  is dynamically stable with reference to  $\text{Dyn}_{\mathcal{I}}$ , a sufficient condition for this is that none of the elements in  $\text{Dyn}'_{\mathcal{I}} \setminus \text{Dyn}_{\mathcal{I}}$  are asymptotically equal to any of the elements of  $\text{Dyn}_{\mathcal{I}}$ . That is:  $\forall \zeta \in \text{Dyn}_{\mathcal{I}}, \forall \zeta' \in \text{Dyn}'_{\mathcal{I}} \setminus \text{Dyn}_{\mathcal{I}}, \zeta(t) \not\rightarrow \zeta'(t)$  as  $t \rightarrow \infty$ .

in terms of its parents. We extend this to the case that our variables do not take fixed values but rather represent entire trajectories.

**Definition 4.** Let  $\text{Dyn} = \bigcup_{\mathcal{I} \subseteq \mathcal{I}} \text{Dyn}_{\mathcal{I}}$  be a modular set of trajectories, where  $\text{Dyn}_{\mathcal{I}} \subseteq \mathcal{R}_{\mathcal{I}}^{\mathbb{R}_{\geq 0}}$ . A deterministic Dynamic Structural Causal Model (DSCM) on the time-indexed variables  $\mathbf{X}_{\mathcal{I}}$  taking values in  $\text{Dyn}$  is a collection of structural equations

$$\mathcal{M} : \{ X_i = F_i(\mathbf{X}_{\text{pa}(i)}) \quad i \in \mathcal{I},$$

where  $\text{pa}(i) \subseteq \mathcal{I} \setminus \{i\}$  and each  $F_i$  is a map  $\text{Dyn}_{\text{pa}(i)} \rightarrow \text{Dyn}_i$  that gives the trajectory of an effect variable in terms of the trajectories of its direct causes.

The point of this paper is to show that, subject to restrictions on  $\mathcal{D}$  and  $\text{Dyn}$ , we can derive a DSCM that allows us to reason about the effect on the asymptotic dynamics of interventions using trajectories in  $\text{Dyn}$ . ‘Traditional’ deterministic SCMs arise as a special case, where all trajectories are constant over time.

In an ODE, the equations  $f_i$  determine the causal relationship between the variable  $X_i(t)$  and its parents  $\mathbf{X}_{\text{pa}(i)}(t)$  at each instant in time. In contrast, we think of the function  $F_i$  of the DSCM as a causal mechanism that determines the entire trajectory of  $X_i$  in terms of the trajectories of the variables  $\mathbf{X}_{\text{pa}(i)}$ , integrating over the instantaneous causal effects over all time. In the case that  $\text{Dyn}$  consists of constant trajectories (and thus the instantaneous causal effects are constant over time), a DSCM reduces to a traditional deterministic SCM.

The rest of this section is laid out as follows. In Section 5.1 we define what it means to make an intervention in a DSCM. In Section 5.2 we show how, subject to certain conditions, a DSCM can be derived from a pair  $(\mathcal{D}, \text{Dyn})$ . The procedure for doing this relies on intervening on all but one variable at a time. In Section 5.3, Theorem 2 states that the DSCM thus derived is capable of modelling the effect of intervening on arbitrary subsets of variables, even though it was constructed by considering the case that we consider interventions on exactly  $D - 1$  variables. Theorem 3 and Corollary 1 in Section 5.4 prove that the notions of intervention in ODE and the derived DSCM coincide. Collectively, these theorems tell us that we can derive a DSCM that allows us to reason about the effects of interventions on the asymptotic dynamics of the ODE. Proofs of these theorems are provided in Section A of the Supplementary Material.

### 5.1 INTERVENTIONS IN A DSCM

Interventions in (D)SCMs are realized by replacing the structural equations of the intervened variables. Given

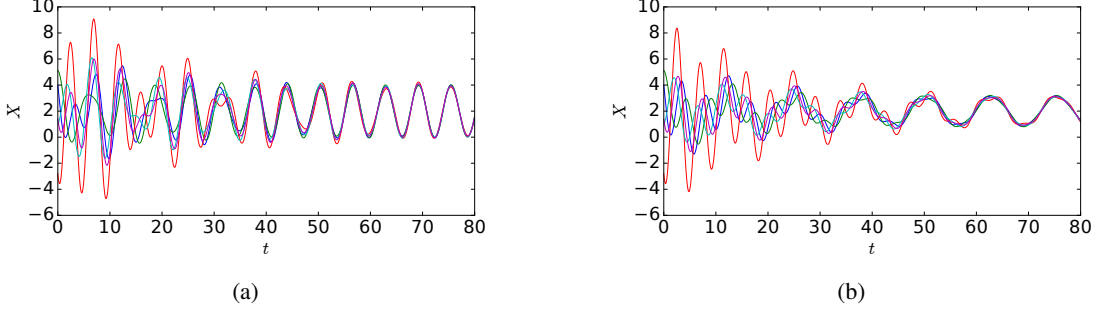


Figure 2: Simulations from the forced simple harmonic oscillator in Example 3 showing the evolution of  $X_1$  with different initial conditions for different forcing terms (interventions on  $X_2$ ). The parameters used were  $m = 1, k = 1, l = 2, F = 2, b = 0.1$ , with (a)  $\omega = 3$  and (b)  $\omega = 2$ . Dynamic stability means that asymptotic dynamics are independent of initial conditions, and the purpose of the DSCM is to quantify how the asymptotic dynamics change under intervention.

$\zeta_I \in \text{Dyn}_I$  for some  $I \subseteq \mathcal{I}$ , the intervened DSCM  $\mathcal{M}_{\text{do}(\mathbf{X}_I = \zeta_I)}$  can be written:

$$\mathcal{M}_{\text{do}(\mathbf{X}_I = \zeta_I)} : \begin{cases} X_i = F_i(\mathbf{X}_{\text{pa}(i)}) & i \in \mathcal{I} \setminus I, \\ X_i = \zeta_i & i \in I. \end{cases}$$

The causal mechanisms determining the non-intervened variables are unaffected, so their structural equations remain the same. The intervened variables are decoupled from their usual causal mechanisms and are forced to take the specified trajectory.

## 5.2 DERIVING DSCMs FROM ODEs

In order to derive a DSCM from an ODE, we require the following consistency property between the asymptotic dynamics of the ODE and the set of interventions.

**Definition 5** (Structural dynamic stability). *Let  $\text{Dyn}$  be modular. The pair  $(\mathcal{D}, \text{Dyn})$  is **structurally dynamically stable** if  $(\mathcal{D}, \text{Dyn}_{\mathcal{I} \setminus \{i\}})$  is dynamically stable with reference to  $\text{Dyn}_{\mathcal{I}}$  for all  $i$ .*

This means that for any intervention trajectory  $\zeta_{\mathcal{I} \setminus \{i\}} \in \text{Dyn}_{\mathcal{I} \setminus \{i\}}$ , the asymptotic dynamics of the intervened ODE  $\mathcal{D}_{\text{do}(\mathbf{X}_{\mathcal{I} \setminus \{i\}} = \zeta_{\mathcal{I} \setminus \{i\}})}$  are expressible uniquely as an element of  $\text{Dyn}_{\mathcal{I}}$ . Since  $\text{Dyn}$  is modular, the asymptotic dynamics of the non-intervened variable can be realised as the trajectory  $\zeta_i \in \text{Dyn}_i$ , and thus  $\text{Dyn}$  is rich enough to allow us to make an intervention which forces the non-intervened variable to take this trajectory. This is a crucial property that allows the construction of the structural equations. In the particular case that  $\text{Dyn}$  consists of all constant trajectories, structural dynamic stability means that after any intervention on all-but-one-variable, the non-intervened variable settles to a unique equilibrium. In the language of Mooij et al. (2013), this would imply that the ODE is *structurally stable*.

It should be noted that  $(\mathcal{D}, \text{Dyn})$  being structurally dynamically stable is a strong assumption in general. If  $\text{Dyn}$  is too small,<sup>6</sup> then it may be possible to find a larger set  $\text{Dyn}' \supset \text{Dyn}$  such that  $(\mathcal{D}, \text{Dyn}')$  is structurally dynamically stable. The procedure described in this section describes how to derive a DSCM capable of modelling all interventions in  $\text{Dyn}'$ , which can thus be used to model interventions in  $\text{Dyn}$ .

Henceforth, we use the notation  $I_i = \mathcal{I} \setminus \{i\}$  for brevity. Suppose that  $(\mathcal{D}, \text{Dyn})$  is structurally dynamically stable. We can **derive structural equations**  $F_i : \text{Dyn}_{\text{pa}(i)} \rightarrow \text{Dyn}_i$  to describe the asymptotic dynamics of children variables as functions of their parents as follows. Pick  $i \in \mathcal{I}$ . The variable  $X_i$  has parents  $\mathbf{X}_{\text{pa}(i)}$ . Since  $\text{Dyn}$  is modular, for any configuration of parent dynamics  $\eta_{\text{pa}(i)} \in \text{Dyn}_{\text{pa}(i)}$  there exists  $\zeta_{I_i} \in \text{Dyn}_{I_i}$  such that  $(\zeta_{I_i})_{\text{pa}(i)} = \eta_{\text{pa}(i)}$ .

By structural dynamic stability, the system  $\mathcal{D}_{\text{do}(\mathbf{X}_{I_i} = \zeta_{I_i})}$  has asymptotic dynamics specified by a unique element  $\eta \in \text{Dyn}_{\mathcal{I}}$ , which in turn defines a unique element  $\eta_i \in \text{Dyn}_i$  specifying the asymptotic dynamics of variable  $X_i$  since  $\text{Dyn}$  is modular.

**Theorem 1.** *Suppose that  $(\mathcal{D}, \text{Dyn})$  is structurally dynamically stable. Then the functions*

$$F_i : \text{Dyn}_{\text{pa}(i)} \rightarrow \text{Dyn}_i : \eta_{\text{pa}(i)} \mapsto \eta_i$$

*constructed as above are well-defined.*

Given the structurally dynamically stable pair  $(\mathcal{D}, \text{Dyn})$  we define the derived DSCM

$$\mathcal{M}_{\mathcal{D}} : \{ X_i = F_i(\mathbf{X}_{\text{pa}(i)}) \quad i \in \mathcal{I},$$

<sup>6</sup>For example, if  $\text{Dyn}$  is not modular or represents interventions on only a subset of the variables.

where the  $F_i : \text{Dyn}_{\text{pa}(i)} \rightarrow \text{Dyn}_i$  are defined as above. Note that structural dynamic stability was a crucial property that ensured  $F_i(\text{Dyn}_{\text{pa}(i)}) \subseteq \text{Dyn}_i$ . If  $(\mathcal{D}, \text{Dyn})$  is not structurally dynamically stable, we cannot build structural equations in this way.

We provide next an example of a DSCM for the mass-spring system of Example 1 with  $D = 2$ . The derivation of this for the general case of arbitrarily many masses is included in the Supplementary Material.

**Example 4.** Consider the system  $\mathcal{D}$  governed by the differential equation of Example 1 with  $D = 2$ . Let  $\text{Dyn}_{\{1,2\}}$  be the modular set of trajectories with

$$\text{Dyn}_{\{i\}} = \left\{ \sum_{j=1}^{\infty} A_i^j \cos(\omega_i^j t + \phi_i^j) : \right. \\ \left. w_i^j, \phi_i^j, A_i^j \in \mathbb{R}, \sum_{j=1}^{\infty} |A_i^j| < \infty \right\}$$

for  $i = 1, 2$ , where for each  $i$  it holds that  $\sum_{j=1}^{\infty} |A_i^j| < \infty$  (so that the series is absolutely convergent). Then  $(\mathcal{D}, \text{Dyn}_{\{1,2\}})$  is structurally dynamically stable and admits the following DSCM.

$$\mathcal{M} : \begin{cases} X_1 &= F_1(X_2) \\ X_2 &= F_2(X_1) \end{cases}$$

where, writing  $C_1^j = [k_1 + k_2 - m_1(\omega_2^j)^2]^2$  and  $C_2^j = [k_1 + k_2 - m_2(\omega_1^j)^2]^2$ , the functionals  $F_1$  and  $F_2$  are given by Equations 2 and 3 overleaf.

### 5.3 SOLUTIONS OF A DSCM

Theorem 1 states that we can construct a DSCM by the described procedure. We constructed each equation by intervening on  $D - 1$  variables at a time. The result of this section states that the DSCM can be used to correctly model interventions on arbitrary subsets of variables. We say that  $\eta_{\mathcal{I}} \in \text{Dyn}_{\mathcal{I}}$  is a solution of  $\mathcal{M}$  if  $\eta_i = F_i(\eta_{\text{pa}(i)}) \forall i \in \mathcal{I}$ .

**Theorem 2.** Suppose that  $(\mathcal{D}, \text{Dyn})$  is structurally dynamically stable. Let  $I \subseteq \mathcal{I}$ , and let  $\zeta_I \in \text{Dyn}_I$ . Then  $\mathcal{D}_{\text{do}(\mathbf{X}_I = \zeta_I)}$  is dynamically stable if and only if the intervened SCM  $\mathcal{M}_{(\mathcal{D}_{\text{do}(\mathbf{X}_I = \zeta_I)})}$  has a unique solution. If there is a unique solution, it coincides with the element of  $\text{Dyn}_{\mathcal{I}}$  describing the asymptotic dynamics of  $\mathcal{D}_{\text{do}(\mathbf{X}_I = \zeta_I)}$ .

*Remark 2.* We could also take  $I = \emptyset$ , in which case the above theorem applies to just  $\mathcal{D}$ .

### 5.4 CAUSAL REASONING IS PRESERVED

We have defined ways to model interventions in both ODEs and DSCMs. The following theorem and its immediate corollary proves that these notions of intervention

coincide, and hence that DSCMs provide a representation to reason about the asymptotic behaviour of the ODE under interventions in  $\text{Dyn}$ . A consequence of these results is that the diagram in Figure 3 commutes.

**Theorem 3.** Suppose that  $(\mathcal{D}, \text{Dyn})$  is structurally dynamically stable. Let  $I \subseteq \mathcal{I}$  and let  $\zeta_I \in \text{Dyn}_I$ . Then  $\mathcal{M}_{(\mathcal{D}_{\text{do}(\mathbf{X}_I = \zeta_I)})} = (\mathcal{M}_{\mathcal{D}})_{\text{do}(\mathbf{X}_I = \zeta_I)}$ .

**Corollary 1.** Suppose additionally that  $J \subseteq \mathcal{I} \setminus I$  and let  $\zeta_J \in \text{Dyn}_J$ . Then

$$\left( \mathcal{M}_{(\mathcal{D}_{\text{do}(\mathbf{X}_I = \zeta_I)})} \right)_{\text{do}(\mathbf{X}_J = \zeta_J)} = (\mathcal{M}_{\mathcal{D}})_{\text{do}(\mathbf{X}_I = \zeta_I, \mathbf{X}_J = \zeta_J)}.$$

To summarise, Theorems 1–3 and Corollary 1 collectively state that if  $(\mathcal{D}, \text{Dyn})$  is dynamically structurally stable then it is possible to derive a DSCM that allows us to reason about the asymptotic dynamics of the ODE under any possible intervention in  $\text{Dyn}$ .

## 5.5 RELATION TO ODEs AND DYNAMIC BAYESIAN NETWORKS

An ODE is capable of modelling arbitrary interventions on the system it describes. At the cost of only modelling a restricted set of interventions, a DSCM can be derived which describes the asymptotic behaviour of the system under these interventions. This may be desirable in cases for which transient behaviour is not important.

We now compare DSCMs to Dynamic Bayesian Networks (DBNs), an existing popular method for causal modelling of dynamical systems (Koller and Friedman, 2009). DBNs are essentially Markov chains, and thus are appropriate for discrete-time systems. When the discrete-time Markov assumption holds, DBNs are a powerful tool capable of modelling arbitrary interventions. However, approximations must be made whenever these assumptions do not hold. In particular, a continuous system must be approximately discretised in order to be modelled by a DBN (Sokol and Hansen, 2014).

By using the Euler method for numerically solving ODEs, we can make such an approximation to derive a DBN describing the system in Example 1, leading to the discrete time equation given in (8) the Supplementary Material. For DBNs, the main choice to be made is how fine the temporal discretisation should be. The smaller the value of  $\Delta$ , the better the discrete approximation will be. Even if there is a natural time-scale on which measurements can be made, choosing a finer discretisation than this will provide a better approximation to the behaviour of the true system. The choice of  $\Delta$  should reflect the natural timescales of the interventions to be considered too; for example, it is not clear how one would model the intervention  $\text{do}(X_1(t) = \cos(\frac{2\pi t}{\Delta}))$  with a discretisation length  $\Delta$ . Another notable disadvantage of DBNs is that the

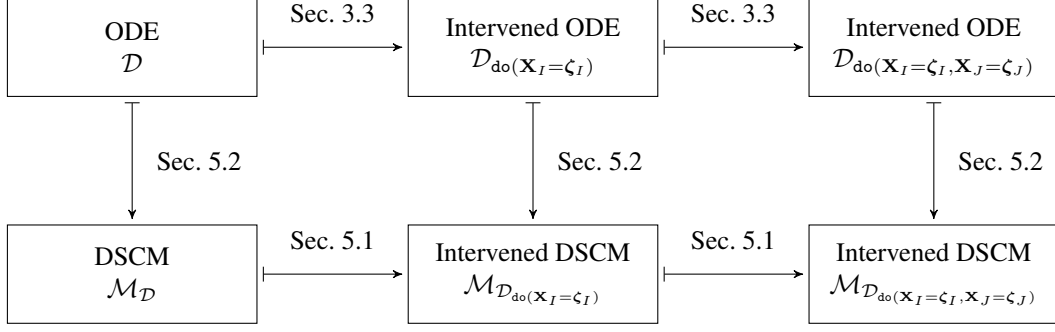


Figure 3: Top-to-bottom arrows: Theorems 1 and 2 together state that if  $(\mathcal{D}, \text{Dyn})$  is structurally dynamically stable then we can construct a DSCM to describe the asymptotic behaviour of  $\mathcal{D}$  under different interventions in the set  $\text{Dyn}$ . Left-to-right arrows: Both ODEs and DSCMs are equipped with notions of intervention. Theorem 3 and Corollary 1 say that these two notions of intervention coincide, and thus the diagram commutes.

$$F_1 \left( \sum_{j=1}^{\infty} A_2^j \cos(\omega_2^j t + \phi_2^j) \right) = \frac{-k_1 l_1}{k_1 + k_0} + \sum_{j=1}^{\infty} \frac{k_1 A_2^j}{\sqrt{C_1^j + b_1 m_1 (\omega_2^j)^2}} \cos \left( \omega_2^j t + \phi_2^j - \arctan \left[ \frac{b_1 \omega_2^j}{C_1^j} \right] \right) \quad (2)$$

$$F_2 \left( \sum_{j=1}^{\infty} A_1^j \cos(\omega_1^j t + \phi_1^j) \right) = \frac{k_1 l_1 - k_2 l_2}{k_1 + k_2} + \frac{k_2 L}{k_2 + k_3} + \sum_{j=1}^{\infty} \frac{k_1 A_1^j}{\sqrt{C_2^j + b_2 m_2 (\omega_1^j)^2}} \cos \left( \omega_1^j t + \phi_1^j - \arctan \left[ \frac{b_2 \omega_1^j}{C_2^j} \right] \right) \quad (3)$$

Figure 4: Equations giving the structural equations for the DSCM describing the mass-spring system of Example 4

computational cost of learning and inference increases for smaller  $\Delta$ , where computational cost becomes infinitely large in the limit  $\Delta \rightarrow 0$ .

In contrast, the starting point for DSCMs is to fix a convenient set of interventions we are interested in modelling. If a DSCM containing these interventions exists, it will model the asymptotic behaviour of the system under each of these interventions *exactly*, rather than approximately modelling the transient and asymptotic behaviour as in the case of a DBN. Computational cost does not relate inversely to accuracy as for DBNs, but depends on the chosen representation of the set of admitted interventions.

## 6 DISCUSSION AND FUTURE WORK

The main contribution of this paper is to show that the SCM framework can be applied to reason about time-dependent interventions on an ODE in a dynamic setting. In particular, we showed that if an ODE is sufficiently well-behaved under a set of interventions, a DSCM can be derived that captures how the asymptotic dynamics change under these interventions. This is in contrast to previous approaches to connecting the language of ODEs with the SCM framework, which used SCMs to describe the stable (constant-in-time) equilibria of the ODE and

how they change under intervention.

We identify three possible directions in which to extend this work in the future. The first is to properly understand how learning DSCMs from data could be performed. This is important if DSCMs are to be used in practical applications. Challenges to be addressed include finding practical parameterizations of DSCMs, the presence of measurement noise in the data and the fact that time-series data are usually sampled at a finite number of points in time. The second is to relax the assumption that the asymptotic dynamics are *independent of initial conditions*, as was done recently for the static equilibrium scenario by Blom and Mooij (2018). The third extension is to move away from deterministic systems and consider Random Differential Equations (Bongers and Mooij, 2018), thereby allowing to take into account model uncertainty, but also to include systems that may be inherently stochastic.

## ACKNOWLEDGEMENTS

Stephan Bongers was supported by NWO, the Netherlands Organization for Scientific Research (VIDI grant 639.072.410). This project has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement n° 639466).

## References

- J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, New York, NY, 2nd edition, 2009.
- Kenneth A Bollen. *Structural equations with latent variables*. John Wiley & Sons, 2014.
- Peter Spirtes. Directed cyclic graphical representations of feedback models. In *Proceedings of the Eleventh conference on Uncertainty in Artificial Intelligence (UAI 1995)*, pages 491–498, 1995.
- Joris M Mooij, Dominik Janzing, Tom Heskes, and Bernhard Schölkopf. On causal discovery with cyclic additive noise models. In *Advances in Neural Information Processing Systems (NIPS 2011)*, pages 639–647, 2011.
- Antti Hyttinen, Frederick Eberhardt, and Patrik O Hoyer. Learning linear cyclic causal models with latent variables. *The Journal of Machine Learning Research*, 13(1):3387–3439, 2012.
- Mark Voortman, Denver Dash, and Marek J Druzdzel. Learning why things change: the difference-based causality learner. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (UAI 2010)*, 2010.
- Gustavo Lacerda, Peter L Spirtes, Joseph Ramsey, and Patrik O Hoyer. Discovering cyclic causal models by independent components analysis. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence (UAI 2008)*, 2008.
- Stephan Bongers, Jonas Peters, Bernhard Schölkopf, and Joris M. Mooij. Theoretical aspects of cyclic structural causal models. *arXiv.org preprint*, arXiv:1611.06221v2 [stat.ME], 2018.
- Yumi Iwasaki and Herbert A Simon. Causality and model abstraction. *Artificial Intelligence*, 67(1):143–194, 1994.
- Denver Dash. Restructuring dynamic causal systems in equilibrium. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics (AISTATS 2005)*, 2005.
- J. Mooij, D. Janzing, and B. Schölkopf. From ordinary differential equations to structural causal models: the deterministic case. In *Proceedings of the Twenty-Ninth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI 2013)*, pages 440–448, 2013.
- Alexander Sokol and Niels Richard Hansen. Causal interpretation of stochastic differential equations. *Electronic Journal of Probability*, 19(100):1–24, 2014.
- Stephan Bongers and Joris M. Mooij. From random differential equations to structural causal models: the stochastic case. *arXiv.org preprint*, arXiv:1803.08784 [cs.AI], March 2018. URL <https://arxiv.org/abs/1803.08784>.
- Tineke Blom and Joris M. Mooij. Generalized structural causal models. *arXiv.org preprint*, <https://arxiv.org/abs/1805.06539> [cs.AI], May 2018. URL <https://arxiv.org/abs/1805.06539>.
- Clive WJ Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: Journal of the Econometric Society*, pages 424–438, 1969.
- David Card and Alan B Krueger. Minimum wages and employment: A case study of the fast food industry in New Jersey and Pennsylvania. Technical report, National Bureau of Economic Research, 1993.
- Sergio Bittanti and Patrizio Colaneri. *Periodic systems: filtering and control*, volume 5108985. Springer Science & Business Media, 2009.
- Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.

---

# Frank-Wolfe Optimization for Symmetric-NMF under Simplicial Constraint

---

**Han Zhao**

Machine Learning Department  
Carnegie Mellon University  
Pittsburgh, PA 15213  
han.zhao@cs.cmu.edu

**Geoff Gordon**

Machine Learning Department  
Carnegie Mellon University  
Pittsburgh, PA 15213  
ggordon@cs.cmu.edu

## Abstract

Symmetric nonnegative matrix factorization has found abundant applications in various domains by providing a symmetric low-rank decomposition of nonnegative matrices. In this paper we propose a Frank-Wolfe (FW) solver to optimize the symmetric nonnegative matrix factorization problem under a simplicial constraint, which has recently been proposed for probabilistic clustering. Compared with existing solutions, this algorithm is simple to implement, and has no hyperparameters to be tuned. Building on the recent advances of FW algorithms in nonconvex optimization, we prove an  $O(1/\varepsilon^2)$  convergence rate to  $\varepsilon$ -approximate KKT points, via a tight bound  $\Theta(n^2)$  on the curvature constant, which matches the best known result in unconstrained nonconvex setting using gradient methods. Numerical results demonstrate the effectiveness of our algorithm. As a side contribution, we construct a simple nonsmooth convex problem where the FW algorithm fails to converge to the optimum. This result raises an interesting question about necessary conditions of the success of the FW algorithm on convex problems.

## 1 INTRODUCTION

Nonnegative matrix factorization (NMF) has found various applications in data mining, natural language processing, and computer vision [Xu et al., 2003, Liu et al., 2003, Kuang et al., 2012], due to its ability to provide low rank approximations and interpretable decompositions. The decision version of NMF is known to be NP-hard [Vavasis, 2009], which also implies that its optimization problem is NP-hard. Recently, a variant of NMF where the input

matrix is constrained to be symmetric has become popular for clustering [He et al., 2011, Kuang et al., 2012, 2015]. The problem is known as symmetric NMF (SymNMF), and its goal is to minimize  $\|A - WW^T\|_F^2$  under the constraint that  $W \geq 0$  elementwise, where  $A$  is a symmetric matrix of cluster affinities. Compared with NMF, SymNMF is applicable even when the algorithm does not have direct access to the data instances, but only their pairwise similarity scores. Note that in general the input matrix  $A$  does not need to be nonnegative [Kuang et al., 2012]. SymNMF has been successfully applied in many different settings and was shown to be competitive with standard clustering algorithms; see [Kuang et al., 2012, 2015] and the references therein for more details.

In this paper we investigate a constrained version of SymNMF where the input matrix is required to be both nonnegative and positive semidefinite. Furthermore, we require that  $W$  is normalized such that each row of  $W$  sums to 1. This problem has an interesting application in probabilistic clustering [Zhao et al., 2015] where the  $i$ th row of  $W$  can be interpreted as the probability that the  $i$ th data point lies in each clusters. Formally, we are interested in the following optimization problem, which we name as simplicial SymNMF (SSymNMF):

$$\begin{aligned} & \underset{W}{\text{minimize}} && \frac{1}{4} \|P - WW^T\|_F^2 \\ & \text{subject to} && W \in \mathbb{R}_+^{n \times k}, \quad W\mathbf{1}_k = \mathbf{1}_n \end{aligned} \tag{1}$$

where  $P \geq 0$  and  $P \in \mathbb{S}_+^n$  is positive semidefinite. (1) was proposed as a formulation for probabilistic clustering [Zhao et al., 2015]. The input matrix  $P$  is interpreted as the co-cluster affinity matrix, i.e., entry  $P_{ij}$  corresponds to the degree to which we encourage data instances  $\mathbf{x}_i$  and  $\mathbf{x}_j$  to be in the same cluster. Each row of  $W$  then corresponds to the probability distribution of instance  $\mathbf{x}_i$  being in different clusters. A similar simplicial constraint has been considered in NMF as well [Nguyen et al., 2013], where the goal is to seek a probabilistic part-based decomposition for clustering.



Previous approaches to solve SymNMF or SSymNMF use the penalty method to convert it into an unconstrained optimization problem and then solve it iteratively, using either first-order or second-order methods [Kuang et al., 2012, Zhao et al., 2015, Kuang et al., 2015]. Such methods usually include two loops where the outer loop gradually increases the penalty coefficients and the inner loop finds a stationary point of each fixed penalized objective, hence they are often computationally expensive and slow to converge. In this paper we first give an equivalent geometric description of (1) and then propose a variant of the classic Frank-Wolfe (FW) algorithm [Frank and Wolfe, 1956], a.k.a. the conditional gradient method [Levitin and Polyak, 1966], to solve it. We also provide a non-asymptotic convergence guarantee of our algorithm under an affine invariant stationarity measure (defined in Sec. 2). More specifically, for a given approximation parameter  $\varepsilon > 0$ , we show that the algorithm converges to an  $\varepsilon$ -approximate KKT point of (1) in  $O(1/\varepsilon^2)$  iterations. This rate is analogous to the one derived by Nesterov [2013] for general unconstrained problems (potentially nonconvex) using the gradient descent method, where the measure of stationarity is given by the norm of the gradient. The  $O(1/\varepsilon^2)$  rate has recently been shown to be optimal [Cartis et al., 2010] in the unconstrained setting for gradient methods, and it also matches the best known rate to a stationary point with (accelerated) projected gradient descent [Ghadimi and Lan, 2016, Ghadimi et al., 2016] in the constrained smooth nonconvex setting.

**Contributions.** We first give a generalized definition of the curvature constant [Jaggi, 2013, Lacoste-Julien et al., 2013, Lacoste-Julien, 2016] that works for both convex and nonconvex functions, and we prove a tight bound of it in (1). We then propose a convergence measure in terms of the duality gap and show that the gap is 0 iff KKT conditions are satisfied. Using these two tools, we propose a FW algorithm to solve (1) and show that it has a non-asymptotic convergence rate  $O(1/\varepsilon^2)$  to KKT points. On the algorithmic side, we give a procedure that has the optimal linear time complexity and constant space complexity to implement the *linear minimization oracle* (LMO) in the FW algorithm. As a side contribution, we construct a piecewise linear example where the FW algorithm fails to converge to the optimum. Surprisingly, we can also show that the FW algorithm works if we slightly change the objective function, despite that the new function remains piecewise linear and has an unbounded curvature constant. These two examples then raise an interesting question w.r.t. the necessary condition of the success of the FW algorithm. At the end, we conduct several numerical experiments to demonstrate the efficiency of the proposed algorithm by comparing it with the penalty method and projected gradient descent.

## 2 PRELIMINARY

### 2.1 SymNMF UNDER SIMPLICIAL CONSTRAINT

One way to understand (1) is through its clustering based explanation: the goal is to find a probabilistic clustering of all the instances such that the given co-cluster affinity  $P_{ij}$  for a pair of instances  $(\mathbf{x}_i, \mathbf{x}_j)$  is close to the true probability that  $\mathbf{x}_i$  and  $\mathbf{x}_j$  reside in the same cluster:

$$\begin{aligned} \Pr(\mathbf{x}_i \sim \mathbf{x}_j) &= \sum_{h=1}^k \Pr(c_i = c_j = h) \\ &= \sum_{h=1}^k \Pr(c_i = h) \Pr(c_j = h) = \mathbf{w}_i^T \mathbf{w}_j \end{aligned}$$

where we use the notation  $\mathbf{x}_i \sim \mathbf{x}_j$  to mean “ $\mathbf{x}_i$  and  $\mathbf{x}_j$  reside in the same cluster”;  $c_i$  is the cluster assignment of  $\mathbf{x}_i$  and  $\mathbf{w}_i$  denotes the  $i$ th row vector of  $W$ . Note that the second equation holds because of the assumption of i.i.d. generation process of instances and their cluster assignments.

As a first note, the optimal solution  $W^*$  to (1) is not unique: for any permutation  $\pi_n$  over  $[n]$ , an equivalent solution can be constructed by  $W_{\pi_n}^* = W^* \Pi_{\pi_n}$ , where  $\Pi_{\pi_n}$  is a permutation matrix specified by  $\pi_n$ . This corresponds to an equivalence class of  $W$  by label switching. Hence for any fixed  $k$ , there are at least  $k!$  optimal solutions to (1). The uniqueness of the solution to (1) up to permutation is still an open problem. Huang et al. [2014] studied sufficient and necessary conditions for the uniqueness of SymNMF, but they are NP-hard to check in general.

Zhao et al. [2015] proposed a penalty method to transform (1) into an unconstrained problem and solve it via sequential minimization. Roughly speaking, the penalty method repeatedly solves an unconstrained problem, and enforces the constraints in (1) by gradually increasing the coefficients of the penalty terms. This process iterates until a solution is both feasible and a stopping criterion w.r.t. the objective function is met; see [Zhao et al., 2015, Algo. 1]. The penalty method contains 6 different hyperparameters to be tuned, and it is not even clear whether it will converge to a KKT point of (1). To the best of our knowledge, no other methods has been proposed to solve (1). To solve SymNMF, Kuang et al. [2012] proposed a projected Newton method and Vandaele et al. [2016] developed a block coordinate descent method. However, due to the coupling of columns of  $W$  introduced by the simplicial constraint, it is not clear how to extend these two algorithms to solve (1). On the other hand, the simplicial constraint in (1) restricts the feasible set to be compact, which makes it possible for us to apply the FW algorithm to solve it.

## 2.2 Frank-Wolfe ALGORITHM

The FW algorithm [Frank and Wolfe, 1956, Levitin and Polyak, 1966] is a popular first-order method to solve constrained convex optimization problems of the following form:

$$\text{minimize}_{\mathbf{x} \in \mathcal{D}} f(\mathbf{x}) \quad (2)$$

where  $f : \mathbb{R}^d \rightarrow \mathbb{R}$  is a convex and continuously differentiable function over the convex and compact domain  $\mathcal{D}$ . The FW method has recently attracted a surge of interest in machine learning due to the fact that it never requires us to project onto the constraint set  $\mathcal{D}$  and its ability to cheaply exploit structure in  $\mathcal{D}$  [Clarkson, 2010, Jaggi, 2011, 2013, Lacoste-Julien et al., 2013, Lan, 2013]. Compared with projected gradient descent, it provides arguably wider applicability since projection can often be computationally expensive (e.g., for the general  $\ell_p$  ball), and in some case even computationally intractable [Collins et al., 2008]. At each iteration, the FW algorithm finds a feasible search corner  $\mathbf{s}$  by minimizing a linear approximation at the current iterate  $\mathbf{x}$  over  $\mathcal{D}$ :

$$w(\mathbf{x}) := \min_{\hat{\mathbf{s}} \in \mathcal{D}} f(\hat{\mathbf{s}}) + \nabla f(\mathbf{x})^T (\hat{\mathbf{s}} - \mathbf{x}) \quad (3)$$

The linear minimization oracle (LMO) at  $\mathbf{x}$  is defined as:

$$\mathbf{s} = \text{LMO}(\mathbf{x}) := \arg \min_{\hat{\mathbf{s}} \in \mathcal{D}} f(\hat{\mathbf{s}}) + \nabla f(\mathbf{x})^T (\hat{\mathbf{s}} - \mathbf{x}) \quad (4)$$

Given  $\mathbf{s} = \text{LMO}(\mathbf{x})$ , the next iterate is then updated as a convex combination of  $\mathbf{s}$  and the current iterate  $\mathbf{x}$ . If the FW algorithm starts with a corner as the initial point, then this property implies that at the  $t$ -th iteration, the current iterate is a convex combination of at most  $t + 1$  corners. The difference between  $f(\mathbf{x})$  and  $w(\mathbf{x})$  is known as the *Frank-Wolfe gap* (FW-gap):

$$g(\mathbf{x}) := f(\mathbf{x}) - w(\mathbf{x}) = \max_{\hat{\mathbf{s}} \in \mathcal{D}} \nabla f(\mathbf{x})^T (\mathbf{x} - \hat{\mathbf{s}}) \quad (5)$$

which turns out to be a special case of the general Fenchel duality gap when we transform (2) into an unconstrained problem by adding an indicator function over  $\mathcal{D}$  into the objective function [Lacoste-Julien et al., 2013, Appendix D]. Due to the convexity of  $f$ , we have the following inequality:  $\forall \mathbf{x}, \mathbf{y} \in \mathcal{D}$ ,  $w(\mathbf{y}) \leq f^* \leq f(\mathbf{x})$ , where  $f^*$  is the globally optimal value of  $f$ . Specifically,  $\forall \mathbf{x} \in \mathcal{D}$ ,  $w(\mathbf{x}) \leq f(\mathbf{x})$  and as a result  $g(\mathbf{x})$  can be used as an upper bound for the optimality gap:  $\forall \mathbf{x} \in \mathcal{D}$ ,  $f(\mathbf{x}) - f^* \leq g(\mathbf{x})$ , so that  $g(\mathbf{x})$  is a certificate for the approximation quality of the current solution. Furthermore, the duality gap  $g(\mathbf{x})$  can be computed *essentially for free* in each iteration:  $g(\mathbf{x}) = \nabla f(\mathbf{x})^T (\mathbf{x} - \mathbf{s})$ .

It is well known that for smooth convex optimization problems the FW algorithm converges to an  $\varepsilon$ -approximate solution in  $O(1/\varepsilon)$  iterations [Frank and Wolfe, 1956,

Dunn and Harshbarger, 1978]. This result has recently been generalized to the setting where the LMO is solved only approximately [Clarkson, 2010, Jaggi, 2013]. The analysis of convergence depends on a crucial concept known as the *curvature constant*  $C_f$  defined for a convex function  $f$ :

$$C_f := \sup_{\substack{\mathbf{x}, \mathbf{s} \in \mathcal{D}, \\ \gamma \in (0, 1], \\ \mathbf{y} = \mathbf{x} + \gamma(\mathbf{s} - \mathbf{x})}} \frac{2}{\gamma^2} (f(\mathbf{y}) - f(\mathbf{x}) - \nabla f(\mathbf{x})^T (\mathbf{y} - \mathbf{x})) \quad (6)$$

The curvature constant measures the relative deviation of a convex function  $f$  from its linear approximation. It is clear that  $C_f \geq 0$ . Furthermore, the definition of  $C_f$  only depends on the inner product of the underlying space, which makes it affine invariant. In fact, the curvature constant, the FW algorithm, and its convergence analysis are all affine invariant [Lacoste-Julien et al., 2013]. Later we shall generalize the curvature constant to a function  $f$  that is not necessarily convex and state our result in terms of the generalized definition. The above definition of the curvature constant still works for nonconvex functions, but for a concave function  $f$ ,  $C_f = 0$ , which loses its geometric interpretation as measuring the curvature of  $f$ .

To proceed our discussion, we first establish some standard terminologies used in this paper. A continuously differentiable function  $f$  is called  $L$ -smooth w.r.t. the norm  $\|\cdot\|$  if  $\nabla f$  is  $L$ -Lipschitz continuous w.r.t. the norm  $\|\cdot\|$ :  $\forall \mathbf{x}, \mathbf{y} \in \mathcal{D}$ ,  $\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|$ . Throughout this paper, we will use  $\|\cdot\|$  to mean the Euclidean norm, i.e., the  $\ell_2$  norm for vectors and the Frobenius norm for matrices if not explicitly specified. It is standard to assume  $f$  to be  $L$ -smooth in the convergence analysis of the FW algorithm [Clarkson, 2010, Jaggi, 2013]. As we will see below, the smoothness of  $f$  also implies the boundedness of  $C_f$  on a compact set.

## 3 Frank-Wolfe FOR SYMMMF UNDER SIMPLICIAL CONSTRAINT

### 3.1 A GEOMETRIC PERSPECTIVE

In this section we complement our discussion of (1) in Sec. 2 with a geometric interpretation, which allows us to make a connection between the decision version of (1) to the well known problem of completely positive matrix factorization [Berman and Shaked-Monderer, 2003, Dickinson, 2013, Dickinson and Gijben, 2014]. The decision version of the optimization problem in (1) is formulated as follows:

**Definition 1 (SSymNMF).** Given a matrix  $P \in \mathbb{R}_+^{n \times n} \cap \mathbb{S}_+^n$ , can it be factorized as  $P = WW^T$  for some integer  $k$  such that  $W \in \mathbb{R}_+^{n \times k}$  and  $W\mathbf{1}_k = \mathbf{1}_n$ ?

Clearly, for  $P \in \mathbb{S}_+^n$ , we can decompose it as  $P = UU^T$ , where  $U \in \mathbb{R}^{n \times r}$ ,  $r = \text{rank}(P)$ . Let  $\mathbf{u}_i$  be the  $i$ th row vector of  $U$ ; then  $P$  is the *Gram matrix* of a set of  $r$  dimensional vectors  $\mathcal{U} = \{\mathbf{u}_1, \dots, \mathbf{u}_n\} \subseteq \mathbb{R}^r$ . Similarly,  $WW^T$  can be understood as the Gram matrix of a set of  $k$  dimensional vectors  $\mathcal{W} = \{\mathbf{w}_1, \dots, \mathbf{w}_n\} \subseteq \mathbb{R}^k$ , where  $\mathbf{w}_i$  is the  $i$ th row vector of  $W$ . The simplex constraint in (1) further restricts each  $\mathbf{w}_i \in \Delta^{k-1}$ , i.e.,  $\mathbf{w}_i$  resides in the  $k - 1$  dimensional probability simplex. Hence equivalently, **SSymNMF** asks the following question:

**Definition 2.** *Given a set of  $n$  instances  $\mathcal{U} = \{\mathbf{u}_i\}_{i=1}^n \subseteq \mathbb{R}^r$ , does there exist an integer  $k$  and an embedding  $\mathcal{T} : \mathbb{R}^r \rightarrow \Delta^{k-1}$ , such that inner product is preserved under  $\mathcal{T}$ , i.e.,  $\forall i, j \in [n], \langle \mathbf{u}_i, \mathbf{u}_j \rangle = \langle \mathcal{T}(\mathbf{u}_i), \mathcal{T}(\mathbf{u}_j) \rangle$ ?*

An affirmative answer to **SSymNMF** will give a certificate of the existence of such embedding  $\mathcal{T}$ :  $\mathcal{T}(\mathbf{u}_i) = \mathbf{w}_i, \forall i \in [n]$ . The goal of (1) can thus be understood as follows: find an embedding into the probability simplex such that the discrepancy of inner products between the image space and the original space is minimized. The fact that inner product is preserved immediately implies that distances between every pair of instances are also preserved. If  $k = r$ , such an embedding is also known as an *isometry*. In this case  $\mathcal{T}$  is unitary. Note in the above definition of **SSymNMF** we do not restrict that  $k = r$ , and  $\mathcal{T}$  does not have to be linear.

**SSymNMF** is closely connected to the strong membership problem for completely positive matrices [Berman and Plemmons, 1994, Berman and Shaked-Monderer, 2003]. A completely positive matrix is a matrix  $P$  that can be factorized as  $P = WW^T$  where  $W \in \mathbb{R}_+^{n \times k}$  for some  $k$ . The set of completely positive matrices forms a convex cone, and is known as the completely positive cone (CP cone). From this definition we can see that the decision version of **SymNMF** corresponds to the strong membership problem of the CP cone, which has recently been shown by Dickinson and Gijben [2014] to be NP-hard. Geometrically, the strong membership problem for CP matrices asks the following question:

**Definition 3 (SMEMCP).** *Given a set of  $n$  instances  $\mathcal{U} = \{\mathbf{u}_i\}_{i=1}^n \subseteq \mathbb{R}^r$ , does there exist an integer  $k$  and an embedding  $\mathcal{T} : \mathbb{R}^r \rightarrow \mathbb{R}_+^k$ , such that inner product is preserved under  $\mathcal{T}$ , i.e.,  $\forall i, j \in [n], \langle \mathbf{u}_i, \mathbf{u}_j \rangle = \langle \mathcal{T}(\mathbf{u}_i), \mathcal{T}(\mathbf{u}_j) \rangle$ ?*

Note that the only difference between **SSymNMF** and **SMEMCP** lies in the range of the image space: the former asks for an embedding in  $\Delta^{k-1}$  while the latter only asks for embedding to reside in  $\mathbb{R}_+^k$ . **SSymNMF** is therefore conjectured to be NP-hard as well, but this assertion has not been formally proved yet. A simple reduction from **SMEMCP** to **SSymNMF** does not work: a “yes” answer to the former does not imply a “yes” answer to the latter.

### 3.2 ALGORITHM

We list the pseudocode of the FW method in Alg. 1 and discuss how Alg. 1 can be efficiently applied and implemented to solve **SSymNMF**. We start by deriving the gradient of  $f(W) = \frac{1}{4} \|P - WW^T\|_F^2$ :

$$\nabla f(W) = (WW^T - P)W \in \mathbb{R}^{n \times k} \quad (7)$$

The normalization constraint keeps the feasible set decomposable (even though the objective function  $f$  is not decomposable): it can be equivalently represented as the product of  $n$  probability simplices  $\Delta^{k-1} \times \dots \times \Delta^{k-1} =: \Pi_n \Delta^{k-1}$ . Hence at the  $t$ -th iteration of the algorithm we solve the following linear optimization problem over  $\Pi_n \Delta^{k-1}$ :

$$\begin{aligned} & \underset{S}{\text{minimize}} && \text{tr} \left( S^T \nabla f(W^{(t)}) \right) \\ & \text{subject to} && S \in \Pi_n \Delta^{k-1} \end{aligned} \quad (8)$$

Because of the special structure of the constraint set in (8), given  $\nabla f(W)$ , we can efficiently compute the two key quantities  $\mathbf{x}^{(t+1)}$  and  $g_t$  in Alg. 1 in  $O(nk)$  time and  $O(1)$  space. The pseudocode is listed in Alg. 2. The key observation that allows us to achieve this efficient implementation is that at each iteration  $t \in [T]$ ,  $\text{LMO}(\mathbf{x}^{(t)})$  is guaranteed to be a sparse matrix that contains exactly one 1 in each row. The time complexity of Alg. 2 is  $3nk$ , which can be further reduced to  $2nk$  by additional  $O(n)$  space to store the index of the nonzero element at each row of  $\text{LMO}(\mathbf{x}^{(t)})$ . Alg. 1, together with Alg. 2 as a sub-procedure, is very efficient while at the same time being simple to implement. Furthermore, it does not have any hyperparameter to be tuned: this is in sharp contrast with the penalty method.

---

#### Algorithm 1 Frank-Wolfe algorithm (non-convex variant)

---

**Input:** Initial point  $\mathbf{x}^{(0)} \in \mathcal{D}$ , approximation parameter  $\varepsilon > 0$

- 1: **for**  $t = 0$  to  $T$  **do**
  - 2:   Compute  $\mathbf{s}^{(t)} = \text{LMO}(\mathbf{x}^{(t)}) := \arg \min_{\mathbf{s} \in \mathcal{D}} \mathbf{s}^T \nabla f(\mathbf{x}^{(t)})$
  - 3:   Compute update direction  $\mathbf{d}_t := \mathbf{s}^{(t)} - \mathbf{x}^{(t)}$
  - 4:   Compute FW-gap  $g_t := -\nabla f(\mathbf{x}^{(t)})^T \mathbf{d}_t$
  - 5:   **if**  $g_t \leq \varepsilon$  **then return**  $\mathbf{x}^{(t)}$
  - 6:   Compute  $\gamma_t := \min\{g_t/C, 1\}$  for any  $C \geq \bar{C}_f$  (defined in (10))
  - 7:   Update  $\mathbf{x}^{(t+1)} := \mathbf{x}^{(t)} + \gamma_t \mathbf{d}_t$
  - 8: **end for**
  - 9: **return**  $\mathbf{x}^{(T)}$
-

---

**Algorithm 2** Compute next iterate and the gap function
 

---

**Input:**  $\nabla f(\mathbf{x}^{(t)}), \mathbf{x}^{(t)}$   
 1:  $g_t := \langle \nabla f(\mathbf{x}^{(t)}), \mathbf{x}^{(t)} \rangle$   
 2: **for**  $i = 1$  to  $n$  **do**  
 3:    $g_t \leftarrow g_t - \min_{j \in [k]} \nabla f(\mathbf{x}^{(t)})_{ij}$   
 4: **end for**  
 5: Compute  $\gamma_t := \min\{g_t/C, 1\}$   
 6:  $\mathbf{x}^{(t+1)} := (1 - \gamma_t)\mathbf{x}^{(t)}$   
 7: **for**  $i = 1$  to  $n$  **do**  
 8:    $j_i := \arg \min_{j \in [k]} \nabla f(\mathbf{x}^{(t)})_{ij}$   
 9:    $\mathbf{x}_{ij_i}^{(t+1)} \leftarrow \mathbf{x}_{ij_i}^{(t+1)} + \gamma_t$   
 10: **end for**  
 11: **return**  $\mathbf{x}^{(t+1)}, g_t$

---

### 3.3 CONVERGENCE ANALYSIS

In this section we provide a non-asymptotic convergence rate for the FW algorithm for solving (1) and derive a tight bound for its curvature constant. Our analysis is based on the recent work [Lacoste-Julien, 2016], where we redefine the curvature constant so that the new definition works in both convex and nonconvex settings. Due to space limit, we only provide partial proofs for theorems and lemmas derived in this paper, and refer readers to supplementary material for all detailed proofs.

When applied to smooth convex constrained optimization problems, the FW algorithm is known to converge to an  $\varepsilon$ -approximate solution in  $O(1/\varepsilon)$  iterations. However the convergence of global optimality is usually unrealistic to hope for in nonconvex optimization, where even checking local optimality itself can be computationally intractable [Murty and Kabadi, 1987]. Clearly, to talk about convergence, we need to first establish a convergence criterion. For unconstrained nonconvex problems, the norm of the gradient  $\|\nabla f\|$  has been used to measure convergence [Ghadimi et al., 2016, Ghadimi and Lan, 2016], since  $\lim_{t \rightarrow \infty} \|\nabla f(\mathbf{x}^{(t)})\| = 0$  means every limit point of the sequence  $\{\mathbf{x}^{(t)}\}$  is a stationary point. But such a convergence criterion is not appropriate for constrained problems because a stationary point can lie on the boundary of the feasible region while not having a zero gradient. To address this issue, we will use the FW-gap  $g(\mathbf{x})$  as a measure of convergence. Note that the gap  $g(\mathbf{x})$  works as an optimality gap only if the original problem is convex. To see why this is also a good measure of convergence for constrained nonconvex problems, we first prove the following theorem:

**Theorem 1.** Let  $f$  be a differentiable function and  $\mathcal{D}$  be a convex compact domain. Define  $g(\mathbf{x}) := \max_{\hat{\mathbf{s}} \in \mathcal{D}} \nabla f(\mathbf{x})^T(\mathbf{x} - \hat{\mathbf{s}})$ . Then  $\forall \mathbf{x} \in \mathcal{D}, g(\mathbf{x}) \geq 0$  and  $g(\mathbf{x}) = 0$  iff  $\mathbf{x}$  is a Karush–Kuhn–Tucker (KKT) point.

*Proof.* To see  $g(\mathbf{x}) \geq 0$ , we have:

$$g(\mathbf{x}) := \max_{\hat{\mathbf{s}} \in \mathcal{D}} \nabla f(\mathbf{x})^T(\mathbf{x} - \hat{\mathbf{s}}) \geq \nabla f(\mathbf{x})^T(\mathbf{x} - \mathbf{x}) = 0$$

Reformulate the original constrained problem in the following way:

$$\begin{aligned} & \underset{\mathbf{x}}{\text{minimize}} && f(\mathbf{x}) \\ & \text{subject to} && \mathbb{I}_{\mathcal{D}}(\mathbf{x}) \leq 0 \end{aligned} \quad (9)$$

where  $\mathbb{I}_{\mathcal{D}}(\mathbf{x})$  is the indicator function of  $\mathcal{D}$  which takes value 0 iff  $\mathbf{x} \in \mathcal{D}$  otherwise  $\infty$ . Define  $\mathcal{N}_{\mathcal{D}}(\mathbf{x})$  as the normal cone at  $\mathbf{x}$  in a convex set  $\mathcal{D}$ :

$$\mathcal{N}_{\mathcal{D}}(\mathbf{x}) := \{\mathbf{z} \mid \mathbf{z}^T \mathbf{x} \geq \mathbf{z}^T \mathbf{y}, \forall \mathbf{y} \in \mathcal{D}\}$$

and realize the fact that the subdifferential of the indicator function when  $\mathbf{x} \in \mathcal{D}$  is precisely the normal cone at  $\mathbf{x}$ , i.e.,  $\partial \mathbb{I}_{\mathcal{D}}(\mathbf{x}) = \mathcal{N}_{\mathcal{D}}(\mathbf{x})$ , we have:

$$\begin{aligned} g(\mathbf{x}) = 0 & \Leftrightarrow \max_{\mathbf{y} \in \mathcal{D}} \nabla f(\mathbf{x})^T(\mathbf{x} - \mathbf{y}) = 0 \\ \Leftrightarrow \forall \mathbf{y} \in \mathcal{D}, \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) & \geq 0 \Leftrightarrow -\nabla f(\mathbf{x}) \in \mathcal{N}_{\mathcal{D}}(\mathbf{x}) \end{aligned}$$

Note that the normal cone for a convex set is a convex cone, which implies that  $\exists \lambda \geq 0$ , s.t.,

$$\nabla f(\mathbf{x}) + \lambda \partial \mathbb{I}_{\mathcal{D}}(\mathbf{x}) = 0 \Leftrightarrow \nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \lambda) = 0$$

where  $\mathcal{L}(\mathbf{x}, \lambda) := f(\mathbf{x}) + \lambda \mathbb{I}_{\mathcal{D}}(\mathbf{x})$  is the Lagrangian of (9). By construction,  $\lambda \geq 0$  satisfies the dual feasibility condition and  $\mathbf{x} \in \mathcal{D}$  satisfies the primal feasibility condition, which also means the complementary slackness is satisfied, i.e.,  $\lambda \mathbb{I}_{\mathcal{D}}(\mathbf{x}) = 0$ . It follows that  $g(\mathbf{x}) = 0$  iff  $\mathbf{x}$  is a KKT point of (9). ■

**Remark.** Note that in Thm. 1 we do not assume  $f$  to be convex. In fact we can also relax the differentiability of  $f$ , as long as the subgradient exists at every point of  $f$ . The proof relies on the fact that  $g(\mathbf{x}) = 0$  implies  $-\nabla f(\mathbf{x})$  is in the normal cone at  $\mathbf{x}$ . Thm. 1 justifies the use of  $g(\mathbf{x})$  as a convergence measure: if  $\lim_{t \rightarrow \infty} g(\mathbf{x}^{(t)}) = 0$ , then by the continuity of  $g$ , every limit point of  $\{\mathbf{x}^{(k)}\}$  is a KKT point of  $f$ . Since being a KKT point is also a sufficient condition for optimality when  $f$  is convex, Thm. 1 also recovers the case where  $g(\mathbf{x})$  is used as convergence measure for convex problems.

For a continuously differentiable function  $f$ , we now extend the definition of curvature constant as follows:

$$\bar{C}_f := \sup_{\substack{\mathbf{x}, \mathbf{s} \in \mathcal{D}, \\ \gamma \in (0, 1], \\ \mathbf{y} = \mathbf{x} + \gamma(\mathbf{s} - \mathbf{x})}} \frac{2}{\gamma^2} \left| f(\mathbf{y}) - f(\mathbf{x}) - \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) \right| \quad (10)$$

Clearly  $\bar{C}_f \geq 0$  and the new definition reduces to  $C_f$  when  $f$  is a convex function. In general we have  $\bar{C}_f \geq$

$C_f$  for any function  $f$ . Again,  $\bar{C}_f$  measures the relative deviation of  $f$  from its linear approximation, and is still affine invariant. The difference of  $\bar{C}_f$  from the original  $C_f$  becomes clear when  $f$  is concave: in this case  $C_f = 0$ , but  $\bar{C}_f > 0$  and  $\bar{C}_f = C_{-f}$ . On the downside, the fact that  $\bar{C}_f \geq C_f$  means our asymptotic bound is of constant times larger than the original one proved by Lacoste-Julien [2016], but they share the same asymptotic rate in terms of the given precision parameter  $\varepsilon$ . Finally,  $\bar{C}_f = 0$  iff  $f$  is affine. As in [Jaggi, 2013, Lacoste-Julien et al., 2013], for a smooth function  $f$  with Lipschitz constant  $L$  over compact set  $\mathcal{D}$ , we can bound  $\bar{C}_f$  in terms of  $L$ :

**Lemma 1.** Let  $f$  be a  $L$ -smooth function over a convex compact domain  $\mathcal{D}$ , and define  $\text{diam}(\mathcal{D}) := \sup_{\mathbf{x}, \mathbf{y} \in \mathcal{D}} \|\mathbf{x} - \mathbf{y}\|$ . Then  $\bar{C}_f \leq \text{diam}^2(\mathcal{D})L$ .

The proof of Lemma 1 does not require  $f$  to be convex. Furthermore,  $f$  does not need to be second-order differentiable — being smooth is sufficient. We proceed to derive a convergence bound for Alg. 1 using our new  $\bar{C}_f$ , which better reflects the geometric nonlinearity of nonconvex functions. The main idea of the proof is to bound the decrease of the gap function by minimizing a quadratic function iteratively.

**Theorem 2.** Consider the problem (2) where  $f$  is a continuously differentiable function that is potentially nonconvex, but has a finite curvature constant  $\bar{C}_f$  as defined by (10) over the compact convex domain  $\mathcal{D}$ . Consider running Frank-Wolfe (Algo. 1), then the minimal FW gap  $\tilde{g}_T := \min_{0 \leq t \leq T} g_t$  encountered by the iterates during the algorithm after  $T$  iterations satisfies:

$$\tilde{g}_T \leq \frac{\max\{2h_0\bar{C}_f, \sqrt{2h_0\bar{C}_f}\}}{\sqrt{T+1}}, \quad \forall T \geq 0 \quad (11)$$

where  $h_0 := f(\mathbf{x}^{(0)}) - \min_{\mathbf{x} \in \mathcal{D}} f(\mathbf{x})$  is the initial global suboptimality. It thus takes at most  $O(1/\varepsilon^2)$  iterations to find an approximate KKT point with gap smaller than  $\varepsilon$ .

We comment that the  $O(1/\varepsilon^2)$  convergence rate is the same as the one provided by Lacoste-Julien [2016] using the original curvature constant  $C_f$  for smooth nonconvex functions, and it is also analogous to the ones derived for gradient descent for unconstrained smooth problems and (accelerated) projected gradient descent for constrained smooth problems. The convergence rate of Alg. 1 depends on the curvature constant of  $f$  over  $\mathcal{D}$ . We now bound the smoothness constant  $L$  and the diameter of the feasible set in (1).

**Bound on smoothness constant.** The objective function in (1) is second-order differentiable, hence we can bound the smoothness constant by bounding the spectral norm of the Hessian instead:

**Lemma 2** (Nesterov [2013]). Let  $f$  be twice differentiable. If  $\|\nabla^2 f(\mathbf{x})\|_2 \leq L$  for all  $\mathbf{x}$  in the domain, then  $f$  is  $L$ -smooth.

Note that (7) is a matrix function of a matrix variable, whose Hessian is a matrix of order  $nk \times nk$ . Although one can compute all the elements of the Hessian by computing all the partial derivatives  $\partial \nabla f(W)_{ij} / \partial W_{st}$  separately, this approach is tedious and may hide the structure of the Hessian matrix. Instead we apply matrix differential calculus [Magnus and Neudecker, 1985, Magnus, 2010] to derive the Hessian:

**Lemma 3.** Let  $f(W) = \frac{1}{4}\|P - WW^T\|_F^2$  and define  $\nabla^2 f(W) := \partial \text{vec} \nabla f(W) / \partial \text{vec} W$ . Then:

$$\begin{aligned} \nabla^2 f(W) &= W^T W \otimes I_n + I_k \otimes (WW^T - P) \\ &\quad + (W^T \otimes W)K_{nk} \end{aligned} \quad (12)$$

where  $K_{nk}$  is a commutation matrix such that  $K_{nk} \text{vec} W = \text{vec} W^T$ .

The derivation of the above lemma uses the matrix differential calculus with basic properties of tensors and commutation matrices. Before we proceed, we first present a lemma that will be useful to bound the spectral norm of the above Hessian matrix:

**Lemma 4.**  $\sup_{\substack{W \geq 0, \\ W\mathbf{1}_k = \mathbf{1}_n}} \|W^T W\|_2 = n$ .

We are now ready to bound the spectral norm of the Hessian  $\nabla^2 f(W)$  and use it to bound the smoothness constant of  $f$ .

**Lemma 5.** Let  $c := \|P\|_2$ .  $f = \frac{1}{4}\|P - WW^T\|_F^2$  is  $(3n + c)$ -smooth on  $\mathcal{D} = \{W \in \mathbb{R}_+^{n \times k} \mid W\mathbf{1}_k = \mathbf{1}_n\}$ .

The above lemma follows from Lemma 2 where we use Lemma 4 to help bound the spectral norm of the Hessian matrix in (12).

**Bound on diameter of  $\mathcal{D}$ .** The following lemma can be easily shown:

**Lemma 6.** Let  $\mathcal{D} = \{W \in \mathbb{R}_+^{n \times k} \mid W\mathbf{1}_k = \mathbf{1}_n\}$ . Then  $\text{diam}^2(\mathcal{D}) = 2n$  with respect to the Frobenius norm.

Combining Lemma 6 with Lemma 5 and assuming  $c$  is a constant that does not depend on  $n$ , we immediately have  $\bar{C}_f \leq 2n(3n + c) = O(n^2)$  by Lemma 1. A natural question to ask is: can we get better dependency on  $n$  in the upper bound for  $\bar{C}_f$  given the special structure that  $\mathcal{D} = \Pi_n \Delta^{k-1}$ ? The answer is negative, as we can prove the following lower bound on the Hessian:

**Lemma 7.**  $\inf_{\substack{W \geq 0, \\ W\mathbf{1}_k = \mathbf{1}_n}} \|\nabla^2 f(W)\|_2 \geq n/k^2 - c$ .

Using Lemma 7, we can prove a tight bound on our curvature constant  $\bar{C}_f$ :

**Theorem 3.** The curvature constant  $\bar{C}_f$  for  $f = \frac{1}{4}\|P - WW^T\|_F^2$  on  $\mathcal{D} = \{W \in \mathbb{R}_+^{n \times k} \mid W\mathbf{1}_k = \mathbf{1}_n\}$  satisfies:

$$2n(n/k^2 - c) \leq \bar{C}_f \leq 2n(3n + c)$$

where  $c := \|P\|_2$ . Specifically, we have  $\bar{C}_f = \Theta(n^2)$ .

*Proof.* The upper bound part is clear by combining Lemma 1 and Lemma 5. We only need to show the lower bound. Since  $f$  is twice-differentiable, we have:

$$\begin{aligned} \bar{C}_f &:= \sup_{\substack{\mathbf{x}, \mathbf{s} \in \mathcal{D}, \\ \gamma \in (0, 1], \\ \mathbf{y} = \mathbf{x} + \gamma(\mathbf{s} - \mathbf{x})}} \frac{2}{\gamma^2} |f(\mathbf{y}) - f(\mathbf{x}) - \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x})| \\ &= \sup_{\substack{\mathbf{x}, \mathbf{s} \in \mathcal{D}, \\ \gamma \in (0, 1], \\ \mathbf{y} = \mathbf{x} + \gamma(\mathbf{s} - \mathbf{x})}} \frac{2}{\gamma^2} \frac{1}{2} |(\mathbf{y} - \mathbf{x})^T \nabla^2 f(\xi)(\mathbf{y} - \mathbf{x})| \\ &\geq \sup_{\substack{\mathbf{x}, \mathbf{s} \in \mathcal{D}, \\ \gamma \in (0, 1], \\ \mathbf{y} = \mathbf{x} + \gamma(\mathbf{s} - \mathbf{x})}} \frac{1}{\gamma^2} \|\mathbf{y} - \mathbf{x}\|_2^2 \cdot \inf_{\xi \in \mathcal{D}} \|\nabla^2 f(\xi)\|_2 \\ &\geq \frac{1}{\gamma^2} \gamma^2 \text{diam}^2(\mathcal{D}) \left( \frac{n}{k^2} - c \right) = 2n \left( \frac{n}{k^2} - c \right) \end{aligned}$$

The second equality is due to the mean-value theorem, and the third inequality holds by the definition of inf. The last inequality follows from Lemma 6 and 7. ■

Combining all the analysis above and using Alg. 2 to implement the linear minimization oracle in Alg. 1, we can bound the time complexity of the FW algorithm to solve (1):

**Corollary 1.** The FW algorithm (Alg. 1) achieves an  $\varepsilon$ -approximate KKT point of (1) in  $O(n^3k/\varepsilon^2)$  time.

*Proof.* In each iteration Alg. 1 takes  $O(nk)$  time to compute the gap function as well as the next iterate. Based on Thm. 2, the iteration complexity to achieve an  $\varepsilon$ -approximate KKT point is  $O(\bar{C}_f/\varepsilon^2)$ . The result follows from Thm. 3 showing that  $\bar{C}_f = \Theta(n^2)$ . ■

#### 4 A FAILURE CASE OF Frank-Wolfe ON A NONSMOOTH CONVEX PROBLEM

For convex problems, the theoretical convergence guarantee of Frank-Wolfe algorithm and its variants depends crucially on the smoothness of the objective function: [Freund and Grigas, 2016, Garber and Hazan, 2015, Harchaoui et al., 2015, Nesterov et al., 2015] require the gradient of the objective function to be Lipschitz continuous or Hölder continuous; the analysis in [Jaggi, 2013] requires a finite curvature constant  $C_f$ . The Lipschitz

continuous gradient condition is sufficient for the convergence analysis, but not necessary: Odor et al. [2016] shows that Frank-Wolfe works for a Poisson phase retrieval problem, where the gradient of the objective is not Lipschitz continuous. As we discuss in the last section, Lipschitz continuous gradient implies a finite curvature constant. Hence an interesting question to ask is: does there exist a constrained convex problem with unbounded curvature constant such that the Frank-Wolfe algorithm can find its optimal solution? Surprisingly, the answer to the above question is affirmative. But before we give the example, we first construct an example where the FW algorithm fails when the objective function is convex but nonsmooth.

We first construct a very simple example where the objective function is a piecewise linear and the constraint set is a polytope. We will show that when applied to this simple problem, the FW algorithm, along with its line search variant and its fully corrective variant, do not even converge to the optimal solution. In fact, as we will see shortly, the limit point of the sequence can be arbitrarily far away from the global optimum. Consider the following convex, constrained optimization problem:

$$\begin{aligned} \underset{x_1, x_2}{\text{minimize}} \quad & \max\{5x_1 + x_2, -5x_1 + x_2\} \quad (13) \\ \text{subject to} \quad & x_2 \leq 3, 3x_1 + x_2 \geq 0, -3x_1 + x_2 \geq 0 \end{aligned}$$

We plot the objective function of this example in the left figure of Fig. 1. The unique global optimum for this problem is given by  $\mathbf{x}^* = (0, 0)$  with  $f(\mathbf{x}^*) = 0$ . The feasible set  $\mathcal{D}$  contains three vertices:  $(-1, 3)$ ,  $(1, 3)$  and  $(0, 0)$ . If we apply the FW algorithm to this problem, it is straightforward to verify that the LMO( $\mathbf{x}$ ) is given by:

$$\text{LMO}(\mathbf{x}) = \begin{cases} (-1, 3) & x_1 > 0 \\ (1, 3) & x_1 < 0 \\ \{(-1, 3), (1, 3), (0, 0)\} & x_1 = 0 \end{cases}$$

Note that when  $x_1 = 0$ , the function is not differentiable, and the subdifferential is given by  $\text{conv}\{(5, 1), (-5, 1)\}$ . In this case the FW algorithm chooses arbitrary subgradient from the subdifferential and computes the corresponding LMO. From the fundamental theorem of linear programming, it is easy to see that when  $x_1 = 0$ , LMO( $\mathbf{x}$ ) can be any of the three corners depending on the choice of subgradient at  $\mathbf{x}$ . Now suppose the FW algorithm stops in  $T$  iterations. For any initial point  $\mathbf{x}^{(0)} = (x_1^{(0)}, x_2^{(0)})$  where  $x_1^{(0)} \neq 0$ , the final point output by the FW algorithm will be a convex combination of  $(-1, 3)$ ,  $(1, 3)$ ,  $(0, 0)$  and  $\mathbf{x}^{(0)}$ . Let  $\{\gamma_t\}_{t=1}^T$  be the sequence of step sizes chosen by the FW algorithm. Then we can easily check that  $x_2^{(T)} \geq 3 - \prod_{t=1}^T (1 - \gamma_t)(1 - x_2^{(0)}/3) \rightarrow 3$  as

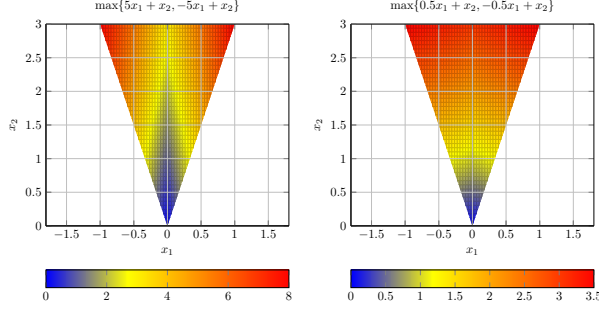


Figure 1: Two convex examples of piecewise linear objective functions on a convex polytope. The FW algorithm fails on the left example but works on the right one. The objective functions of both examples have unbounded curvature constants.

$T \rightarrow \infty$ . Note that we can readily change this example by extending  $\mathcal{D}$  so that the distance between  $\mathbf{x}^{(T)}$  and the optimum  $\mathbf{x}^*$  becomes arbitrarily large. Furthermore, both the line search and the fully-corrective variants fail since the vertices picked by the algorithm remains the same:  $\{(-1, 3), (1, 3)\}$ . Finally, for any regular probability distribution that is absolutely continuous w.r.t. the Lebesgue measure, with probability 1 the initial points sampled from the distribution will converge to suboptimal solutions. As a comparison, it can be shown that the sub-gradient method works for this problem since the function  $f$  itself is Lipschitz continuous [Nesterov, 2013].

Pick  $\mathbf{x} = (-\varepsilon, \delta)$ ,  $\mathbf{s} - \mathbf{x} = (1, 0)$  and  $\mathbf{y} = \mathbf{x} + \gamma(\mathbf{s} - \mathbf{x}) = (\gamma - \varepsilon, \delta)$ , where  $\gamma > \varepsilon$ , and plug them in the definition of the curvature constant  $C_f$ . We have:

$$\begin{aligned} C_f &\geq \lim_{\varepsilon \rightarrow 0^+, \gamma > \varepsilon} \frac{2}{\gamma^2} (f(\mathbf{y}) - f(\mathbf{x}) - \nabla f(\mathbf{x})^T (\mathbf{y} - \mathbf{x})) \\ &= \lim_{\gamma \rightarrow 0^+} \frac{20}{\gamma} = \infty \end{aligned}$$

i.e., the curvature constant of this piecewise linear function is unbounded. The problem for this failure case of FW lies in the fact that the curvature constant is infinity.

On the other hand, we can also show that FW works even when  $C_f = \infty$  by slightly changing the objective function while keeping the constraint set:

$$\begin{aligned} \underset{x_1, x_2}{\text{minimize}} \quad & \max\left\{\frac{1}{2}x_1 + x_2, -\frac{1}{2}x_1 + x_2\right\} \quad (14) \\ \text{subject to} \quad & x_2 \leq 3, 3x_1 + x_2 \geq 0, -3x_1 + x_2 \geq 0 \end{aligned}$$

The objective function of the second example is shown in the right figure of Fig. 1. Still, the unique global optimum for the new problem is given by  $\mathbf{x}^* = (0, 0)$  with  $f(\mathbf{x}^*) = 0$ , but now the LMO( $\mathbf{x}$ ) is:

$$\text{LMO}(\mathbf{x}) = (0, 0), \quad \forall \mathbf{x} \in \mathcal{D}$$

It is not hard to see that FW converges to the global optimum, and the curvature constant  $C_f = \infty$  as well for this new problem. Combining with example (14), we can see that  $C_f < \infty$  is not a necessary condition for the success of FW algorithm on convex problems, either. Piecewise linear functions form a rich class of objectives that are frequently encountered in practice, while depending on the structure of the problem, the FW algorithm may or may not work for them. This thus raises an interesting problem: *can we develop a necessary condition for the success of the FW algorithm on convex problems?* Another interesting question is, *can we develop sufficient conditions for piecewise linear functions under which the FW algorithm converges to global optimum?*

## 5 NUMERICAL RESULTS

We evaluate the effectiveness of the FW algorithm (Alg. 1) in solving (1) by comparing it with the penalty method [Zhao et al., 2015] and the projected gradient descent method (PGD) on 4 datasets (Table 1). These datasets are standard for clustering analysis: two of them are used in [Zhao et al., 2015], and we add two more datasets with various sizes to make a more comprehensive evaluation. The instances in each dataset are associated with true class labels. For each data set, we use the number of classes as the true number of clusters.

Table 1: Statistics about datasets.

Dataset	# inst. ( $n$ )	# feats. ( $p$ )	# clusters ( $k$ )
blood	748	4	10
yeast	1,484	8	10
satimage	4,435	36	6
pendigits	10,992	16	100

Given a data matrix  $X \in \mathbb{R}^{n \times p}$ , we use a Gaussian kernel with fixed bandwidth 1.0 to construct the co-cluster matrix  $P$  as  $P_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2)$ . For each dataset, we use its number of clusters as the rank of the decomposition, i.e.,  $W \in \mathbb{R}^{n \times k}$ . We implement the penalty method based on [Zhao et al., 2015], where we set the maximum number of inner loops to be 50, and choose the step factor for the coefficients of the two penalty terms to be 2. In each iteration of PGD, we use backtracking line search to choose the step size. For all three algorithms, the stop conditions are specified as follows: if the difference of function values in two consecutive iterations is smaller than the fixed gap  $\varepsilon = 10^{-3}$ , or the number of (outer) iterations exceed 50, then the algorithms will stop. Also, for each dataset, all three algorithms share the same initial point.

We plot the convergence speed to local optimum of these

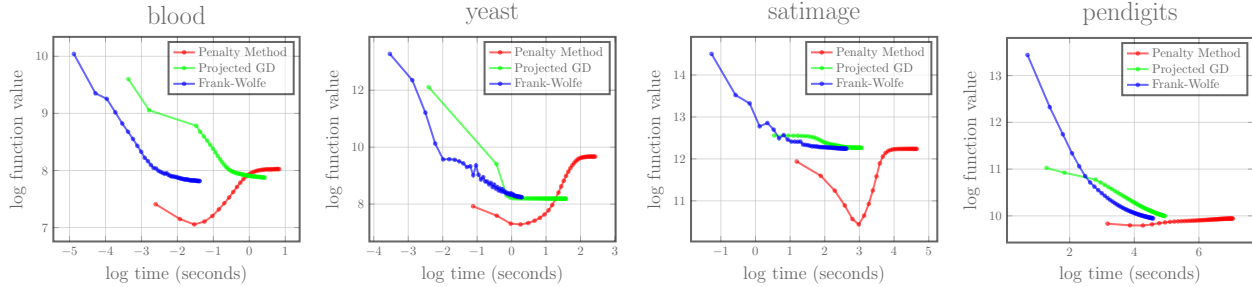


Figure 2: Convergence speed of three algorithms. The  $x$ -axis measures log-seconds and the  $y$ -axis measures log of objective function value. Note that the intermediate iterates of the penalty method are not feasible solutions, so we should only compare the convergent point of the penalty method with the other two.

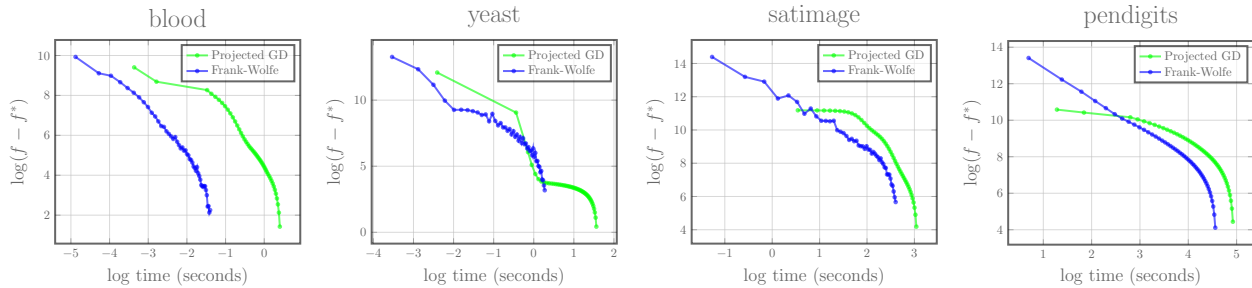


Figure 3: Convergence speed comparison between the projected gradient descent method and the FW algorithm. The  $x$ -axis measures log-seconds and the  $y$ -axis measures  $\log(f - f^*)$ , where  $f$  is the objective function value and  $f^*$  is the local optimum achieved by the algorithm. Note that the local optimum  $f^*$  achieved by PGD and FW can be different.

three algorithms in Fig. 2. Clearly, the penalty method is orders of magnitude slower than both PGD and the FW algorithm, and it usually converges to a worse solution. On the other hand, although the FW algorithm tends to have more iterations before it converges, due to its cheap computation in each iteration, it consistently takes less time than PGD. Another distinction between PGD and FW is that PGD, when implemented with backtracking line search, is a monotone descent method, while FW is not. For a better visualization to compare between the PGD and the FW algorithms, we omit the penalty method in Fig. 3, and draw the log-gap plot of the four datasets. We can confirm from Fig. 3 that both PGD and the FW algorithm have roughly the same order of convergence speed for solving (1), which is consistent with the theoretical result proved in the previous section (Thm. 2). However, the FW algorithm often converges faster than the PGD method, in terms of the gap between the objective function value and local optimum.

## 6 CONCLUSION

We propose a FW algorithm to solve the SymNMF problem under a simplicial constraint. Compared with existing solutions, the proposed algorithm enjoys a non-

asymptotic convergence guarantee to KKT points, is simple to implement, contains no hyperparameter to be tuned, and is also demonstrated to be much more efficient in practice. Theoretically, we establish a close connection of this problem to the famous completely positive matrix factorization by providing an equivalent geometric description. We also derive a tight bound on the curvature constant of this problem. As a side contribution, we give a pair of nonsmooth convex examples where the FW algorithm converges or fails to converge to its optimum. This result raises an interesting question w.r.t. the necessary condition of the success of the FW algorithm.

## Acknowledgements

HZ would like to thank Simon Lacoste-Julien for his insightful comments and helpful discussions. HZ and GG gratefully acknowledge support from ONR, award number N000141512365.

## References

- A. Berman and R. J. Plemmons. *Nonnegative matrices in the mathematical sciences*. SIAM, 1994.
- A. Berman and N. Shaked-Monderer. *Completely positive matrices*. World Scientific, 2003.



- C. Cartis, N. I. Gould, and P. L. Toint. On the complexity of steepest descent, newton’s and regularized newton’s methods for nonconvex unconstrained optimization problems. *SIAM Journal on Optimization*, 20(6):2833–2852, 2010.
- K. L. Clarkson. Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm. *ACM Transactions on Algorithms (TALG)*, 6(4):63, 2010.
- M. Collins, A. Globerson, T. Koo, X. Carreras, and P. L. Bartlett. Exponentiated gradient algorithms for conditional random fields and max-margin markov networks. *Journal of Machine Learning Research*, 9(Aug):1775–1822, 2008.
- P. J. Dickinson and L. Gijben. On the computational complexity of membership problems for the completely positive cone and its dual. *Computational optimization and applications*, 57(2):403–415, 2014.
- P. J. C. Dickinson. *The copositive cone, the completely positive cone and their generalisations*. Citeseer, 2013.
- J. C. Dunn and S. Harshbarger. Conditional gradient algorithms with open loop step size rules. *Journal of Mathematical Analysis and Applications*, 62(2):432–444, 1978.
- M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.
- R. M. Freund and P. Grigas. New analysis and results for the frank-wolfe method. *Mathematical Programming*, 155(1-2):199–230, 2016.
- D. Garber and E. Hazan. Faster Rates for the Frank-Wolfe Method over Strongly-Convex Sets. In *ICML*, pages 541–549, 2015.
- S. Ghadimi and G. Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Mathematical Programming*, 156(1-2):59–99, 2016.
- S. Ghadimi, G. Lan, and H. Zhang. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Mathematical Programming*, 155(1-2):267–305, 2016.
- Z. Harchaoui, A. Juditsky, and A. Nemirovski. Conditional gradient algorithms for norm-regularized smooth convex optimization. *Mathematical Programming*, 152(1-2):75–112, 2015.
- Z. He, S. Xie, R. Zdunek, G. Zhou, and A. Cichocki. Symmetric nonnegative matrix factorization: Algorithms and applications to probabilistic clustering. *IEEE Transactions on Neural Networks*, 22(12):2117–2131, 2011.
- K. Huang, N. D. Sidiropoulos, and A. Swami. Non-negative matrix factorization revisited: Uniqueness and algorithm for symmetric decomposition. *IEEE Transactions on Signal Processing*, 62(1):211–224, 2014.
- M. Jaggi. *Sparse convex optimization methods for machine learning*. PhD thesis, ETH Zürich, 2011.
- M. Jaggi. Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization. In *ICML*, pages 427–435, 2013.
- D. Kuang, C. Ding, and H. Park. Symmetric nonnegative matrix factorization for graph clustering. In *SDM*, pages 106–117. SIAM, 2012.
- D. Kuang, S. Yun, and H. Park. SymNMF: nonnegative low-rank approximation of a similarity matrix for graph clustering. *Journal of Global Optimization*, 62(3):545–574, 2015.
- S. Lacoste-Julien. Convergence rate of Frank-Wolfe for non-convex objectives. *arXiv preprint arXiv:1607.00345*, 2016.
- S. Lacoste-Julien, M. Jaggi, M. Schmidt, and P. Pletscher. Block-Coordinate Frank-Wolfe Optimization for Structural SVMs. In *ICML*, pages 53–61, 2013.
- G. Lan. The complexity of large-scale convex programming under a linear optimization oracle. *arXiv preprint arXiv:1309.5550*, 2013.
- E. S. Levitin and B. T. Polyak. Constrained minimization methods. *USSR Computational mathematics and mathematical physics*, 6(5):1–50, 1966.
- W. Liu, N. Zheng, and X. Lu. Non-negative matrix factorization for visual coding. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP’03). 2003 IEEE International Conference on*, volume 3, pages III–293. IEEE, 2003.
- J. R. Magnus. On the concept of matrix derivative. *Journal of Multivariate Analysis*, 101(9):2200–2206, 2010.
- J. R. Magnus and H. Neudecker. Matrix differential calculus with applications to simple, hadamard, and kronecker products. *Journal of Mathematical Psychology*, 29(4):474–492, 1985.
- K. G. Murty and S. N. Kabadi. Some np-complete problems in quadratic and nonlinear programming. *Mathematical programming*, 39(2):117–129, 1987.
- Y. Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.
- Y. Nesterov et al. Complexity bounds for primal-dual methods minimizing the model of objective function. *Center for Operations Research and Econometrics, CORE Discussion Paper*, 2015.
- D. K. Nguyen, K. Than, and T. B. Ho. Simplicial nonnegative matrix factorization. In *Computing and Communication Technologies, Research, Innovation, and Vision*

- for the Future (RIVF)*, 2013 *IEEE RIVF International Conference on*, pages 47–52. IEEE, 2013.
- G. Odor, Y.-H. Li, A. Yurtsever, Y.-P. Hsieh, Q. Tran-Dinh, M. El Halabi, and V. Cevher. Frank-Wolfe works for non-Lipschitz continuous gradient objectives: Scalable poisson phase retrieval. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 6230–6234. IEEE, 2016.
- A. Vandaele, N. Gillis, Q. Lei, K. Zhong, and I. Dhillon. Efficient and non-convex coordinate descent for symmetric nonnegative matrix factorization. *IEEE Transactions on Signal Processing*, 64(21):5571–5584, 2016.
- S. A. Vavasis. On the complexity of nonnegative matrix factorization. *SIAM Journal on Optimization*, 20(3):1364–1377, 2009.
- W. Xu, X. Liu, and Y. Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 267–273. ACM, 2003.
- H. Zhao, P. Poupard, Y. Zhang, and M. Lysy. Sof: Soft-cluster matrix factorization for probabilistic clustering. In *AAAI*, pages 3188–3195, 2015.

---

# Learning Time Series Segmentation Models from Temporally Imprecise Labels

---

Roy J. Adams and Benjamin M. Marlin  
University of Massachusetts, Amherst

## Abstract

This paper considers the problem of learning time series segmentation models when the labeled data are subject to temporal uncertainty or noise. Our approach augments the semi-Markov conditional random field (semi-CRF) model with a probabilistic model of the label observation process. This augmentation allows us to estimate the parameters of the semi-CRF from timestamps corresponding roughly to the occurrence of transitions between segments. We show how exact marginal inference can be performed in the augmented model in polynomial time, enabling learning based on marginal likelihood maximization. Our experiments on two activity detection problems show that the proposed approach can learn models from temporally imprecise labels, and can successfully refine imprecise segmentations through posterior inference. Finally, we show how inference complexity can be reduced by a factor of 40 using static and model-based pruning of the inference dynamic program.

## 1 INTRODUCTION

Structured prediction frameworks (e.g. conditional random fields [7]) are well-established approaches that often improve on independent prediction models when applied to problems with structured output spaces. In this paper, we consider the problem of learning and inference in structured prediction models for time series segmentation when the labels are subject to temporal imprecision. In this setting, supervision is provided in the form of timestamps that roughly correspond to segment boundaries.

This problem is an instance of weakly supervised learning [5] motivated by real-world data analytic challenges that

arise in the area of mobile health (mHealth) research. A central problem in mHealth research is learning accurate models for detecting health behaviors like eating, smoking, and sleeping from mobile sensor data [14, 18]. A key challenge in such problems is the high cost of obtaining accurately annotated data. mHealth researchers often must estimate the parameters of detection models from limited amounts of data collected in a lab setting where subjects perform scripted activities. This collection process allows researchers to observe subject behavior in detail, but severely limits the amount of data that can be gathered. Further, such data can differ systematically from data collected under real-world scenarios, leading to a lab-to-field generalization gap [10].

An alternative to gathering data in the lab is having subjects self-report their activities in the field, but this suffers from a variety of problems from a modeling perspective including limited frequency of the reports and recall bias. In both lab and field settings, annotations are generally provided in continuous time and may be subject to temporal imprecision. If ignored, such temporal imprecision in the labels may lead to the degraded performance of models trained on these labels. This is the primary problem we address in this paper.

This work makes three main contributions. First, we propose a framework for estimating the parameters of discrete time series segmentation models from temporally imprecise, continuous-time labels. Our approach augments a conditional random field (CRF) model with a probabilistic model of the label observation process. We focus on two classes of CRF models: the semi-Markov CRF (semi-CRF) [15], which models sequence segmentations, and the *hierarchical nested segmentation* (HNS) model [2], an extension of the semi-CRF that models activities composed of repeated short duration events such as eating. Our proposed observation model can account for both temporal imprecision and missingness in the labels. We evaluate this framework on sleep and smoking detection problems using data gathered in both lab

and field settings, demonstrating improved performance compared with ignoring label imprecision.

Second, we enable the synthesis of self-report and wearable sensor data. To the best of our knowledge, current methods for synthesizing these two types of observations are ad hoc and domain specific (e.g. [11]). In this work, we combine sensor data with imprecise continuous-time observations of activity segment boundaries by performing posterior inference in the proposed observation model. This leads to improved predictive performance over treating test-time observations as ground truth.

Finally, we enable the practical application of the proposed framework to long sequences. The model that we present supports exact inference via dynamic programming, but the complexity scales quadratically in the length of the input sequence. We achieve a 40 times speedup by applying a combination of static and model-based pruning techniques, while matching the performance of a model trained on hand-aligned labels.

## 2 RELATED WORK

In this section, we briefly describe related work on weakly supervised learning in the independent classification and structured prediction settings.

**Weakly supervised classification:** Reducing the cost of acquiring labeled data is a fundamental problem in supervised learning. This can often be achieved by lowering the quality of labels in some way. For example, *multiple instance learning* generalizes supervised learning by allowing for sets (or “bags”) of instances to be labeled instead of single instances. It is assumed that a positive bag contains at least one positive instance [9]. Similarly, the *label proportions* framework provides the proportion of each type of label for a group of instances [13]. These approaches avoid the need to label individual instances.

More closely related problems include learning independent classifiers in the presence of label noise [6], and learning independent sequence labeling models from temporally imprecise labels [1]. In both of these frameworks, the true instance labels are assumed to be unobserved. In the label noise framework, noisy instances of the labels are observed, while in the temporally imprecise labels framework, timestamps roughly corresponding to positive instances are observed. Approaches to both problems exist that are based on models of the noisy labeling process that marginalize over the unobserved instance labels during learning. The main difference between these models and the model presented in this work is that these models assume the true labels are independent given the features. In this paper, we consider a more complex structured

prediction setting, which in turn requires more complex observation models and inference algorithms.

**Weakly supervised structured prediction:** There has also been significant research in the area of weakly supervised structured prediction, particularly for computer vision applications. Various standard weak supervision frameworks, such as multiple instance learning, have been extended to structured prediction. [16] extend the multiple instance SVM framework to structured SVMs by considering an image to be a bag of pixels or overlapping sub-windows. [4] extend multiple instance learning to an auto-regressive HMM. While applicable to the problem considered in this paper, these methods would require discarding temporal information that was shown to be valuable in [1].

Another common approach is to assume that only a subset of the label variables in the model are exactly observed. This can be handled by marginalizing out the unobserved variables [17]; however, this framework cannot incorporate auxiliary observations such as continuous observation timestamps. [8] incorporate domain knowledge in the form of constraints on the marginal label distributions. These constraints can be enforced on unlabeled data, allowing for weak supervision. [12] use a similar constraint based approach where image tags are used to form constraints on the set of possible image segmentations. The approach in this work can be interpreted as using observation timestamps to place soft constraints on the set of segmentations; however, by using soft constraints, we can explicitly model the notion of temporal proximity.

## 3 NOTATION AND BACKGROUND

Many mHealth detection problems involve inferring activity segments from sensor data. Past work has shown improved performance when using conditional random field-based structured models to infer such segmentations [2]. We begin by the defining notation used for the input sequences and output structures in this type of problem. We then briefly review the Semi-Markov CRF model that this work extends.

### 3.1 Notation

We assume that the input data consists of  $N$  multivariate time series that we will call **sessions**. Each session contains a set of time-aligned signals gathered from one or more sensors. Separate sessions may correspond to data from different subjects data or to data from the same subject collected at different times. We assume that each session  $n$  has been discretized into a sequence of  $L_n$  potentially overlapping sub-windows and that a feature

vector  $\mathbf{x}_{ni} \in \mathbb{R}^D$  has been extracted for each sub-window  $i$ . We refer to each sub-window  $i$  as an **instance**. Further, each instance  $i$  in session  $n$  is associated with a timestamp  $t_{ni}$  which may correspond to the start, end, or other point of interest associated with instance  $i$ . We refer to the complete sequence of feature vectors  $\mathbf{x}_n = \{\mathbf{x}_{ni}\}_{i=1, \dots, L_n}$  as the **input sequence** and the complete sequence of timestamps  $\mathbf{t}_n = \{t_{ni}\}_{i=1, \dots, L_n}$  as the **timestamp sequence**. Where it does not cause ambiguity, we will drop the session index  $n$ . We use the notation  $\mathbf{x}_{j:k} = \{\mathbf{x}_i\}_{i=j, \dots, k}$  to refer to the subsequence of  $\mathbf{x}$  beginning at  $j$  and ending at  $k$  (this applies to any sequence).

In this work, our goal is to learn a model that produces a labeled segmentation of the input sequence  $\mathbf{x}$ . We represent such a segmentation as a sequence  $\mathbf{y} = \{y_s\}_{s=1, \dots, S}$  of segments where a segment  $y_s = (c_s, j_s, k_s)$  is a tuple containing a label  $c_s \in \mathcal{C}$ , a start position  $j_s \in \{1, \dots, L\}$ , and an end position  $k_s \in \{1, \dots, L\}$ . To ensure only valid segmentations, we assume  $j_1 = 1$ ,  $k_S = L$ , and  $k_s = j_{s+1}$  for all  $1 \leq s \leq S - 1$ . Our goal, then, is to learn the distribution  $p(\mathbf{y}|\mathbf{x}, \mathbf{t})$ . We will parameterize this distribution as a semi-Markov CRF.

### 3.2 Semi-Markov Conditional Random Fields

The semi-CRF [15] associates each segment  $y_s$  with a feature function  $\mathbf{f}(y_s, c_{s-1}, \mathbf{x}, \mathbf{t})$  which may depend on the segment  $y_s$ , the label of the previous segment  $c_{s-1}$ , and the complete feature and timestamp sequences  $\mathbf{x}$  and  $\mathbf{t}$ . The function  $f$  maps these inputs to a length  $F$  feature vector. Given a parameter vector  $\theta \in \mathbb{R}^F$ , the distribution over segmentations is given by

$$p_\theta(\mathbf{y}|\mathbf{x}, \mathbf{t}) = \frac{\prod_s \exp(\langle \theta, \mathbf{f}(y_s, c_{s-1}, \mathbf{x}, \mathbf{t}) \rangle)}{Z_\theta(\mathbf{x}, \mathbf{t})} \quad (1)$$

Both maximum a posteriori (MAP) and marginal inference can be performed in this model by dynamic programs with complexity  $\mathcal{O}(|\mathcal{C}|^2 L^2)$  [15]. The parameters  $\theta$  are typically estimated using maximum likelihood estimation, however, this requires observing the ground truth segmentations. In settings such as mHealth, acquiring the exact segmentation boundaries may be costly or even impossible. We next present our proposed method for estimating the parameters of the semi-CRF model from timestamps corresponding roughly to segment boundaries.

## 4 LEARNING SEMI-CRF MODELS FROM TEMPORALLY IMPRECISE LABELS

Let  $\mathbf{z} = \{z_m\}_{m=1, \dots, M}$  be a sequence of **observations** where each observation  $z_m$  is a timestamp corresponding to a particular kind of transition. For example, each  $z_m$

may be the time a subject reported going to sleep marking the start of a sleep segment. For ease of exposition, we will assume that there is only one type of observation and will later generalize to multiple observation types. To map between our labels  $\mathbf{y}$  and our observations  $\mathbf{z}$ , let  $\mathbf{o} = \{o_i\}_{i=1, \dots, L}$  be a sequence of latent binary variables where  $o_i = 1$  if and only if instance  $i$  is associated with an observation. Under the assumption that observations are recorded in the order they actually occurred and  $\sum_i o_i = M$ ,  $\mathbf{o}$  defines a matching between instances in the input sequence and observations in the observation sequence.

We model the observation sequence using a generative model with three components. The base segmentation model  $p_\theta(\mathbf{y}|\mathbf{x}, \mathbf{t})$  is the semi-CRF model whose parameters we are interested in estimating. The observation indicator distribution  $p_\pi(o|y_s, c_{s-1}, i)$  models the probability that instance  $i$  is associated with an observation given the segment it is contained in and the label of the previous segment. Finally, the observation timestamp density  $p_\phi(z|t)$  models the timestamp of an observation  $z$  given the timestamps  $t$  with which it is associated. For example, we may use a simple Bernoulli distribution for  $p_\pi(o|y_s, c_{s-1}, i)$  and a normal distribution centered at  $t$  for  $p_\phi(z|t)$ . The specific choices for these distributions are domain specific and we demonstrate a couple different choices in section 5. With these distributions, we can now write the observation generation process as shown below:

```

1:  $M \leftarrow 0$ 
2:  $\mathbf{y} \sim p_\theta(\mathbf{y}|\mathbf{x})$ 
3: for  $s = 1, \dots, S$  do
4:   for  $i = j_s, \dots, k_s$  do
5:      $o_i \sim p_\pi(o|y_s, c_{s-1}, i)$ 
6:     if  $o_i = 1$  then
7:        $M \leftarrow M + 1$ 
8:        $z_M \sim p_\phi(z|t_i)$ 

```

This generative process asserts that a complete segmentation is first sampled according to the semi-CRF model. Next, each instance either generates an observation or not according to  $p_\pi(o|y_s, c_{s-1}, i)$ . Finally, if instance  $i$  does generate an observation, an observation timestamp is sampled from  $p_\phi(z|t_i)$ . The variable  $M$  counts the number of generated observations. We note that additional structure could be encoded into the label observation process at the cost of a potentially more complex inference algorithm.

$$p_\omega(\mathbf{z}, \mathbf{y}, \mathbf{o}|\mathbf{x}, \mathbf{t}) = p_\theta(\mathbf{y}|\mathbf{x})p_\pi(\mathbf{o}|\mathbf{y})p_\phi(\mathbf{z}|\mathbf{o}, \mathbf{t}) \quad (2)$$

$$p_\pi(\mathbf{o}|\mathbf{y}) = \prod_s \prod_{i=j_s}^{k_s} p_\pi(o_i|y_s, c_{s-1}, i) \quad (3)$$

$$p_\phi(\mathbf{z}|\mathbf{o}, \mathbf{t}) = \prod_{m=1}^M p_\phi(z_m|t_{i(m)}) \quad (4)$$

The joint model implied by this generative process is

given in Equation 2 where the set of all parameters in the model is  $\omega = \{\theta, \pi, \phi\}$ . The distributions  $p_\pi(\mathbf{o}|\mathbf{y})$  and  $p_\phi(\mathbf{z}|\mathbf{o}, \mathbf{t})$  are defined in Equations 3 and 4. We define  $i(m)$  as the function mapping observation  $m$  to the instance that generated it.

#### 4.1 Inference and Learning

To learn the parameters of this model, we maximize the log marginal likelihood  $\mathcal{L}(\omega|\mathcal{D})$ :

$$\mathcal{L}(\omega|\mathcal{D}) = \sum_{n=1}^N \log p_\omega(\mathbf{z}_n|\mathbf{x}_n, \mathbf{t}_n) \quad (5)$$

$$p_\omega(\mathbf{z}|\mathbf{x}, \mathbf{t}) = \sum_{\mathbf{y} \in \mathcal{Y}} \sum_{\mathbf{o} \in \mathcal{O}} p_\omega(\mathbf{z}, \mathbf{y}, \mathbf{o}|\mathbf{x}, \mathbf{t}) \quad (6)$$

where  $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{t}_n, \mathbf{z}_n)\}_{n=1, \dots, N}$  consists of the observed data for all sessions. We perform this optimization using standard gradient methods. Here, we consider the gradient equation for each of the three parameter groups:  $\theta$ ,  $\pi$ , and  $\phi$ . These equations are presented primarily to give intuition for what maximum likelihood estimation is doing in this model. The gradient equations for  $\pi$  and  $\phi$  are shown below.

$$\begin{aligned} \nabla_\phi \log p_\omega(\mathbf{z}|\mathbf{x}, \mathbf{t}) \\ = \sum_{m=1}^M \mathbb{E}_{p_\omega(i(m)|\mathbf{z}, \mathbf{x}, \mathbf{t})} [\nabla_\phi \log p_\phi(z_m|t_{i(m)})] \end{aligned} \quad (7)$$

$$\begin{aligned} \nabla_\pi \log p_\omega(\mathbf{z}|\mathbf{x}, \mathbf{t}) \\ = \sum_{i=1}^L \mathbb{E}_{p_\omega(o_i, \mathbf{y}|\mathbf{z}, \mathbf{x}, \mathbf{t})} [\nabla_\pi \log p_\pi(o_i|\mathbf{y})] \end{aligned} \quad (8)$$

Both gradient equations take the form of a posterior expectation of the log gradient of the relevant distribution. The gradient with respect to the base classifier parameters also takes the form of an expected gradient of a log density and is shown below.

$$\begin{aligned} \nabla_\theta \log p_\omega(\mathbf{z}|\mathbf{x}, \mathbf{t}) &= \mathbb{E}_{p_\omega(\mathbf{y}|\mathbf{z}, \mathbf{x}, \mathbf{t})} [\nabla_\theta \log p_\theta(\mathbf{y}|\mathbf{x})] \\ &= \mathbb{E}_{p_\omega(\mathbf{y}|\mathbf{z}, \mathbf{x}, \mathbf{t})} [\nabla_\theta \langle \theta, \mathbf{f}(\mathbf{x}, \mathbf{t}, \mathbf{y}) \rangle] - \nabla_\theta Z_\theta(\mathbf{x}) \quad (9) \\ &= \mathbb{E}_{p_\omega(\mathbf{y}|\mathbf{z}, \mathbf{x}, \mathbf{t})} [\mathbf{f}(\mathbf{x}, \mathbf{t}, \mathbf{y})] - \mathbb{E}_{p_\theta(\mathbf{y}|\mathbf{x})} [\mathbf{f}(\mathbf{x}, \mathbf{t}, \mathbf{y})] \end{aligned}$$

where  $\mathbf{f}(\mathbf{x}, \mathbf{t}, \mathbf{y})$  denotes the sufficient statistics function for the semi-CRF model. In this case, the log-linear form of the semi-CRF model gives us the further interpretation that the learning algorithm is trying to match the expected sufficient statistics under the base semi-CRF model to the posterior expected sufficient statistics given by the observation model. This is in contrast to typical maximum likelihood estimation for a log-linear model which would match the expected sufficient statistics under the model to the observed sufficient statistics.

The primary computational challenge of this learning pro-

cedure is calculating the log marginal likelihood. This can be done exactly using a dynamic program for calculating  $p_\omega(\mathbf{z}|\mathbf{x}, \mathbf{t})$ . An entry in the dynamic programming table  $\alpha$  has the following interpretation:  $\alpha(k, c, m)$  is the unnormalized probability that the input subsequence  $\mathbf{x}_{1:k}$  generated the observation subsequence  $\mathbf{z}_{1:m}$  given that the last segment in  $\mathbf{y}$  has label  $c$ . Or, written mathematically:

$$\alpha(k, c, m) \propto p_\omega(\mathbf{z}_{1:m}|\mathbf{x}_{1:k}, \mathbf{t}_{1:k}, c_{|\mathbf{y}|} = c) \quad (10)$$

Filling in this table has complexity  $\mathcal{O}(|\mathcal{C}|^2 L^2 M)$  where  $L$  is the length of the input sequence,  $\mathcal{C}$  is the set of possible segment labels, and  $M$  is the length of the observation sequence. A full description of this dynamic program can be found in section 2 of the supplementary materials. We use reverse-mode automatic differentiation [3] to derive a dynamic program with the same complexity to calculate the necessary gradients for learning.

#### 4.2 MAP Inference

Our second goal is to combine temporal observations, such as self-reported activities, and wearable sensor input to infer behaviors. That is, we would like to infer the most likely segmentation of the input sequence given  $\mathbf{x}$ ,  $\mathbf{t}$ , and  $\mathbf{z}$ . To do this, we perform full maximum a posteriori (MAP) inference over both  $\mathbf{y}$  and  $\mathbf{o}$

$$\mathbf{y}^*, \mathbf{o}^* = \arg \max_{\mathbf{y}, \mathbf{o}} p_\omega(\mathbf{y}, \mathbf{o}|\mathbf{z}, \mathbf{x}, \mathbf{t}) \quad (11)$$

The same dynamic program used to calculate the marginal likelihood can be used to perform MAP inference by swapping summation over  $\mathbf{y}$  and  $\mathbf{o}$  for maximization with no change in the computational complexity.

#### 4.3 Multiple Observation Types

In some settings, it may be desirable to allow for multiple types of observations. For example, we may want to include observations of both the beginning and end of sleep. This can be handled by including multiple observation sequences  $\mathbf{z}^{(l)}$  each with length  $M^{(l)}$  and observation indicator sequences  $\mathbf{o}^{(l)}$  where  $l$  indicates the observation type. Observation sequences of each type are assumed to be independent conditioned on the segmentation  $\mathbf{y}$  and the ordering assumption need not hold between types. The complexity of inference in this setup is  $\mathcal{O}(|\mathcal{C}|^2 L^2 \prod_l M^{(l)})$ .

## 5 EXPERIMENTS AND RESULTS

We evaluated the proposed framework’s ability to accommodate the temporal label imprecision that arises in both the lab and field settings on two mHealth detection problems: sleep detection and cigarette smoking detection. In

this section we describe the datasets and models used as well as the results of these evaluations.

## 5.1 Sleep detection

We evaluated our framework’s performance on data from the field using the Extrasensory<sup>1</sup> dataset [18]. This dataset contains signals from a variety of sensors including the accelerometer, gyroscope, GPS, and microphone on a mobile device as well as a wrist-worn accelerometer. Subjects carried these sensors during daily activities and self-reported a range of activities such as sleeping, eating, and exercising. We focus on the sleep detection problem, as this was one of the more abundantly reported activities. Signals from all sensors were recorded for 20 seconds every minute leading to a natural one minute discretization, which we downsampled to one instance every two minutes in order to run a large number of experiments (a 2x downsample results in a 4x inference speedup). We partitioned the data into 24 hour sessions beginning and ending at 2:00pm and dropped any session with less than four hours of recorded data or less than one hour of reported sleep. This resulted in 80 sessions from 28 subjects. While the researchers corrected obvious conflicts in the self-reported activities, there is no ground truth for this data, so we evaluated against the cleaned self-reported sleep. To simulate extra noise in the observation process, we added further synthetic noise (described below) to the observation timestamps.

**Instance Features:** We used the full set of instance features reported in [18] which include a number of statistical features calculated on the various accelerometer and gyroscope sensors, relative features calculated on the GPS positions, and discrete time-of-day features.

**Model:** Our goal in the sleep detection problem is to segment the input sequence into periods of sleep and non-sleep. We use a binary semi-CRF with a constraint that consecutive segments may not have the same label. We included as features the sum of all instance level features within segment  $x_{jk} = \sum_{i=j}^k x_i$  and duration based features  $\mathbb{I}[c_m = 1](t_k - t_j)$  and  $\mathbb{I}[c_m = 1](t_k - t_j)^2$ , which are similar to putting a normal distribution on the duration of sleeping activities<sup>2</sup>. We placed a zero-mean gaussian prior with tuned variance on the parameters of the semi-CRF model (i.e.  $\ell_2$  regularization).

We included two types of observations: the beginning of sleep  $\mathbf{z}^{(1)}$  and the end of sleep  $\mathbf{z}^{(2)}$ . Because sleep was observed in all sessions, we used a deterministic observation indicator distribution. If instance  $i$  is the beginning

of a sleep segment, it must generate an observation  $z_m^{(1)}$  and likewise for the end of a sleep segment. No other instances may generate observations in this model.

To model the procedure of self-reporting when you go to sleep and when you wake up, we used a one-sided distribution to model the observation timestamp noise. We used the following exponential distributions to model observation timestamp noise:

$$p_\phi(z_m^{(1)}|t_{i(m)}) = \text{Exp}(t_{i(m)} - z_m^{(1)}; \lambda)$$

$$p_\phi(z_m^{(2)}|t_{i(m)}) = \text{Exp}(z_m^{(2)} - t_{i(m)}; \lambda)$$

We placed an inverse-Gamma prior with shape  $\alpha = 1$  and scale  $\beta = 1$  on  $\lambda$ . We found parameter estimation to be fairly insensitive to changes in the settings of this prior distribution and used informative values for  $\alpha$  and  $\beta$ .

**Train and Test Procedures:** We evaluated performance using a 10-fold cross-validation procedure, where folds were calculated at the session level. The strength of the  $\ell_2$  regularizer was tuned to maximize instance-level  $F_1$  over a logarithmic grid using a further 9-fold evaluation. This procedure is equivalent to assuming that some of the data has been labeled for tuning purposes. Predictions were evaluated against the self-reported labels.

**Experiments:** We compared semi-CRF models trained in two ways. First, we trained a semi-CRF model based on a naive alignment defined by mapping each augmented observation to the nearest instance (semi-NV). Second, we trained a semi-CRF model using the proposed weak supervision framework applied to the augmented observations (semi-WS). To test these models under a variety of noise conditions, we augmented the observation timestamps by adding different amounts of exponentially distributed noise and trained both models using these augmented observations. Finally, we tested each model when provided with different amounts of information. At test time, each model was given either all segment start observations (Start), all segment end observations (End), neither observations (None), or both observations (Start+End). Observations were incorporated into semi-WS as described in section 4.2, and were incorporated into semi-NV by mapping the provided observations to the nearest instance and performing MAP inference over the label set constrained to agree with the mapped observations.

The results from these experiments are shown in figure 1. The three plots correspond to models trained and tested on observations augmented with standard deviation  $\lambda = 0, 30, 60$  minutes of temporal noise. Within each plot, the performance for each model when conditioned on different amounts of information is shown. In all cases, semi-WS outperforms semi-NV. The performance gap grows both as the standard deviation of the observation

<sup>1</sup><http://extrasensory.ucsd.edu/>

<sup>2</sup> $\mathbb{I}[\cdot]$  is the indicator function

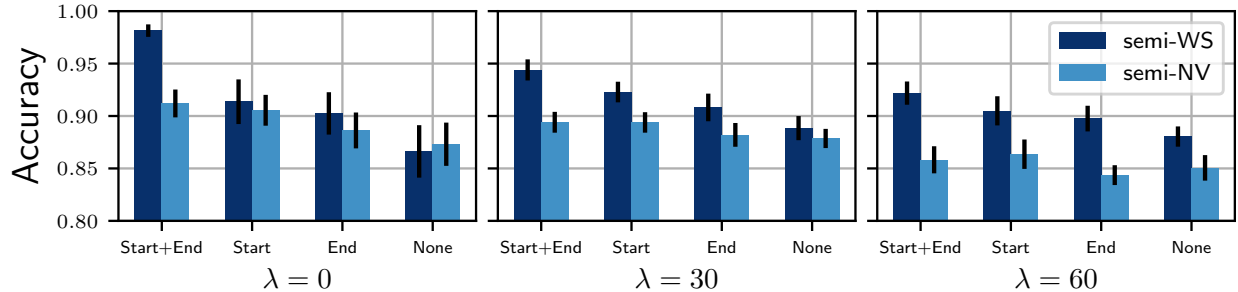


Figure 1: Performance for the semi-WS and semi-NV models on the sleep detection problem when trained on data with  $\text{Exp}(\lambda)$  distributed noise (measured in minutes) added to the observation timestamps. Each plot shows the performance of both models when conditioned on all segment start observations (Start), all segment end observations (End), neither (None), or both (Start+End) at test time.

noise increases and as the amount of information available at test time increases. This indicates that semi-WS is better able to learn from temporally imprecise labels, and that using an explicit observation model is useful when incorporating imprecise observations.

## 5.2 Smoking detection

We evaluated the proposed framework’s ability to handle the types of imprecision that arise in a laboratory setting using the puffMarker smoking dataset [14]. This data was collected in a lab setting where subjects were fitted with chest-worn respiration monitors and wrist-worn actigraphy sensors and asked to smoke a cigarette while an observer marked the occurrence of smoking puffs using a mobile phone app. The respiration signal was discretized into a sequence of non-overlapping respiration cycles (a single inhalation and exhalation) and the goal is to label each respiration cycle as a smoking puff or not and segment the respiration cycles into periods of smoking and non-smoking activities. We created sessions by including random amounts of non-smoking on either side of each recorded smoking activity resulting in 23 sessions from five subjects. In addition to the raw observation timestamps, researchers visualized the respiration signal and hand-aligned the observation timestamps to respiration signal. We treat these hand aligned labels as ground truth for the purposes of evaluation, though we acknowledge that there may be errors in the alignment process. While most experiments on this data were conducted using the true observation timestamps, we also tested the robustness of our framework to extra noise which was generated synthetically and added to the raw observation timestamps.

**Instance Features:** Features were extracted from the respiration monitor data for each respiration cycle according to [14]. Further, we extracted features from the actigraphy data using the following procedure: Let  $t_i$  be the timestamp of the maximum peak in respiration cycle  $i$ .

For each actigraphy channel, extract a window beginning 8 seconds before  $t_i$  and ending 1 second after  $t_i$  and calculate as features the mean, max, min, standard deviation, median, and five bin histogram of the channel’s signal within this window. The actigraphy channels included were accelerometer x, y, and z, accelerometer magnitude, gyroscope x, y, and z, gyroscope magnitude, and pitch and roll angles for a total of 100 actigraphy based features. Pitch and roll calculations using accelerometer data are only valid when the hand is stationary, so these signals were filtered using the procedure described in [14].

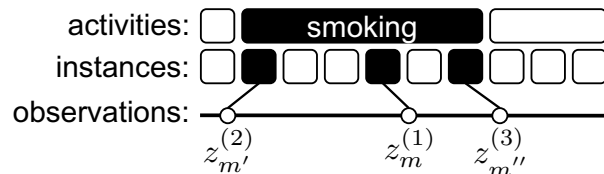
Respiration and actigraphy-based features have different properties as a function of time. Due to the method we used to extract actigraphy-based features, these features tend to be smooth through time, particularly as compared to the respiration features extracted from non-overlapping windows. The smooth noise model we propose tends to over-emphasize temporally smooth features at the expense of less smooth features. To combat this effect, we use the actigraphy features to augment the respiration features in a manner similar to the filtering approach used in [14].

We trained a logistic regression model on a small set of instances with hand-aligned labels using only the actigraphy features, then took the predictions from this model and augmented the respiration features as  $\mathbf{x}_{aug} = [\hat{c}_{act}\mathbf{x}_{resp} \ (1 - \hat{c}_{act})\mathbf{x}_{resp}]$  where  $\hat{c}_{act} \in \{0, 1\}$  is a prediction from the filter model and  $\mathbf{x}_{resp}$  is the vector containing only respiration features. A similar effect might be achieved by including interaction effects between the actigraphy and respiration features; however, this would result in more than 10,000 features. The filtering approach can therefore also be thought of as first doing a supervised compression of the actigraphy features and then doing a polynomial basis expansion. For more details, see section 3 in the supplemental materials.

**Model:** Our goal in the smoking detection problem is to label each respiration cycle as smoking or non-smoking



and to segment the input sequence into periods of smoking and non-smoking; however, smoking detection differs from typical segmentation problems in that a complete smoking activity contains a mix of smoking puffs and non-smoking respiration cycles. In terms of the model, this means that the instances contained in a positive segment may be both positive or negative (an example of this structure with example observations is shown below). To address this we used an extension of the standard semi-CRF called the hierarchical nested segmentation (HNS) model [2]. Rather than segment a sequence into positive and negative activities, the HNS model segments the sequence into periods between positive instances, termed inter-event spans. Further, the HNS model includes a cardinality potential that models the number of positive instances that make up a positive activity (or the number of consecutive positive inter-event spans). In addition to the instance level features, we included the segment duration,  $t_k - t_j$  and segment duration squared  $(t_k - t_j)^2$  to model the time between positive instances (i.e. the time between puffs on a cigarette). For full details of the HNS model for smoking detection, see [2], and for details on how the HNS model can be written as a semi-CRF, see section 4 of the supplementary materials. We placed a zero-mean gaussian prior with tuned variance on the parameters of the HNS model (i.e.  $\ell_2$  regularization).



As seen in the figure above, we included three types of observations: positive instance observations associated with any inter-event span  $\mathbf{z}^{(1)}$ , activity start observations associated only with the first inter-event span in a complete activity  $\mathbf{z}^{(2)}$ , and activity end observations associated only with the last inter-event span in a complete activity  $\mathbf{z}^{(3)}$ . We used the following Bernoulli distributions for our observation indicator model  $p_\pi(o_i^{(l)}|\mathbf{y})$ :

$$\begin{aligned} p_\pi(o_i^{(1)} = 1 | i \text{ is the start of an inter-event span}) &= \pi_1^{(1)} \\ p_\pi(o_i^{(2)} = 1 | i \text{ is the start of a smoking activity}) &= \pi_1^{(2)} \\ p_\pi(o_i^{(3)} = 1 | i \text{ is the end of a smoking activity}) &= \pi_1^{(2)} \end{aligned}$$

where  $\pi_1^{(1)}, \pi_1^{(2)} \in [0, 1]$ . For the observation timestamp density, we used the following normal distribution:

$$p_\phi(z_m^{(l)}|t_{i(m)}) = \mathcal{N}(z_m^{(l)}; t_{i(m)} + \mu_l, \sigma_l^2)$$

for  $l \in \{1, 2, 3\}$  where  $\phi = \{\mu, \sigma\}$ . This density was chosen to match the empirical noise distribution [1]. We placed a Uniform(0, 1) prior on each  $\pi^{(l)}$ , a standard normal prior on each  $\mu_l$ , and an inverse-Gamma prior

with shape  $\alpha = 1$  and scale  $\beta = 1$  on each  $\sigma_l^2$ .

**Train and Test Procedures:** We evaluated performance using a leave-one-session-out cross-validation procedure. All tuned hyperparameters were tuned to maximize instance level  $F_1$  over a logarithmic grid using a further nested leave-one-session-out evaluation. Predictions were evaluated against the hand-aligned labels.

**Experiment 1 - Pruning Strategies:** While the inference algorithm described in section 4.1 is at most quadratic in the size of each input, the overall run time can be quite high, particularly for long sequences or models with a large label set  $\mathcal{C}$  such as the HNS model. In order to improve inference run times, we consider three strategies to prune the inference dynamic program.

**Maximum segment length:** One straightforward way to constrain the label space is to place a bound on segment lengths. For example, in the case of smoking detection, we might say that two smoking puffs separated by five minutes (or approximately 50 respiration cycles) constitute two separate smoking activities. Adding this constraint reduces the complexity of inference to  $\mathcal{O}(|\mathcal{C}|^2 L B M)$  where  $B$  is the maximum segment length.

**Maximum observation distance:** Depending on the observation process, we might also place a constraint on the maximum time between a true event and an associated timestamp. This corresponds to using a truncated distribution for  $p_\phi(z|t)$ . Given a maximum observation distance of  $r$ , we can upper bound the inference complexity by  $\mathcal{O}(|\mathcal{C}|^2 L \tilde{B} M)$  where  $\tilde{M}$  is the maximum number of observations that could be associated with a single instance or  $\tilde{M} = \max_i \sum_m \mathbb{I}[t_i - r \leq z_m \leq t_i + r]$ . In practice, the average improvement in runtime is better than this.

**Negative instance filtering:** In cascaded classification, a simple classifier is used to filter the label set for a more complex classifier [19]. This technique has been successfully applied to structured prediction problems (e.g. [20]) and we apply it here to filter the space of possible segmentations. Due to the heavy instance level class imbalance in many mHealth problems, it is often easy to learn a high recall instance-level classifier, which can then be used to clamp instance labels to the negative class. Given an instance level classifier, let  $\tilde{c}_i$  be the filter model's prediction for instance  $i$ . Then, during inference, we constrain the set of possible segmentations to agree with the negative predictions of the filter model. Using this filtering procedure, the worst case complexity remains unchanged (it is possible that the filter model filters nothing), but the average case complexity becomes  $\mathcal{O}(\gamma|\mathcal{C}|^2 L B M)$  where  $\gamma$  is the proportion of instances that pass the filter.

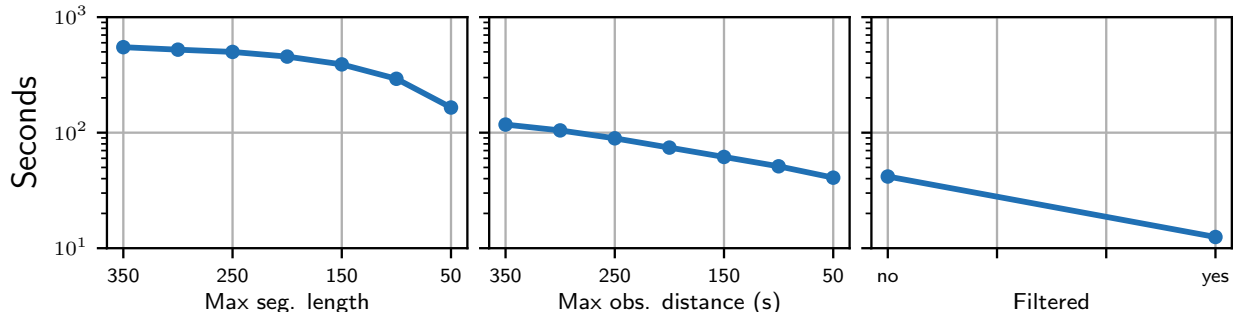


Figure 2: This figure shows the effect of changing the maximum segment length with no observation depth pruning or filtering (left), the effect of changing the maximum observation distance with no filtering (center), and the further marginal effect of filtering approximately 85% of instances (right). The maximum pruning configuration results in a 40x speedup.

To test the effect of the proposed pruning strategies, we ran an ablation experiment to assess the time required to run marginal inference in the HNS model augmented with an observation model using different combinations of pruning techniques. First, we varied the maximum segment length from 350 to 50. Next, with the maximum segment length fixed at 50, we varied the maximum observation distance from 350 to 50. Finally with the maximum segment length and maximum observation distance fixed at 50, we ran inference with and without negative instance filtering. For the filtering model, we used the same actigraphy-based logistic regression model used to perform feature augmentation (Section 5.2, Instance Features). Figure 2 shows the run time in seconds for each of these settings<sup>3</sup>. Using all pruning strategies, the runtime of marginal inference is decreased from approximately 600 seconds to approximately 15 seconds, a 40 times speedup. We use the most aggressive pruning settings in all subsequent experiments.

**Experiment 2 - Prior predictive performance:** We next evaluated the ability of the proposed framework to learn the parameters of the base classifier from imprecise lab data by comparing the HNS model trained in three different ways. First, we trained the HNS model directly on the hand-aligned labels (HNS-HA). This represents the gold standard performance that we would like to achieve. Second, we trained the HNS model on labels generated by associating each observation timestamp with the closest respiration cycle (HNS-NV). This represents the naive baseline and we would expect our procedure to fall somewhere between HNS-HA and HNS-NV. Third, we trained the HNS model using the weak supervision framework proposed above (HNS-WS). Figure 3 shows the performance of all three models on the instance labeling and

segmentation tasks. The HNS-WS model performs approximately as well as the HNS-HA model at both the instance labeling and segmentation tasks while the HNS-NV model performs worse than either. A paired t-test indicates that the improvement in the HNS-WS results over the HNS-NV results is statistically significant in terms of both instance labeling and segmentation ( $p \leq 0.05$ ). These results indicate that we have achieved our primary aim of enabling learning of the HNS model from data with noisy observation timestamps.

### Experiment 3 - Posterior predictive performance:

We evaluated the ability of the HNS-WS model to combine sensor data with timestamp observations at test time. As in the sleep detection experiments, we evaluated all three models when given either all activity start observations (Start), all activity end observations (End), neither (None), or both (Start+End) at test time. The results are shown in Figure 4 (left). Unlike in our sleep detection experiments, all noise present in these observations was real and all evaluations were made against carefully hand aligned labels. While conditioning on segment observations results in improvements for all three models, these gains are much larger for the HNS-WS model. In particular, conditioning on both the segment start and end timestamps results in a 6% error reduction for the HNS-HA model and a 16% error reduction for the HNS-NV model whereas conditioning on the same information results in an 89% error reduction for the HNS-WS model.

In general, we cannot expect the noise we observe in the field to look like the noise we observe in the lab, therefore it is valuable to know how sensitive the HNS-WS model is to the correctness of the observation timestamp model. To test this, we generated synthetic observation timestamps from a normal distribution centered at the true activity start or end and varied the standard deviation of the distribution. The segmentation accuracy of the HNS-WS model when conditioning on these synthetic observations at test

<sup>3</sup>Runtime experiments were performed on a 2.8 GHz Intel Core i7 processor with 8GB of RAM and the inference algorithm was coded in Cython.

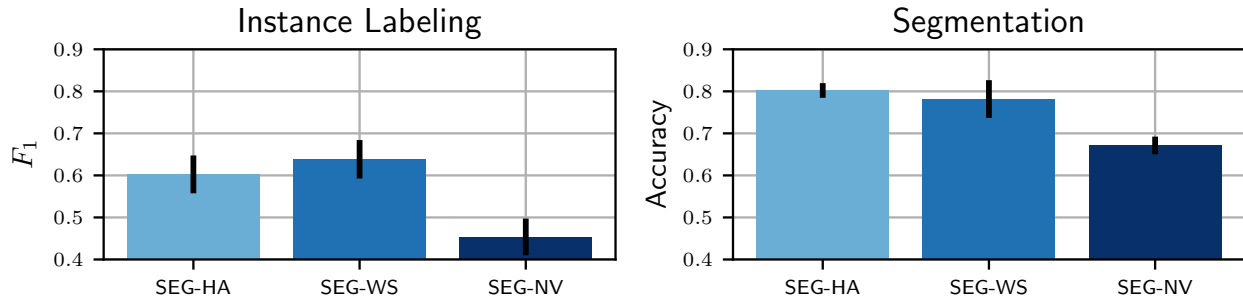


Figure 3: The left plot shows instance level  $F_1$  score for all three models. The right plot shows the segmentation accuracy for all three models. The proposed HNS-WS model significantly outperforms the HNS-NV model. Error bars show one standard error.

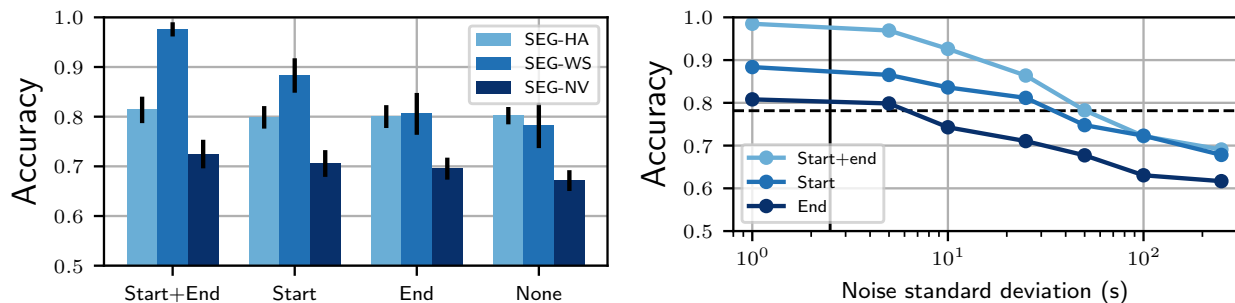


Figure 4: The left plot shows the segmentation accuracy when each HNS model is conditioned on different combinations of observations (segment start, segment end or both). The right plot shows the HNS-WS model when conditioned on segment observations with different amounts of synthetic noise added. The dashed line shows the segmentation accuracy of the HNS-WS model when conditioned on no observations (None) and the solid line shows the empirical standard deviation of the timestamp noise in the data, which reflects what HNS-WS was trained on.

time is shown in Figure 4 (right). The results show that the HNS-WS model can successfully incorporate observations with up to an order of magnitude more noise than was observed at train time. As expected, adding sufficient noise to the observations eventually causes performance to degrade; However, even with large amounts of noise, posterior segmentation accuracy plateaus between 0.6 and 0.7 compared to an accuracy of approximately 0.8 when not conditioning on any observations.

## 6 CONCLUSIONS

In this work, we have addressed the problem of learning time series segmentation models from noisy observation timestamps. We extended the weakly supervised learning framework of [1] to the semi-Markov CRF and HNS models and derived exact and approximate inference algorithms based on dynamic programming. We showed using real sleeping and smoking data that learning the segmentation models in this way can recover the performance of models trained on more expensive hand-aligned labels, while significantly out-performing the naive alignment strategy. Further, we showed that this framework can be

used to combine noisy observations with sensor input at test time in a principled way.

This work suggests a several of interesting research directions for future research. First, in many cases it is much cheaper to gather large amounts self-report data than it is to gather lab data. The proposed framework is capable of incorporating both lab data and self-report data gathered in the field to train or fine-tune a model in a noisy semi-supervised-like learning framework. Second, personalizing detection models is an important goal in mHealth research, but is typically not practical due to the cost of obtaining labels. Our approach opens the possibility of personalizing models using less costly (but more noisy) self-report data from the field.

## Acknowledgments

The authors would like to thank members of the MD2K Center (<http://www.md2k.org>) for helping to enable this research. This work was partially supported by the National Institutes of Health under award 1U54EB020404, and the National Science Foundation under award IIS-1350522.

## References

- [1] Roy Adams and Ben Marlin. Learning time series detection models from temporally imprecise labels. In *Artificial Intelligence and Statistics*, pages 157–165, 2017.
- [2] Roy Adams, Nazir Saleheen, Edison Thomaz, Abhinav Parate, Santosh Kumar, and Benjamin Marlin. Hierarchical span-based conditional random fields for labeling and segmenting events in wearable sensor data streams. In *International Conference on Machine Learning*, pages 334–343, 2016.
- [3] Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *arXiv preprint arXiv:1502.05767*, 2015.
- [4] Xinze Guan, Raviv Raich, and Weng-Keen Wong. Efficient multi-instance learning for activity recognition from time series data using an auto-regressive hidden markov model. In *International Conference on Machine Learning*, pages 2330–2339, 2016.
- [5] Jerónimo Hernández-González, Iñaki Inza, and Jose A Lozano. Weak supervision and other non-standard classification problems: a taxonomy. *Pattern Recognition Letters*, 69:49–55, 2016.
- [6] Rong Jin and Zoubin Ghahramani. Learning with multiple labels. In *Advances in neural information processing systems*, pages 897–904, 2002.
- [7] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 282–289, 2001.
- [8] Gideon S Mann and Andrew McCallum. Generalized expectation criteria for semi-supervised learning with weakly labeled data. *Journal of machine learning research*, 11(Feb):955–984, 2010.
- [9] Oded Maron and Tomás Lozano-Pérez. A framework for multiple-instance learning. *Advances in neural information processing systems*, pages 570–576, 1998.
- [10] Annamalai Natarajan, Gustavo Angarita, Edward Gaiser, Robert Malison, Deepak Ganesan, and Benjamin M Marlin. Domain adaptation methods for improving lab-to-field generalization of cocaine detection using wearable ecg. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 875–885. ACM, 2016.
- [11] Sanjay R Patel, Jia Weng, Michael Rueschman, Katherine A Dudley, Jose S Loredo, Yasmin Mossavar-Rahmani, Maricelle Ramirez, Alberto R Ramos, Kathryn Reid, Ashley N Seiger, et al. Reproducibility of a standardized actigraphy scoring algorithm for sleep in a us hispanic/latino population. *Sleep*, 38(9):1497–1503, 2015.
- [12] Deepak Pathak, Philipp Krahenbuhl, and Trevor Darrell. Constrained convolutional neural networks for weakly supervised segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1796–1804, 2015.
- [13] Novi Quadrianto, Alex J Smola, Tiberio S Caetano, and Quoc V Le. Estimating labels from label proportions. *Journal of Machine Learning Research*, 10(Oct):2349–2374, 2009.
- [14] Nazir Saleheen, Amin Ahsan Ali, Syed Monowar Hossain, Hillol Sarker, Soujanya Chatterjee, Benjamin Marlin, Emre Ertin, Mustafa Al’Absi, and Santosh Kumar. puffMarker: A Multi-sensor Approach for Pinpointing the Timing of First Lapse in Smoking Cessation. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 999–1010, 2015.
- [15] Sunita Sarawagi and William W Cohen. Semi-markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems*, pages 1185–1192, 2004.
- [16] Hyun Oh Song, Ross B Girshick, Stefanie Jegelka, Julien Mairal, Zaid Harchaoui, Trevor Darrell, et al. On learning to localize objects with minimal supervision. In *ICML*, pages 1611–1619, 2014.
- [17] Bill Triggs and Jakob J Verbeek. Scene segmentation with crfs learned from partially labeled images. In *Advances in neural information processing systems*, pages 1553–1560, 2008.
- [18] Yonatan Vaizman, Katherine Ellis, and Gert Lanckriet. Recognizing detailed human context in the wild from smartphones and smartwatches. *IEEE Pervasive Computing*, 16(4):62–74, 2017.
- [19] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2001.
- [20] David Weiss and Benjamin Taskar. Structured prediction cascades. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 916–923, 2010.

---

# Multi-Target Optimisation via Bayesian Optimisation and Linear Programming

---

Alistair Shilton   Santu Rana   Sunil Gupta   Svetha Venkatesh

Deakin University, Geelong, Australia,  
Centre for Pattern Recognition and Data Analytics  
{alistair.shilton,santu.rana,sunil.gupta,svetha.venkatesh}@deakin.edu.au

## Abstract

In Bayesian Multi-Objective optimisation, expected hypervolume improvement is often used to measure the goodness of candidate solutions. However when there are many objectives the calculation of expected hypervolume improvement can become computationally prohibitive. An alternative approach measures the goodness of a candidate based on the distance of that candidate from the Pareto front in objective space. In this paper we present a novel distance-based Bayesian Many-Objective optimisation algorithm. We demonstrate the efficacy of our algorithm on three problems, namely the DTLZ2 benchmark problem, a hyper-parameter selection problem, and high-temperature creep-resistant alloy design.

## 1 INTRODUCTION

Bayesian optimisation (Brochu et al., 2010) is a method for maximising black-box functions that are expensive to evaluate either in terms of time or cost. Bayesian optimisation works by modelling the objective function (typically) using a Gaussian process (GP) (Rasmussen and Williams, 2006). At each iteration a point (called a recommendation) is selected to maximise an acquisition function, where the acquisition function is a measure of the *goodness* of a proposed point. Unlike the black-box function, the acquisition function is cheap to evaluate and therefore amenable to global optimisation.

In the context of multi-objective optimisation, Bayesian optimisation is typically applied using an acquisition function based on expected hypervolume improvement (EHI) (Ponweiser et al., 2008; Emmerich and Klöppel, 2008; Shir et al., 2007; Zaefferer et al., 2013; Shimoyama et al., 2013), which is the expected change in

the hypervolume dominated by the estimated Pareto front (the set of dominant evaluations of prior recommendations in objective space - see figure 1). However this can be expensive to evaluate, particularly in the many-objective case where the number of objectives is large (Wagner et al., 2010; Zaefferer et al., 2013). While optimised algorithms have been developed for calculating EHI for up to 3 dimensions (Hupkens et al., 2015) the general (many-objective (Ishibuchi et al., 2008)) case remains computationally challenging.

An alternative approach is to use a distance-based acquisition function (or score function) (Miranda and Von Zuben, 2015; Yun et al., 2004). Distance-based acquisition functions seek to maximise the signed distance of a point from the estimated Pareto front, as shown in figure 1. Unlike EHI this acquisition function is cheap to evaluate, making its global optimisation (and hence Bayesian optimisation) practical in the many-objective case. While the underlying concept is old (e.g. (Yun et al., 2004)) it has only recently been formalised in a rigorous manner (Miranda and Von Zuben, 2015) in the form of conditions that must be met by a distance (score) function measuring the signed distance *in advance of (dominating)* the Pareto front; whereas (Yun et al., 2004) for example defines the signed distance from the estimated feasible region in any direction, dominating or otherwise. However, while (Miranda and Von Zuben, 2015) defines the conditions that must be met by such a score function, the method implemented therein - namely a GP model with a probability distribution over the gradient - only approximately meets these requirements.

In the present paper we introduce an alternative model based on a modified 1-norm support vector machine (SVM) that is able to *exactly* satisfy the conditions laid down in (Miranda and Von Zuben, 2015). To be precise, we use a restricted 1-norm, 1-class SVM to define a signed distance function which is strictly positive for points that dominate the estimated Pareto front, strictly negative for points dominated by the estimated Pareto

front, and for which signed distance and the dominance relation are congruent.<sup>1</sup> This distance function forms the basis for an acquisition function that is computationally cheap to evaluate and scales well with the number of objectives, thereby making feasible Bayesian many-objective optimisation.

To test our proposed algorithm we have applied it to one benchmark problem and two practical problems. The benchmark problem used is taken from the DTLZ suite of benchmarks (Deb et al., 2005). For practical problems we have chosen a hyper-parameter selection problem and an experimental problem involving high-temperature, creep resistant alloy design.

The first practical problem considered is hyperparameter selection for a multi-class classifier where the relative weights (importance) of the various classes is unknown. While the default assumption often made for such problems is that all classes should have equal weight (or alternatively that their weight should be proportional to their class density) this will not be valid in general. Instead the accuracy of the classifier with respect to each class of training data forms an independent objective, and the problem of hyper-parameter selection in the absence of additional information regarding relative weight is one of multi-objective optimisation.

The second practical problem considered is the design of high-temperature, creep-resistant alloys. High-temperature creep resistant Ni-superalloy is used for making boilers of super-critical thermal power plants. In a joint project with metallurgists we were asked to optimize the current alloy recipe to obtain superior creep resistance than the industry standard. This involves using phase simulation (via ThermoCalc) to design an alloy with maximum good phases (those that improved creep-resistance) and minimum bad phases (those that made the alloy less creep-resistant) over a range of temperatures. The total number of objectives for this experiment is 12, each corresponding to a particular phase and temperature, which leads to recommendation times of up to 1 day/recommendation if EHI is used. We demonstrate that our approach is able to provide a range of potential alloys, each Pareto-optimal in terms of phase contents, for further assessment by the metallurgist.

We note that there exists an abundance of such many-objective optimisation problems in physical systems - for example advanced fibre production (Li et al., 2017). By making many-objective Bayesian optimisation feasible we envisage that such problems will be able to be formulated and solved.

<sup>1</sup>That is, if  $\mathbf{y}$  dominates  $\mathbf{y}'$  then the distance of  $\mathbf{y}$  from the estimated Pareto front, as measured by the score function, is greater than the distance of  $\mathbf{y}'$  from the estimated Pareto front.

## 2 NOTATION

Column vectors are written  $\mathbf{a}, \mathbf{b}, \dots$  with elements  $a_i, b_i, \dots$ . Matrices are written  $\mathbf{A}, \mathbf{B}, \dots$ , with elements  $A_{i,j}, B_{i,j}, \dots$ . If  $\mathbf{f} : \mathbb{X} \subset \mathbb{R}^m \rightarrow \mathbb{R}^n$  is a map from design to objective space then  $\forall \mathbf{x}, \mathbf{x}' \in \mathbb{X}$  we say  $\mathbf{x}$  dominates  $\mathbf{x}'$ , written  $\mathbf{x} \succeq_{\mathbf{f}} \mathbf{x}'$ , if  $f_i(\mathbf{x}) \geq f_i(\mathbf{x}') \forall i$ ; and  $\mathbf{x}$  strongly dominates  $\mathbf{x}'$ , written  $\mathbf{x} \succ_{\mathbf{f}} \mathbf{x}'$ , if  $\mathbf{x} \succeq_{\mathbf{f}} \mathbf{x}' \wedge \mathbf{f}(\mathbf{x}) \neq \mathbf{f}(\mathbf{x}')$ . Analogously,  $\forall \mathbf{y}, \mathbf{y}' \in \mathbb{R}^n$  we say  $\mathbf{y}$  dominates  $\mathbf{y}'$ , written  $\mathbf{y} \succeq \mathbf{y}'$ , if  $y_i \geq y'_i \forall i$ ; and  $\mathbf{y}$  strongly dominates  $\mathbf{y}'$ , written  $\mathbf{y} \succ \mathbf{y}'$ , if  $\mathbf{y} \succeq \mathbf{y}' \wedge \mathbf{y} \neq \mathbf{y}'$ .

## 3 BACKGROUND

Multi-objective optimisation (Deb, 2001; Coello et al., 2002; Miettinen, 1999) extends standard single-objective optimisation to the case where there are multiple, potentially conflicting objectives. The multi-objective optimisation problem is:

$$\operatorname{argmax}_{\mathbf{x} \in \mathbb{X}} \mathbf{f}(\mathbf{x}) \quad (1)$$

where  $\mathbf{f} : \mathbb{X} \rightarrow \mathbb{R}^n$  maps from design space to objective space;  $\mathbb{X} \subset [0, r]^m \subset \mathbb{R}^m$  is the feasible region; and  $\operatorname{argmax}$  is defined in the Pareto sense described below. This is known as a *many-objective optimisation problem* (Ishibuchi et al., 2008) if the number of objectives is sufficiently large to cause difficulties with standard multi-objective optimisation algorithms (as shown in our experiments, as few as 6 objectives can cause difficulties).

Our aim is to find a representation of the Pareto set:

$$\mathbb{X}^* = \{ \mathbf{x}^* \in \mathbb{X} \mid \nexists \mathbf{x} \in \mathbb{X} : \mathbf{x} \succ_{\mathbf{f}} \mathbf{x}^* \}$$

where  $\succ_{\mathbf{f}}$  is the dominance relation as defined in section 2 ( $\mathbf{x}$  strongly dominates  $\mathbf{x}'$ , written  $\mathbf{x} \succ_{\mathbf{f}} \mathbf{x}'$ , if  $f_i(\mathbf{x}) \geq f_i(\mathbf{x}') \forall i$  and  $\mathbf{f}(\mathbf{x}) \neq \mathbf{f}(\mathbf{x}')$ ). This is the set of all Pareto-optimal  $\mathbf{x} \in \mathbb{X}$ , where a vector is Pareto-optimal if it cannot be changed without causing a decrease in at least one objective  $f_i : \mathbb{X} \rightarrow \mathbb{R}$ . The Pareto front is the image of the Pareto set in objective space:

$$\mathbb{Y}^* = \{ \mathbf{y}^* \in \mathbb{Y} \mid \nexists \mathbf{y} \in \mathbb{Y} : \mathbf{y} \succ \mathbf{y}^* \}$$

where  $\mathbb{Y} = \mathbf{f}(\mathbb{X})$  and  $\succ$  is the dominance relation in objective space ( $\mathbf{y}$  strongly dominates  $\mathbf{y}'$ , written  $\mathbf{y} \succ \mathbf{y}'$ , if  $y_i \geq y'_i \forall i$  and  $\mathbf{y} \neq \mathbf{y}'$ ). The solution to (1) is a finite set of Pareto-optimal solutions  $\mathcal{X}^* \subset \mathbb{X}^*$ .

### 3.1 GAUSSIAN PROCESSES

We assume the many-objective case where, for all  $i$ ,  $f_i(\mathbf{x}) \sim \text{GP}(0, k(\mathbf{x}, \mathbf{x}'))$  is a sample from a zero-mean Gaussian process (Rasmussen and Williams, 2006;

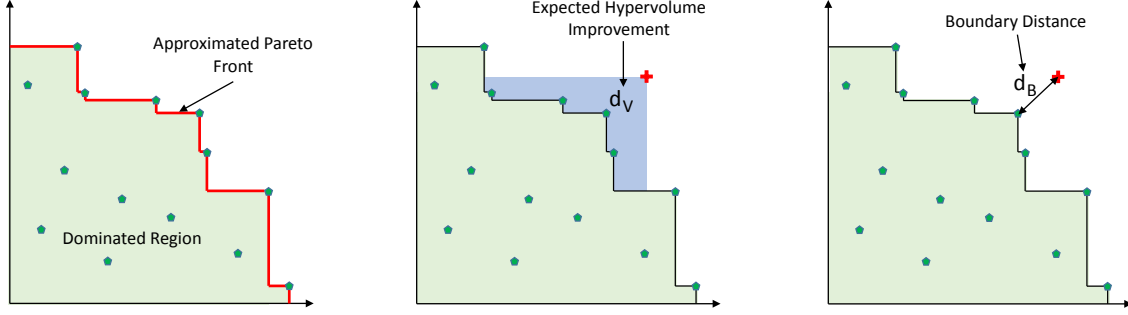


Figure 1: Pareto front (left), expected hypervolume improvement (EHI, middle) and boundary distance (right) for a simple two-objective problem.

MacKay, 1998) (we assume the objectives are non-correlated) that is costly to evaluate. Evaluations of  $\mathbf{f}$  are presumed noisy, so  $y_i = f_i(\mathbf{x}) + \epsilon$ , where  $\epsilon \in \mathcal{N}(0, \sigma^2)$ . Given observations  $\mathcal{D}_t = \{(\mathbf{x}, \mathbf{y}) | \mathbf{y} = \mathbf{f}(\mathbf{x}) + \epsilon\}$  we have  $\mathbf{f}(\mathbf{x}) | \mathcal{D}_t \sim \mathcal{N}(\boldsymbol{\mu}_t(\mathbf{x}), \mathbf{I}\sigma_t^2(\mathbf{x}))$ , where:

$$\begin{aligned} \boldsymbol{\mu}_t(\mathbf{x}) &= \mathbf{Y}_t^T (\mathbf{K}_t + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_t(\mathbf{x}) \\ \sigma_t^2(\mathbf{x}) &= k(\mathbf{x}, \mathbf{x}) - \mathbf{k}_t^T(\mathbf{x}) (\mathbf{K}_t + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_t(\mathbf{x}) \end{aligned} \quad (2)$$

where  $\mathbf{Y}_t = [\mathbf{y}^T]_{(-, \mathbf{y}) \in \mathcal{D}_t}$ ,  $\mathbf{k}_t(\mathbf{x}) = [k(\mathbf{x}, \mathbf{x}')]_{(\mathbf{x}', -) \in \mathcal{D}_t}$  and  $\mathbf{K}_t = [k(\mathbf{x}, \mathbf{x}')]_{(\mathbf{x}, -), (\mathbf{x}', -) \in \mathcal{D}_t}$ .<sup>2</sup> Given  $\mathcal{D}_t$  the estimated Pareto set  $\mathcal{X}_t^*$  and Pareto front  $\mathcal{Y}_t^*$  at iteration  $t$  are the dominant subsets of  $\mathcal{D}_t$ :

$$\begin{aligned} \mathcal{X}_t^* &= \{\mathbf{x}^* \in \mathcal{X}_t | \nexists \mathbf{x} \in \mathcal{X}_t : \mathbf{x} \succ_{\mathbf{f}} \mathbf{x}^*\} \\ \mathcal{Y}_t^* &= \{\mathbf{y}^* \in \mathcal{Y}_t | \nexists \mathbf{y} \in \mathcal{Y}_t : \mathbf{y} \succ \mathbf{y}^*\} \\ \text{where: } \mathcal{X}_t &= \{\mathbf{x} \in \mathbb{X} | (\mathbf{x}, -) \in \mathcal{D}_t\} \\ \mathcal{Y}_t &= \{\mathbf{y} \in \mathbb{Y} | (-, \mathbf{y}) \in \mathcal{D}_t\} \end{aligned} \quad (3)$$

### 3.2 BAYESIAN OPTIMISATION

Bayesian optimisation (Brochu et al., 2010) is an optimisation method designed for problems where the function being optimised is expensive to evaluate in terms of time or monetary cost. A typical Bayesian optimisation algorithm is presented in algorithm 1. For each iteration  $t$  we maximise a (cheap) acquisition function  $a_t : \mathbb{X} \rightarrow \mathbb{R}$  based on  $\boldsymbol{\mu}_{t-1}$  and  $\sigma_{t-1}$ , and the resulting recommendation is evaluated to obtain  $y_t = f(\mathbf{x}_t) + \epsilon$ . GP models are updated, and the algorithm continues. Standard acquisition functions include expected improvement (EI) (Mockus et al., 1978), probability of improvement (PI) (Kushner, 1964), and GP upper confidence bound (GP-UCB) (Jones et al., 1998; Srinivas et al., 2012; Brochu et al., 2010).

<sup>2</sup>We write  $(\mathbf{x}, -) \in \mathcal{D}_t$  if  $\exists \mathbf{y} \in \mathbb{Y} : (\mathbf{x}, \mathbf{y}) \in \mathcal{D}_t$ ; and likewise  $(-, \mathbf{y}) \in \mathcal{D}_t$  if  $\exists \mathbf{x} \in \mathbb{X} : (\mathbf{x}, \mathbf{y}) \in \mathcal{D}_t$ .

---

#### Algorithm 1 Generic Bayesian Optimisation

---

**input**  $\mathcal{D}_0 := \{(\mathbf{x}_i, y_i) | y_i = f(\mathbf{x}_i) + \epsilon, i = 1, 2, \dots\}$ .  
**for**  $t = 1, 2, \dots, T$  **do**  
  Select test point  $\mathbf{x}_t = \operatorname{argmax}_{\mathbf{x}} a_t(\mathbf{x})$ .  
  Perform Experiment  $y_t = f(\mathbf{x}_t) + \epsilon$ .  
  Update  $\mathcal{D}_t := \mathcal{D}_{t-1} \cup \{(\mathbf{x}_t, y_t)\}$ .  
**end for**

---

### 3.3 MULTI-OBJECTIVE BAYESIAN OPTIMISATION

Adding an observation  $\mathbf{y}_t$  to  $\mathcal{Y}_{t-1}$  will either cause no change to the estimated Pareto front  $\mathcal{Y}_{t-1}^*$  (if  $\exists \mathbf{y} \in \mathcal{Y}_{t-1}^* : \mathbf{y} \succ \mathbf{y}_t$ ) or push it closer to the actual Pareto front  $\mathbb{Y}^*$  (if  $\nexists \mathbf{y} \in \mathcal{Y}_{t-1}^* : \mathbf{y} \succ \mathbf{y}_t$ ). The acquisition function  $a_t(\mathbf{x})$  is designed to measure this expected change. Two popular measures used, as shown in figure 1, are:

- Expected hypervolume improvement (EHI):

$$a_t(\mathbf{x}) = \mathbb{E} [S(\mathcal{Y}_{t-1}^* \cup \{\mathbf{f}(\mathbf{x})\}) - S(\mathcal{Y}_{t-1}^*)]$$

(Shir et al., 2007; Zaefferer et al., 2013; Shimoyama et al., 2013). This is the expected change in hypervolume dominated by  $\mathcal{Y}_{t-1}^*$ , where  $S(\mathcal{Y})$  is the hypervolume dominated by  $\mathcal{Y}$  (Zitzler, 1999; Huband et al., 2003; Purshouse, 2003; Laumanns et al., 2000; Fleischer, 2000).

- Boundary distance:

$$a_t(\mathbf{x}) = \mathbb{E} (d(\mathcal{Y}_{t-1}^*, \mathbf{f}(\mathbf{x})))$$

(Yun et al., 2004; Keane, 2006). This is the expected (signed) distance between the the estimated Pareto front and  $\mathbf{f}(\mathbf{x})$ .

It has been noted that calculating the EHI is non-trivial (Wagner et al., 2010; Zaefferer et al., 2013) and, while heavily optimised algorithms are available for up to 3

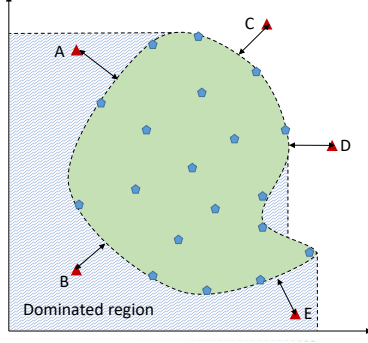


Figure 2: Distance maximisation of 1-class SVM (Yun et al., 2004). The set of observations (blue) are used to train a 1-class SVM, giving the boundary of the green region. Points A-E all give a positive boundary distance, but only points C and D dominate the observations as the boundary does not satisfy consistency requirements.

objectives (Hupkens et al., 2015), the computational cost in the many-objective case remains prohibitive, making EHI unsuitable in a many-objectives context. Similarly, calculating the precise distance to the Pareto front is often computationally intractable, particularly in the many-objective case. Hence when calculating the boundary distance the estimated Pareto front is usually approximated (smoothed) using a score function such as the 1-class SVM (Yun et al., 2004), and the signed distance to this front used. We note that the Pareto front approximation used by (Yun et al., 2004) is in fact a hypersurface surrounding the set of observations and may contain pairs of points where one dominates the other (i.e. it does not satisfy the consistency requirements discussed in section 4). Thus, as shown in figure 2, maximising this measure will not necessarily maximise change to the Pareto front as points may be selected that are not in advance of (dominating) the set of observations.

## 4 PROPOSED METHOD

In the present paper we will be using an acquisition function based on GP-UCB (Srinivas et al., 2012) that we call AD-GP-UCB (approximated distance GP-UCB):

$$a_t(\mathbf{x}) = g_t(\boldsymbol{\mu}_{t-1}(\mathbf{x})) + \sqrt{\beta_t} \eta_t(\boldsymbol{\mu}_{t-1}(\mathbf{x}), \sigma_{t-1}(\mathbf{x})) \quad (4)$$

In this expression  $g_t(\boldsymbol{\mu}_{t-1}(\mathbf{x}))$  is the approximate mean distance of  $\mathbf{f}(\mathbf{x})$  from the estimated Pareto set and  $\eta_t(\boldsymbol{\mu}_{t-1}(\mathbf{x}), \sigma_{t-1}(\mathbf{x}))$  the approximate variance. The constants  $\beta_t$  control the trade-off between exploitation (selecting recommendations with high predicted distance from the estimated Pareto set) and exploration (exploring unexplored regions of the feasible set  $\mathbb{X}$ ) as per the GP-

UCB method (Srinivas et al., 2012):

$$\beta_t = \begin{cases} 2 \log\left(\frac{\pi^2 t^2}{6\delta} |\mathbb{X}|\right) & \text{if } |\mathbb{X}| < \infty \\ 2 \log\left(\frac{2\pi^2 t^2}{3\delta} \left(t^2 m b r (\log(\frac{2ma}{\delta}))^{\frac{1}{2}}\right)^{2m}\right) & \text{otherwise} \end{cases} \quad (5)$$

where  $0 < \delta \ll 1$  and in the infinite case we assume  $\mathbf{f}$  satisfies  $\Pr\{\sup_{\mathbf{x} \in \mathbb{X}} |\partial f_i / \partial x_j| > L\} \leq a e^{-(L/b)^2} \forall i, L$ . We have chosen GP-UCB here as it is explicitly designed to balance exploration and exploitation; and because, in the single objective case, there exist convergence bounds to show that, with probability  $1 - \delta$ , the optimisation procedure is guaranteed to converge (as measured by cumulative risk) sub-linearly as  $T \rightarrow \infty$ . While our method is not a “true” GP-UCB method ( $g_t$  and  $\eta_t$  are only approximations of the mean and variance of the predicted distance from the estimated Pareto front; and moreover the function approximated by  $g_t$  changes over time) our experimental results demonstrate its efficacy.

### 4.1 APPROXIMATING THE MEAN DISTANCE

The score function  $g_t(\boldsymbol{\mu}_{t-1}(\mathbf{x}))$  is used to approximate the signed distance between the estimated Pareto set  $\mathcal{Y}_{t-1}^*$  and the sample evaluation  $\mathbf{f}(\mathbf{x})$  for a given  $\mathbf{x} \in \mathbb{X}$ . We use the GP posterior mean  $\boldsymbol{\mu}_{t-1}(\mathbf{x})$  to estimate the mean of  $\mathbf{f}(\mathbf{x})$  and  $g_t$  to approximate the distance of this from the Pareto front  $\mathcal{Y}_{t-1}^*$ . Motivated by the “standard form” of the trained SVM in dual form, the score function  $g_t$  is defined as:

$$g_t(\mathbf{y}) = 1 - 2 \sum_{i=1}^{N_t} \alpha_i^t L(\mathbf{y}_i, \mathbf{y}) \quad (6)$$

where the indices  $i$  applied to all  $\mathbf{y}_i \in \mathcal{Y}_{t-1}$  correspond to the indices  $i$  on  $\alpha_i^t$ ,  $N_t = |\mathcal{Y}_{t-1}|$ ,  $\boldsymbol{\alpha}^t \geq \mathbf{0}$ , and  $L$  is defined to ensure that  $g_t$  satisfies consistency conditions:

1. Observational Consistency:

$$g_t(\mathbf{y}) \leq 0 \quad \forall \mathbf{y} \in \mathcal{Y}_{t-1}^*$$

2. Dominance Consistency:

$$g_t(\mathbf{y}) > g_t(\mathbf{y}') \quad \forall \mathbf{y}, \mathbf{y}' \in \mathbb{Y} : \mathbf{y} \succ \mathbf{y}'$$

Observational consistency is required to ensure that the reported distance is never positive for existing observations that are by definition dominated by the current estimated Pareto set  $\mathcal{Y}_{t-1}^*$ . Dominance consistency ensures that,  $\forall \mathbf{y}, \mathbf{y}' \in \mathbb{Y}$ , the dominant vector will receive the higher “score”. Thus  $g_t$  is a score function in the sense of (Miranda and Von Zuben, 2015) and may be said to define an estimated Pareto set:

$$\mathbb{Y}^{g_t^*} = \{\mathbf{y} \in \mathbb{R}^m \mid g(\mathbf{y}) = 0\} \quad (7)$$



that dominates all points in  $\mathcal{Y}_{t-1}$  (that is,  $\forall \mathbf{y} \in \mathcal{Y}_{t-1} \exists \mathbf{y}' \in \mathbb{Y}^{g_t^*} : \mathbf{y}' \succeq \mathbf{y}$ ). The distance reported by  $g_t$  is the signed distance from  $\mathbb{Y}^{g_t^*}$  as measured by some metric. Motivated by this we let  $L(\mathbf{y}, \mathbf{y}') = \kappa(\min_q (y_q - y'_q))$ , where:

$$\begin{aligned} \kappa(0) &= \frac{1}{2} && \text{(centred)} \\ \kappa(y + \delta) &> \kappa(y) \quad \forall y \in \mathbb{R}, \delta \in \mathbb{R}^+ && \text{(increasing)} \end{aligned} \quad (8)$$

Many standard neural activation functions are suitable choices (e.g. the logistic function  $\kappa(y) = 1/(1 + \exp(-vy))$ ). It is straightforward to see that  $g_t$  defined by (6) satisfies dominance consistency if  $\alpha^t \neq \mathbf{0}$ . To satisfy observational consistency  $\alpha^t$  is selected to solve the linear programming problem:

$$\begin{aligned} \min_{\alpha} \quad & \|\alpha^t\|_1 = \mathbf{1}^T \alpha^t \\ \text{such that:} \quad & \sum_{i=1}^{N_t} \alpha_i^t L(\mathbf{y}_i, \mathbf{y}_j) \geq \frac{1}{2} \quad \forall 1 \leq j \leq N_t \\ & \alpha^t \geq \mathbf{0} \end{aligned} \quad (9)$$

which will be referred to this as the score-function optimisation problem. It may be noted that the score-function optimisation problem, and the form of the score function  $g_t$ , are closely related to the 1-norm SVM (Bradley and Mangasarian, 1998; Zhu et al., 2004), which is a variant of the standard SVM that retains the standard (dual) form of the trained machine but minimises  $\|\alpha\|_1$  rather than  $\|\alpha\|_{\mathcal{H}_L}$ . This form has two distinct advantages: the kernel<sup>3</sup>  $L$  may be any function (not just positive definite) and the solution tends to be more sparse than the standard form. Our approach also borrows from the 1-class SVM (Schölkopf et al., 1999), but rather than using a *bias-forcing* term to achieve margin minimalisation (rather than maximising the margin of separation, the 1-class SVM seeks to minimise the margin) we instead use a fixed bias ( $b = -1$ ) and restrict  $L$  using (8) so that the margin minimalisation occurs as a direct result from minimising the regularisation term  $\|\alpha\|_1$ .

## 4.2 APPROXIMATING THE DISTANCE VARIANCE

The function  $\eta_t(\boldsymbol{\mu}_{t-1}(\mathbf{x}), \sigma_{t-1}(\mathbf{x}))$  in the acquisition function (4) approximates the variance in the estimate  $g_t(\boldsymbol{\mu}_{t-1}(\mathbf{x}))$  of the distance between  $\mathbf{f}(\mathbf{x})$  and the estimated Pareto front  $\mathcal{Y}_{t-1}^*$ . We approximate this using a simple first-order Taylor approximation of the second moment about  $\boldsymbol{\mu}_{t-1}(\mathbf{x})$  - that is:

$$\eta_t(\boldsymbol{\mu}_{t-1}(\mathbf{x}), \sigma_{t-1}(\mathbf{x})) = \|\nabla_{\mathbf{x}} g_t(\boldsymbol{\mu}_{t-1}(\mathbf{x}))\| \sigma_{t-1}(\mathbf{x})$$

<sup>3</sup>In general we have tried to avoid using the word kernel to refer to  $L$  to avoid potential confusion with the covariance function (kernel)  $k$  used for Gaussian Processes.

where, defining  $q_i = \operatorname{argmin}_q (y_{i,q} - \mu_{t-1,q}(\mathbf{x})) \forall i$ :

$$\begin{aligned} \frac{\partial}{\partial x_i} g_t(\boldsymbol{\mu}_{t-1}(\mathbf{x})) &= \dots \\ &- 2 \sum_i \alpha_i^t \kappa'(y_{i,q_i} - \mu_{t-1,q_i}(\mathbf{x})) \frac{\partial}{\partial x_i} \mu_{t-1,q_i}(\mathbf{x}) \end{aligned}$$

where  $\kappa'(y) = \partial \kappa(y) / \partial y$  and:

$$\frac{\partial}{\partial x_i} \boldsymbol{\mu}_t(\mathbf{x}) = \mathbf{Y}_t^T (\mathbf{K}_t + \sigma^2 \mathbf{I})^{-1} \frac{\partial}{\partial x_i} \mathbf{k}_t(\mathbf{x})$$

We note that the accuracy of this approximation degrades as the non-linearity of  $g_t(\boldsymbol{\mu}_{t-1}(\bullet))$  increases. This is a necessary trade-off as calculating the actual variance is not feasible. Assuming an squared-exponential kernel  $k(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|_2^2/l)$  for the GP model and a sigmoid function for the score function kernel  $\kappa(y) = 1/(1 + \exp(-vy))$  we see that the variance approximation is best when the length scale  $l$  of  $k$  is large and the constant  $v$  of  $\kappa$  is small.

## 5 THEORETICAL ANALYSIS

It is useful at this point to analyse the theoretical properties of our proposed algorithm. We have already noted that the score function  $g_t$  satisfies both observational and dominance consistency and so provides a sensible approximation of distance from the estimated Pareto front. Applying SVM techniques we find the following properties (all proofs presented in the supplementary material):

**Theorem 1 (Non-triviality)** *Let  $\alpha^t$  be the solution to the score-function optimisation problem (9). Then  $\alpha^t \neq \mathbf{0}$ .*

**Theorem 2 (Margin Minimisation)** *Let  $\alpha^t$  be the solution to the score-function optimisation problem (9). Let  $\mathbb{Y}^{g_t^*}$  be the estimated Pareto front defined by  $g_t$ . The minimum distance between  $\mathcal{Y}_{t-1}$  and the estimated Pareto front  $\mathbb{Y}^{g_t^*}$  is zero:*

$$\min_{\mathbf{y} \in \mathcal{Y}_{t-1}, \mathbf{y}' \in \mathbb{Y}^{g_t^*}} \|\mathbf{y} - \mathbf{y}'\| = 0$$

**Theorem 3 (Sparsity)** *Let  $\alpha^t$  be the solution to the score-function optimisation problem (9). Then  $\alpha_i^t = 0 \forall i : \mathbf{y}_i \notin \mathcal{Y}_{t-1}^*$  (ie. points not in the estimated Pareto front cannot be support vectors).*

**Theorem 4 (Heaviside Limit)** *Let  $\kappa = \kappa_{\perp}$ , where*

$$\kappa_{\perp}(y) = \lim_{v \rightarrow \infty} \frac{1}{1 + \exp(-vy)} = \frac{1}{2} (1 + \operatorname{sgn}(y)),$$

*and  $\nexists i \neq j : \mathbf{y}_i = \mathbf{y}_j$ . Then  $\alpha_i^t = 1 \forall i : \mathbf{y}_i \in \mathcal{Y}_{t-1}^*$ ,  $\alpha_i^t = 0$  otherwise (ie. in the limiting case the support vectors are precisely the estimated Pareto set).*

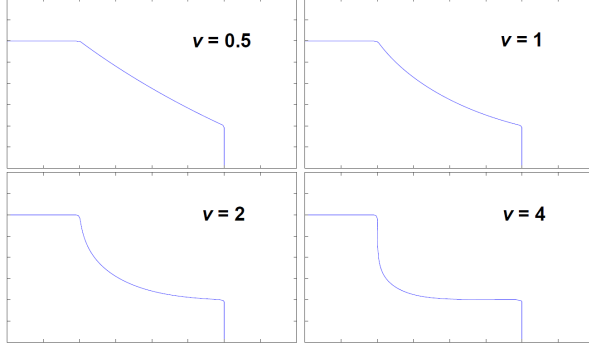


Figure 3: Estimated front  $\mathbb{Y}^{g_t^*}$  (objective space) given  $\mathbf{y}_1 = [-1; 1]$ ,  $\mathbf{y}_2 = [1; -1]$ , where  $\kappa(y) = \frac{1}{1+\exp(-vy)}$  and  $v = 0.5, 1, 2, 4$ , respectively.

It follows that solving (9) will define a score function, and hence an estimated Pareto front  $\mathbb{Y}^{g_t^*}$ , that is as tight as possible (insofar as it lies as close to  $\mathcal{Y}_t$  as possible while maintaining observational consistency) and sparsely represented. As shown in figure 3 the parameter  $v$  in the  $\kappa$  function acts as a smoothing parameter on the estimated Pareto front, where smaller  $v$  will tend to favour smoother fronts while larger  $v$  will attempt to achieve a tighter “fit” to the observations  $\mathcal{Y}_t$ . In the limiting case  $v \rightarrow \infty$  the Pareto front becomes stepwise, as shown by theorem 4.

## 6 EXPERIMENTS

We consider three experiments in this section: standard test function optimisation, hyper-parameter selection in multi-class SVM classification in the absence of relative class weighting, and high-temperature alloy design. All SVM and related code was written in C++ with linking to ThermoCalc via a Matlab interface. Where relevant EHI estimation was performed using the IRS algorithm (Hupkens et al., 2015). Global optimisation on our acquisition function was carried out using the DIRECT algorithm (Jones et al., 1993). The objective function  $\mathbf{f}$  was modelled using a GP with a squared-exponential kernel. For the score function we use  $\kappa(y) = \frac{1}{1+\exp(-vy)}$ .

### 6.1 STANDARD TEST FUNCTION

In our first experiment we have evaluated the performance of AD-GP-UCB on the standard DTLZ2 multi-objective test function (Deb et al., 2005). We have run simulations for Bayesian optimisation using both EHI and AD-GP-UCB acquisition functions over a budget of  $T = 200$  iterations for  $n = 2, 3, \dots, 10$  objectives.

We have evaluated performance on four criteria:

1. How close the elements of the estimated Pareto front  $\mathcal{Y}_t^*$  are to the actual Pareto front  $\mathbb{Y}^*$  (how optimal the elements of the  $\mathcal{Y}_t^*$  are):

$$d_{\mathbb{Y}^*} = \sup_{\mathbf{y} \in \mathcal{Y}_t^*} d(\mathbf{y}, \mathbb{Y}^*) = \sup_{\mathbf{y} \in \mathcal{Y}_t^*} \left( \inf_{\mathbf{y}' \in \mathbb{Y}^*} \|\mathbf{y} - \mathbf{y}'\| \right)$$

2. The maximum distance between any point on the actual Pareto front  $\mathbb{Y}^*$  and the closest point to it in the estimated Pareto front (how well  $\mathcal{Y}_t^*$  approximates  $\mathbb{Y}^*$ ):

$$d_{\mathcal{Y}_T^*} = \sup_{\mathbf{y}' \in \mathbb{Y}^*} d(\mathcal{Y}_t^*, \mathbf{y}') = \sup_{\mathbf{y}' \in \mathbb{Y}^*} \left( \inf_{\mathbf{y} \in \mathcal{Y}_t^*} \|\mathbf{y} - \mathbf{y}'\| \right)$$

3. The Hausdorff distance between the estimated Pareto front  $\mathcal{Y}_t^*$  and the actual Pareto front  $\mathbb{Y}^*$ :

$$d_{\mathcal{H}} = d(\mathcal{Y}_T^*, \mathbb{Y}^*) = \max(d_{\mathbb{Y}^*}, d_{\mathcal{Y}_T^*})$$

4. Simulation time per recommendation produced.

Results are summarised in table 1. It may be seen from this that in terms of the Hausdorff distance between estimated and actual Pareto fronts AD-GP-UCB consistently outperformed EHI in this experiment. Moreover although the EHI estimated Pareto front was closer to the actual Pareto front (measure 1) the AD-GP-UCB estimated Pareto front better approximated the actual Pareto front in terms of coverage (measure 2). We note that the hypervolume dominated by the AD-GP-UCB estimated Pareto front was consistently higher than the hypervolume dominated by the EHI estimated Pareto front. Figure 5 shows the average time required per recommendation for each of the algorithms. We have chosen this measure to factor out extraneous fluctuations observed in the estimated Pareto set size generated by the EHI method as  $n$  varied.

To aid visualisation the estimated Pareto fronts for AD-GP-UCB and EHI in the case  $n = 3$  are shown in figure 4, where the Pareto front for DTLZ2 consists of a first-quadrant unit sphere in objective space (Deb et al., 2005). From this figure it may be seen that EHI constructs a cluster of recommendations that are close to a fragment of the actual Pareto front; whereas AD-GP-UCB creates a more diverse coverage of the Pareto front. We postulate that this results from the fact that AD-GP-UCB explicitly incorporates an exploration term  $\sqrt{\beta_t} \eta_t$  in the acquisition function, encouraging greater exploration and hence more diversity in  $\mathcal{Y}_T^*$ .

As noted previously, and as may be seen from table 1, the size of the estimated Pareto front found by EHI was surprisingly small for larger  $n$ . It is unclear why this occurs; however it appears to contribute to the significant fluctuation in the total time  $\tau$  for required EHI to complete  $T = 200$  iterations.

$n$	EHI-based Bayesian Optimisation							AD-GP-UCB						
	$N^*$	$d_{\mathbb{Y}^*} \downarrow$	$d_{\mathcal{Y}_T^*} \downarrow$	$d_{\mathcal{H}} \downarrow$	HV $\uparrow$	$\tau \downarrow$	$\frac{\tau}{N^*} \downarrow$	$N^*$	$d_{\mathbb{Y}^*} \downarrow$	$d_{\mathcal{Y}_T^*} \downarrow$	$d_{\mathcal{H}} \downarrow$	HV $\uparrow$	$\tau \downarrow$	$\frac{\tau}{N^*} \downarrow$
2	124	<b>0.007</b>	0.48	0.48	2.52	<b>162</b>	<b>1.31</b>	66	0.25	<b>0.13</b>	<b>0.25</b>	<b>3.18</b>	815	12.35
3	185	<b>0.027</b>	0.66	0.66	5.67	<b>174</b>	<b>0.94</b>	107	0.25	<b>0.25</b>	<b>0.25</b>	<b>7.33</b>	824	7.70
4	198	<b>0.25</b>	0.90	0.90	11.1	<b>358</b>	<b>1.81</b>	117	<b>0.25</b>	<b>0.35</b>	<b>0.35</b>	<b>15.5</b>	986	8.43
5	14	<b>0.25</b>	1.26	1.26	25.2	<b>243</b>	17.4	121	<b>0.25</b>	<b>0.49</b>	<b>0.49</b>	<b>31.5</b>	1012	<b>8.36</b>
6	187	<b>0.25</b>	1.16	1.16	51.5	3950	21.1	126	<b>0.25</b>	<b>0.71</b>	<b>0.71</b>	<b>63.3</b>	<b>992</b>	<b>7.87</b>
7	167	<b>0.25</b>	1.32	1.32	90.3	27546	165	141	<b>0.25</b>	<b>0.92</b>	<b>0.92</b>	<b>127</b>	<b>1024</b>	<b>7.26</b>
8	60	<b>0.24</b>	1.39	1.39	207	1483	24.7	180	0.25	<b>0.99</b>	<b>0.99</b>	<b>253</b>	<b>1227</b>	<b>6.82</b>
9	46	<b>0.20</b>	1.38	1.38	378	2025	44.0	174	0.24	<b>1.15</b>	<b>1.15</b>	<b>499</b>	<b>1444</b>	<b>8.30</b>
10	32	<b>0.20</b>	1.41	1.41	818	<b>973</b>	30.4	200	0.23	<b>1.13</b>	<b>1.13</b>	<b>978</b>	2795	<b>13.98</b>

Table 1: Results summary for DTLZ2 optimisation over range of  $n$ . In this table  $d_{\mathbb{Y}^*}$  measures how close the estimated Pareto front is to the actual Pareto front (optimality);  $d_{\mathcal{Y}_T^*}$  measures the maximum distance from any point on the actual Pareto front to any point in the estimated Pareto front (coverage) (for calculation  $\mathbb{Y}^*$  is approximated as a projected grid);  $d_{\mathcal{H}}$  is the Hausdorff distance between the estimated and actual Pareto fronts; and HV is the dominated hypervolume.  $T = 200$  iterations were used, producing an estimated Pareto front  $\mathcal{Y}_T^*$  containing  $N^* = |\mathcal{Y}_T^*|$  recommendations in  $\tau$  seconds - ie.  $\tau/N^*$  recommendations per second.  $\uparrow$  indicates that larger values are preferable and  $\downarrow$  that smaller values are preferable.

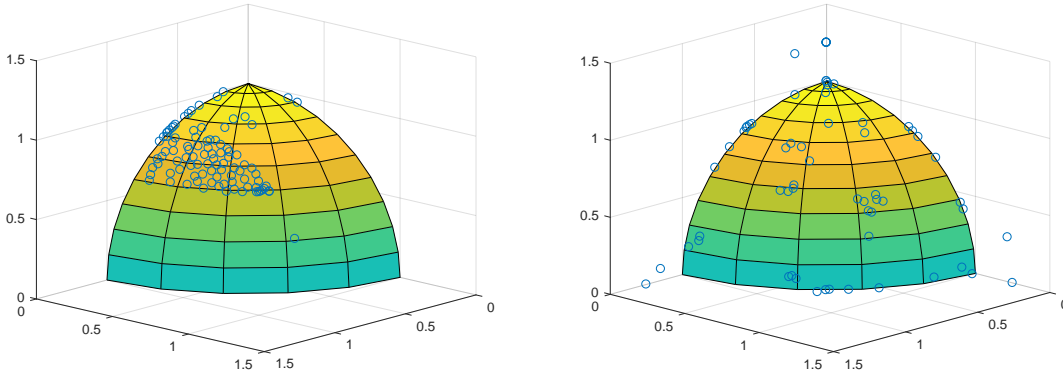


Figure 4: Estimated Pareto fronts for EHI (left) and AD-GP-UCB (right) for  $n = 3$  objectives. Note that DTLZ2 is a minimisation problem, so the BO maximises its negative.

## 6.2 HYPERPARAMETER SELECTION

In this experiment we have compared our algorithm to Bayesian multi-objective optimisation with an expected hypervolume improvement (EHI) based acquisition function. We consider hyper-parameter selection for multi-class classifiers in the absence of information about the relative importance of classes. As there is no objective way to compare (weight) the cost of misclassification for the different classes this is an example of a multi-objective optimisation, where the classification accuracy with respect to each class is a single objective.

For multi-class classification we used the CS-SVM algorithm (Shilton et al., 2012) in SVMHeavy (Shilton, 2001)

with an RBF kernel with length-scale  $g$ . Performance on each class was measured using 10-fold cross-validation. The hyper-parameters being tuned were the CS-SVM trade-off parameter  $C \in [0.1, 10]$  and the kernel parameter  $g \in [0.1, 10]$ . Three datasets from the UCI collection (Dheeru and Karra Taniskidou, 2017) were used: SAT (6 classes,  $N = 4435$  vectors), SEG (7 classes,  $N = 2310$ ), and WAV (3 classes,  $N = 5000$ ).

Results of simulations are shown in figure 6. These figures show both hypervolume as a function of iteration number (the hypervolume is used as a measure of the optimality of the Pareto set) and also the time required to recommend the next sample at each iteration. As may be seen from the graphs our proposed method is significantly faster than EHI. In fact, in the higher-dimensional

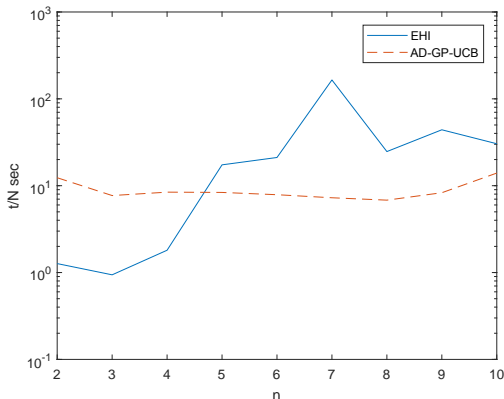


Figure 5: Average time taken to produce Pareto front in seconds/recommendation ( $\tau/N^*$ ).

cases - namely SAT (6 classes/objectives) and SEG (7 classes/objectives) - the EHI simulations had to be terminated early due to excessive computation time when computing the next recommendation (up to 1 day to produce a single recommendation). With regard to optimality (as measured by dominated hypervolume) it is somewhat difficult to say with certainty, but based on the WAV dataset at least our algorithm is certainly competitive (particularly given that the EHI alternative was unable to finish for either the SAT or SEG datasets due to excessive computational load resulting from EHI calculations made inside the global optimisation DIRECT call).

### 6.3 ALLOY DESIGN

High-temperature creep resistant Ni-superalloy is used for making boilers of super-critical thermal power plants. In a joint project with metallurgist we were asked to optimize the current alloy recipe to obtain superior creep resistance to the industry standard. The alloy consist of Ni, Cr, Co, Al, Ti, Mo, Ta, W and V. We use ThermoCalc software for phase simulation i.e. to predict what compounds (phases) get formed at a given temperature. Based on the existing knowledge, phases were clubbed into either good or bad for creep resistance. The phase simulation is performed at 6 different temperatures and a total of 12 objectives are created. Recommendation times for EHI were found to be excessive ( $\sim 1$  day), whereas our method was able to complete the task without difficulty.

Results for our simulation are shown in figure 7. In these figures dominated hypervolume has been used as a measure of convergence (Zitzler, 1999; Huband et al., 2003; Purshouse, 2003; Laumanns et al., 2000; Fleischer, 2000). The GP length scale in these results is 20 and was selected experimentally to optimise the rate of conver-

gence; and the time budget  $T = 100$  was chosen for practical reasons. A total of 21 Pareto-optimal alloys were found by our simulation. As may be seen our algorithm was able to calculate a set of Pareto-optimal recommendations within a reasonable time-frame despite the high number of objectives to provide the experimentalist with a good selection of options for further investigation.

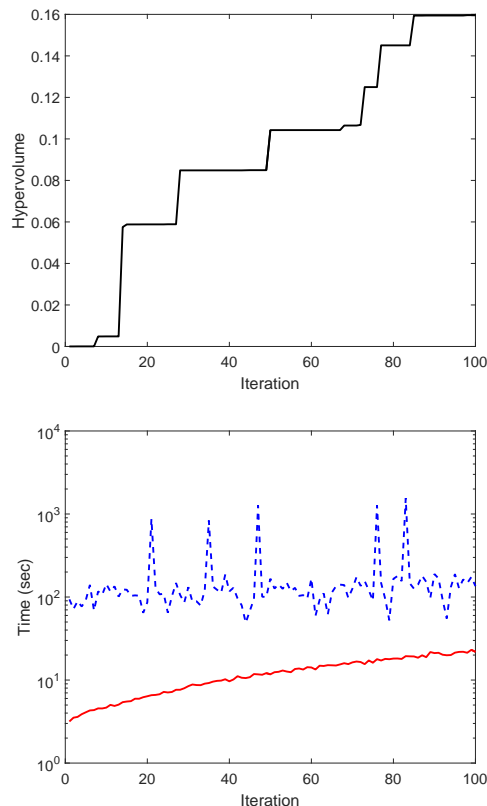


Figure 7: Simulation results for alloy design. Top: dominated hypervolume. Bottom: recommendation time (solid red), ThermoCalc simulation time (dashed blue).

## 7 CONCLUSIONS

In this paper we have proposed a method for Bayesian multi-objective optimisation based on score functions. Our proposed method is particularly well suited to the many-objective case where the number of objectives is significant and renders alternative methods such as EHI unsuitable due to reasons of computational infeasibility (for example, on 2 of our datasets we found that the EHI method failed early as the time required for a single recommendation grew to over 1 day). We have analysed the theoretical properties of our method and shown that it possesses properties such as sparseness, inherited from the 1-norm SVM, that make it well suited to the task. For

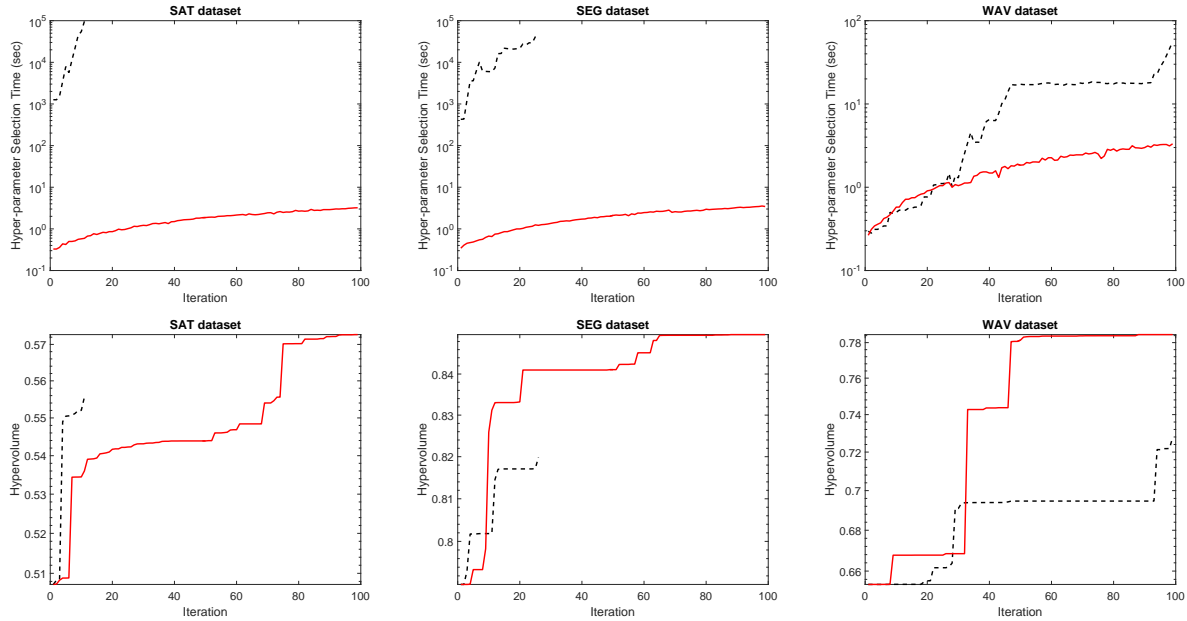


Figure 6: Hyper-parameter tuning results. Top row: recommendation times per iteration. Bottom row: enclosed hypervolume. EHI shown as black (dashed) line, our method as red (solid) line. EHI Simulations for SAT/SEG datasets had to be terminated early due to excessive recommendation times (we estimate for example that running the SAT dataset simulation to completion using the EHI method would have taken at least 3 months, which is clearly impractical).

experimental validation we have applied our proposed method to high-temperature alloy design and hyperparameter selection in the multi-class case where no information is provided with regard to the relative weight (or importance) of the classes. Our results clearly showed that our method is able to continue in cases where EHI breaks down due to computational complexity, and moreover that the results achieved by our method are competitive with those achieved by EHI.

## ACKNOWLEDGEMENTS

This research was partially funded by the Australian Government through the Australian Research Council (ARC) and the Telstra-Deakin Centre of Excellence in Big Data and Machine Learning. Prof Venkatesh is the recipient of an ARC Australian Laureate Fellowship (FL170100006).

## References

Paul S. Bradley and Olvi L. Mangasarian. Feature selection via concave minimization and support vector machines. In J. Shavlik, editor, *Proceedings of the 15th International Conference on Machine Learning (ICML'98)*, pages 82–90, San Fansisco, 1998. Kaufmann.

Eric Brochu, Vlad M. Cora, and Nando de Freitas. A tutorial on bayesian optimization of expensive cost functions, with applications to active user modeling and heirarchical reinforcement learning. eprint arXiv:1012.2599, arXiv.org, December 2010.

Coello A. Coello Coello, David A. Van Veldhuizen, and Gary B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishing, New York, 2002.

Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, UK, 2001.

Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. Scalable multi-objective optimization test problems. *Evolutionary Multiobjective Optimization. Theoretical Advances and Applications*, pages 105–145, 2005.

Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository, 2017. URL <http://archive.ics.uci.edu/ml>.

Michael Emmerich and Jan-Willem Klinkenberg. The computation of the expected improvement in dominated hypervolume of pareto front approximations. Technical report, Leiden University, 2008.

Mark Fleischer. The measure of pareto optima: Applications to multi-objective metaheuristics. In *Inter-*

- national Conference on Evolutionary Multi-Criterion Optimization*, pages 519–533, 2000.
- Simon Huband, Phil Hingston, Lyndon While, and Luigi Barone. An evolution strategy with probabilistic mutation for multi-objective optimisation. In *Proceedings of the IEEE Congress on Evolutionary Computation*, volume 4, pages 2284–2291, 2003.
- Iris Hupkens, André Deutz, Kaifeng Yang, and Michael Emmerich. Faster exact algorithms for computing expected hypervolume improvement. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 65–79. Springer, 2015.
- Hisao Ishibuchi, Noritaka Tsukamoto, and Yusuke Nojima. Evolutionary many-objective optimization: A short review. In *Proceedings of the 2008 IEEE Congress on Evolutionary Computation*, pages 2424–2431, June 2008.
- Donald R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, 1993. ISSN 1573-2878. doi: 10.1007/BF00941892. URL <http://dx.doi.org/10.1007/BF00941892>.
- Donald R. Jones, Matthias Schonlau, and William J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.
- Andy J. Keane. Statistical improvement criteria for use in multiobjective design optimization. *Journal of the American Institute of Aeronautics and Astronautics AIAA*, 44(4):879–891, 2006.
- Harold J. Kushner. A new method of locating the maximum point of an arbitrary multiplex curve in the presence of noise. *Journal of Basic Engineering*, 86(1):97–106, 1964.
- Marco Laumanns, Eckart Zitzler, and Lothar Thiele. A unified model for multi-objective evolutionary algorithms with elitism. In *Proceedings of the 2000 Congress on Evolutionary Computation*, volume 1, pages 46–53, 2000.
- Cheng Li, David Rubin de Celis, Santu Rana, Sunil Gupta, Alessandra Sutti, Stewart Greenhill, Teo Slezak, Murray Height, and Svetha Venkatesh. Rapid bayesian optimisation for synthesis of short polymer fiber materials. *Nature Scientific Reports*, 7, 2017.
- David J. C. MacKay. Introduction to gaussian processes. *NATO ASI Series F Computer and Systems Sciences*, 168, 1998.
- Kaisa Miettinen. *Multi-Objective Optimization Using Evolutionary Algorithms*. Springer US, 1999.
- Conrado S. Miranda and Fernando J. Von Zuben. Necessary and sufficient conditions for surrogate functions of pareto frontiers and their synthesis using gaussian processes. *IEEE Transactions on Evolutionary Computation*, PP(99):1–1, May 2015.
- Jonas Mockus, Vytautas Tiesis, and Antanas Zilinskas. The application of bayesian methods for seeking the extremum. In *Towards Global Optimization*, volume 2, pages 117–129. September 1978. ISBN 0-444-85171-2.
- Wolfgang Ponweiser, Tobias Wagner, Dirk Biermann, and Markus Vincze. Multiobjective optimization on a limited budget of evaluations using model-assisted S-Metric selection. In *Proceedings of the 10th International Conference on Parallel Problem Solving from Nature: PPSN X*, pages 784–794, Berlin, Heidelberg, 2008. Springer-Verlag.
- Robin Charles Purshouse. *On the Evolutionary Optimization of Many Objectives*. PhD thesis, University of Sheffield, 2003.
- Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.
- Bernhard Schölkopf, John C. Platt, John C. Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. Technical Report MSR-TR-99-87, Microsoft Research, Redmond, 1999.
- Alistair Shilton. SVMHeavy: a support vector machine optimiser, 2001.
- Alistair Shilton, Daniel T. H. Lai, and Marimuthu Palaniswami. The conic-segmentation support vector machine - a target space method for multiclass classification. In *Neural Networks (IJCNN), The 2012 International Joint Conference on*, pages 1–8, June 2012.
- Koji Shimoyama, Shinkyu Jeong, and Shigeru Obayashi. Kriging-surrogate-based optimization considering expected hypervolume improvement in non-constrained many-objective test problems. In *Proceedings of 2013 IEEE Congress on Evolutionary Computation*, 2013.
- Ofer M. Shir, Michael Emmerich, Thomas Back, and Marc J. J. Vrakking. The application of evolutionary multi-criteria optimization to dynamic molecular alignment. In *Proceedings of 2007 IEEE Congress on Evolutionary Computation*, 2007.
- Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias W. Seeger. Information-theoretic regret bounds for gaussian process optimization in the bandit setting. *IEEE Transactions on Information Theory*, 58(5):3250–3265, May 2012.

- Tobias Wagner, Michael Emmerich, André Deutz, and Wolfgang Ponweiser. On expected-improvement criteria for model-based multi-objective optimization. In *Proceedings of the 2010 International Conference on Parallel Problem Solving from Nature*, pages 718–727, 2010.
- Yeboon Yun, Hirotaka Nakayama, and Masao Arakava. Generation of pareto frontiers using support vector machines. In *International Convergence on Multiple Criteria Decision Making*, 2004.
- Martin Zaeferrer, Thomax Bartz-Beielstein, Boris Naujoks, Tobias Wagner, and Michael Emmerich. A case study on multi-criteria optimization of an event detection software under limited budgets. In *Proceedings of the 2013 International Conference on Evolutionary Multi-Criterion Optimization*, pages 756–770. Springer, 2013.
- Ji Zhu, Saharon Rosset, Trevor Hastie, and Rob Tibshirani. 1-norm support vector machines. In *The Annual Conference on Neural Information Processing Systems* 16, 2004.
- Eckart Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology Zurich, 1999.

---

# Stochastic Learning for Sparse Discrete Markov Random Fields with Controlled Gradient Approximation Error

---

Sinong Geng \*  
UW-Madison

Zhaobin Kuang \*  
UW-Madison

Jie Liu  
University of Washington

Stephen Wright  
UW-Madison

David Page  
UW-Madison

## Abstract

We study the  $L_1$ -regularized maximum likelihood estimator/estimation (MLE) problem for discrete Markov random fields (MRFs), where efficient and scalable learning requires both sparse regularization and approximate inference. To address these challenges, we consider a stochastic learning framework called stochastic proximal gradient (SPG; Honorio 2012a, Atchade et al. 2014, Miasojedow and Rejchel 2016). SPG is an *inexact* proximal gradient algorithm [Schmidt et al., 2011], whose inexactness stems from the stochastic oracle (Gibbs sampling) for gradient approximation – exact gradient evaluation is infeasible in general due to the NP-hard inference problem for discrete MRFs [Koller and Friedman, 2009]. Theoretically, we provide novel *verifiable* bounds to inspect and control the quality of gradient approximation. Empirically, we propose the *tighten asymptotically* (TAY) learning strategy based on the verifiable bounds to boost the performance of SPG.

## 1 INTRODUCTION

Markov random fields (MRFs, a.k.a. Markov networks, undirected graphical models) are a compact representation of the joint distribution among multiple variables, with each variable being a node and an edge between two nodes indicating conditional dependence between the two corresponding variables. Sparse discrete MRF learning is proposed in the seminal work of

Lee et al. [2006]. By considering an  $L_1$ -regularized MLE problem, many components of the parameterization are driven to zero, yielding a sparse solution to structure learning. However, in general, solving an  $L_1$ -regularized MLE problem exactly for a discrete MRF is infamously difficult due to the NP-hard inference problem posed by exact gradient evaluation [Koller and Friedman, 2009]. We hence inevitably have to compromise accuracy for the gain of efficiency and scalability via *inexact* learning techniques [Liu and Page, 2013, Liu et al., 2014b, 2016, Geng et al., 2018].

In this paper, we consider stochastic proximal gradient (SPG; Honorio 2012a, Atchade et al. 2014, Miasojedow and Rejchel 2016), a stochastic learning framework for  $L_1$ -regularized discrete MRFs. SPG hinges on a stochastic oracle for gradient approximation of the log-likelihood function (inexact inference). However, both the theoretical guarantees and the practical performances of existing algorithms are unsatisfactory.

The stochastic oracle behind SPG is Gibbs sampling [Levin et al., 2009], which is an effective approach to draw samples from an intractable probability distribution. With enough samples, the intractable distribution can be approximated effectively by the empirical distribution, and hence many quantities (e.g., the gradient of the log-likelihood function) related to the intractable distribution can be estimated efficiently. Since SPG uses Gibbs sampling for gradient approximation, it can be viewed as an inexact proximal gradient method [Schmidt et al., 2011], whose success depends on whether the gradient approximation error can be effectively controlled. While previous works [Honorio, 2012a, Atchade et al., 2014, Miasojedow and Rejchel, 2016] have shown that the quality of the gradient approximation can be improved *in the long run* with increasingly demanding computational resources, such long term guarantees might not translate to satisfactory performance in practice (see Section 7). Therefore, it is desirable to estimate and control the

---

\* Sinong Geng and Zhaobin Kuang contribute equally. Their names are listed in alphabetical order. Corresponds to: sgeng2@wisc.edu.



gradient approximation error of SPG meticulously in each iteration so that a more refined approximation to the exact gradient will be rewarded with a higher gain of efficiency and accuracy in practice.

Careful analysis and control of the quality of the gradient approximation of SPG call for the cross-fertilization of theoretical and empirical insights from stochastic approximate inference [Bengio and Delalleau, 2009, Fischer and Igel, 2011], inexact proximal methods [Schmidt et al., 2011], and statistical sampling [Mitliagkas and Mackey, 2017]. Our contributions are hence both theoretical and empirical. Theoretically, we provide novel *verifiable* bounds (Section 4) to inspect and control the gradient approximation error induced by Gibbs sampling. Also, we provide a proof sketch for the main results in Section 5. Empirically, we propose the *tighten asymptotically* (TAY) learning strategy (Section 6) based on the verifiable bounds to boost the performance of SPG.

## 2 BACKGROUND

We first introduce  $L_1$ -regularized discrete MRFs in Section 2.1. We then briefly review SPG as a combination of proximal gradient for sparse statistical learning and Gibbs sampling for addressing the intractable exact gradient evaluation problem.

### 2.1 $L_1$ -Regularized Discrete MRF

For the derivation, we focus on the binary pairwise case and we illustrate that our framework can be generalized to other models in Section 6. Let  $\mathbf{X} = [X_1, X_2, \dots, X_p]^\top \in \{0, 1\}^p$  be a  $p \times 1$  binary random vector. We use an uppercase letter such as  $X$  to denote a random variable and the corresponding lowercase letter to denote a particular *assignment* of the random variable, i.e.,  $X = x$ . We also use boldface letters to represent vectors and matrices and regular letters to represent scalars. We define the function  $\psi : \{0, 1\}^p \rightarrow \{0, 1\}^m$ ,  $\mathbf{x} \rightarrow \psi(\mathbf{x})$  to represent the *sufficient statistics* (a.k.a. *features*) whose values depend on the assignment  $\mathbf{x}$  and compose an  $m \times 1$  vector  $\psi(\mathbf{x})$ , with its  $j^{\text{th}}$  component denoted as  $\psi_j(\mathbf{x})$ . We use  $\mathbb{X}$  to represent a dataset with  $n$  independent and identically distributed (i.i.d.) samples.

With the notation introduced above, the  $L_1$ -regularized discrete MRF problem can be formulated as the following convex optimization problem:

$$\hat{\theta} = \arg \min_{\theta \in \Theta} -\frac{1}{n} \sum_{\mathbf{x} \in \mathbb{X}} \theta^\top \psi(\mathbf{x}) + A(\theta) + \lambda \|\theta\|_1, \quad (1)$$

---

### Algorithm 1 Gibbs Sampling (Gibbs-1)

---

**Require:** initial samples  $\mathbb{S}_0$  and  $\theta$ .

**Ensure:**  $\mathbb{S}$ .

```

1: function GIBBS-1( $\mathbb{S}_0, \theta$ )
2:    $\mathbb{S} \leftarrow \mathbb{S}_0$ , and decide  $p$  from  $\mathbb{S}_0$ .
3:   for  $i \in \{1, \dots, p\}$  do
4:     for  $\mathbf{x} \in \mathbb{S}$  do
5:       Compute  $P_\theta(X_i | \mathbf{x}_{-i})$  according to (5).
6:       Update  $x_i$  by  $P_\theta(X_i | \mathbf{x}_{-i})$ .
7:     end for
8:   end for
9:   return  $\mathbb{S}$ .
10: end function

```

---



---

### Algorithm 2 Gradient Approximation (GRAD)

---

**Require:**  $\theta$ ,  $\mathbb{E}_{\mathbb{X}} \psi(\mathbf{x})$ , and  $q$ .

**Ensure:**  $\Delta f(\theta)$ .

```

1: function GRAD( $\theta, \mathbb{E}_{\mathbb{X}} \psi(\mathbf{x}), q$ )
2:   Initialize  $\mathbb{S}$  with  $q$  samples.
3:   while true do
4:      $\mathbb{S} \leftarrow \text{GIBBS-1}(\mathbb{S}, \theta)$ .
5:     if stopping criteria met then
6:       Compute  $\mathbb{E}_{\mathbb{S}} \psi(\mathbf{x})$  according to (6).
7:        $\Delta f(\theta) \leftarrow \mathbb{E}_{\mathbb{S}} \psi(\mathbf{x}) - \mathbb{E}_{\mathbb{X}} \psi(\mathbf{x})$ .
8:       break.
9:     end if
10:  end while
11:  return  $\Delta f(\theta)$ .
12: end function

```

---



---

### Algorithm 3 Stochastic Proximal Gradient (SPG)

---

**Require:**  $\mathbb{X}$ ,  $\lambda$ , and  $q$ .

**Ensure:**  $\tilde{\theta}$ .

```

1: function SPG( $\mathbb{X}, \lambda, q$ )
2:   Compute  $\mathbb{E}_{\mathbb{X}} \psi(\mathbf{x})$  according to (4).
3:   Initialize  $\theta^{(0)}$  randomly and  $k \leftarrow 0$ .
4:   Choose step length  $\alpha$ .
5:   while true do
6:      $\Delta f(\theta^{(k)}) \leftarrow \text{GRAD}(\theta^{(k)}, \mathbb{E}_{\mathbb{X}} \psi(\mathbf{x}), q)$ .
7:      $\theta^{(k+1)} \leftarrow \mathcal{S}_{\alpha\lambda}(\theta^{(k)} - \alpha \Delta f(\theta^{(k)}))$ .
8:     if Stopping criteria met then
9:        $\tilde{\theta} = \theta^{(k+1)}$ , return  $\tilde{\theta}$ .
10:    end if
11:     $k \leftarrow k + 1$ 
12:  end while
13: end function

```

---

with

$$A(\theta) = \log \sum_{\mathbf{x} \in \{0, 1\}^p} \exp(\theta^\top \psi(\mathbf{x})),$$

where  $\Theta \subseteq \mathbb{R}^m$  is the parameter space of  $\theta$ 's,  $\lambda \geq 0$ ,

and  $A(\boldsymbol{\theta})$  is the *log partition function*. We denote the differentiable part of (1) as

$$f(\boldsymbol{\theta}) = -\frac{1}{n} \sum_{\mathbf{x} \in \mathbb{X}} \boldsymbol{\theta}^\top \boldsymbol{\psi}(\mathbf{x}) + A(\boldsymbol{\theta}). \quad (2)$$

Solving (1) requires evaluating the gradient of  $f(\boldsymbol{\theta})$ , which is given by:

$$\nabla f(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{\theta}} \boldsymbol{\psi}(\mathbf{x}) - \mathbb{E}_{\mathbb{X}} \boldsymbol{\psi}(\mathbf{x}), \quad (3)$$

with

$$\mathbb{E}_{\boldsymbol{\theta}} \boldsymbol{\psi}(\mathbf{x}) = \sum_{\mathbf{x} \in \{0,1\}^p} P_{\boldsymbol{\theta}}(\mathbf{x}) \boldsymbol{\psi}(\mathbf{x}), \quad \mathbb{E}_{\mathbb{X}} \boldsymbol{\psi}(\mathbf{x}) = \frac{1}{n} \sum_{\mathbf{x} \in \mathbb{X}} \boldsymbol{\psi}(\mathbf{x}). \quad (4)$$

$\mathbb{E}_{\boldsymbol{\theta}} \boldsymbol{\psi}(\mathbf{x})$  represents the expectation of the sufficient statistics under  $P_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{\exp(\boldsymbol{\theta}^\top \boldsymbol{\psi}(\mathbf{x}))}{\exp(A(\boldsymbol{\theta}))}$ , which is a discrete MRF probability distribution parameterized by  $\boldsymbol{\theta}$ .  $\mathbb{E}_{\mathbb{X}} \boldsymbol{\psi}(\mathbf{x})$  represents the expectation of the sufficient statistics under the empirical distribution. Computing  $\mathbb{E}_{\mathbb{X}} \boldsymbol{\psi}(\mathbf{x})$  is straightforward, but computing  $\mathbb{E}_{\boldsymbol{\theta}} \boldsymbol{\psi}(\mathbf{x})$  exactly is intractable due to the entanglement of  $A(\boldsymbol{\theta})$ . As a result, various approximations have been made [Wainwright et al., 2007, Höfling and Tibshirani, 2009, Viallon et al., 2014].

## 2.2 Stochastic Proximal Gradient

To efficiently solve (1), many efforts have been made in combining Gibbs sampling [Levin et al., 2009] and proximal gradient descent [Parikh et al., 2014] into SPG, a method that adopts the proximal gradient framework to update iterates, but uses Gibbs sampling as a stochastic oracle to approximate the gradient when the gradient information is needed [Honorio, 2012a, Atchade et al., 2014, Miasojedow and Rejchel, 2016].

Specifically, Gibbs sampling with  $q$  chains running  $\tau$  steps (Gibbs- $\tau$ ) can generate  $q$  samples for  $P_{\boldsymbol{\theta}}(\mathbf{x})$ . Gibbs- $\tau$  is achieved by iteratively applying Gibbs-1 for  $\tau$  times. Gibbs-1 is summarized in Algorithm 1, where

$$P_{\boldsymbol{\theta}}(X_i | \mathbf{x}_{-i}) = P_{\boldsymbol{\theta}}(x_i | x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_p) \quad (5)$$

represents the conditional distribution of  $X_i$  given the assignment of the remaining variables  $\mathbf{x}_{-i}$  under the parameterization  $\boldsymbol{\theta}$ . Denoting the set of these  $q$  (potentially repetitive) samples as  $\mathbb{S}$ , we can approximate  $\mathbb{E}_{\boldsymbol{\theta}} \boldsymbol{\psi}(\mathbf{x})$  by the easily computable

$$\mathbb{E}_{\mathbb{S}} \boldsymbol{\psi}(\mathbf{x}) = \frac{1}{q} \sum_{\mathbf{x} \in \mathbb{S}} \boldsymbol{\psi}(\mathbf{x}) \quad (6)$$

and thus reach the approximated gradient  $\Delta f(\boldsymbol{\theta}) = \mathbb{E}_{\mathbb{S}} \boldsymbol{\psi}(\mathbf{x}) - \mathbb{E}_{\mathbb{X}} \boldsymbol{\psi}(\mathbf{x})$  with the gradient

approximation error:

$$\delta(\boldsymbol{\theta}) = \Delta f(\boldsymbol{\theta}) - \nabla f(\boldsymbol{\theta}).$$

By replacing  $\nabla f(\boldsymbol{\theta})$  with  $\Delta f(\boldsymbol{\theta})$  in proximal gradient, the update rule for SPG can be derived as  $\boldsymbol{\theta}^{(k+1)} = \mathcal{S}_{\alpha\lambda}(\boldsymbol{\theta}^{(k)} - \alpha \Delta f(\boldsymbol{\theta}^{(k)}))$ , where  $\alpha > 0$  is the step length and  $\mathcal{S}_{\lambda}(\mathbf{a})$  is the soft-thresholding operator whose value is also an  $m \times 1$  vector, with its  $i^{\text{th}}$  component defined as  $\mathcal{S}_{\lambda}(\mathbf{a})_i = \text{sgn}(a_i) \max(0, |a_i| - \lambda)$  and  $\text{sgn}(a_i)$  is the sign function.

By defining

$$\begin{aligned} \mathbf{G}_{\alpha}(\boldsymbol{\theta}^{(k)}) &:= \frac{1}{\alpha} \left( \boldsymbol{\theta}^{(k)} - \boldsymbol{\theta}^{(k+1)} \right) \\ &= \frac{1}{\alpha} \left( \boldsymbol{\theta}^{(k)} - \mathcal{S}_{\alpha\lambda} \left( \boldsymbol{\theta}^{(k)} - \alpha \Delta f(\boldsymbol{\theta}^{(k)}) \right) \right), \end{aligned} \quad (7)$$

we can rewrite the previous update rule in a form analogous to the update rule of a standard gradient descent, resulting in the update rule of a *generalized gradient descent* algorithm:

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \alpha \mathbf{G}_{\alpha}(\boldsymbol{\theta}^{(k)}). \quad (8)$$

SPG is summarized in Algorithm 3. Its gradient evaluation procedure based on Algorithm 1 is given in Algorithm 2.

## 3 MOTIVATION

Both practical performance and theoretical guarantees of SPG are still far from satisfactory. Empirically, there are no convincing schemes for selecting  $\tau$  and  $q$ , which hinders the efficiency and accuracy of SPG. Theoretically, to the best of our knowledge, existing non-asymptotic convergence rate guarantees can only be achieved for SPG with an averaging scheme [Schmidt et al., 2011, Honorio, 2012a, Atchade et al., 2014] (see also Section 3.3), instead of ordinary SPG. In contrast, in the exact proximal gradient descent method, the objective function value is non-decreasing and convergent to the optimal value under some mild assumptions [Parikh et al., 2014]. In Section 3.2, we identify that the absence of non-asymptotic convergence rate guarantee for SPG primarily comes from the existence of gradient approximation error  $\delta(\boldsymbol{\theta})$ . In Section 3.3, we further validate that the objective function value achieved by SPG is also highly dependent on  $\delta(\boldsymbol{\theta})$ . These issues bring about the demand of inspecting and controlling  $\delta(\boldsymbol{\theta})$  in each iteration.

### 3.1 Setup and Assumptions

For the ease of presentation, we rewrite the objective function in (1) as  $g(\boldsymbol{\theta}) = f(\boldsymbol{\theta}) + h(\boldsymbol{\theta})$ , where  $h(\boldsymbol{\theta}) = \lambda \|\boldsymbol{\theta}\|_1$ , and  $f(\boldsymbol{\theta})$  is given in (2). Since  $\nabla f(\boldsymbol{\theta})$  is Lipschitz continuous [Honorio, 2012b], we denote its Lipschitz constant as  $L$ . We also make the same assumption that  $\alpha \leq 1/L$  as Schmidt et al. [2011].

### 3.2 Decreasing Objective

It is well-known that exact proximal gradient enjoys a  $O\left(\frac{1}{k}\right)$  convergence rate [Parikh et al., 2014]. One premise for this convergence result is that the objective function value decreases in each iteration. However, satisfying the decreasing condition is much more intricate in the context of SPG. Theorem 1 clearly points out that  $\boldsymbol{\delta}(\boldsymbol{\theta})$  is one main factor determining whether the objective function decreases in SPG.

**Theorem 1.** Let  $\boldsymbol{\theta}^{(k)}$  be the iterate of SPG after the  $k^{\text{th}}$  iteration. Let  $\boldsymbol{\theta}^{(k+1)}$  be defined as in (8). With  $\alpha \leq 1/L$ , we have

$$g(\boldsymbol{\theta}^{(k+1)}) - g(\boldsymbol{\theta}^{(k)}) \leq \alpha \boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})^\top \mathbf{G}_\alpha(\boldsymbol{\theta}^{(k)}) - \frac{\alpha}{2} \|\mathbf{G}_\alpha(\boldsymbol{\theta}^{(k)})\|_2^2.$$

Furthermore, a sufficient condition for  $g(\boldsymbol{\theta}^{(k+1)}) < g(\boldsymbol{\theta}^{(k)})$  is

$$\|\boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})\|_2 < \frac{1}{2} \|\mathbf{G}_\alpha(\boldsymbol{\theta}^{(k)})\|_2.$$

According to Theorem 1, if the magnitude of the noise, quantified by  $\|\boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})\|_2$ , is reasonably small, the objective function value decreases in each iteration. Under this condition, we can further construct a theoretical support for the convergence rate of the objective function value in the Section 3.3.

### 3.3 Convergence Rate

Assuming that  $\boldsymbol{\delta}(\boldsymbol{\theta})$  is small enough in each iteration to generate a decreasing objective value sequence, we can derive Theorem 2 following Proposition 1 in Schmidt et al. [2011]:

**Theorem 2.** Let  $\mathcal{K} = (\boldsymbol{\theta}^{(0)}, \boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \dots, \boldsymbol{\theta}^{(\kappa)})$  be the iterates generated by Algorithm 3. Then if  $g(\boldsymbol{\theta}^{(k+1)}) \leq g(\boldsymbol{\theta}^{(k)})$  with  $k \in \{1, 2, \dots, \kappa - 1\}$ , we have

$$g(\boldsymbol{\theta}^{(\kappa)}) - g(\hat{\boldsymbol{\theta}}) \leq \frac{L}{2\kappa} \left( \|\boldsymbol{\theta}^{(0)} - \hat{\boldsymbol{\theta}}\|_2 + \frac{2}{L} \sum_{k=1}^{\kappa} \|\boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})\|_2 \right)^2. \quad (9)$$

Recall that  $\hat{\boldsymbol{\theta}}$  is an optimal solution to the sparse MLE problem defined in (1). From (9), it is obvious that if the gradient approximation error is reasonably small, then during the early iterations of SPG,  $\|\boldsymbol{\theta}^{(0)} - \hat{\boldsymbol{\theta}}\|_2$  dominates  $\frac{2}{L} \sum_{k=1}^{\kappa} \|\boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})\|_2$ . Therefore, in the beginning, the convergence rate is  $O(1/\kappa)$ . However, as the iteration proceeds,  $\frac{2}{L} \sum_{k=1}^{\kappa} \|\boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})\|_2$  accumulates and hence in practice SPG can only maintain a convergence rate of  $O(1/\kappa)$  up to some noise level that is closely related to  $\boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})$ . Therefore,  $\boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})$  plays an importance role in the performance of SPG.

Notice that Theorem 2 offers convergence analysis of the objective function value in the last iteration  $g(\boldsymbol{\theta}^{(\kappa)})$ . This result is different from the existing non-asymptotic analysis on  $g(\sum_{k=1}^{\kappa} \boldsymbol{\theta}^{(k)}/\kappa)$ , the objective function evaluated on the average of all the visited solutions [Schmidt et al., 2011, Honorio, 2012a, Atchade et al., 2014]. Theorem 2 is more practical than previous analysis, since  $\sum_{k=1}^{\kappa} \boldsymbol{\theta}^{(k)}/\kappa$  is a dense parameterization not applicable to structure learning.

According to the analysis above, we need to control  $\boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})$  in each iteration to achieve a decreasing and  $O\left(\frac{1}{k}\right)$ -converging objective function value sequence. Therefore, we focus on checkable bounds for gradient approximation error in Section 4.

## 4 MAIN RESULTS

In this section, we derive an asymptotic and a non-asymptotic bound to control the gradient approximation error  $\boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})$  in each iteration. For this purpose, we consider an arbitrary  $\boldsymbol{\theta}$ , and perform gradient approximation via Gibbs- $\tau$  using Algorithm 2, given an initial value for the Gibbs sampling algorithm,  $\tilde{\mathbf{x}}_0$ . By bounding  $\boldsymbol{\delta}(\boldsymbol{\theta})$ , we can apply the same technique to address  $\boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})$ .

We first provide a bound for the magnitude of the conditional expectation of  $\boldsymbol{\delta}(\boldsymbol{\theta})$ ,  $\|\mathbb{E}_{\tilde{\mathbf{x}}_\tau}[\boldsymbol{\delta}(\boldsymbol{\theta}) \mid \tilde{\mathbf{x}}_0]\|_2$ , in Section 4.1. Based on this result, we further draw a non-asymptotic bound for the magnitude of the gradient approximation error,  $\|\boldsymbol{\delta}(\boldsymbol{\theta})\|_2$ , in Section 4.2. Both results are *verifiable* in each iteration.

For the derivation of the conclusions, we will focus on binary pairwise Markov networks (BPMNs). Let  $\mathbf{x} \in \{0, 1\}^p$  and  $\boldsymbol{\theta}$  be given, a binary pairwise Markov network [Höfling and Tibshirani, 2009, Geng et al., 2017] is defined as:

$$P_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{1}{Z(\boldsymbol{\theta})} \exp \left( \sum_{i=1}^p \sum_{j \geq i}^p \theta_{ij} x_i x_j \right), \quad (10)$$

where  $Z(\boldsymbol{\theta}) = \exp(A(\boldsymbol{\theta}))$  is the partition function.

$\theta_{ij}$  is a component of  $\boldsymbol{\theta}$  that represents the strength of conditional dependence between  $X_i$  and  $X_j$ .

#### 4.1 An Asymptotic Bound

We first consider the magnitude of the conditional expectation of  $\boldsymbol{\delta}(\boldsymbol{\theta})$  with respect to  $\tilde{\mathbf{x}}_\tau$ ,  $\|\mathbb{E}_{\tilde{\mathbf{x}}_\tau}[\boldsymbol{\delta}(\boldsymbol{\theta}) \mid \tilde{\mathbf{x}}_0]\|_2$ . To this end, we define  $\mathbf{U}$  a  $p \times p$  computable matrix that is related to  $\boldsymbol{\theta}$  and the type of MRF in question.  $U_{ij}$ , the component in the  $i^{\text{th}}$  row and the  $j^{\text{th}}$  column of  $\mathbf{U}$ , is defined as follows:

$$U_{ij} = \frac{|\exp(-\xi_{ij}) - 1|b^*}{(1 + b^* \exp(-\xi_{ij}))(1 + b^*)}, \quad (11)$$

where

$$b^* = \max \left\{ r, \min \left\{ s, \exp \left( \frac{\xi_{ij}}{2} \right) \right\} \right\},$$

$$s = \exp \left( -\xi_{ii} - \sum_{k \neq i, k \neq j} \xi_{ik} \max \{ -\text{sgn}(\xi_{ik}), 0 \} \right)$$

$$r = \exp \left( -\theta_{ii} - \sum_{k \neq i, k \neq j} \xi_{i,k} \max \{ \text{sgn}(\xi_{i,k}), 0 \} \right),$$

and  $\text{sgn}(\xi_{ik})$  is the sign function evaluated on  $\xi_{ij} = \theta_{\min\{i,j\}, \max\{i,j\}}$ .

We then define  $\mathbf{B}_i$  as a  $p \times p$  identity matrix except that its  $i^{\text{th}}$  row is replaced by the  $i^{\text{th}}$  row of  $\mathbf{U}$ , with  $i \in \{1, 2, \dots, p\}$ . We further define

$$\mathbf{B} = \mathbf{B}_p \mathbf{B}_{p-1} \mathbf{B}_{p-2} \cdots \mathbf{B}_i \cdots \mathbf{B}_1$$

and the grand sum  $\mathcal{G}(\mathbf{B}) = \sum_{i=1}^p \sum_{j=1}^p B_{ij}$ , where  $B_{ij}$  is the entry in the  $i^{\text{th}}$  row and the  $j^{\text{th}}$  column of  $\mathbf{B}$ . With the definitions above,  $\|\mathbb{E}_{\tilde{\mathbf{x}}_\tau}[\boldsymbol{\delta}(\boldsymbol{\theta}) \mid \tilde{\mathbf{x}}_0]\|_2$  can be upper bounded by Theorem 3.

**Theorem 3.** Let  $\tilde{\mathbf{x}}_\tau$  be the sample generated after running Gibbs sampling for  $\tau$  steps (Gibbs- $\tau$ ) under the parameterization  $\boldsymbol{\theta}$  initialized by  $\tilde{\mathbf{x}}_0 \in \{0, 1\}^p$ ; then with  $m$  denoting the size of sufficient statistics, the following inequality holds:

$$\|\mathbb{E}_{\tilde{\mathbf{x}}_\tau}[\boldsymbol{\delta}(\boldsymbol{\theta}) \mid \tilde{\mathbf{x}}_0]\|_2 \leq 2\sqrt{m}\mathcal{G}(\mathbf{B}^\tau), \quad (12)$$

where  $\mathbf{B}^\tau$  represents the  $\tau^{\text{th}}$  power of  $\mathbf{B}$ .

In Theorem 3, the bound provided is not only observable in each iteration, but also efficient to compute, offering a convenient method to inspect the quality of the gradient approximation. When the spectral norm of  $\mathbf{U}$  is less than 1, the left hand side of (12) will converge to 0. Thus, by increasing  $\tau$ , we can decrease  $\|\mathbb{E}_{\tilde{\mathbf{x}}_\tau}[\boldsymbol{\delta}(\boldsymbol{\theta}) \mid \tilde{\mathbf{x}}_0]\|_2$  to an arbitrarily small value.

Theorem 3 is derived by bounding the influence of a variable on another variable in  $\mathbf{X}$  (i.e., the Dobrushin influence defined in 2) with  $\mathbf{U}$ . Furthermore,  $\mathbf{U}$  defined in (11) is a sharp bound of the Dobrushin influence whenever  $b^* \neq \exp\left(\frac{\xi_{ij}}{2}\right)$ , explaining why (12) using the definition of  $\mathbf{U}$  is tight enough for practical applications.

#### 4.2 A Non-Asymptotic Bound

In order to provide a non-asymptotic guarantee for the quality of the gradient approximation, we need to concentrate  $\|\boldsymbol{\delta}(\boldsymbol{\theta})\|_2$  around  $\|\mathbb{E}_{\tilde{\mathbf{x}}_\tau}[\boldsymbol{\delta}(\boldsymbol{\theta}) \mid \tilde{\mathbf{x}}_0]\|_2$ . Let  $q$  defined in Section 2.2 be given. Then,  $q$  trials of Gibbs sampling are run, resulting in  $q$  samples,  $\{\tilde{\mathbf{x}}_\tau^{(1)}, \tilde{\mathbf{x}}_\tau^{(2)}, \dots, \tilde{\mathbf{x}}_\tau^{(q)}\}$ . That is to say, for each sufficient statistic,  $\psi_j(\boldsymbol{\theta})$ , with  $j \in \{1, 2, \dots, m\}$ , we have  $q$  samples,  $\{\psi_j^{(1)}(\boldsymbol{\theta}), \psi_j^{(2)}(\boldsymbol{\theta}), \dots, \psi_j^{(q)}(\boldsymbol{\theta})\}$ . Defining the sample variance of the corresponding sufficient statistics as  $V_{\psi_j}$ , we have Theorem 4 to provide a non-asymptotic bound for  $\|\boldsymbol{\delta}(\boldsymbol{\theta})\|_2$ :

**Theorem 4.** Let  $\boldsymbol{\theta}$ ,  $q$ , and an arbitrary  $\tilde{\mathbf{x}}_0 \in \{0, 1\}^p$  be given. Let  $m$  represent the dimension of  $\boldsymbol{\theta}$  and  $\|\boldsymbol{\delta}(\boldsymbol{\theta})\|_2$  represent the magnitude of the gradient approximation error by running  $q$  trials of Gibbs- $\tau$  initialized by  $\tilde{\mathbf{x}}_0$ . Compute  $\mathbf{B}$  according to Section 4.1 and choose  $\epsilon_j > 0$ . Then, with probability at least  $1 - 2\sum_{j=1}^m \beta_j$ , where  $\beta_j > 0$ ,  $j \in \{1, 2, \dots, m\}$ ,

$$\|\boldsymbol{\delta}(\boldsymbol{\theta})\|_2 \leq 2\sqrt{m} \left( \mathcal{G}(\mathbf{B}^\tau) + \sqrt{\frac{\sum_{j=1}^m \epsilon_j^2}{4m}} \right), \quad (13)$$

with  $\beta_j$  satisfying

$$\epsilon_j = 2 \left( \sqrt{\frac{V_{\psi_j} \ln 2 / \beta_j}{2q}} + \frac{7 \ln 2 / \beta_j}{3(q-1)} \right). \quad (14)$$

Notice that the bound in Theorem 4 is easily *checkable*, i.e., given  $\tau$ ,  $q$ ,  $V_{\psi_j}$ 's, and  $\boldsymbol{\theta}$ , we can determine a bound for  $\|\boldsymbol{\delta}(\boldsymbol{\theta})\|_2$  that holds with high probability. Furthermore, Theorem 4 provides the sample complexity needed for gradient estimation. Specifically, given small enough  $\beta_j$ 's, if we let

$$\mathcal{G}(\mathbf{B}^\tau) = \sqrt{\sum_{j=1}^m \epsilon_j^2 / 4m},$$

we can show that

$$2\sqrt{m} \left( \mathcal{G}(\mathbf{B}^\tau) + \sqrt{\sum_{j=1}^m \epsilon_j^2 / 4m} \right) = O\left(\frac{1}{q}\right).$$

That is to say, by assuming that  $\mathcal{G}(\mathbf{B}^\tau)$  and  $\sqrt{\sum_{j=1}^m \epsilon_j^2 / 4m}$  share the same scale, the upper bound of the gradient approximation error converges to 0 as  $q$  increases. Moreover, we include sample variance,  $V_{\psi_j}$ 's, in (13). This is because the information provided by sample variance leads to an improved data dependent bound.

## 5 PROOF SKETCH OF MAIN RESULTS

As mentioned in Section sec:non-asy-bound, the non-asymptotic result in Theorem 4 is derived from the asymptotic bound in Theorem 3 by concentration inequalities, we therefore only highlight the proof of Theorem 3 in this section, and defer other technical results to Supplements. Specifically, the proof of Theorem 3 is divided into two parts: bounding  $\|\mathbb{E}_{\tilde{\mathbf{x}}_\tau}[\delta(\boldsymbol{\theta}) | \tilde{\mathbf{x}}_0]\|_2$  by the total variation distance (Section 5.1) and bounding the total variation distance (Section 5.2).

### 5.1 Bounding $\|\mathbb{E}_{\tilde{\mathbf{x}}_\tau}[\delta(\boldsymbol{\theta}) | \tilde{\mathbf{x}}_0]\|_2$ by the Total Variation Distance

To quantify  $\|\mathbb{E}_{\tilde{\mathbf{x}}_\tau}[\delta(\boldsymbol{\theta}) | \tilde{\mathbf{x}}_0]\|_2$ , we first introduce the concept of total variation distance [Levin et al., 2009] that measures the distance between two distributions over  $\{0, 1\}^p$ .

**Definition 1.** Let  $u(\mathbf{x})$ , and  $v(\mathbf{x})$  be two probability distributions of  $\mathbf{x} \in \{0, 1\}^p$ . Then the total variation distance between  $u(\mathbf{x})$  and  $v(\mathbf{x})$  is given as:

$$\|u(\mathbf{x}) - v(\mathbf{x})\|_{\text{TV}} = \frac{1}{2} \sum_{\mathbf{x} \in \{0, 1\}^p} |u(\mathbf{x}) - v(\mathbf{x})|.$$

With the definition above,  $\|\mathbb{E}_{\tilde{\mathbf{x}}_\tau}[\delta(\boldsymbol{\theta}) | \tilde{\mathbf{x}}_0]\|_2$  can be upper bounded by the total variation distance between two distributions ( $P_\tau(\mathbf{x} | \tilde{\mathbf{x}}_0)$  and  $P_\theta(\mathbf{x})$ ) using the following lemma:

**Lemma 1.** Let  $\tilde{\mathbf{x}}_\tau$  be the sample generated after running Gibbs sampling for  $\tau$  steps (Gibbs- $\tau$ ) under the parameterization  $\boldsymbol{\theta}$  initialized by  $\tilde{\mathbf{x}}_0 \in \{0, 1\}^p$ , then the following is true:

$$\|\mathbb{E}_{\tilde{\mathbf{x}}_\tau}[\delta(\boldsymbol{\theta}) | \tilde{\mathbf{x}}_0]\|_2 \leq 2\sqrt{m} \|P_\tau(\mathbf{x} | \tilde{\mathbf{x}}_0) - P_\theta(\mathbf{x})\|_{\text{TV}}.$$

With Lemma 1, bounding  $\|\mathbb{E}_{\tilde{\mathbf{x}}_\tau}[\delta(\boldsymbol{\theta}) | \tilde{\mathbf{x}}_0]\|_2$  can be achieved by bounding the total variation distance  $\|P_\tau(\mathbf{x} | \tilde{\mathbf{x}}_0) - P_\theta(\mathbf{x})\|_{\text{TV}}$ . Recent advances in the quality control of Gibbs samplers offer us emphverifiable upper bounds for  $\|P_\tau(\mathbf{x} | \tilde{\mathbf{x}}_0) - P_\theta(\mathbf{x})\|_{\text{TV}}$  on the learning of a variety of MRFs [Mitliagkas and Mackey,

2017]. However, they can not be applied to BPMNs because of the positive constraint on parameters. We describe these next.

### 5.2 Bounding $\|P_\tau(\mathbf{x} | \tilde{\mathbf{x}}_0) - P_\theta(\mathbf{x})\|_{\text{TV}}$

Now we generalize the analysis in Mitliagkas and Mackey [2017] to BPMNs without constraints on the sign of parameters by introducing the definition of the Dobrushin influence matrix and a technical lemma.

**Definition 2** (Dobrushin influence matrix). The Dobrushin influence matrix of  $P_\theta(\mathbf{x})$  is a  $p \times p$  matrix  $\mathbf{C}$  with its component in the  $i^{\text{th}}$  row and the  $j^{\text{th}}$  column,  $C_{ij}$ , representing the influence of  $X_j$  on  $X_i$  given as:

$$C_{ij} = \max_{(\mathbf{X}, \mathbf{Y}) \in N_j} \|P_\theta(X_i | \mathbf{X}_{-i}) - P_\theta(Y_i | \mathbf{Y}_{-i})\|_{\text{TV}},$$

where  $(\mathbf{X}, \mathbf{Y}) \in N_j$  represents  $X_l = Y_l$  for all  $l \neq j$ .

**Lemma 2.** Let  $P_\theta(\mathbf{x})$  represent a binary pairwise Markov network defined in (10) that is parameterized by  $\boldsymbol{\theta}$ . An upper bound of the total influence matrix is given by  $\mathbf{U}$  defined in Section 4.1.

It should be noticed that, similar to the Theorem 12 in Mitliagkas and Mackey [2017], Lemma 2 provides an exact calculation except when  $b^* = \exp\left(\frac{\xi_{i,j}}{2}\right)$ .

Therefore, we can consider the  $\mathbf{U}$  defined in Section 4.1 as an upper bound for Dobrushin influence matrix in BPMN and thus apply  $\mathbf{U}$  to Theorem 9 in Mitliagkas and Mackey [2017]. Then, we have

$$\|P_\tau(\mathbf{x} | \tilde{\mathbf{x}}_0) - P_\theta(\mathbf{x})\|_{\text{TV}} \leq \mathcal{G}(\mathbf{B}^\tau),$$

where  $\mathbf{B}^\tau$  represents the  $\tau^{\text{th}}$  power of  $\mathbf{B}$ . Theorem 3 follows this combined with Lemma 1

## 6 STRUCTURE LEARNING

With the two bounds introduced in Section 4, we can easily examine and control the quality of gradient approximation in each iteration by choosing  $\tau$ . In detail, we introduce a criterion for the selection of  $\tau$  in each iteration. Satisfying the proposed criterion, the objective function is guaranteed to decrease asymptotically. That is to say, the difference between  $g(\boldsymbol{\theta}^{(k+1)})$  and  $g(\hat{\boldsymbol{\theta}})$  is asymptotically *tightened*, compared with the difference between  $g(\boldsymbol{\theta}^{(k)})$  and  $g(\hat{\boldsymbol{\theta}})$ . Therefore, we refer to the proposed criterion as TAY-CRITERION. Furthermore, using TAY-CRITERION we provide an improved SPG method denoted by TAY for short.

Specifically, starting from  $\tau = 1$ , TAY stops increasing  $\tau$  when the following is satisfied:

$$2\sqrt{m}\mathcal{G}(\mathbf{B}^\tau) < \frac{1}{2}\|\mathbf{G}_\alpha(\boldsymbol{\theta}^{(k)})\|_2. \quad (\text{TAY-CRITERION})$$

We can also derive a non-asymptotic counterpart of TAY-CRITERION by combining the results of Theorem 1 and Theorem 4:

$$0 < 2\sqrt{m} \left( \mathcal{G}(\mathbf{B}^\tau) + \sqrt{\frac{\sum_{j=1}^m \epsilon_j^2}{4m}} \right) \leq \frac{1}{2} \|\mathbf{G}_\alpha(\boldsymbol{\theta}^{(k)})\|_2,$$

$$\epsilon_j = 2 \left( \sqrt{\frac{2V_{\psi_j} \ln 2/\beta_j}{4q}} + \frac{7 \ln 2/\beta_j}{3(q-1)} \right), \quad (15)$$

where the  $V_{\psi_j}$ 's and  $\beta_j$ 's are defined in Theorem 4. (15) provides the required sample complexity,  $q$ , for TAY in each iteration. However, the selection of  $q$  according to (15) is conservative, because it includes the worst-case scenario where the gradient approximation errors in any two iterations cannot offset each other.

In Section 6.1 and 6.2, we theoretically analyze the performance guarantees of TAY-CRITERION and the convergence of TAY, respectively.

### 6.1 Guarantees of TAY-CRITERION

The theorem below provides the performance guarantee for TAY-CRITERION in each iteration.

**Theorem 5.** Let  $\boldsymbol{\theta}^{(k)}$  and  $\tilde{\mathbf{x}}_0$  be given. Let  $q$  and  $\mathbf{B}$  defined in Theorem 4 be given. For  $\boldsymbol{\theta}^{(k+1)}$  generated in Algorithm 3 using TAY-CRITERION, the following is true:

$$\lim_{q \rightarrow \infty} \mathbb{P} \left( g(\boldsymbol{\theta}^{(k+1)}) < g(\boldsymbol{\theta}^{(k)}) \mid 2\sqrt{m} \mathcal{G}(\mathbf{B}^\tau) < \frac{1}{2} \|\mathbf{G}_\alpha(\boldsymbol{\theta}^{(k)})\|_2 \right) = 1.$$

Theorem 5 makes a statement that the objective function value decreases with large  $q$ . Specifically, TAY-CRITERION assumes that the upper bound of the conditional expectation of  $\|\boldsymbol{\delta}(\boldsymbol{\theta})\|_2$  is small enough to satisfy the sufficient condition proven in Theorem 1. When the number of samples  $q$  is large enough,  $\|\boldsymbol{\delta}(\boldsymbol{\theta})\|_2$  itself is very likely to meet the condition and hence the objective function is also likely to decrease with TAY-CRITERION satisfied.

### 6.2 Convergence of TAY

Finally, based on Theorem 2 and Theorem 5, we derive the following theorem on the convergence of TAY.

**Theorem 6.** Let  $\mathcal{K} = (\boldsymbol{\theta}^{(0)}, \boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \dots, \boldsymbol{\theta}^{(\kappa)})$  be the iterates generated by TAY. Then, with  $k \in \{1, 2, \dots, \kappa - 1\}$ , the following is true:

$$\lim_{q \rightarrow \infty} \mathbb{P} \left[ g(\boldsymbol{\theta}^{(\kappa)}) - g(\hat{\boldsymbol{\theta}}) \leq \frac{L}{2\kappa} \left( \|\boldsymbol{\theta}^{(0)} - \hat{\boldsymbol{\theta}}\|_2 + \frac{2}{L} \sum_{k=1}^{\kappa} \|\boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})\|_2 \right)^2 \right] = 1,$$

where  $\hat{\boldsymbol{\theta}}$  is defined in (1).

### 6.3 Generalizations

As we demonstrate in Section 4 and Section 5, the derivation of our main results relies on bounding the Dobrushin influence with  $\mathbf{U}$  and we show a procedure to construct  $\mathbf{U}$  in the context of BPMNs. Moreover, Mitliagkas and Mackey [2017] and Liu and Domke [2014] provide upper bounds  $\mathbf{U}$ 's for other types of discrete pairwise MRFs. Therefore, combined with their results, our framework can also be applied to other discrete pairwise Markov networks. Dealing with pairwise MRFs is without any loss of generality, since any discrete MRF can be transformed into a pairwise one [Wainwright et al., 2008, Ravikumar et al., 2010].

## 7 EXPERIMENTS

We demonstrate that the structure learning of discrete MRFs benefits substantially from the application of TAY with synthetic data and that the bound provided on the gradient estimation error by Theorem 3 is tighter than existing bounds. To illustrate that TAY is readily available for practical problems, we also run TAY using a real world dataset. Because of the limit of space, we only report the experiments under one set of representative experiment configurations. Exhaustive results using different experiment configurations are presented in the Supplements.

### 7.1 Structure Learning

In order to demonstrate the utility of TAY for effectively learning the structures of BPMNs, we simulate two BPMNs (one with 10 nodes and the other one with 20 nodes):

- We set the number of features to  $p = 10$  ( $p = 20$ ). Components of  $\boldsymbol{\theta}$  in the ground truth model are randomly chosen to be nonzero with an edge generation probability of 0.3. The non-zero components of the real parameter have a uniform distribution on  $[-2, -1] \cup [1, 2]$
- 1000 (2000 for 20 nodes) samples are generated by Gibbs sampling with 1000 burn-in steps.

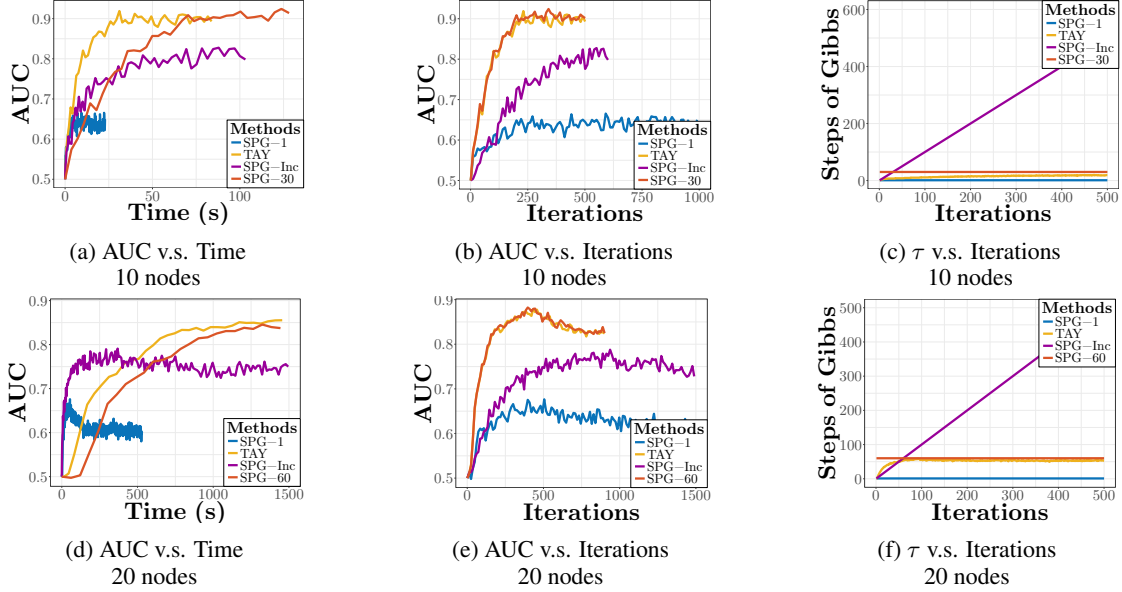


Figure 1: Area under curve (AUC) and the steps of Gibbs sampling ( $\tau$ ) in each iteration for structure learning of a 20-node network.

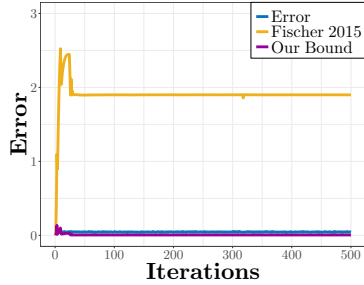


Figure 2: The gradient approximation error, the existing bound and the bound (12) in the structure learning of a 10-node network.

- The results are averaged over 10 trials.

The sizes of the BPMNs generated in this paper are comparable to those in [Honorio, 2012a, Atchade et al., 2014, Miasojedow and Rejchel, 2016].

Then, using the generated samples, we consider SPG and TAY. According to the analysis in Section 4, the quality of the gradient approximation is closely related to the number of Gibbs sampling steps  $\tau$ . However, for SPG, there are no convincing schemes for selecting  $\tau$ . Therefore, we consider a large enough  $\tau = 30$  ( $\tau = 60$  for 20 nodes) to make sure that the gradient approximation error is small enough. Furthermore, we also evaluate the performance of the algorithm using an increasing  $\tau$  ( $\tau = k$  in the  $k^{th}$  iteration), suggested by Atchade et al. [2014] (SPG-Inc).

To strike a fair comparison, we use the same step length  $\alpha = 0.4$  and regularization parameter  $\lambda = 0.025$  ( $\lambda = 0.017$  for 20 nodes) for different methods. We do not tune the step length individually for each method, since Atchade et al. [2014] has shown that various learning rate selection schemes have minimal impact on the performance in the context of SPG. The number of chains used in Gibbs sampling,  $q$ , is not typically a tunable parameter either, since it indicates the allocation of the computational resources. For each method, it can be easily noticed that the larger the number of samples is the slower but more accurate the method will be. Furthermore, if the  $q$ 's are different for different methods, it would be difficult to distinguish the effect of  $\tau$  from that of  $q$ . Therefore, we set it to 2000 for 10-node networks and 5000 for 20-node networks. Performances of different methods are compared using the area under curve (AUC) of receiver operating characteristic (ROC) for structure learning in Figure 1. The Gibbs sampling steps in each method are also compared in Figure 1.

Notice that we plot AUCs against both time (Figure 1a and Figure 1d) and iterations (Figure 1b and Figure 1e). The two kinds of plots provide different information about the performances of different methods: the former ones focus on overall complexity and the latter illustrate iteration complexity. We run each method until it converges. Using much less time, TAY achieves a similar AUC to SPG with  $\tau = 30$  and  $\tau = 60$ . Moreover, SPG with  $\tau = 1$  reaches the lowest AUC, since the quality of the gradient approximation cannot be guaranteed with such a small  $\tau$ . Therefore, the experimental results

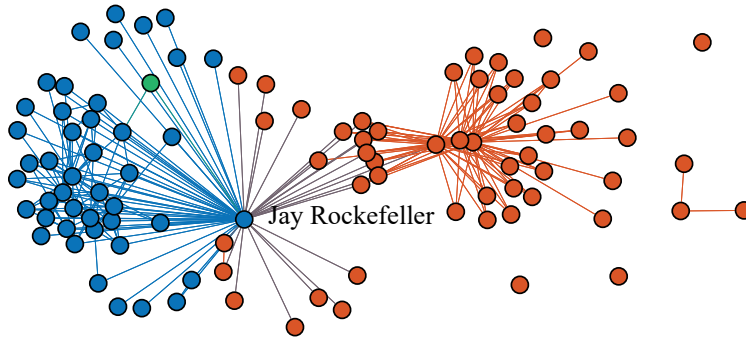


Figure 3: The result of TAY on the senator voting data: Red vertices denote Republicans, blue Democrats, and green Independent. The figure is rendered by Gephi [Bastian et al., 2009].

indicate that TAY adaptively chooses a  $\tau$  achieving reasonable accuracy as well as efficiency for structure learning in each iteration. For a more thorough comparison, we also contrast the performance of TAY and a non-SPG-based method, i.e., the pseudo-likelihood method [Höfling and Tibshirani, 2009, Geng et al., 2017], in the Supplements. As a result, the two methods achieve comparable AUCs.

## 7.2 Tightness of the Proposed Bound

According to the empirical results above, TAY needs a  $\tau$  only on the order of ten, suggesting that the bound in Theorem 3 is tight enough for practical applications. To illustrate this more clearly, we compare (12) with another bound on the expectation of the gradient approximation error derived by Fischer [2015]. Specifically, we calculate the gradient approximation error, the bound (12), and Fischer [2015]’s bound, in each iteration of learning a 10-node network. The results are reported in Figure 2. Notice that the bound in Fischer [2015] gets extraordinarily loose with more iterations. Considering this, we may need run Gibbs chains for thousands of steps if we use this bound. In contrast, bound (12) is close to and even slightly less than the real error. This is reflective of the fact that the proposed bound is on the expectation instead of the error itself. As a result, (12) is much tighter and thus more applicable.

## 7.3 Real World Data

In our final experiment, we run TAY using the Senate voting data from the second session of the 109<sup>th</sup> Congress [USS]. The dataset has 279 samples and 100 variables. Each sample represents the vote cast by each of the 100 senators for a particular bill, where 0 represents nay, and 1 represents yea. Missing data are imputed as 0’s. The task of interest is to learn a

BPMN model that identifies some clusters that represent the dependency between the voting inclination of each senator and the party with which the senator is affiliated.

We use TAY with  $\alpha = 0.4$ . 5000 Markov chains are used for Gibbs sampling. Since our task is exploratory analysis,  $\lambda = 0.1$  is selected in order to deliver an interpretable result. The proposed algorithm is run for 100 iterations. The resultant BPMN with only edges corresponding to the positive parameters is shown in Figure 3, where each node represents the voting record of a senator and the edges represent some positive dependency between the pair of senators connected. The nodes in red represent Republicans and the nodes in blue represents Democrats. The clustering effects of voting consistency within a party are captured, coinciding with conventional wisdom. More interestingly, Jay Rockefeller, as a Democrat, has many connections with Republicans. This is consistent with the fact that his family has been a “traditionally Republican dynasty” [Wikipedia, 2017].

## 8 CONCLUSION

We consider SPG for  $L_1$ -regularized discrete MRF estimation. Furthermore, we conduct a careful analysis of the gradient approximation error of SPG and provide upper bounds to quantify its magnitude. With the aforementioned analysis, we introduce a learning strategy called TAY and show that it can improve the accuracy and efficiency of SPG.

**Acknowledgement:** Sinong Geng, Zhaobin Kuang, and David Page would like to gratefully acknowledge the NIH BD2K Initiative grant U54 AI117924 and the NIGMS grant 2R01 GM097618. Stephen Wright would like to gratefully acknowledge NSF Awards IIS-1447449, 1628384, 1634597, and 1740707; AFOSR Award FA9550-13-1-0138; Subcontracts 3F-30222 and 8F-30039 from Argonne National Laboratory; and DARPA Award N660011824020.



## References

- U.S. Senate. <http://www.senate.gov/index.htm>. (Accessed on 10/11/2016).
- Y. F. Atchade, G. Fort, and E. Moulines. On stochastic proximal gradient algorithms. 2014.
- M. Bastian, S. Heymann, M. Jacomy, et al. Gephi: an open source software for exploring and manipulating networks. *ICWSM*, 8:361–362, 2009.
- Y. Bengio and O. Delalleau. Justifying and generalizing contrastive divergence. *Neural computation*, 21(6):1601–1621, 2009.
- A. Fischer. Training restricted boltzmann machines. *KI-Künstliche Intelligenz*, 29(4):441–444, 2015.
- A. Fischer and C. Igel. Bounding the bias of contrastive divergence learning. *Neural computation*, 23(3):664–673, 2011.
- S. Geng, Z. Kuang, and D. Page. An efficient pseudo-likelihood method for sparse binary pairwise Markov network estimation. *arXiv preprint arXiv:1702.08320*, 2017.
- S. Geng, Z. Kuang, P. Peissig, and D. Page. Temporal square root graphical models. In *Proceedings of the Thirty-Fifth International Conference on Machine Learning (2018)*, 2018.
- H. Höfling and R. Tibshirani. Estimation of sparse binary pairwise Markov networks using pseudo-likelihoods. *Journal of Machine Learning Research*, 10(Apr):883–906, 2009.
- J. Honorio. Convergence rates of biased stochastic optimization for learning sparse ising models. In J. Langford and J. Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, ICML '12, pages 257–264, New York, NY, USA, July 2012a. Omnipress. ISBN 978-1-4503-1285-1.
- J. Honorio. Lipschitz parametrization of probabilistic graphical models. *arXiv preprint arXiv:1202.3733*, 2012b.
- D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.
- S.-I. Lee, V. Ganapathi, and D. Koller. Efficient structure learning of Markov networks using  $l_1$ -regularization. In *Proceedings of the 19th International Conference on Neural Information Processing Systems*, pages 817–824. MIT Press, 2006.
- D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov chains and mixing times*. American Mathematical Soc., 2009.
- J. Liu and D. Page. Bayesian estimation of latently-grouped parameters in undirected graphical models. In *Advances in Neural Information Processing Systems*, pages 1232–1240, 2013.
- J. Liu, D. Page, H. Nassif, J. Shavlik, P. Peissig, C. McCarty, A. A. Onitilo, and E. Burnside. Genetic variants improve breast cancer risk prediction on mammograms. In *AMIA Annual Symposium Proceedings*, volume 2013, page 876. American Medical Informatics Association, 2013.
- J. Liu, D. Page, P. Peissig, C. McCarty, A. A. Onitilo, A. Trentham-Dietz, and E. Burnside. New genetic variants improve personalized breast cancer diagnosis. *AMIA Summits on Translational Science Proceedings*, 2014:83, 2014a.
- J. Liu, C. Zhang, E. Burnside, and D. Page. Learning heterogeneous hidden Markov random fields. In *Artificial Intelligence and Statistics*, pages 576–584, 2014b.
- J. Liu, C. Zhang, D. Page, et al. Multiple testing under dependence via graphical models. *The Annals of Applied Statistics*, 10(3):1699–1724, 2016.
- X. Liu and J. Domke. Projecting markov random field parameters for fast mixing. In *Advances in Neural Information Processing Systems*, pages 1377–1385, 2014.
- A. Maurer and M. Pontil. Empirical bernstein bounds and sample variance penalization. *arXiv preprint arXiv:0907.3740*, 2009.
- B. Miasojedow and W. Rejchel. Sparse estimation in ising model via penalized Monte Carlo methods. *arXiv preprint arXiv:1612.07497*, 2016.
- I. Mitliagkas and L. Mackey. Improving Gibbs sampler scan quality with DoGS, 2017.
- N. Parikh, S. Boyd, et al. Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239, 2014.
- P. Ravikumar, M. J. Wainwright, J. D. Lafferty, et al. High-dimensional ising model selection using  $l_1$ -regularized logistic regression. *The Annals of Statistics*, 38(3):1287–1319, 2010.
- M. Schmidt, N. L. Roux, and F. R. Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In *Advances in neural information processing systems*, pages 1458–1466, 2011.
- L. Vandenberghe. Lecture notes in ee236c-optimization methods for large-scale systems (spring 2016), 2016.
- V. Viallon, O. Banerjee, E. Jouglu, G. Rey, and J. Coste. Empirical comparison study of approximate methods for structure selection in binary graphical models. *Biometrical Journal*, 56(2):307–331, 2014.
- M. J. Wainwright, P. Ravikumar, and J. D. Lafferty. High-dimensional graphical model selection using  $l_1$ -regularized logistic regression. *Advances in neural information processing systems*, 19:1465, 2007.
- M. J. Wainwright, M. I. Jordan, et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2008.
- Wikipedia. Jay rockefeller—Wikipedia, the free encyclopedia, 2017. URL [https://en.wikipedia.org/wiki/Jay\\_Rockefeller](https://en.wikipedia.org/wiki/Jay_Rockefeller). (Accessed on 05/06/2017).

## Supplements

### A Proofs

#### A.1 Proof of Theorem 1

We first introduce the following technical lemma.

**Lemma 3.** Let  $g(\boldsymbol{\theta})$ ,  $f(\boldsymbol{\theta})$ , and  $h(\boldsymbol{\theta})$  be defined as in Section 2.1; hence  $f(\boldsymbol{\theta})$  is convex and differentiable, and  $\nabla f(\boldsymbol{\theta})$  is Lipschitz continuous with Lipschitz constant  $L$ . Let  $\alpha \leq 1/L$ . Let  $\mathbf{G}_\alpha(\boldsymbol{\theta})$  and  $\Delta f(\boldsymbol{\theta})$  be defined as in Section (2.2). Then for all  $\boldsymbol{\theta}_1$  and  $\boldsymbol{\theta}_2$ , the following inequality holds:

$$g(\boldsymbol{\theta}_1^\dagger) \leq g(\boldsymbol{\theta}_2) + \mathbf{G}_\alpha^\top(\boldsymbol{\theta}_1)(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2) + (\nabla f(\boldsymbol{\theta}_1) - \Delta f(\boldsymbol{\theta}_1))^\top (\boldsymbol{\theta}_1^\dagger - \boldsymbol{\theta}_2) - \frac{\alpha}{2} \|\mathbf{G}_\alpha(\boldsymbol{\theta}_1)\|_2^2, \quad (16)$$

where  $\boldsymbol{\theta}_1^\dagger = \boldsymbol{\theta}_1 - \alpha \mathbf{G}_\alpha(\boldsymbol{\theta}_1)$ .

*Proof.* The proof is based on the convergence analysis of the standard proximal gradient method [Vandenberghe, 2016].  $f(\boldsymbol{\theta})$  is a convex differentiable function whose gradient is Lipschitz continuous with Lipschitz constant  $L$ . By the quadratic bound of the Lipschitz property:

$$f(\boldsymbol{\theta}_1^\dagger) \leq f(\boldsymbol{\theta}_1) - \alpha \nabla^\top f(\boldsymbol{\theta}_1) \mathbf{G}_\alpha(\boldsymbol{\theta}_1) + \frac{\alpha^2 L}{2} \|\mathbf{G}_\alpha(\boldsymbol{\theta}_1)\|_2^2.$$

With  $\alpha \leq 1/L$ , and adding  $h(\boldsymbol{\theta}_1^\dagger)$  on both sides of the quadratic bound, we have an upper bound for  $g(\boldsymbol{\theta}_1^\dagger)$ :

$$g(\boldsymbol{\theta}_1^\dagger) \leq f(\boldsymbol{\theta}_1) - \alpha \nabla^\top f(\boldsymbol{\theta}_1) \mathbf{G}_\alpha(\boldsymbol{\theta}_1) + \frac{\alpha}{2} \|\mathbf{G}_\alpha(\boldsymbol{\theta}_1)\|_2^2 + h(\boldsymbol{\theta}_1^\dagger).$$

By convexity of  $f(\boldsymbol{\theta})$  and  $h(\boldsymbol{\theta})$ , we have:

$$\begin{aligned} f(\boldsymbol{\theta}_1) &\leq f(\boldsymbol{\theta}_2) + \nabla^\top f(\boldsymbol{\theta}_1)(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2), \\ h(\boldsymbol{\theta}_1^\dagger) &\leq h(\boldsymbol{\theta}_2) + (\mathbf{G}_\alpha(\boldsymbol{\theta}_1) - \Delta f(\boldsymbol{\theta}_1))^\top (\boldsymbol{\theta}_1^\dagger - \boldsymbol{\theta}_2), \end{aligned}$$

which can be used to further upper bound  $g(\boldsymbol{\theta}_1^\dagger)$ , and results in (16). Note that we have used the fact that  $\mathbf{G}_\alpha(\boldsymbol{\theta}_1) - \Delta f(\boldsymbol{\theta}_1)$  is a subgradient of  $h(\boldsymbol{\theta}_1^\dagger)$  in the last inequality.  $\square$

With Lemma 3, we are now able to prove Theorem 1. In Lemma 3, let  $\boldsymbol{\theta}_1 = \boldsymbol{\theta}_2 = \boldsymbol{\theta}^{(k)}$ . Then by (8),  $\boldsymbol{\theta}_1^\dagger = \boldsymbol{\theta}^{(k+1)}$ . The inequality in (16) can then be simplified as:

$$g(\boldsymbol{\theta}^{(k+1)}) - g(\boldsymbol{\theta}^{(k)}) \leq \alpha \boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})^\top \mathbf{G}_\alpha(\boldsymbol{\theta}^{(k)}) - \frac{\alpha}{2} \|\mathbf{G}_\alpha(\boldsymbol{\theta}^{(k)})\|_2^2.$$

By the Cauchy-Schwarz inequality and the sufficient condition that  $\|\boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})\|_2 < \frac{1}{2} \|\mathbf{G}_\alpha(\boldsymbol{\theta}^{(k)})\|_2$ , we can further simplify the inequality and conclude  $g(\boldsymbol{\theta}^{(k+1)}) < g(\boldsymbol{\theta}^{(k)})$ .

#### A.2 Proof of Theorem 2

To prove Theorem 2, we first review Proposition 1 in Schmidt et al. [2011]:

**Theorem 7** (Convergence on Average, Schmidt et al. [2011]). Let  $\mathcal{K} = (\boldsymbol{\theta}^{(0)}, \boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \dots, \boldsymbol{\theta}^{(\kappa)})$  be the iterates generated by Algorithm 3, then

$$g\left(\frac{1}{\kappa} \sum_{k=1}^{\kappa} \boldsymbol{\theta}^{(k)}\right) - g(\hat{\boldsymbol{\theta}}) \leq \frac{L}{2\kappa} \left( \|\boldsymbol{\theta}^{(0)} - \hat{\boldsymbol{\theta}}\|_2 + \frac{2}{L} \sum_{k=1}^{\kappa} \|\boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})\|_2 \right)^2.$$

Furthermore, according to the assumption that  $g(\boldsymbol{\theta}^{(k+1)}) \leq g(\boldsymbol{\theta}^{(k)})$  with  $k \in \{1, 2, \dots, \kappa\}$ , we have:  $g\left(\frac{1}{\kappa} \sum_{k=1}^{\kappa} \boldsymbol{\theta}^{(k)}\right) \geq g(\boldsymbol{\theta}^{(\kappa)})$ . Therefore,

$$g(\boldsymbol{\theta}^{(\kappa)}) - g(\hat{\boldsymbol{\theta}}) \leq \frac{L}{2\kappa} \left( \|\boldsymbol{\theta}^{(0)} - \hat{\boldsymbol{\theta}}\|_2 + \frac{2}{L} \sum_{k=1}^{\kappa} \|\boldsymbol{\delta}(\boldsymbol{\theta}^{(k)})\|_2 \right)^2.$$

---

# Active Information Acquisition for Linear Optimization

---

**Shuran Zheng**  
School of Engineering  
and Applied Sciences  
Harvard University  
Cambridge, MA

**Bo Waggoner**  
The Warren Center for  
Network and Data Sciences  
University of Pennsylvania  
Philadelphia, PA

**Yang Liu**  
School of Engineering  
and Applied Sciences  
Harvard University  
Cambridge, MA

**Yiling Chen**  
School of Engineering  
and Applied Sciences  
Harvard University  
Cambridge, MA

## Abstract

We consider partially-specified optimization problems where the goal is to actively, but efficiently, acquire missing information about the problem in order to solve it. An algorithm designer wishes to solve a linear program (LP),  $\max c^T \mathbf{x}$  s.t.  $\mathbf{Ax} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}$ , but does not initially know some of the parameters. The algorithm can iteratively choose an unknown parameter and gather information in the form of a noisy sample centered at the parameter's (unknown) value. The goal is to find an approximately feasible and optimal solution to the underlying LP with high probability while drawing a small number of samples. We focus on two cases. (1) When the parameters  $\mathbf{b}$  of the constraints are initially unknown, we propose an efficient algorithm combining techniques from the ellipsoid method for LP and confidence-bound approaches from bandit algorithms. The algorithm adaptively gathers information about constraints only as needed in order to make progress. We give sample complexity bounds for the algorithm and demonstrate its improvement over a naive approach via simulation. (2) When the parameters  $\mathbf{c}$  of the objective are initially unknown, we take an information-theoretic approach and give roughly matching upper and lower sample complexity bounds, with an (inefficient) successive-elimination algorithm.

## 1 INTRODUCTION

Many real-world settings are modeled as optimization problems. For example, a delivery company plans driver routes to minimize the driver's total travel time; an airline

assigns vehicles to different origin-destination pairs to maximize profit. However, in practice, some parameters of the optimization problems may be initially unknown. The delivery company may not know the average congestion or travel time of various links of the network, but has ways to poll Waze<sup>1</sup> drivers to get samples of travel times on links of the network. The delivery company may not know the demand and the revenues for each origin-destination pair, but can get estimates of them by selling tickets on chosen origin-destination pairs.

To capture such settings, this paper proposes a model of optimization wherein the algorithm can iteratively choose a parameter and draw a "sample" that gives information about that parameter; specifically, the sample is an independent draw from a subgaussian random variable centered at the true value of the parameter. This models, for instance, observing the congestion on a particular segment of road on a particular day. Drawing each sample is presumed to be costly, so the goal of the algorithm is to draw the fewest samples necessary in order to find a solution that is approximately feasible and approximately optimal.

Thus, the challenge falls under an umbrella we term *active information acquisition for optimization (AIAO)*. The key feature of the AIAO setting is the structure of the optimization problem itself, i.e. the objective and constraints. The challenge is to understand how the difficulty of information acquisition relates to this underlying structure. For example, are there information-theoretic quantities relating the structure to the sample complexity? Meanwhile, the opportunity of AIAO is to exploit algorithms for the underlying optimization problem. For example, can one interface with the algorithm to reduce sample complexity by only acquiring the information needed, as it is needed?

These are the questions investigated in this paper, which focuses on active information acquisition for linear opti-

---

<sup>1</sup><https://www.waze.com>

mization problems. Specifically, we consider linear programs in the form

$$\max_{\mathbf{x}} \mathbf{c}^T \mathbf{x}, \text{ s.t. } \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0} \quad (1)$$

with  $\mathbf{A} \in \mathbb{R}^{m \times n}$ ,  $\mathbf{c} \in \mathbb{R}^n$ , and  $\mathbf{b} \in \mathbb{R}^m$ . We will consider either the case that the  $\mathbf{b}$  in the constraints is unknown or else the case that the  $\mathbf{c}$  in the objective is unknown, with all other parameters initially known to the algorithm. The algorithm can iteratively choose an unknown parameter, e.g.  $b_i$ , and draw a “sample” from it, e.g. observing  $b_i + \eta$  for an independent, zero-mean, sub-gaussian  $\eta$ . The algorithm must eventually output a solution  $\mathbf{x}$  such that, with probability  $1 - \delta$ ,  $\mathbf{A}\mathbf{x} \leq \mathbf{b} + \varepsilon_1 \mathbf{1}$  and  $\mathbf{c}^T \mathbf{x} \geq \mathbf{c}^T \mathbf{x}^* - \varepsilon_2$ , where  $\mathbf{x}^*$  is the optimal solution. The goal is for the algorithm to achieve this while using as few total samples as possible.

There is a natural “naive” or “static” approach: draw samples for all unknown parameters until they are known to high accuracy with high probability, then solve the “empirical” linear program. However, we can hope to improve by leveraging known algorithms and properties of linear programs. For example, in the case that  $\mathbf{b}$  is unknown, if a linear program has an optimal solution, it has an optimal solution that is an extreme point (a corner point of the feasible region); and at this extremal optimal solution, several constraints are binding. These suggest that it is more important to focus on binding constraints and to gather information on the differing objective values of extreme points. Algorithms developed in this paper leverage these properties of linear programs to decide how much information to acquire for each unknown parameter.

## 1.1 APPROACHES AND RESULTS

**Two settings and our approaches.** The paper investigates two settings: unknown  $\mathbf{b}$  but known  $\mathbf{c}$ , and unknown  $\mathbf{c}$  but known  $\mathbf{b}$ . We always suppose  $\mathbf{A}$  is known and assume that the linear program has an optimal solution.

It might initially appear that these cases are “equivalent” via duality theory, but we argue that the two cases are quite different when we do not know the parameters of LP exactly. Given a *primal* linear program of the form (1), the *dual* program is given by  $\min_{\mathbf{y}} \mathbf{b}^T \mathbf{y}$  s.t.  $\mathbf{A}^T \mathbf{y} \geq \mathbf{c}, \mathbf{y} \geq \mathbf{0}$ , which is easily transformed into the maximization format of (1). In particular, the parameters  $\mathbf{c}$  in the objective function of a primal LP becomes the parameters in the constraints of the dual LP. By duality theory, the (exact) optimal solutions to the primal and dual are connected by *complementary slackness* conditions. However, this approach breaks down in

the approximate setting for two reasons. First, approximately optimal solutions do not satisfy complementary slackness; and second, even knowing which constraints bind does not suffice to determine the optimal solution  $\mathbf{x}^*$  when some of the constraint or objective parameters are unknown.<sup>2</sup> We hence take two different approaches toward our two settings.

**Unknown- $\mathbf{b}$  case.** In the unknown  $\mathbf{b}$  setting, the uncertainty is over the constraints. Our algorithm combines two main ideas: the ellipsoid method for solving linear programs, and multi-armed bandit techniques for gathering data. The ellipsoid algorithm only requires the information of one violated constraint at each iteration, if there exists a violated one. We then leverage the multi-armed bandits method to find the most violated constraint (if there exists one) using as few samples as possible.

We theoretically bound the number of samples drawn by our algorithm as a function of the parameters of the problem. In simulations on generated linear programs, UCB-Ellipsoid far outperforms the naive approach of sampling all parameters to high accuracy, and approaches the performance of an oracle that knows the binding constraints in advance and needs only to sample these. In other words, the algorithm spends very few resources on unimportant parameters.

**Unknown- $\mathbf{c}$  case.** In the unknown-objective setting, we know the feasible region exactly. Our algorithm focuses only on the set of extreme points of the feasible region. For each of the extreme point, there are a set of possible values for  $\mathbf{c}$  such that if  $\mathbf{c}$  takes any value in the set, this extreme point is the optimal solution to the LP. The algorithm hence draws just enough samples to determine with high probability which is actually the case for the true  $\mathbf{c}$ .

We define an information-theoretic measure,  $Low(\mathcal{I})$  for an instance  $\mathcal{I}$ . We show that this quantity essentially characterizes the sample complexity of a problem instance and we give an algorithm, not necessarily efficient, for achieving it up to low-order factors.

## 2 RELATED WORK

The setting considered in this paper, active information acquisition for optimization, is related to a conceptual

<sup>2</sup>Nor does knowing which constraints bind even necessarily help, as approximately satisfying them may still lead to large violations of other constraints. Thus, while we do not rule out some future approach that connects approximate solutions of the primal and dual, the evidence suggests to us that the two settings are quite different and we approach each differently in this paper.

level to a large number of lines of work that deal with optimization and uncertainty. But it differs from them mostly in either the active aspect or the optimization aspect. We'll discuss some of these related lines of work and the differences below.

**Optimization under uncertainty** Our problem considers optimization with uncertain model parameters, which is also a theme in stochastic programming [Heyman and Sobel, 2003, Neely, 2010], chance-constrained programming [Ben-Tal et al., 2009], and robust optimization [Ben-Tal et al., 2009, Bertsimas et al., 2004]. In both stochastic optimization and chance-constrained programming, there are no underlying true, fixed values of the parameters; instead, a probabilistic, distributional model of the parameters is used to capture the uncertainty and such model of uncertainty becomes part of the optimization formulation. Hence, optimality in expectation or approximate optimality is sought after under the probabilistic model. But in our problem underlying fixed parameters exist, and the problem is only how much information to gather about them. Meanwhile, robust optimization models deterministic uncertainty (e.g. parameters come from a known set) and often seeks for a worst-case solution, *i.e.* a solution that is feasible in the worst case over a set of possible constraints. A key distinction of our model is that there are underlying true values of the parameters and we do not incorporate any probabilistic or deterministic model of the uncertainty into the optimization problem itself. Instead, we take an "active querying" approach to approximately solve the true optimization problem with high probability.

**Artificial intelligence.** Several existing lines of work in the artificial intelligence literature, deal with actively acquiring information about parameters of an optimization problem in order to solve it. Preference elicitation [Braziunas, 2006, Blum et al., 2004] typically focuses on acquiring information about parameters of the *objective* by querying a user about his preferences, this is similar to our goal for the unknown-*c* setting. Relevant to our unknown-*b* case, for more combinatorial problems, the constraint acquisition literature [OConnell et al., 2002, Bessiere et al., 2015] is closer to our problem in some respects, as it posits an optimization problem with unknown constraints that must be learned via interactive querying. We emphasize that a central feature of the model in this paper is *noisy observations*: the observations of the algorithm are only noisy samples of a true underlying parameter. The key challenge is to choose how many repeated samples of each parameter to draw. This aspect of the problem is not to our knowledge present in preference elicitation or model/constraint acquisition.

### **Machine learning and theoretical computer science.**

Much work in *active learning* considers acquiring data points iteratively with a goal of low sample complexity [Balcan et al., 2006, 2007, Castro and Nowak, 2008]. The key difference to AIAO is between *data* and *parameters*. In learning, the goal is to minimize the average or expectation of some loss function over a distribution of data points. Other than its likelihood, each data point plays the same role in the problem. Here, the focus is on how much information about each of various parameters is necessary to solve a structured optimization problem to a desired level of accuracy. In other words, the key question here, which is how much an optimization algorithm needs to know about the parameters of the problem it is solving, does not apply in active learning.

A line of work on sample complexity of reinforcement learning (or approximate reinforcement learning) [Kakade et al., 2003, Lattimore and Hutter, 2012, Azar et al., 2012, Wang, 2017, Chen and Wang, 2016] also bears some resemblance to our problem. A typical setting considered is solving a model-free Markov Decision Processes (MDP), where the transition probabilities and the reward functions are initially unknown but the algorithm can query an oracle to get samples. This problem is a special case of our AIALO problem with unknown *A* and *b* because an MDP can be formulated as an linear program. The solutions provided focuses on the particular structure of MDP, while we consider general linear programs.

Broadly related is recent work on *optimization from samples* Balkanski et al. [2016], which considers the sample complexity of a two-stage process: (1) draw some number of i.i.d. data points; (2) optimize some loss function or submodular function on the data. In that setting, the algorithm sees a number of input-output pairs of the function, randomly distributed, and must eventually choose a particular input to optimize the function. Therefore, it is quite different from our setting in both important ways: (1) the information collected are data points (and evaluations), as in ML above, rather than parameters as in our problem; (2) (so far) it is not active, but acquires information in a batch.

A line of work that is closely related to our unknown-*c* problem is the study of *combinatorial pure exploration* (CPE) problem, where a learner collects samples of unknown parameters of an objective function to identify the optimal member in a solution set. The problem was first proposed in Chen et al. [2014], and subsequently studied by Gabillon et al. [2016], Chen et al. [2016a,b, 2017]. CPE only considers combinatorial optimization problems whose solution set contains only binary vectors of length *n*. A recent work by Chen et al. [2017]

extended CPE to a *General-Sampling* problem by allowing general solution sets. Our unknown- $c$  problem can be fitted into the setting of General-Sampling. Our algorithm for unknown- $c$  was inspired by the work of [Chen et al. \[2017\]](#), but leverages the structure of LP and hence has better sample complexity performance than directly treating it as a General-Sampling problem. The General-Sampling problem does not encompass all AIAO settings, e.g., our unknown- $b$  case.

### 3 MODEL AND PRELIMINARIES

#### 3.1 THE AIALO PROBLEM

We now formally define an instance  $\mathcal{I}$  of the *active information acquisition for linear optimization (AIALO)* problem. We then describe the format of algorithms for solving this problem. Note that one can easily extend this into a more general formal definition of AIAO, for more general optimization problems, but we leave this for future work.

An instance  $\mathcal{I}$  consists of three components. The first component consists of the *parameters* of the underlying linear program on  $n$  variables and  $m$  constraints: a vector  $\mathbf{c} \in \mathbb{R}^n$ , a vector  $\mathbf{b} \in \mathbb{R}^m$ , and a matrix  $\mathbf{A} \in \mathbb{R}^{n \times m}$ . Naturally, these specify the program<sup>3</sup>

$$\max_{\mathbf{x}} \mathbf{c}^T \mathbf{x} \text{ s.t. } \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}. \quad (2)$$

We assume for simplicity in this paper that all linear programs are feasible and are known *a priori* to have a solution of norm at most  $R$ . The second component specifies which parameters are *initially known* and which are *initially unknown*. The third and final component specifies, for each unknown parameter (say  $c_i$ ), of a  $\sigma^2$ -subgaussian distribution with mean equal to the value of the parameter.<sup>4</sup>

Given  $\mathcal{I}$ , we define the following sets of approximately-feasible, approximately-optimal solutions.

**Definition 3.1.** *Given an instance  $\mathcal{I}$ , let  $\mathbf{x}^*$  be an optimal solution to the LP. Define  $OPT(\mathcal{I}; \varepsilon_1, \varepsilon_2)$  to be the set of solutions  $\mathbf{x}$  satisfying  $\mathbf{c}^T \mathbf{x} \geq \mathbf{c}^T \mathbf{x}^* - \varepsilon_1$  and  $\mathbf{A}\mathbf{x} \leq \mathbf{b} + \varepsilon_2 \mathbf{1}$ . We use  $OPT(\mathcal{I})$  as shorthand for  $OPT(\mathcal{I}; 0, 0)$ .*

<sup>3</sup>Note that any linear program can be transformed into the given format with at most a factor 2 increase in  $n$  and  $m$ .

<sup>4</sup>Distribution  $\mathcal{D}$  with mean  $\mu$  is  $\sigma^2$ -subgaussian if, for  $X \sim \mathcal{D}$ , we have  $\mathbb{E}[e^{t(X-\mu)}] \leq e^{\sigma^2 t^2/2}$  for all  $t$ . The family of sub-Gaussian distributions with parameter  $\sigma$  encompasses all distributions that are supported on  $[0, \sigma]$  as well as many unbounded distributions such as Gaussian distributions with variance  $\sigma^2$ .

#### 3.2 ALGORITHM SPECIFICATION

An algorithm for the AIALO problem, run on an instance  $\mathcal{I}$ , functions as follows. The algorithm is given as input  $n$  (number of variables),  $m$  (number of constraints), and  $\sigma^2$  (subgaussian parameter). It is also given the second component of the instance  $\mathcal{I}$ , i.e. a specification of which parameters are known and which are unknown. For each parameter that is specified as “known”, the algorithm is given the value of that parameter, e.g. it is given “ $A_{11} = 42$ .” Finally, the algorithm is given an optimality parameter  $\varepsilon_1$ , a feasibility parameter  $\varepsilon_2$ , and a failure probability parameter  $\delta$ .

The algorithm may iteratively choose an unknown parameter and *sample* that parameter: observe an independent and identically-distributed draw from the distribution corresponding to that parameter (as specified in the third component of the instance  $\mathcal{I}$ ). At some point, the algorithm stops and outputs a *solution*  $\mathbf{x} \in \mathbb{R}^n$ .

**Definition 3.2** ( $(\delta, \varepsilon_1, \varepsilon_2)$ -correct algorithm). *An algorithm  $\mathcal{A}$  is  $(\delta, \varepsilon_1, \varepsilon_2)$ -correct if for any instance  $\mathcal{I}$  and inputs  $(\delta, \varepsilon_1, \varepsilon_2)$ , with probability at least  $1 - \delta$ ,  $\mathcal{A}$  outputs a solution  $\mathbf{x} \in OPT(\mathcal{I}; \varepsilon_1, \varepsilon_2)$ . In the case  $\varepsilon_1 = \varepsilon_2 = 0$ , we say  $\mathcal{A}$  is  $\delta$ -correct.*

Our goal is to find algorithms with low *sample complexity*, i.e., the number of samples drawn by the algorithm.

### 4 UNKNOWN CONSTRAINTS

We will first consider the *unknown-b case* where every parameter of the constraint vector  $\mathbf{b}$  is initially unknown, and all other parameters are initially known. Geometrically, the algorithm is given an objective “direction” ( $\mathbf{c}$ ) and a set of constraint “orientations” ( $\mathbf{A}$ ), but does not initially know the “offset” or displacement  $b_i$  of each constraint  $i$ .

In this setting, we do not expect to attain either exact feasibility or exact optimality, as the exact constraints can never be known, and in general an arbitrarily small change in constraints of an LP leads to a nonzero change in the value of the optimal solution.

The section begins with a lower bound of the sample complexity (across all LP instances) of any correct algorithm.

**Theorem 4.1** (Lower bound for unknown  $\mathbf{b}$ ). *Suppose we have a  $(\delta, \varepsilon_1, \varepsilon_2)$ -correct algorithm  $\mathcal{A}$  where  $\delta \in (0, 0.1)$ ,  $\varepsilon_1 > 0$ ,  $\varepsilon_2 > 0$ . Then for any  $n > 0$ , there exists infinitely many instances of the AIALO problem with unknown- $\mathbf{b}$  with  $n$  variables with objective function  $\|\mathbf{c}\|_\infty = 1$  such that  $\mathcal{A}$  must draw at least*

$$\Omega(n \ln(1/\delta) \cdot \max\{\varepsilon_1, \varepsilon_2\}^{-2})$$

samples in expectation on each of them.

The idea of the lower bound (proof in Appendix C.1) is that in the worst case, an algorithm must accurately estimate at least all  $n$  binding constraints (in general with  $n$  variables, up to  $n$  constraints bind at the optimal solution). It remains open whether we can get a tighter lower bound which also captures the difficulty of ruling out non-binding constraints.

#### 4.1 ELLIPSOID-UCB ALGORITHM

**Background.** The standard ellipsoid algorithm for linear programming begins with an ellipsoid  $\mathcal{E}^{(0)}$  known to contain the optimal solution. Then, it checks two cases: (1) The center of this ellipsoid  $\mathbf{x}^{(0)}$  is feasible, or (2) it is not feasible. If (2), say it violates constraint  $i$ , then the algorithm considers the halfspace defined by the constraint  $\mathbf{A}_i \mathbf{x}^{(0)} \leq b_i$ . If (1), the algorithm considers the halfspace defined by the “constraint”  $\mathbf{c}^T \mathbf{x} \geq \mathbf{c}^T \mathbf{x}^{(0)}$ , as the optimal solution must satisfy this constraint. In either case, it updates to a new ellipsoid  $\mathcal{E}^{(1)}$  defined as the minimal ellipsoid containing the intersection of  $\mathcal{E}^{(0)}$  with the halfspace under consideration.

The obstacle is that, now,  $\mathbf{b}$  is initially unknown. A first observation is that we only need to find a single violated constraint, so there may be no need to sample most parameters at a given round. A second observation is that it suffices to find the **most violated** constraint. This can be beneficial as it may require only a few samples to find the most violated constraint; and in the event that no constraint is violated, we still need to find an upper bound on “closest to violated” constraint in order to certify that no constraint is violated.

To do so, we draw inspiration from algorithms for *bandits* problems (whose details are not important to this paper). Suppose we have  $m$  distributions with means  $\mu_1, \dots, \mu_m$  and variances  $\sigma_1^2, \dots, \sigma_m^2$ , and we wish to find the largest  $\mu_i$ . After drawing a few samples from each distribution, we obtain estimates  $\hat{\mu}_i$  along with confidence intervals given by tail bounds. Roughly, an “upper confidence bound” (UCB) algorithm (see *e.g.* Jamieson and Nowak [2014]) for finding  $\max_i \mu_i$  proceeds by always sampling the  $i$  whose upper confidence bound is the highest.

We therefore will propose a UCB-style approach to doing so, but with the advantage that we can re-use any samples from earlier stages of the ellipsoid algorithm.

**Algorithm and results.** Ellipsoid-UCB is given in Algorithm 1. At each round  $k = 1, 2, \dots$ , we choose the center point  $\mathbf{x}^{(k)}$  of the current ellipsoid  $\mathcal{E}^{(k)}$  and call the subroutine Algorithm 2 to draw samples and check for

violated constraints. We use the result of the oracle to cut the current space exactly as in the standard ellipsoid method, and continue.

Some notation:  $t$  is used to track the total number of samples drawn (from all parameters) and  $T_i(t)$  denotes the number of samples of  $b_i$  drawn up to “time”  $t$ . The average of these samples is:

**Definition 4.1.** Let  $X_{i,s}$  denote the  $s$ -th sample of  $b_i$  and let  $T_i(t)$  denote the number of times  $b_i$  is sampled in the first  $t$  samples. Define  $\hat{b}_{i,T_i(t)} = \sum_{s=1}^{T_i(t)} X_{i,s} / T_i(t)$  to be the empirical mean of  $b_i$  up to “time”  $t$ .

---

#### Algorithm 1 Modified ellipsoid algorithm

---

Let  $\mathcal{E}^{(0)}$  be the initial ellipsoid containing the feasible region.  
 Draw one sample for each  $b_i$ ,  $i \in [m]$ .  
 Let  $k = 0$  and  $t = m$ .  
 Let  $T_i(t) = 1$  for all  $i$ .  
**while** stopping criterion is not met<sup>5</sup> **do**  
 Let  $\mathbf{x}^{(k)}$  be the center of  $\mathcal{E}^{(k)}$   
 Call UCB method to get constraint  $i$  or “feasible”  
**if**  $\mathbf{x}^{(k)}$  is feasible **then**  
 $\mathbf{x} \leftarrow \mathbf{x}^{(k)}$  if  $\mathbf{x}$  not initialized or  $\mathbf{c}^T \mathbf{x}^{(k)} > \mathbf{c}^T \mathbf{x}$ .  
 $\mathbf{y} \leftarrow -\mathbf{c}$   
**else**  
 $\mathbf{y} \leftarrow \mathbf{A}_i^T$   
**end if**  
 Let  $\mathcal{E}^{(k+1)}$  be the minimal ellipsoid that contains  $\mathcal{E}^{(k)} \cap \{\mathbf{p} : \mathbf{y}^T \mathbf{p} \leq \mathbf{y}^T \mathbf{x}^{(k)}\}$   
 Let  $k \leftarrow k + 1$   
**end while**  
 Output  $\mathbf{x}$  or “failure” if it was never set.

---

The performance of the algorithm is measured by how many samples (observations) it needs to draw. To state our theoretical results, define  $V_i(k) = \mathbf{A}_i \mathbf{x}^{(k)} - b_i$  to be the amount by which the  $i$ -th constraint is violated by  $\mathbf{x}^{(k)}$ , and  $V^*(k) = \max_i V_i(k)$ . Define  $gap_{i,\varepsilon}(k) = \max\{|V_i(k)|, V^*(k) - V_i(k), \varepsilon\}$  and  $\Delta_{i,\varepsilon} = \min_k gap_{i,\varepsilon}(k)$ .

**Theorem 4.2** (Ellipsoid-UCB algorithm). *The Ellipsoid-UCB algorithm is  $(\delta, \varepsilon_1, \varepsilon_2)$ -correct and with probability  $1 - \delta$ , draws at most the following number of samples:*

$$O\left(\sum_{i=1}^m \frac{\sigma_i^2}{\Delta_{i,\varepsilon_2/2}^2} \log \frac{m}{\delta} + \sum_{i=1}^m \frac{\sigma_i^2}{\Delta_{i,\varepsilon_2/2}^2} \log \log \left(\frac{\sigma_i^2}{\Delta_{i,\varepsilon_2/2}^2}\right)\right).$$

*Specifically, the number of samples used for  $b_i$  is at most  $\frac{\sigma_i^2}{\Delta_{i,\varepsilon_2/2}^2} \left(\log(m/\delta) + \log \log(\sigma_i^2/\Delta_{i,\varepsilon_2/2}^2)\right)$ .*

---

<sup>5</sup>Our stopping criterion is exactly the same as in the standard ellipsoid algorithm, for which there are a variety of possible criteria that work. In particular, one is  $\sqrt{\mathbf{c}^T \mathbf{P}^{-1} \mathbf{c}} \leq \min\{\varepsilon_1, \varepsilon_2\}$ , where  $\mathbf{P}$  is the matrix corresponding to ellipsoid  $\mathcal{E}^{(k)}$  as discussed above.

---

**Algorithm 2** UCB-method

---

Input  $\mathbf{x}^{(k)}$ Output either index  $j$  of a violated constraint, or “feasible”.Set  $\delta' = \left(\frac{\delta}{20m}\right)^{2/3}$ **loop**

1. Let
- $j$
- be the constraint with the largest index,

$$j = \arg \max_i \mathbf{A}_i \mathbf{x}^{(k)} - \widehat{b}_{i, T_i(t)} + U_i(T_i(t)),$$

where  $U_i(s) = 3\sqrt{2\sigma_i^2 \log(\log(3s/2)/\delta')/s}$  and  $\widehat{b}_{i, T_i(t)}$  as in Definition 4.1.

2. If  $\mathbf{A}_j \mathbf{x}^{(k)} - \widehat{b}_{j, T_j(t)} - U_j(T_j(t)) > 0$  return  $j$ .
3. If  $\mathbf{A}_j \mathbf{x}^{(k)} - \widehat{b}_{j, T_j(t)} + U_j(T_j(t)) < 0$  return “feasible”.
4. If  $U_j(T_j(t)) < \varepsilon_2/2$  return “feasible”.
5. Let  $t \leftarrow t + 1$
6. Draw a sample of  $b_j$ .
7. Let  $T_j(t) = T_j(t-1) + 1$ .
8. Let  $T_i(t) = T_i(t-1)$  for all  $i \neq j$ .

**end loop**

---

*Proof Sketch:* Define event  $\mathcal{A}$  to be the event that  $|\widehat{b}_{i,s} - b_i| \leq U_i(s)$  for all  $s$  and  $i \in [m]$ . According to Lemma 3 in Jamieson et al. [2014],  $\mathcal{A}$  holds with probability at least  $1 - \delta$ .

**Correctness:** Conditioning on event  $\mathcal{A}$  holds, our UCB method will only return a constraint that is violated (line 2) and when it returns “feasible”, no constraint is violated more than  $\varepsilon_2$  (line 3 and 4).

**Number of samples:** We bound the number of samples used on each constraint separately. Consider a fixed ellipsoid iteration  $k$  in which UCB method is given input  $\mathbf{x}^{(k)}$ , the key idea is to prove that if  $b_i$  is sampled in this iteration at “time”  $t$ ,  $U_i(T_i(t))$  should be larger than  $\text{gap}_{i, \varepsilon_2/2}(k)$ . This gives an upper bound of  $T_i(t)$ . Taking the maximum of them, we get the final result.  $\square$

**Discussion.** To understand the bound, suppose for simplicity that each  $\sigma_i = 1$ . We observe that the first term will dominate in all reasonable parameter settings, so we can ignore the second summation in this discussion.

Next, note that each term in the sum reflects a bound on how many times constraint  $i$  must be sampled over the course of the algorithm. This depends inversely on  $\Delta_{i, \varepsilon_2/2}$ , which is a minimum over all stages  $k$  of the “precision” we need of constraint  $i$  at stage  $k$ . We only need a very precise estimate if both of the following conditions are satisfied: (1)  $|V_i(k)|$  is small, meaning that the ellipsoid center  $\mathbf{x}^{(k)}$  is very close to binding constraint  $i$ ; (2) There is no other constraint that is significantly violated, meaning that  $i$  is very close to the most-violated

constraint for  $\mathbf{x}^{(k)}$  if any. Because this is unlikely to happen for most constraints, we expect  $\Delta_{i, \varepsilon_2/2}$  to generally be large (leading to a good bound), although we do not have more precise theoretical bounds. The only constraints where we might expect  $\Delta_{i, \varepsilon_2/2}$  to be small are the binding constraints, which we expect to come close to satisfying the above two conditions at some point. Indeed, this seems inevitable for any algorithm, as we explore in our experiments.

**Comparison to static approach.** Again suppose each  $\sigma_i = 1$  for simplicity. Note that each  $\Delta_{i, \varepsilon_2/2} \geq \varepsilon_2/2$ . This implies that our bound is always better than  $O\left(\frac{m \log(m/\delta)}{\varepsilon_2^2}\right)$ , ignoring the dominated second term.

The static approach is to measure each  $b_i$  with enough samples to obtain a good precision so that relaxed feasibility can be satisfied with high probability, then solve the linear program using the estimated constraints. This uses  $\frac{4m \log(m/\delta)}{\varepsilon_2^2}$  samples. (This number comes from using tail bounds to ensure good precision is achieved on every  $b_i$ .)

Therefore, the UCB-Ellipsoid algorithm dominates the static approach up to some constant factors and can show dramatic instance-wise improvements. Indeed, in some simple cases, such as the number of variables equal to the number of constraints, we do not expect any algorithm to be able to improve over the static approach. However, a nice direction for future work is to show that, if  $m$  is very large compared to  $n$ , then the UCB-Ellipsoid algorithm (or some other algorithm) is guaranteed to asymptotically improve on the static approach.

## 5 UNKNOWN OBJECTIVE FUNCTION

In this section, we consider the *unknown-c case*. Here, every parameter of the objective  $\mathbf{c}$  is initially unknown, and all other parameters are initially known. Geometrically, the algorithm is initially given an exact description of the feasible polytope, in the form of  $\mathbf{A}\mathbf{x} \leq \mathbf{b}$  and  $\mathbf{x} \geq 0$ , but no information about the “direction” of the objective.

Because the constraints are known exactly, we focus on exact feasibility in this setting, i.e.  $\varepsilon_2 = 0$ . We also focus on an information-theoretic understanding of the problem, and produce an essentially optimal but computationally inefficient algorithm. We assume that there is a unique optimal solution  $\mathbf{x}^*$ ,<sup>6</sup> and consider the problem of

---

<sup>6</sup>If we only aim for a  $\varepsilon_1$ -suboptimal solution, we can terminate our algorithm when  $\varepsilon^{(r)}$  (defined in Line 5 of Algorithm 3) becomes smaller than  $\varepsilon_1/2$ , such that the algorithm no longer requires the best point to be unique.



finding an exact optimal solution with confidence  $\delta$  (i.e., a  $\delta$ -correct algorithm). We also make the simplifying assumption that each parameter’s distribution is a Gaussian of variance 1 (in particular is 1-subgaussian). Our results can be easily extend to the general case.

Our approaches are based on the techniques used in [Chen et al. \[2017\]](#), but address a different class of optimization problems. We thus use the same notations as in [Chen et al. \[2017\]](#). We first introduce a function  $Low(\mathcal{I})$  that characterizes the sample complexity required for an LP instance  $\mathcal{I}$ . The function  $Low(\mathcal{I})$  is defined by the solution of a convex program. We then give an instance-wise lower bound in terms of the  $Low(\mathcal{I})$  function and the failure probability parameter  $\delta$ . We also formulate a worst-case lower bound of the problem, which is polynomially related to the instance-wise lower bound. Finally, we give an algorithm based on successive elimination that matches the worst-case lower bound within a factor of  $\ln(1/\Delta)$ , where  $\Delta$  is the gap between the objective function value of the optimal extreme point ( $\mathbf{x}^*$ ) and the second-best.

## 5.1 LOWER BOUNDS

The function  $Low(\mathcal{I})$  is defined as follows.

**Definition 5.1** ( $Low(\mathcal{I})$ ). *For any instance  $\mathcal{I}$  of AIALO (or more generally, for any linear program), we define  $Low(\mathcal{I}) \in \mathbb{R}$  to be the optimal solution to the following convex program.*

$$\begin{aligned} \min_{\tau} \quad & \sum_{i=1}^n \tau_i & (3) \\ \text{s.t.} \quad & \sum_{i=1}^n \frac{(s_i^{(k)} - x_i^*)^2}{\tau_i} \leq \left( \mathbf{c}^T(\mathbf{x}^* - \mathbf{s}^{(k)}) \right)^2, \forall k \\ & \tau_i \geq 0, \forall i \end{aligned}$$

Here  $\mathbf{x}^*$  is the optimal solution to the LP in  $\mathcal{I}$  and  $\mathbf{s}^{(1)}, \dots, \mathbf{s}^{(k)}$  are the extreme points of the feasible region  $\{\mathbf{x} : \mathbf{Ax} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$ .

For intuition about  $Low(\mathcal{I})$ , consider a thought experiment where we are given an extreme point  $\mathbf{x}^*$ , and we want to check whether or not  $\mathbf{x}^*$  is the optimal solution using as few samples as possible. Given our empirical estimate  $\hat{\mathbf{c}}$  we would like to have enough samples so that with high probability, for each  $\mathbf{s}^{(k)} \neq \mathbf{x}^*$ , we have

$$\hat{\mathbf{c}}^T(\mathbf{x}^* - \mathbf{s}^{(k)}) > 0 \iff \mathbf{c}^T(\mathbf{x}^* - \mathbf{s}^{(k)}) > 0.$$

This will hold by a standard concentration bound (Lemma D.2) if enough samples of each parameter are drawn; in particular, “enough” is given by the  $k$ -th constraint in (3).

**Theorem 5.1** (Instance lower bound). *Let  $\mathcal{I}$  be an instance of AIALO in the unknown- $\mathbf{c}$  case. For  $0 < \delta < 0.1$ , any  $\delta$ -correct algorithm  $\mathcal{A}$  must draw*

$$\Omega(Low(\mathcal{I}) \ln \delta^{-1})$$

*samples in expectation on  $\mathcal{I}$ .*

We believe that it is unlikely for an algorithm to match the instance-wise lower bound without knowing the value of  $\mathbf{c}$  and  $\mathbf{x}^*$  in the definition of  $Low(\mathcal{I})$ . To formally prove this claim, for any  $\delta$ -correct algorithm  $\mathcal{A}$ , we construct a group of LP instances that share the same feasible region  $\mathbf{Ax} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}$  but have different objective functions and different optimal solutions. We prove that  $\mathcal{A}$  will have unmatched performance on at least one of these LP instances.

Our worst-case lower bound can be stated as follows.

**Theorem 5.2** (Worst-case lower bound for unknown  $\mathbf{c}$ ). *Let  $n$  be a positive integer and  $\delta \in (0, 0.1)$ . For any  $\delta$ -correct algorithm  $\mathcal{A}$ , there exists an infinite sequence of LP instances with  $n$  variables,  $\mathcal{I}_1, \mathcal{I}_2, \dots$ , such that  $\mathcal{A}$  takes*

$$\Omega \left( Low(\mathcal{I}_k) (\ln |S_k^{(1)}| + \ln \delta^{-1}) \right)$$

*samples in expectation on  $\mathcal{I}_k$ , where  $S_k^{(1)}$  is the set of all extreme points of the feasible region of  $\mathcal{I}_k$ , and  $Low(\mathcal{I}_k)$  goes to infinity.*

## 5.2 SUCCESSIVE ELIMINATION ALGORITHM

Before the description of the algorithm, we first define a function  $LowAll(S, \varepsilon, \delta)$  that indicates the number of samples we should take for each  $c_i$ , such that the difference in objective value between any two points in  $S$  can be estimated to an accuracy  $\varepsilon$  with probability  $1 - \delta$ . Define  $LowAll(S, \varepsilon, \delta)$  to be the optimal solution of the following convex program,

$$\begin{aligned} \min_{\tau} \quad & \sum_{i=1}^n \tau_i & (4) \\ \text{s.t.} \quad & \sum_{i=1}^n \frac{(x_i - y_i)^2}{\tau_i} \leq \frac{\varepsilon^2}{2 \ln(2/\delta)}, \forall \mathbf{x}, \mathbf{y} \in S \\ & \tau_i \geq 0, \forall i. \end{aligned}$$

Our algorithm starts with a set  $S^{(1)}$  that contains all extreme points of the feasible region  $\{\mathbf{x} : \mathbf{Ax} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$ , which is the set of all possible optimal solutions. We first draw samples so that the difference between each pairs in  $S^{(1)}$  is estimated to accuracy  $\varepsilon^{(1)}$ . Then we delete all points that are not optimal with high probability. In the next iteration, we halve the accuracy

$\varepsilon^{(2)} = \varepsilon^{(1)}/2$  and repeat the process. The algorithm terminates when the set contains only one point.

---

**Algorithm 3** A successive elimination algorithm

---

- 1:  $S^{(1)} \leftarrow$  set of all extreme points of feasible region  $\{\mathbf{x} : \mathbf{Ax} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$
  - 2:  $r \leftarrow 1$
  - 3:  $\lambda \leftarrow 10$
  - 4: **while**  $|S^{(r)}| > 1$  **do**
  - 5:    $\varepsilon^{(r)} \leftarrow 2^{-r}, \delta^{(r)} \leftarrow \delta/(10r^2|S^{(1)}|^2)$
  - 6:    $(t_1^{(r)}, \dots, t_n^{(r)}) \leftarrow \text{LowAll}(S^{(r)}, \varepsilon^{(r)}/\lambda, \delta^{(r)})$
  - 7:   Sample  $c_i$  for  $t_i^{(r)}$  times. Let  $\hat{c}_i^{(r)}$  be its empirical mean
  - 8:   Let  $\mathbf{x}^{(r)}$  be the optimal solution in  $S^{(r)}$  with respect to  $\hat{\mathbf{c}}^{(r)}$
  - 9:   Eliminate the points in  $S^{(r)}$  that are  $\varepsilon^{(r)}/2 + 2\varepsilon^{(r)}/\lambda$  worse than  $\mathbf{x}^{(r)}$  when the objective function is  $\hat{\mathbf{c}}^{(r)}$ ,
 
$$S^{(r+1)} \leftarrow \{\mathbf{x} \in S^{(r)} : \langle \mathbf{x}, \hat{\mathbf{c}}^{(r)} \rangle \geq \langle \mathbf{x}^{(r)}, \hat{\mathbf{c}}^{(r)} \rangle - \varepsilon^{(r)}/2 - 2\varepsilon^{(r)}/\lambda\} \quad (5)$$
  - 10:    $r \leftarrow r + 1$
  - 11: **end while**
  - 12: Output  $\mathbf{x} \in S^{(r)}$
- 

The algorithm has the following sample complexity bound.

**Theorem 5.3** (Sample complexity of Algorithm 3). *For the AIALO with unknown-c problem, Algorithm 3 is  $\delta$ -correct and, on instance  $\mathcal{I}$ , with probability  $1 - \delta$  draws at most the following number of samples:*

$$O\left(\text{Low}(\mathcal{I}) \ln \Delta^{-1} (\ln |S^{(1)}| + \ln \delta^{-1} + \ln \ln \Delta^{-1})\right),$$

where  $S^{(1)}$  is the set of all extreme points of the feasible region and  $\Delta$  is the gap in objective value between the optimal extreme point and the second-best,

$$\Delta = \max_{\mathbf{x} \in S^{(1)}} \mathbf{c}^T \mathbf{x} - \max_{\mathbf{x} \in S^{(1)} \setminus \mathbf{x}^*} \mathbf{c}^T \mathbf{x}.$$

*Proof Sketch:* We prove that conditioning on a good event  $\mathcal{E}$  that holds with probability at least  $1 - \delta$ , the algorithm will not delete the optimal solution and will terminate before  $\lceil \log(\Delta^{-1}) \rceil + 1$  iterations. Then we bound the number of samples used in iteration  $r$  by showing that the optimal solution of  $\text{Low}(\mathcal{I})$  times  $\alpha^{(r)} = 32\lambda^2 \ln(2/\delta^{(r)})$  is a feasible solution of the convex program that defines  $\text{LowAll}(S^{(r)}, \varepsilon^{(r)}/\lambda, \delta^{(r)})$ . Therefore the number of samples used in iteration  $r$  is no more than  $\alpha^{(r)} \text{Low}(\mathcal{I})$ .  $\square$

This matches the worst-case lower bound within a problem-dependent factor  $\ln(1/\Delta)$ . Notice, however, that the size of  $|S^{(1)}|$  can be exponentially large, and so

is the size of the convex program (4). So Algorithm 3 is computationally inefficient if implemented straightforwardly, and it remains open whether the algorithm can be implemented in polynomial time or an alternative algorithm with similar sample complexity and better performance can be found.

## 6 EXPERIMENTS

In this section, we investigate the empirical number of samples used by Ellipsoid-UCB algorithm for the unknown-b case of AIALO. We fix  $\delta = 0.1$  and focus on the impact of the other parameters<sup>7</sup>, which are more interesting.

We compare three algorithms on randomly generated LP problems. The first is Ellipsoid-UCB. The second is the naive “static approach”, namely, draw  $4\sigma^2 \log(m/\delta)/\varepsilon_2^2$  samples of each constraint, then solve the LP using estimated means of the parameters. (This is the same approach mentioned in the previous section, except that previously we discussed the case  $\sigma = 1$  for simplicity.) The third is designed to intuitively match the lower bound of Theorem 4.1: Draw  $4\sigma^2 \log(d/\delta)/\varepsilon_2^2$  samples of each of only the binding constraints, where there are  $d$  of them, then solve the LP using estimated means of the  $b_i$ . (For a more fair comparison, we use the same tail bound to derive the number of samples needed for high confidence, so that the constants match more appropriately.)

We generate instances as follows.  $\mathbf{c}$  is sampled from  $[-10, 10]^n$  uniformly at random.  $\mathbf{b}$  is uniformly drawn from  $[0, 10]^n$ . Each  $\mathbf{A}_i$  is sampled from unit ball uniformly at random. Notice that the choice of  $b_i \geq 0$  guarantees feasibility because the origin is always a feasible solution. We also add additional constraints  $x_i \leq 500$  to make sure that the LP generated is bounded. When the algorithm makes an observation, a sample is drawn from Gaussian distribution with variance  $\sigma^2$ .

In Figure 1, each algorithm’s number of samples (average of 50 instances) is plotted as function of different parameters. The number of samples used by Ellipsoid-UCB is proportional to  $n, \sigma^2$  and  $\varepsilon^{-2}$ . However, it does not change much as  $m$  increases.<sup>8</sup> This will not be surprising if ellipsoid uses most of its samples on binding constraints, just as the lower bound does. This is shown in Table 1, where it can be seen that Ellipsoid-UCB re-

<sup>7</sup>99.5 percent of the outputs turn out to satisfy relaxed feasibility and relaxed optimality.

<sup>8</sup>Indeed, the standard ellipsoid algorithm for linear programming requires a number of iterations that is bounded in terms of the number of variables regardless of the number of constraints.

quires much fewer samples of non-binding constraints than binding constraints.

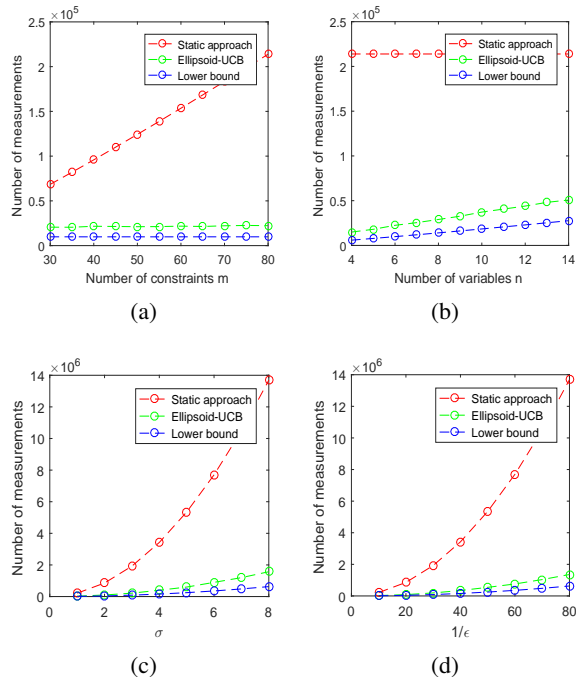


Figure 1: Number of samples as we vary  $m$ ,  $n$ ,  $\sigma$  and  $1/\epsilon$ . Every data point is the mean of 50 randomly drawn problem instances. The baseline parameters are  $m = 80$ ,  $n = 6$ ,  $\sigma = 1$ ,  $\epsilon_1 = \epsilon_2 = 0.1$ . In figure (d),  $\epsilon_1 = \epsilon_2 = \epsilon$ .

	Binding	Non-binding
Static approach	2674	2674
Ellipsoid-UCB	3325	11.7
Lower bound	1476	0

Table 1: Average number of samples used per binding constraint and per non-binding constraint. Numbers are average from 100 trials. Here,  $m = 80$ ,  $n = 4$ ,  $\sigma = 1$ ,  $\epsilon_1 = \epsilon_2 = 0.1$ .

Figure 2 addresses the variance in the number of samples drawn by Ellipsoid-UCB by plotting its empirical CDF over 500 random trials. The horizontal axis is the ratio of samples required by Ellipsoid-UCB to those of the lower bound. For comparison, we also mention  $R$ , the ratio between the performances of the static approach and the lower bound. These results suggest that the variance is quite moderate, particularly when the total number of samples needed grows.

## 7 DISCUSSION AND FUTURE WORK

One question is whether we can extend our results to situations when the constraint matrix  $\mathbf{A}$  is unknown as well as  $\mathbf{b}$ . The goal is again to solve the problem with few

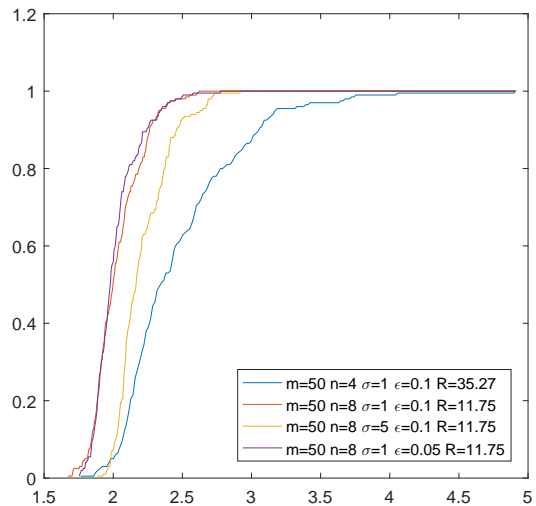


Figure 2: Empirical cumulative distribution function of Ellipsoid-UCB’s number of samples, in units of the “lower bound”, over 500 trials. Note that the lower bound varies when parameters change.  $R = \frac{m \log(m)}{d \log(d)}$  is the ratio between the number of samples used by static approach and lower bound.

total observations. This extended model will allow us to study a wider range of problems. For example, the sample complexity problem in Reinforcement Learning studied by Kakade et al. [2003], Wang [2017], Chen and Wang [2016] is a special case of our AIALO problem with unknown  $\mathbf{A}$  and  $\mathbf{b}$ .

A second extension to the model would allow algorithms access to varying qualities of samples for varying costs. For instance, perhaps some crowd workers can give very low-variance estimates for high costs, while some workers can give cheaper estimates, but have larger variance. In this case, some preliminary theoretical investigations suggest picking the worker that minimizes the product (price)(variance). A direction for future work is for the algorithm to select samples dynamically depending on the payment-variance tradeoffs currently available. A final interested direction is a more mechanism-design approach where the designer collects bids from the agents and selects a winner whose data is used to update the algorithm.

## Acknowledgements

The authors are grateful to anonymous reviewers for their thorough reviews. This work is supported in part by National Science Foundation under grant CCF-1718549 and the Harvard Data Science Initiative.

## References

- Daniel P Heyman and Matthew J Sobel. *Stochastic models in operations research: stochastic optimization*, volume 2. Courier Corporation, 2003.
- Michael J Neely. Stochastic network optimization with application to communication and queueing systems. *Synthesis Lectures on Communication Networks*, 3(1): 1–211, 2010.
- Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust optimization*. Princeton University Press, 2009.
- Dimitris Bertsimas, Dessislava Pachamanova, and Melvyn Sim. Robust linear optimization under general norms. *Operations Research Letters*, 32(6):510–516, 2004.
- Darius Braziunas. Computational approaches to preference elicitation. Technical report, 2006.
- Avrim Blum, Jeffrey Jackson, Tuomas Sandholm, and Martin Zinkevich. Preference elicitation and query learning. *Journal of Machine Learning Research*, 5 (Jun):649–667, 2004.
- Sarah O’Connell, Barry O’Sullivan, and Eugene C Freuder. Strategies for interactive constraint acquisition. In *Proceedings of the CP-2002 Workshop on User-Interaction in Constraint Satisfaction*, pages 62–76, 2002.
- Christian Bessiere, Frédéric Koriche, Nadjib Lazaar, and Barry O’Sullivan. Constraint acquisition. *Artificial Intelligence*, 2015.
- Maria-Florina Balcan, Alina Beygelzimer, and John Langford. Agnostic active learning. In *ICML*, 2006.
- Maria-Florina Balcan, Andrei Broder, and Tong Zhang. Margin based active learning. In *COLT*, 2007.
- Rui M Castro and Robert D Nowak. Minimax bounds for active learning. *Information Theory, IEEE Transactions on*, 54(5):2339–2353, 2008.
- Sham Machandranath Kakade et al. *On the sample complexity of reinforcement learning*. PhD thesis, University of London London, England, 2003.
- Tor Lattimore and Marcus Hutter. Pac bounds for discounted mdps. In *International Conference on Algorithmic Learning Theory*, pages 320–334. Springer, 2012.
- Mohammad Gheshlaghi Azar, Rémi Munos, and Bert Kappen. On the sample complexity of reinforcement learning with a generative model. *arXiv preprint arXiv:1206.6461*, 2012.
- Mengdi Wang. Primal-dual  $\pi$  learning: Sample complexity and sublinear run time for ergodic markov decision problems. *CoRR*, abs/1710.06100, 2017. URL <http://arxiv.org/abs/1710.06100>.
- Yichen Chen and Mengdi Wang. Stochastic primal-dual methods and sample complexity of reinforcement learning. *arXiv preprint arXiv:1612.02516*, 2016.
- Eric Balkanski, Aviad Rubinfeld, and Yaron Singer. The power of optimization from samples. In *Advances in Neural Information Processing Systems*, pages 4017–4025, 2016.
- Shouyuan Chen, Tian Lin, Irwin King, Michael R Lyu, and Wei Chen. Combinatorial pure exploration of multi-armed bandits. In *Advances in Neural Information Processing Systems*, pages 379–387, 2014.
- Victor Gabillon, Alessandro Lazaric, Mohammad Ghavamzadeh, Ronald Ortner, and Peter Bartlett. Improved learning complexity in combinatorial pure exploration bandits. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 1004–1012, 2016.
- Wei Chen, Wei Hu, Fu Li, Jian Li, Yu Liu, and Pinyan Lu. Combinatorial multi-armed bandit with general reward functions. In *Advances in Neural Information Processing Systems*, pages 1659–1667, 2016a.
- Lijie Chen, Anupam Gupta, and Jian Li. Pure exploration of multi-armed bandit under matroid constraints. In *Conference on Learning Theory*, pages 647–669, 2016b.
- Lijie Chen, Anupam Gupta, Jian Li, Mingda Qiao, and Ruosong Wang. Nearly optimal sampling algorithms for combinatorial pure exploration. In *Proceedings of the 2017 Conference on Learning Theory*, pages 482–534, 2017.
- Kevin Jamieson and Robert Nowak. Best-arm identification algorithms for multi-armed bandits in the fixed confidence setting. In *Information Sciences and Systems (CISS), 2014 48th Annual Conference on*, pages 1–6. IEEE, 2014.
- Kevin G Jamieson, Matthew Malloy, Robert D Nowak, and Sébastien Bubeck. lil’ucb: An optimal exploration algorithm for multi-armed bandits. In *COLT*, volume 35, pages 423–439, 2014.
- Emilie Kaufmann, Olivier Cappé, and Aurélien Garivier. On the complexity of best arm identification in multi-armed bandit models. *Journal of Machine Learning Research*, 17(1):1–42, 2016.
- John Duchi. Derivations for linear algebra and optimization.

---

# Transferable Meta Learning Across Domains

---

**Bingyi Kang & Jiashi Feng**

Department of Electrical and Computer Engineering  
National University of Singapore  
*kang@u.nus.edu, elefjia@nus.edu.sg*

## Abstract

Meta learning algorithms are effective at obtaining meta models with the capability of solving new tasks quickly. However, they critically require sufficient tasks for meta model training and the resulted model can only solve new tasks similar to the training ones. These limitations make them suffer performance decline in presence of insufficiency of training tasks in target domains and task heterogeneity—the source (model training) tasks presents different characteristics from target (model application) tasks. To overcome these two significant limitations of existing meta learning algorithms, we introduce the cross-domain meta learning framework and propose a new transferable meta learning (TML) algorithm. TML performs meta task adaptation jointly with meta model learning, which effectively narrows divergence between source and target tasks and enables transferring source meta-knowledge to solve target tasks. Thus, the resulted transferable meta model can solve new learning tasks in new domains quickly. We apply the proposed TML to cross-domain few-shot classification problems and evaluate its performance on multiple benchmarks. It performs significantly better and faster than well-established meta learning algorithms and fine-tuned domain-adapted models.

## 1 Introduction

Meta learning aims at obtaining a model that can capture common characteristics across different learning tasks, such that the learned model can adapt to new tasks quickly. Recently, various meta learning methods (Hariharan & Girshick, 2016; Koch et al., 2015; Lake

et al., 2013; Ravi & Larochelle, 2016; Santoro et al., 2016b; Vinyals et al., 2016) have been developed to solve multiple challenging problems, *e.g.*, few-shot classification (Fei-Fei et al., 2006), and achieved promising performance. Those methods devise different approaches to train a meta model that can be applied to new tasks via simple fine-tuning. In contrast to conventional supervised learning methods that suffer poor generalization performance, meta learning methods explicitly optimize the model generalization ability to new tasks and therefore achieve better performance.

Under the standard meta learning paradigm, the meta model is trained on a meta-training dataset consisting of sufficient training tasks and evaluated on another dataset with novel tasks. However, existing meta learning methods usually assume the training and test tasks have similar characteristics. For instance, for few-shot classification tasks, the samples of different tasks are usually from splits of the *same* dataset (Finn et al., 2017; Hariharan & Girshick, 2016; Vinyals et al., 2016). This actually deviates from the real world scenarios where a pre-trained meta model usually needs to be applicable to heterogeneous tasks in different domains. Moreover, constructing the meta-training set demands sufficiently many labeled examples, which are usually not available in practice considering the “few-shot” nature of meta learning problems. Existing meta learning methods generally ignore such discrepancy between the traditional meta learning paradigm and realistic application scenarios, leading to poor generalization ability of the obtained model to new tasks in new domains.

To extend applicability of meta learning methods, we propose a new framework to utilize data from another (source) domain to construct the meta-training set and aim to develop a new meta learning algorithm to learn a meta model from the source domain that can be directly applied to target domains, without requiring further meta-training. We term this new framework as the cross-domain meta learning. See Fig. 1 for an exam-

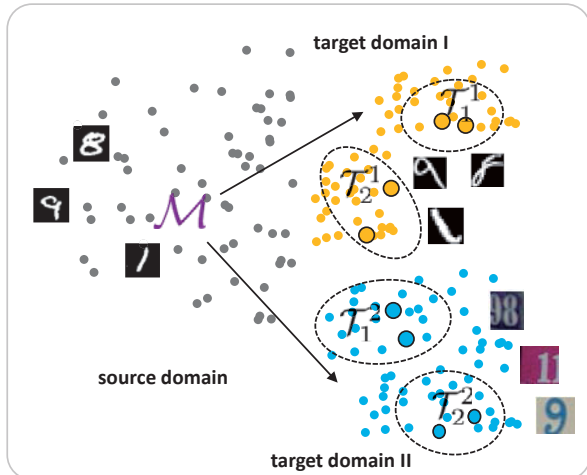


Figure 1: Illustration of the cross-domain meta learning. The pre-trained meta learning model  $\mathcal{M}$  in the source domain is applied to solve new learning tasks  $\mathcal{T}_1^1, \dots, \mathcal{T}_2^2$  in different target domains. The target domains have too few labeled samples to construct sufficient meta-training tasks. Our proposed TML algorithm solves this problem by learning a transferable meta model which can be directly applied to target tasks.

ple. Developing cross-domain meta learning algorithms is difficult due to the scarcity of meta-training examples in target domains and task heterogeneity caused by domain shift. As far as we know, none of existing meta learning methods can deal with these challenging issues.

To address the above challenges, we propose a novel transferable meta learning (TML) algorithm, which provides a meta model capable of fast adapting to new learning tasks in different domains via a few simple gradient descent fine-tuning. Inspired by state-of-the-art meta learning methods (Finn et al., 2017; Vinyals et al., 2016), TML introduces a new learning scheme. It first organizes available training data, very few of which are from target domains, to form a collection of cross-domain meta learning tasks. Taking these tasks as training examples, TML explicitly optimizes the capability of “learning to fast adapt” of the meta model. By taking sensitivity of model parameters to different tasks and domain shift as the joint learning objective, TML effectively trains the meta model to learn task representations robust to domain-shift, enables cross-domain meta-knowledge transfer and makes the model fast adaptable to novel target tasks of different characteristics from the source ones.

TML trains a meta model in two alternating phases. In *meta learning*, TML optimizes the meta model to minimize the loss over all its task-specific fine-tuned models, *i.e.*, minimizing the meta loss. In *meta adaptation*, TML

reinforces the meta model by adapting task representations to minimize domain divergence and thus facilitates meta-knowledge transfer across heterogeneous tasks. We use a domain discriminative loss for measuring domain divergence. Through these two phases, TML effectively minimizes the source domain meta loss and domain divergence jointly, which together serve as an accurate surrogate for learning to minimize the meta loss in the target domain. Therefore, a meta model trained by TML is readily applicable to solving new tasks in target domains.

We apply and evaluate TML for cross-domain few-shot learning problems, on multiple datasets with various domain shift issues. The results demonstrate that TML surpasses fine-tuning based methods and other meta learning models significantly in terms of few-shot classification accuracy and adaptation speed. To our best knowledge, TML is the first one that considers the illness of current meta learning frameworks and explicitly pursues generalization across heterogeneous tasks in different domains. It substantially extends existing meta learning algorithms and mitigates the gap between meta learning frameworks and realistic application scenarios.

## 2 Related Work

**Meta Learning** Recently, some meta learning (Koch et al., 2015; Ravi & Larochelle, 2016; Santoro et al., 2016b; Snell et al., 2017; Vinyals et al., 2016; Wang & Hebert, 2016) works are developed to solve the few-shot learning problems. A meta model is usually trained over a set of similar tasks to capture generalizable properties across tasks, such that it can fast adapt to new similar tasks. Several different strategies for designing meta-learning algorithms are adopted (Andrychowicz et al., 2016; Ravi & Larochelle, 2016). For instance, “learning to compare” aims to learn a comparison metric that can be used to find the most similar labeled sample for each unlabeled input (Koch et al., 2015; Mishra et al., 2017; Vinyals et al., 2016). Some meta learning methods adopt external memory to augment the model (Munkhdalai & Yu, 2017; Santoro et al., 2016a). For example, (Santoro et al., 2016a) builds a meta model upon a Neural Turing Machine (Graves et al., 2014), which encodes and writes labeled examples into the memory and retrieve relevant information from the memory for classifying an unlabeled sample. Differently, Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017) tries to find a proper intermediate model, which can be fine-tuned for several steps to produce a task-specific model given very few samples. However, all these existing models assume that training and testing tasks have similar characteristics, and suffer performance decline in presence of task heterogeneity. Moreover, they all require suffi-

ciently many training tasks. Our proposed TML algorithm is the first one that tries to achieve fast adaptation to new meta learning tasks in presence of varying in the task characteristics in applied domains.

**Few-shot Learning** Few-shot learning (Fei-Fei et al., 2006; Hariharan & Girshick, 2016; Lake et al., 2013) is proposed to learn to recognize new categories with few examples. (Fei-Fei et al., 2006) provides a solution based on Bayesian inference over a pre-trained model to capture general knowledge from previously learned categories, whose generalization ability however is limited by heavy dependency on the relation between previously seen and new objects. Recently, (Hariharan & Girshick, 2016; Luo et al., 2017) propose to transfer intra-class features from base classes to new classes. This method achieves good performance on new examples while maintaining the accuracy on original training classes. But all these conventional few-shot learning methods require retraining the model from scratch when applied to new few-shot learning tasks with randomly assigned labels, thus are incapable of fast adapting to multiple new tasks.

**Domain Adaptation** Many works have been developed for domain adaptation learning (Ganin et al., 2016; Hoffman et al., 2013; Liu & Tuzel, 2016; Motiian et al., 2017a,b; Tzeng et al., 2015, 2017). Maximum Mean Discrepancy (MMD) Tzeng et al. (2014) measures the distribution difference between the source and target domains by computing norm of the mean feature difference between two domains. (Long et al., 2015; Sun & Saenko, 2016) have shown that combining MMD with popular deep learning models is effective. More recently, Generative Adversarial Network (GAN) (Goodfellow et al., 2014) based models have achieved remarkable success, *e.g.* Adversarial Discriminative Domain Adaptation (ADDA) (Tzeng et al., 2017) and CoGAN (Liu & Tuzel, 2016). ADDA adapts a well-learned source CNN by learning a target CNN that maps target-domain images into a feature space, where they are indistinguishable from the source feature space by the GAN discriminator. However, existing domain adaptation methods cannot be applied to solve meta learning tasks.

### 3 The Proposed Algorithm

Meta learning aims to learn a meta model that captures generalizable properties across tasks, such that the model can adapt to solving new similar tasks quickly. In this work, we consider the few-shot classification tasks in particular, which are widely adopted for evaluating meta learning methods. We first define the problem of meta learning for few-shot classification. Then we elaborate

on our target problem, cross-domain meta learning, and our proposed TML algorithm.

#### 3.1 Problem Definition

Let  $\mathcal{X}$  denote the input space and  $\mathcal{Y}$  be the label space. We are interested in meta learning for the  $N$ -category  $k$ -shot learning tasks, where only a small number of  $k$  annotated samples per category (*e.g.*,  $k \leq 5$ ) are available for training a classification model within each task.

Let  $f_\theta(\cdot): \mathcal{X} \rightarrow \mathcal{Y}$  denote the meta learning model with learnable parameter  $\theta$  which is optimized to solve the following few-shot learning tasks:

$$\mathcal{T} \triangleq \{ \underbrace{(x_1, y_1), \dots, (x_{Nk}, y_{Nk})}_{\text{training samples}}, \underbrace{(x_t, y_t)}_{\text{test samples}}, f_\theta, \ell \}. \quad (1)$$

More specifically, each task is to learn a specific classification model  $f_{\theta'}(\cdot)$  from only  $Nk$  training samples such that the following task-specific classification loss on test samples  $(x_t, y_t)$  can be minimized:

$$\mathcal{L}_{\mathcal{T}}(f_{\theta'}) \triangleq \ell(f_{\theta'}(x_t), y_t), \quad (2)$$

where  $\ell$  is the classification loss function.

Existing meta learning approaches generally learn a meta model  $f_\theta(\cdot)$  through meta-training over a collection of tasks  $\mathcal{T} \sim p(\mathcal{T})$  with similar distribution. In meta training, the meta model parameter  $\theta$  is learned to minimize the *meta loss* computed from all the training tasks:

$$\theta = \arg \min_{\theta} \sum_{i=1}^m \mathcal{L}_{\mathcal{T}_i}(f_{\theta'_i}), \mathcal{T}_i \sim p(\mathcal{T}), \quad (3)$$

where  $\theta'_i$  is derived from the meta model  $\theta$  through task-specific adaptation, *e.g.*, by fine-tuning  $\theta$  on training samples of task  $\mathcal{T}_i$ . When there are sufficiently many meta-training tasks  $\mathcal{T}_i$  from the same distribution  $p(\mathcal{T})$ , *i.e.*,  $m$  is sufficiently large, one can reliably obtain a well-performing meta model that can solve new task  $\mathcal{T}_t \sim p(\mathcal{T})$  with satisfactory task-specific loss.

However, in many realistic few-shot learning problems, only a few labeled data are available which are not sufficient to form many tasks for performing meta-training in Eqn. (3). Thus the resulted meta model would suffer from insufficient meta-training and would not generalize well to new tasks. In this work, we propose to address this problem by constructing a meta-training set from another (source) domain of different characteristics where rich labeled data are available. Despite being promising and fitting realistic scenarios better, such a method brings a cross-domain meta learning problem as defined below. This problem is challenging to existing meta learning

methods as they usually assume the meta-training and meta-test tasks are from the same distribution. We aim to solve the following problem in this work, whose solution would also bring significant practical benefits in extending application of meta learning models to other heterogeneous tasks in different domains.

**Definition 1** (Cross-domain Meta Learning). *Suppose there are two different datasets of  $\mathcal{X} \times \mathcal{Y}$  with domain shift in  $\mathcal{X}$ , called source data  $D_S$  and target data  $D_T$ , and  $D_T$  only provides very few labeled samples. We aim to learn a meta model  $f_\theta(\cdot)$  by leveraging the sufficiently many source data  $D_S$  and their formed meta-training tasks  $\mathcal{T}_i \in p_S(\mathcal{T})$ , such that the model can generalize well to new tasks  $\mathcal{T}_i \in p_T(\mathcal{T})$  in target dataset  $D_T$  with small task loss. Here  $p_T(\mathcal{T})$  is different from  $p_S(\mathcal{T})$  in terms of label space  $\mathcal{Y}$  and data distribution  $\mathcal{X}$ .*

Directly training a meta model via minimizing the meta loss (Eqn. (3)) in target dataset  $D_T$  is infeasible due to limited labeled data and consequently insufficient meta-training tasks. On the other hand, although the source domain data is enough for training a meta model, directly applying it to the target domain will suffer poor generalization performance due to domain shift (verified by experiments in Sec. 4). To address this problem, we aim to fully utilize cross-domain knowledge to learn a powerful meta model  $f_\theta$ , which is well prepared for fast adaptation to new few-shot learning tasks in target dataset  $D_T$ . To this end, we develop the transferable meta learning algorithm in this work.

### 3.2 Model-Agnostic Meta-Learning (MAML)

We develop our transferable meta learning (TML) algorithm from the state-of-the-art MAML algorithm (Finn et al., 2017). While we extend MAML here, our proposed idea is applicable to other meta-learning methods.

MAML solves above few-shot learning problems by learning the parameter  $\theta$  such that  $f_\theta$  can solve a new task rapidly via several gradient descent steps on few-shot task-related training examples. To this end, MAML forms a set of training tasks  $\mathcal{T} = \{\mathcal{T}_1, \dots, \mathcal{T}_m\}$ , where each task instantiates an  $N$ -category  $k$ -shot classification problem  $\mathcal{T}_i = \{(x_1^{(i)}, y_1^{(i)}), \dots, (x_{Nk}^{(i)}, y_{Nk}^{(i)}), (x_t^{(i)}, y_t^{(i)}), f_\theta, \ell\}$  as in Eqn. (1).

MAML fine-tunes the meta model  $f_\theta$  to a particular task  $\mathcal{T}_i$  by gradient descent:

$$\theta'_i \leftarrow \theta - \alpha \nabla \mathcal{L}_{\mathcal{T}_i}(f_\theta) \quad (4)$$

where  $\mathcal{L}_{\mathcal{T}_i}(f_\theta) = \frac{1}{Nk} \sum_{j=1}^{Nk} \ell(f_\theta(x_j^{(i)}), y_j^{(i)})$  is the task-related training loss and  $\alpha$  is a universal learning rate.

By treating each task as a training example, MAML optimizes meta model parameter  $\theta$  such that the total loss for the task-wise fine-tuned parameter  $\theta'_i$  over testing samples  $(x_t^{(i)}, y_t^{(i)})$  can be minimized:

$$\min_{\theta} \sum_{i=1}^m \mathcal{L}_{\mathcal{T}_i}(f_{\theta'}) = \sum_{i=1}^m \mathcal{L}_{\mathcal{T}_i}(f_{\theta - \alpha \nabla_{\theta} \mathcal{L}_{\mathcal{T}_i}(f_{\theta})}),$$

where  $\mathcal{L}_{\mathcal{T}_i}(f_{\theta'}) = \ell(f_{\theta'}(x_t^{(i)}), y_t^{(i)})$ , i.e., classification loss on the reserved testing samples. The meta parameter  $\theta$  is then updated by gradient descent  $\theta \leftarrow \theta - \beta \nabla_{\theta} \sum_{i=1}^m \mathcal{L}_{\mathcal{T}_i}(f_{\theta'})$ . The trained meta model  $f_\theta$  can be applied directly to a new similar  $N$ -category  $k$ -shot learning task through gradient descent fine-tuning in Eqn. (4) and performs remarkably well.

MAML inspires us in two aspects for solving few-shot learning problems. First, instead of training a single model on all available training data at once (which is a common practice in most few-shot learning methods (Fei-Fei et al., 2006; Hariharan & Girshick, 2016; Lake et al., 2013)), we should construct learning tasks exactly matching the testing case for model training, which is a more suitable learning scheme for obtaining strong generalization ability from few examples. Second, compared with optimizing the classification accuracy, optimizing the model adaptive capability to new tasks better fits the nature of few-shot learning.

Although MAML provides promising solutions to few-shot learning, its performance highly depends on the similarity of training and testing tasks. It cannot handle discrepancies among tasks—in particular the domain shift between source and target data we aim to address—and suffers performance decline.

### 3.3 Transferable Meta Learning Algorithm

Our proposed Transferable Meta Learning (TML) algorithm solves cross-domain meta learning problems by learning a meta model from the source data  $D_S$  along with a few *unlabeled* target data, which can fast adapt to various few-shot classification tasks in target data  $D_T$ . Beyond existing meta-learning algorithms (like MAML), TML entails the meta model with two-fold fast adaptation capability. First, the model can learn from few training examples fast through simple fine-tuning, solving the few-shot learning tasks. Second, the model can adapt to tasks in different domains, addressing the task heterogeneity issues caused by domain shift.

TML is developed following a simple intuition: the loss function computed in the source domain is expected to be a good indicator of the target loss when both tasks are similar. The main idea of TML is to learn



a meta model that is capable of adapting to new tasks fast and meanwhile learning domain-invariant representations such that source tasks can provide useful meta-knowledge for training models in target domains. To this end, we develop a new learning scheme and propose a novel meta model architecture. The meta model  $f_{\varphi,\theta}$  learned by TML includes two components, a domain-invariant representation learner  $f_{\varphi}$  parameterized by  $\varphi$  and a meta-classifier  $f_{\theta}$  with parameters  $\theta$ , as illustrated in Fig. 2. TML optimizes these two components jointly such that the domain divergence can be reduced in a way favorable for few-shot learning and facilitate cross-domain meta-knowledge transfer.

**TML Learning Scheme** We apply TML to train a meta model on a collection of source training tasks, with a new learning scheme suiting cross-domain meta learning. For notational simplicity, we use  $(\mathbf{x}_S, \mathbf{y}_S)$  and  $\mathbf{x}_T$  to collectively denote source data and unlabeled target data respectively, and let  $(\mathbf{x}_{t,S}, \mathbf{y}_{t,S})$  denote another source sample reserved for evaluating the loss in Eqn. (2). Then, we define a cross-domain few-shot learning task for TML as

$$\mathcal{T}_i \triangleq \{(\mathbf{x}_S^{(i)}, \mathbf{y}_S^{(i)}), (\mathbf{x}_{t,S}^{(i)}, \mathbf{y}_{t,S}^{(i)}), \mathbf{x}_T^{(i)}, f_{\varphi,\theta}, \ell\}, \quad (5)$$

which includes two model training steps. First, fine-tune the meta-classifier  $\theta$  and representation learner  $\varphi$  with few-shot source training samples  $(\mathbf{x}_S^{(i)}, \mathbf{y}_S^{(i)})$  by gradient descent:

$$\varphi'_i, \theta'_i \leftarrow (\varphi, \theta) - \alpha \nabla_{\varphi,\theta} \mathcal{L}_{\mathcal{T}_i}(f_{\varphi,\theta}), \quad (6)$$

and evaluate classification loss on  $(\mathbf{x}_{t,S}^{(i)}, \mathbf{y}_{t,S}^{(i)})$  based on  $\ell$  as in Eqn. (2). Second, optimize representation learner  $\varphi$  to minimize distribution divergence (see below) between source data  $\mathbf{x}_S^{(i)}$  and target data  $\mathbf{x}_T^{(i)}$ . This new task formulation distinguishes TML from existing meta learning algorithms. TML explicitly meta-learns both few-shot classification and task adaptation.

When applying the meta model  $f_{\varphi,\theta}$  to few-shot learning tasks in target domain  $\mathcal{D}_T$ , we apply gradient descent to fine-tune the model parameters  $\varphi$  and  $\theta$  over the few labeled target data, following Eqn. (6).

**TML Algorithm** We explain how TML trains a meta model on the training tasks  $\{\mathcal{T}_i\}$  constructed as above, which alternates between two optimization sub-procedures, as illustrated in Fig. 2.

The first is *meta learning* step, where TML tries to learn a domain-specific meta-classifier  $\theta$  and representation learner  $\varphi$  such that fine-tuning over them can minimize

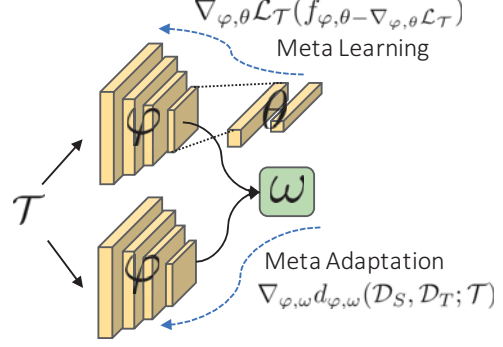


Figure 2: TML for meta model training. TML performs meta learning and task adaptation jointly to optimize the meta model (consisting of representation learner  $\varphi$  and classifier  $\theta$ ) and the discriminative model  $\omega$ .

the loss over source test data:

$$\begin{aligned} \min_{\varphi,\theta} \sum_{\mathcal{T}_i \in \mathcal{T}} \mathcal{L}_{\mathcal{T}_i}(f_{\varphi'_i, \theta'_i}) \\ = \sum_{\mathcal{T}_i \in \mathcal{T}} \mathcal{L}_{\mathcal{T}_i}(f_{(\varphi - \alpha \nabla_{\varphi} \mathcal{L}_{\mathcal{T}_i}(f_{\varphi,\theta}), (\theta - \alpha \nabla_{\theta} \mathcal{L}_{\mathcal{T}_i}(f_{\varphi,\theta}))}) \end{aligned}$$

where the inner loss  $\mathcal{L}_{\mathcal{T}_i}(f_{\varphi,\theta})$  is the total cross-entropy loss over the training samples  $(\mathbf{x}_S^{(i)}, \mathbf{y}_S^{(i)})$  in task  $i$ :

$$\begin{aligned} \mathcal{L}_{\mathcal{T}_i}(f_{\varphi,\theta}) = \sum_{(x_j, y_j) \in (\mathbf{x}_S^{(i)}, \mathbf{y}_S^{(i)})} y_j \log f_{\varphi,\theta}(x_j) \\ + (1 - y_j) \log(1 - f_{\varphi,\theta}(x_j)). \end{aligned} \quad (7)$$

The outer meta loss  $\mathcal{L}_{\mathcal{T}_i}(f_{\varphi'_i, \theta'_i})$  is the cross-entropy loss defined on the task-specific testing samples  $(\mathbf{x}_{t,S}^{(i)}, \mathbf{y}_{t,S}^{(i)}) \in \mathcal{T}_i$  for the fine-tuned model after one gradient descent step  $f_{\varphi'_i, \theta'_i}$ :

$$\begin{aligned} \mathcal{L}_{\mathcal{T}_i}(f_{\varphi'_i, \theta'_i}) = \sum_{(x_{t,S}, y_{t,S}) \in \mathcal{T}_i} y_{t,S} \log f_{\varphi'_i, \theta'_i}(x_{t,S}) \\ + (1 - y_{t,S}) \log(1 - f_{\varphi'_i, \theta'_i}(x_{t,S})). \end{aligned}$$

The involved meta model parameters are updated by gradient descent:

$$\begin{aligned} \varphi \leftarrow \varphi - \beta \nabla_{\varphi} \sum_{\mathcal{T}_i \in \mathcal{T}} \mathcal{L}_{\mathcal{T}_i}(f_{\varphi'_i, \theta'_i}), \\ \theta \leftarrow \theta - \beta \nabla_{\theta} \sum_{\mathcal{T}_i \in \mathcal{T}} \mathcal{L}_{\mathcal{T}_i}(f_{\varphi'_i, \theta'_i}). \end{aligned}$$

This meta learning step is similar to MAML but it decouples the representation learner  $\varphi$  and classifier  $\theta$  for developing the following meta adaptation learning.

The second step in TML is *meta adaptation* with a target to make the meta model fast adaptable to the target

domain  $\mathcal{D}_T$  which is different from the source  $\mathcal{D}_S$  used for extensive meta model training. In particular, TML trains the representation learner  $f_\varphi$  in this step such that it can be adapted through fine-tuning to minimize the distribution divergence between  $\mathcal{D}_T$  and  $\mathcal{D}_S$ , to alleviate domain-shift issues when applying the meta-classifier  $f_\theta$ . Specifically, we use a domain adversarial discriminative loss to measure divergence between domains  $\mathcal{D}_S$  and  $\mathcal{D}_T$ , in the feature space produced by the representation learner  $f_\varphi$ , inspired by (Ganin et al., 2016).

Formally, we use  $\mathcal{U}_S$  to denote the source feature space derived by passing source data  $\mathbf{x}_S$  through the representation learner  $f_\varphi$ . The target feature space  $\mathcal{U}_T \leftarrow \mathbf{x}_T : f_\varphi$  is derived similarly. A domain discriminator  $D_\omega$  parameterized by  $\omega$  is trained on tasks  $\mathcal{T}_i$  to distinguish whether a sample  $x$  is from  $\mathcal{U}_S$  or  $\mathcal{U}_T$ :

$$\omega = \arg \max \log D_\omega(f_\varphi(\mathbf{x}_S)) + \log(1 - D_\omega(f_\varphi(\mathbf{x}_T))),$$

where we give label 1 to the data from source domain and 0 otherwise. We use the negative cross-entropy loss of  $D_\omega$  as a measure over the domain divergence—a larger discriminative loss means the samples are indistinguishable w.r.t. domain shift, indicating a small domain divergence. The domain divergence is calculated as below, dependent on the meta model parameter  $\varphi$  and discriminator  $\omega$ :

$$d_{\varphi,\omega}(\mathcal{D}_S, \mathcal{D}_T) := -\mathbb{E}_{x_S \in \mathcal{D}_S} [\log D_\omega(f_\varphi(x_S))] - \mathbb{E}_{x_T \in \mathcal{D}_T} [\log(1 - D_\omega(f_\varphi(x_T)))].$$

In meta adaptation, TML learns  $\varphi$  and  $\omega$  jointly to minimize the domain divergence derived from samples provided in training tasks  $\mathcal{T}_i$ , *i.e.*,

$$d_{\varphi,\omega}(\mathcal{D}_S, \mathcal{D}_T; \mathcal{T}_i) := -\sum_{x_S \in \mathbf{x}_S^{(i)}} [\log D_\omega(f_\varphi(x_S))] - \sum_{x_T \in \mathbf{x}_T^{(i)}} [\log(1 - D_\omega(f_\varphi(x_T)))].$$

In all, the learning objective for TML is

$$\begin{aligned} \min_{\theta, \varphi} \max_{\omega} \sum_{\mathcal{T}_i \in \mathcal{T}} \mathcal{L}_{\mathcal{T}_i}(f_{\varphi'_i, \theta'_i}) - d_{\varphi,\omega}(\mathcal{D}_S, \mathcal{D}_T; \mathcal{T}_i), \\ \text{s.t. } \varphi'_i, \theta'_i \leftarrow (\varphi, \theta) - \alpha \nabla_{(\varphi, \theta)} \mathcal{L}_{\mathcal{T}_i}(f_{\varphi, \theta}). \end{aligned}$$

TML updates meta model parameters  $\varphi, \theta$  and discriminator  $\omega$  by gradient descent. Details for our proposed TML algorithm are summarized in Alg. 1. The output meta model has following attractive advantages. First, it is well prepared for fast adapting to new tasks and domains through gradient based fine-tuning. Second, its representation learner maps the heterogeneous-domain

---

### Algorithm 1: Transferable Meta Learning

---

**Input:** Source domain data  $\mathcal{D}_S$ , target domain data  $\mathcal{D}_T$ , task set  $\mathcal{T}$ , learning rates  $\alpha, \beta, \gamma$ , max iteration  $I$

**Output:** Representation learner  $\varphi$ , classifier  $\theta$ , domain discriminator  $\omega$

```

1 Randomly initialize  $\theta, \varphi, \omega$ 
2 for  $i = 1, \dots, I$  do
3   Sample task  $\mathcal{T}_i \in \mathcal{T}$ .
4   Estimate  $\nabla \mathcal{L}_{\mathcal{T}_i}(f_{\varphi, \theta})$  using task provided training
   samples  $(\mathbf{x}, \mathbf{y})$  based on Eqn. (7)
5   Fine-tune the parameters  $\varphi, \theta$  as Eqn. (6):
    $(\varphi'_i, \theta'_i) = (\varphi, \theta) - \alpha \nabla_{(\varphi, \theta)} \mathcal{L}_{\mathcal{T}_i}(f_{\varphi, \theta})$ 
6   Estimate meta learning gradient w.r.t.  $(\varphi, \theta)$  on the
   testing examples in task  $\mathcal{T}_i$  by:
    $(\Delta_\varphi^{cls}, \Delta_\theta^{cls}) = \nabla_{(\varphi, \theta)} \sum_{\mathcal{T}_i \in \mathcal{T}} \mathcal{L}_{\mathcal{T}_i^S}(f_{\varphi'_i, \theta'_i})$ 
7   Sample source data  $\mathbf{x}_S \in \mathcal{D}_S$ , target data  $\mathbf{x}_T \in \mathcal{D}_T$ 
8   Estimate meta adaptation gradient w.r.t.  $\varphi$  based on
    $(\mathbf{x}_S, \mathbf{x}_T)$  by:  $\Delta_\varphi^{adpt} = \nabla_\varphi d_{\varphi, \omega}(\mathcal{D}_S, \mathcal{D}_T)$ 
9   Update model parameters:
10   $\varphi \leftarrow \varphi - \beta \Delta_\varphi^{cls} - \gamma \Delta_\varphi^{adpt}$ 
11   $\theta \leftarrow \theta - \beta \Delta_\theta^{cls}$ 
12   $\omega \leftarrow \omega - \gamma \nabla_\omega d_{\varphi, \omega}(\mathcal{D}_S, \mathcal{D}_T)$ 
13 end

```

---

data into a common space with minimized domain divergence such that model can effectively transfer meta-knowledge for few-shot learning across domains. In the experiments, we also verify that TML is superior to the approach that performs domain adaptation and meta learning separately.

## 4 Experiments

**Datasets** We first conduct experiments on learning a meta few-shot classification model with TML across three digits datasets, *i.e.*, MNIST (LeCun et al., 1998), USPS (Le Cun et al., 1989) and SVHN (Netzer et al., 2011). Each dataset contains 10 categories of digit images with varying characteristics. Then we evaluate TML on the office dataset (Saenko et al., 2010), which is a more challenging benchmark with more complex image contents and more significant domain shift. The dataset contains 31 classes of office supplies from three distinct domains, *i.e.*, Amazon, DSLR and Webcam.

**Experiment Settings** All experiments follow the standard  $N$ -way  $K$ -shot protocol (Vinyals et al., 2016) for few-shot learning, which is widely used for evaluating meta learning algorithms. Under this protocol, samples from one dataset are split and re-organized into multiple tasks. Each task provides  $N$  selected classes with  $K$  labeled training instances per class. Each task requires training a few-shot learning model on the provided la-

beled samples. See Sec. 3.1 for the formal definition of the task. We form the learning tasks for TML in the way described in Sec. 3.3. Note the labels for the  $N$  selected classes are randomly assigned under the few-shot learning protocol. The purpose is to evaluate whether the model indeed gains the capability of learning to recognize from few examples, instead of memorizing training examples from all the tasks. In our implementation, both the  $N$  classes and  $K$  samples are randomly selected from the whole dataset. Performance of a few-shot learning model is measured by the classification accuracy on another  $K \times N$  unseen samples.

As we are interested in the cross-domain setting, in the experiments we train a model with access to the source data but evaluate its performance on tasks from the specified target domain. For instance, under the cross-domain setting of “MNIST  $\Rightarrow$  USPS”, we train a model on MNIST data and evaluate its performance in USPS few-shot learning tasks.

**Baselines** Since the problem of cross-domain few-shot classification is new, few valid methods are available for solving it. Here we compare TML with following strong baselines, which leverage the state-of-the-art meta learning algorithms and domain adaptation methods, for obtaining the meta model. 1) Train a standard supervised-learning classifier on the source dataset, and fine-tune it for each specific target task. Comparing with this baseline aims to verify effectiveness of TML in training a meta few-shot learning model with strong generalization ability from few samples, compared with standard supervised learning methods. 2) The state-of-the-art meta learning algorithm, MAML (Finn et al., 2017). In particular, we use MAML to train the meta model in the source domain following the few-shot learning protocols and directly apply MAML to solve target tasks. We aim to demonstrate the advantage of TML over MAML in handling cross-domain few-shot learning problems. 3) The “oracle” MAML. It is trained using the full target datasets and provides performance upper bound for all the cross-domain trained models. 4) MAML+ADDA. Concretely, apply state-of-the-art domain adaptation method ADDA (Tzeng et al., 2017) to align target domain samples with the source domain at first, and then apply MAML on the adapted sample representations. For this baseline, the domain adaptation is blind and unaware of few-shot learning tasks. Comparing TML with it verifies the benefits of end-to-end meta-adaptation and meta-learning in TML.

**Implementation Details** The meta model in TML consists of two components, a meta representation learner and a meta classifier. The former contains four cascade convolutional units, while the latter is built with a linear transformation layer followed by a softmax layer, follow-

ing similar architectures in (Finn et al., 2017; Vinyals et al., 2016). The architecture of the convolutional units varies along with the number of input image channels. For gray-scale images (*e.g.*, digit images), each convolutional unit is composed of 1)  $3 \times 3$  2D convolution with 64 filters and stride  $2 \times 2$ , 2) batch normalization (Ioffe & Szegedy, 2015), and 3) ReLU nonlinear activation function. After the convolutions, a mean pooling layer is used to transform multiple 2D feature maps into a linear feature vector. For color images (*e.g.*, office images), the convolutional unit changes to 1)  $3 \times 3$  2-D convolution with 32 filters and stride 1, 2) batch normalization, 3)  $2 \times 2$  max pooling layer, and 4) ReLU nonlinearity. Then the feature maps are simply flatten to produce a feature vector for classification. The domain discriminator consists of 3 fully connected layers. Each of the two hidden layers has 500 neurons and is followed by a ReLU activation function. One single unit in the output layer is used to indicate the input is from the source or target domain.

To train all network models, we adopt Adam (Kingma & Ba, 2014) as the optimizer. The meta learning rate  $\beta$  is set as 0.001, while the adaptation learning rate  $\gamma$  is set as  $2 \times 10^{-4}$ . The update (or fine-tuning) learning rate  $\alpha$  is fixed as 0.4 for training on gray images and 0.01 for color images. The task batch size is 32 for gray images and 4 for color image due to GPU memory limitation. All models are trained on a single GeForce GTX TITAN Black GPU with 12G memory.

#### 4.1 Results on Digit Datasets

We present results on the digit datasets with multiple cross-domain directions. We first convert all images to gray scale and resize them to  $28 \times 28$ . When building learning tasks, we rotate images class-wise by  $90^\circ$  randomly for data augmentation (Santoro et al., 2016a).

We test our TML in following few-shot settings: 5-way 1-shot and 5-way 5-shot, and four cross-domain directions: MNIST  $\Rightarrow$  USPS, MNIST  $\Rightarrow$  SVHN, USPS  $\Rightarrow$  MNIST and SVHN  $\Rightarrow$  USPS. For the fine-tuning baseline, we first train three classifiers of the same architecture as our model on the three full digit datasets (using the training set) individually. Then the learned classifier on source domain is fine-tuned on the other two target domains under the few-shot setting, *i.e.*, fine-tuning the model on a few training samples and evaluating it on the testing samples for each task individually. For effectively preventing over-fitting, we carefully select the fine-tune learning rate which is set as  $2 \times 10^{-4}$ .

Table 1 reports the few-shot classification accuracy averaged over 500 randomly sampled tasks. One can observe that MAML, MMAL+ADDA and TML all surpass the fine-tune baseline by a large margin in all settings,

Table 1: Few-shot classification results for digit images. The left-most column shows the cross-domain direction, where **M**, **U**, **S** denotes MNIST, USPS, SVHN respectively. The results are reported in terms of classification accuracy (%) averaged over 500 tasks. The gray number in parentheses for MAML is the averaged accuracy obtained by training MAML on the full target datasets. “M + A” denotes the MAML+ADDA baseline.

Direction	5-way 1-shot				5-way 5-shot			
	Fine-tune	MAML	M + A	TML	Fine-tune	MAML	M + A	TML
<b>M</b> $\Rightarrow$ <b>U</b>	39.12	86.33 (97.97)	86.83	<b>91.70</b>	63.42	93.43 (97.95)	93.44	<b>94.43</b>
<b>M</b> $\Rightarrow$ <b>S</b>	21.08	22.30 (83.00)	24.03	<b>29.60</b>	24.56	28.82 (89.70)	29.43	<b>29.68</b>
<b>U</b> $\Rightarrow$ <b>M</b>	37.48	83.30 (99.30)	81.80	<b>88.90</b>	52.99	89.42 (99.59)	<b>90.03</b>	90.01
<b>S</b> $\Rightarrow$ <b>U</b>	23.36	<b>84.60</b> (97.97)	82.70	82.67	61.21	87.23 (97.95)	87.86	<b>89.23</b>

Direction	10-way 1-shot				10-way 5-shot			
	Fine-tune	MAML	M + A	TML	Fine-tune	MAML	M + A	TML
<b>M</b> $\Rightarrow$ <b>U</b>	24.50	74.15 (94.60)	75.05	<b>80.45</b>	49.50	81.86 (95.85)	82.54	<b>86.43</b>
<b>M</b> $\Rightarrow$ <b>S</b>	11.14	12.52 (67.57)	13.13	<b>13.55</b>	13.09	15.36 (84.41)	14.22	<b>15.44</b>
<b>U</b> $\Rightarrow$ <b>M</b>	17.64	53.98 (98.68)	59.05	<b>65.58</b>	31.49	73.09 (98.98)	72.96	<b>75.99</b>
<b>S</b> $\Rightarrow$ <b>U</b>	21.46	54.88 (94.60)	59.30	<b>68.80</b>	43.23	78.71 (95.85)	78.96	<b>80.04</b>

Table 2: Few-shot classification accuracy (in %) of TML in source domain of the digit datasets, averaged over 500 randomly sampled tasks.

	5-way 1-shot	5-way 5-shot
<b>M</b>	99.47	99.42
<b>U</b>	97.40	97.05
<b>S</b>	83.87	89.82

proving the benefits of considering the nature of few-shot learning in model training. More importantly, TML outperforms MAML for almost all settings, by a margin up to 8%. This shows effectiveness of TML in solving domain-shift issues for few-shot learning tasks. The second-best baseline ADDA+MAML also tries to solve domain-shift explicitly by applying ADDA to align domains at first. However its performance is inferior to TML as it conducts domain adaptation blindly which may harm the few-shot learning performance. In contrast, our proposed TML performs meta-adaptation and meta-learning jointly, providing fast adaptation abilities to both new tasks and new domains. Thus it improves ADDA+MAML by up to 9.5%. The superiority of TML over ADDA+MAML becomes more significant when training samples are very limited.

For the  $U \Rightarrow M$  (5-way, 5-shot) setting, TML performs comparably well as ADDA+MAML, where the target domain data are sufficient for ADDA to achieve good domain adaptation. For the  $S \Rightarrow U$  (5-way, 1-shot) setting, MAML performs slightly better than TML. This is because the domain divergence between **S** and **U** is large and target data from a single task are limited for TML to perform meta-adaptation sufficiently well.

For better understanding the domain shift challenge to few-shot learning, we also evaluate the oracle MAML

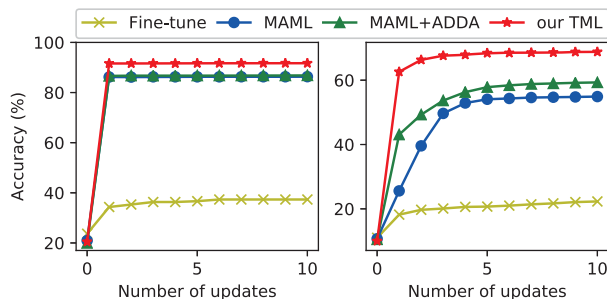


Figure 3: Adaptation speed comparison. Left: MNIST  $\Rightarrow$  USPS, 5-way 1-shot. Right: SVHN  $\Rightarrow$  USPS, 10-way 1-shot. The fine-tuning baseline updates for 300 steps in total, *i.e.*, one step amounts to 30 actual steps.

baseline which has full access to samples in the target domain. The results in Table 1 show that domain shift brings a significant performance drop to MAML, demonstrating the necessity of addressing domain-shift in few-shot learning. TML can reduce the performance gap moderately. TML is effective at minimizing the domain divergence without harming performance in source domain. To show this, we also evaluate its few-shot classification performance on the source domain. The results in Table 2 demonstrate that TML performs as well as MAML in the source domain, proving TML can learn domain-invariant representations benefiting applications in both source and target domains.

Moreover, fast adaptation is important in practical applications. Therefore, we evaluate adaptation speed (in terms of adaptation steps) of different methods. Given a new task from the target domain, each model is updated for several steps (*e.g.*, 10) with gradient descent using the task-provided few-shot training data. We plot the few-shot classification performance against model updating steps for the naive fine-tuning model, MAML,

Table 3: Few-shot classification results for office images. The left-most column shows the cross-domain direction, where **A**, **D**, **W** denote Amazon, DSLR, Webcam respectively. The results are reported in terms of classification accuracy (in %), averaged over 500 tasks. “M + A” denotes the MAML+ADDA baseline.

Direction	5-way 1-shot				5-way 5-shot			
	Fine-tune	MAML	M + A	TML	Fine-tune	MAML	M + A	TML
<b>A ⇒ D</b>	42.44	45.90	49.43	<b>54.50</b>	74.12	71.74	71.63	<b>80.72</b>
<b>A ⇒ W</b>	41.67	46.43	48.50	<b>53.83</b>	69.71	70.70	69.70	<b>78.59</b>
<b>D ⇒ W</b>	46.48	72.77	72.20	<b>76.70</b>	74.02	77.51	78.78	<b>88.97</b>
<b>W ⇒ D</b>	49.28	80.20	80.93	<b>82.83</b>	78.91	90.48	89.55	<b>91.25</b>

Direction	10-way 1-shot				10-way 5-shot			
	Fine-tune	MAML	M + A	TML	Fine-tune	MAML	M + A	TML
<b>A ⇒ D</b>	32.62	32.90	35.05	<b>41.92</b>	64.37	60.20	60.04	<b>71.47</b>
<b>A ⇒ W</b>	31.47	32.72	35.02	<b>38.80</b>	60.39	60.13	61.36	<b>66.22</b>
<b>D ⇒ W</b>	34.78	51.43	50.33	<b>58.67</b>	65.01	80.08	80.22	<b>82.08</b>
<b>W ⇒ D</b>	38.66	58.58	57.07	<b>60.92</b>	69.09	81.02	81.17	<b>83.88</b>

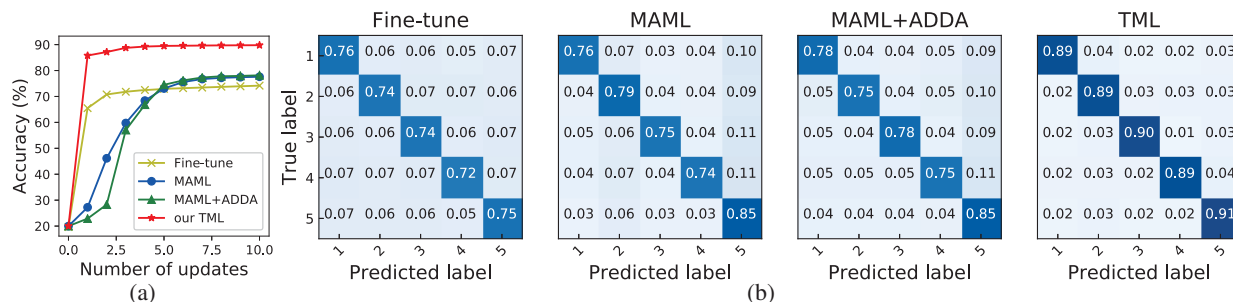


Figure 4: (a) Adaptation speed comparison on office datasets ( $D \Rightarrow W$ , 5-way 5-shot). (b) Corresponding few-shot classification confusion matrices on task-specific testing set.

MAML+ADDA and TML in Fig. 3. TML presents much faster adaptation than others and achieves the best performance. Notably, TML only needs one step adaptation to achieve better performance than all the baselines.

## 4.2 Results on Office Datasets

We then evaluate TML on the more challenging office dataset for four few-shot settings and three cross-domain directions: Amazon  $\Rightarrow$  DSLR, Amazon  $\Rightarrow$  Webcam and Webcam  $\Rightarrow$  DSLR. Since Amazon provides sufficient training examples, we always take it as source domain. The experimental results are shown in Table 1. Similar to the digit images, TML brings improvement up to 10.72% over MAML and MAML+ADDA, and performs significantly better than the fine-tuning baseline.

We also analyze the adaptation speed of different approaches to multiple 5-way 5-shot learning tasks from Dslr to Webcam domains, which is visualized in Fig. 4a. TML adapts significantly faster than all the baselines, demonstrating the meta-adaptation is effective for augmenting model adaptation ability. For understanding few-shot classification performance more transparently, we also plot classification confusion matrix for all the approaches in Fig. 4b. TML provides more accurate clas-

sification for all the 5 categories than baselines, showing its effectiveness in overcoming challenges from both domain-shift and limited training examples. In contrast, MAML+ADDA degrades the performance of MAML for the second category, demonstrating blind domain adaptation may confuse some categories and harm the few-shot learning performance.

## 5 Conclusion

This work introduced the new cross-domain meta learning problems challenged by insufficiency of training examples and varying characteristics of tasks. We developed the first transferable meta learning (TML) algorithm which substantially extends existing meta learning algorithms to solve new tasks in different domains and relieve the issues brought by insufficient training tasks.

## Acknowledgement

This work was partially supported by NUS startup R-263-000-C08-133, MOE Tier-I R-263-000-C21-112, NUS IDS R-263-000-C67-646, ECRA R-263-000-C87-133 and MOE Tier-II R-263-000-D17-112.

## References

- Andrychowicz, Marcin, Denil, Misha, Gomez, Sergio, Hoffman, Matthew W, Pfau, David, Schaul, Tom, and de Freitas, Nando. Learning to learn by gradient descent by gradient descent. In *NIPS*, 2016.
- Fei-Fei, Li, Fergus, Rob, and Perona, Pietro. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006.
- Finn, Chelsea, Abbeel, Pieter, and Levine, Sergey. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.
- Ganin, Yaroslav, Ustinova, Evgeniya, Ajakan, Hana, Germain, Pascal, Larochelle, Hugo, Laviolette, François, Marchand, Mario, and Lempitsky, Victor. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.
- Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and Bengio, Yoshua. Generative adversarial nets. In *NIPS*, 2014.
- Graves, Alex, Wayne, Greg, and Danihelka, Ivo. Neural Turing machines. *arXiv preprint arXiv:1410.5401*, 2014.
- Hariharan, Bharath and Girshick, Ross. Low-shot visual recognition by shrinking and hallucinating features. *arXiv preprint arXiv:1606.02819*, 2016.
- Hoffman, Judy, Tzeng, Eric, Donahue, Jeff, Jia, Yangqing, Saenko, Kate, and Darrell, Trevor. One-shot adaptation of supervised deep convolutional models. *arXiv preprint arXiv:1312.6204*, 2013.
- Ioffe, Sergey and Szegedy, Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.
- Kingma, Diederik and Ba, Jimmy. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- Koch, Gregory, Zemel, Richard, and Salakhutdinov, Ruslan. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, volume 2, 2015.
- Lake, Brenden M, Salakhutdinov, Ruslan R, and Tenenbaum, Josh. One-shot learning by inverting a compositional causal process. In *NIPS*, 2013.
- Le Cun, Yann, Jackel, LD, Boser, B, Denker, JS, Graf, HP, Guyon, I, Henderson, D, Howard, RE, and Hubbard, W. Handwritten digit recognition: Applications of neural network chips and automatic learning. *IEEE Communications Magazine*, 27(11):41–46, 1989.
- LeCun, Yann, Bottou, Léon, Bengio, Yoshua, and Haffner, Patrick. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- Liu, Ming-Yu and Tuzel, Oncel. Coupled generative adversarial networks. In *NIPS*, 2016.
- Long, Mingsheng, Cao, Yue, Wang, Jianmin, and Jordan, Michael I. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791*, 2015.
- Luo, Zelun, Zou, Yuliang, Hoffman, Judy, and Fei-Fei, Li F. Label efficient learning of transferable representations across domains and tasks. In *NIPS*, 2017.
- Mishra, Nikhil, Rohaninejad, Mostafa, Chen, Xi, and Abbeel, Pieter. Meta-learning with temporal convolutions. *arXiv preprint arXiv:1707.03141*, 2017.
- Motiian, Saeid, Jones, Quinn, Iranmanesh, Seyed, and Doretto, Gianfranco. Few-shot adversarial domain adaptation. In *NIPS*, 2017a.
- Motiian, Saeid, Piccirilli, Marco, Adjeroh, Donald A, and Doretto, Gianfranco. Unified deep supervised domain adaptation and generalization. In *ICCV*, 2017b.
- Munkhdalai, Tsendsuren and Yu, Hong. Meta networks. *arXiv preprint arXiv:1703.00837*, 2017.
- Netzer, Yuval, Wang, Tao, Coates, Adam, Bissacco, Alessandro, Wu, Bo, and Ng, Andrew Y. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, pp. 5, 2011.
- Ravi, Sachin and Larochelle, Hugo. Optimization as a model for few-shot learning. 2016.
- Saenko, Kate, Kulis, Brian, Fritz, Mario, and Darrell, Trevor. Adapting visual category models to new domains. *ECCV*, 2010.
- Santoro, Adam, Bartunov, Sergey, Botvinick, Matthew, Wierstra, Daan, and Lillicrap, Timothy. Meta-learning with memory-augmented neural networks. In *ICML*, 2016a.
- Santoro, Adam, Bartunov, Sergey, Botvinick, Matthew, Wierstra, Daan, and Lillicrap, Timothy. One-shot learning with memory-augmented neural networks. *arXiv preprint arXiv:1605.06065*, 2016b.
- Snell, Jake, Swersky, Kevin, and Zemel, Richard. Prototypical networks for few-shot learning. In *NIPS*, 2017.
- Sun, Baochen and Saenko, Kate. Deep coral: Correlation alignment for deep domain adaptation. In *European Conference on Computer Vision*, pp. 443–450. Springer, 2016.

- Tzeng, Eric, Hoffman, Judy, Zhang, Ning, Saenko, Kate, and Darrell, Trevor. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.
- Tzeng, Eric, Hoffman, Judy, Darrell, Trevor, and Saenko, Kate. Simultaneous deep transfer across domains and tasks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4068–4076, 2015.
- Tzeng, Eric, Hoffman, Judy, Saenko, Kate, and Darrell, Trevor. Adversarial discriminative domain adaptation. *arXiv preprint arXiv:1702.05464*, 2017.
- Vinyals, Oriol, Blundell, Charles, Lillicrap, Tim, Wierstra, Daan, et al. Matching networks for one shot learning. In *NIPS*, 2016.
- Wang, Yu-Xiong and Hebert, Martial. Learning to learn: Model regression networks for easy small sample learning. In *ECCV*, 2016.

---

# Learning the Causal Structure of Copula Models with Latent Variables

---

**Ruifei Cui**  
Data Science  
Radboud University  
r.cui@science.ru.nl

**Perry Groot**  
Data Science  
Radboud University  
perry.groot@cs.ru.nl

**Moritz Schauer**  
Mathematical Institute  
University of Leiden  
m.r.schauer@math.leidenuniv.nl

**Tom Heskes**  
Data Science  
Radboud University  
t.heskes@science.ru.nl

## Abstract

A common goal in psychometrics, sociology, and econometrics is to uncover causal relations among latent variables representing hypothetical constructs that cannot be measured directly, such as attitude, intelligence, and motivation. Through measurement models, these constructs are typically linked to measurable indicators, e.g., responses to questionnaire items. This paper addresses the problem of causal structure learning among such latent variables and other observed variables. We propose the ‘Copula Factor PC’ algorithm as a novel two-step approach. It first draws samples of the underlying correlation matrix in a Gaussian copula factor model via a Gibbs sampler on rank-based data. These are then translated into an average correlation matrix and an effective sample size, which are taken as input to the standard PC algorithm for causal discovery in the second step. We prove the consistency of our ‘Copula Factor PC’ algorithm, and demonstrate that it outperforms the PC-MIMBuild algorithm and a greedy step-wise approach. We illustrate our method on a real-world data set about children with Attention Deficit Hyperactivity Disorder.

## 1 INTRODUCTION

Social scientists, psychologists, and many other scientists are usually interested in learning causal relations between latent variables that cannot be measured directly, e.g., attitude, intelligence, and motivation (see [15, 24], and Chapter 10 of [27] for more details). In order to get a grip on these latent concepts, one commonly-used strategy is to construct a measurement model for such a

latent variable, in the sense that domain experts design a set of measurable “items” or survey “questions” that are considered to be indicators of the latent variable. For instance, in the study of Attention Deficit Hyperactivity Disorder (ADHD), 18 questions are designed to measure three latent variables: inattention, hyperactivity, and impulsivity [29]. In some other cases where it is difficult to design a measurement model due to the absence of domain knowledge or for other reasons, there are some off-the-shelf algorithms, e.g., BPC [24] and FOFC [15], for learning the measurement models from indicator data. In this paper, we focus on inferring the causal structure among latent variables, assuming that the measurement models are given. We also allow interactions between these latent variables and other (explicit) variables, e.g., subject characteristics like gender and age. Another issue we consider is that there are diverse types of variables in most real-world data: the questionnaire data in a survey is typically ordinal, whereas other variables might be binary, or continuous.

In this paper, we use a Gaussian copula factor model (the formal definition is given in Section 3) to describe such situations, in which a factor can be connected to either one or more observed variables (indicators). Factors with multiple indicators are used to model latent variables corresponding to psychological traits, such as attitude and intelligence. The copula model provides a good way of analyzing diverse types of variables, where the associations between variables are parameterized separately from their marginal distributions [13].

We propose the ‘Copula Factor PC’ algorithm for estimating the causal structure among factors of a Gaussian copula factor model, which is based on a two-step approach. The first step draws samples of the underlying correlation matrix, where the Gibbs sampler by [13] for Gaussian copula models is extended to Gaussian copula factor models by replacing the Wishart prior with the  $G$ -Wishart prior and adding a new strategy to sample latent factors. These samples are then translated into an aver-



age correlation matrix, and an effective sample size that is used to account for information loss incurred by discrete variables [9]. The second step takes the estimated correlation matrix and effective sample size as input to the standard PC algorithm [27] for causal discovery.

The rest of this paper is organized as follows. Section 2 reviews necessary knowledge and related work. Section 3 gives the definition of a Gaussian copula factor model. Section 4 describes our ‘Copula Factor PC’ algorithm, and introduces two alternative approaches: the PC-MIMBuild algorithm [24] and a greedy step-wise approach. Section 5 compares the ‘Copula Factor PC’ algorithm with the two alternative approaches on simulated data, and Section 6 gives an illustration on real-world data of ADHD patients. Section 7 concludes this paper and gives some discussion.

## 2 BACKGROUND

**Causal discovery** A graphical model is a graph  $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ , where the vertices  $\mathbf{V} = \{X_1, \dots, X_d\}$  correspond to random variables and the edges  $\mathbf{E}$  represent dependence structure among the variables. A graph is *directed* if it just contains directed edges and *undirected* if all edges are undirected. A graph that contains both directed and undirected edges is called a *partially directed graph*. Graphs without directed cycles (e.g.,  $X_i \rightarrow X_j \rightarrow X_i$ ) are *acyclic*. We refer to a graph as a *Directed Acyclic Graph* (DAG) if it is both directed and acyclic. If there is a directed edge  $X_i \rightarrow X_j$ ,  $X_i$  is called a parent of  $X_j$ . A distribution over a random vector  $\mathbf{X}$  with  $X_i \in \mathbf{V}$  is said to be Markov w.r.t. a DAG  $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ , if  $\mathbf{X}$  satisfies the *Causal Markov Condition*: each variable in the DAG  $\mathcal{G}$  is independent of its non-descendants given its parents, which is also implied by the so-called *d-separation* [20]. A distribution is *faithful* w.r.t. a DAG  $\mathcal{G}$  if there are no conditional independencies in the distribution that are not encoded by the Causal Markov Condition. If a distribution is both Markov and faithful w.r.t. a DAG  $\mathcal{G}$ , the DAG is called a *perfect map* of the distribution.

Several DAGs may, via *d-separation*, correspond to the same set of conditional independencies. The set of such DAGs is called a Markov equivalence class, which can be represented by a *completed partially directed acyclic graph* (CPDAG). Arcs in a CPDAG suggest a cause-effect relationship between pairs of variables since the same arc appears in all members of the CPDAG. An undirected edge  $X_i - X_j$  in a CPDAG implies that some of its members contain an arc  $X_i \rightarrow X_j$  while others contain an arc  $X_j \rightarrow X_i$ . Causal discovery aims to learn the Markov equivalence class of the underlying DAG from observations.

**The PC algorithm** The PC algorithm [27], a reference algorithm for causal discovery, consists of two stages: adjacency search and orientation. The adjacency search starts with a fully connected undirected graph, and then recursively removes the edges according to conditional independence decisions, yielding the skeleton and separation sets. In the orientation stage, we first orient the unshielded triples according to the separation sets, and then orient as many of the remaining undirected edges as possible by applying the orientation rules repeatedly.

A key part of the procedure is to test for conditional independencies. When a random vector  $\mathbf{X} \sim \mathcal{N}(0, C)$ , the PC algorithm considers the so-called partial correlation, denoted by  $\rho_{uv|S}$ , which can be obtained by the correlation matrix  $C$  [1]. Given observations of  $\mathbf{X}$  and significance level  $\alpha$ , classical decision theory yields

$$X_u \perp\!\!\!\perp X_v | \mathbf{X}_S \Leftrightarrow \sqrt{n - |S| - 3} \left| \frac{1}{2} \log \left( \frac{1 + \hat{\rho}_{uv|S}}{1 - \hat{\rho}_{uv|S}} \right) \right| \leq \Phi^{-1}(1 - \alpha/2), \quad (1)$$

where  $u \neq v$ ,  $S \subseteq \{1, \dots, d\} \setminus \{u, v\}$  and  $\Phi$  is the cumulative distribution function of the standard Gaussian. Hence, the PC algorithm requires the correlation matrix  $C$  (to compute partial correlations  $\rho_{uv|S}$ ) and the sample size  $n$  as input. Uniform consistency of the PC algorithm for Gaussian data is shown under some relatively mild assumptions on the sparsity of the true underlying structure [14].

Harris & Drton [11] use rank correlations, typically Spearman’s  $\rho$  and Kendall’s  $\tau$ , to replace the Pearson correlation, which extends the PC algorithm to the so-called nonparanormal models. The resulting ‘Rank PC’ algorithm performs as well as the PC algorithm using Pearson correlations on Gaussian data, yet much better on nonparanormal data. The PC algorithms using both Pearson and rank correlations require all univariate marginal distributions to be continuous. Cui et al. [9] extend the PC algorithm to mixed discrete and continuous data assumed to be drawn from a Gaussian copula model, where each observed variable is assumed to be induced by a latent Gaussian variable and the dependence between observed variables is determined by the correlation matrix of the latent variables. The resulting ‘Copula PC’ algorithm works well for mixed data, but requires each latent variable to have only a single indicator. Silva et al. [24] propose the PC-MIMBuild algorithm, which allows a latent variable to have multiple indicators, but it is limited to continuous observations and assumes that each latent variable has at least two indicators.

In this paper, we aim to generalize the PC algorithm to handle latent variables having one or more indicators and observations being either discrete or continuous.

### 3 GAUSSIAN COPULA FACTOR MODEL

#### Definition 1 (Gaussian Copula Factor Model).

Consider a latent random (factor) vector  $\boldsymbol{\eta} = (\eta_1, \dots, \eta_k)^T$ , a response random vector  $\mathbf{Z} = (Z_1, \dots, Z_p)^T$  and an observed random vector  $\mathbf{Y} = (Y_1, \dots, Y_p)^T$ , satisfying

$$\boldsymbol{\eta} \sim \mathcal{N}(0, C), \quad (2)$$

$$\mathbf{Z} = \Lambda \boldsymbol{\eta} + \boldsymbol{\epsilon}, \quad (3)$$

$$Y_j = F_j^{-1}(\Phi[Z_j/\sigma(Z_j)]), \forall j = 1, \dots, p, \quad (4)$$

with  $\Lambda = (\lambda_{ij})$  a  $p \times k$  matrix of factor loadings ( $k \leq p$ ),  $\boldsymbol{\epsilon} \sim \mathcal{N}(0, D)$  Gaussian noise with  $D = \text{diag}(\sigma_1^2, \dots, \sigma_p^2)$ ,  $\sigma(Z_j)$  the standard deviation of  $Z_j$ , and  $F_j^{-1}(t) = \inf\{x : F_j(x) \geq t\}$  the pseudo-inverse of a cumulative distribution function  $F_j$ . This model is called a *Gaussian Copula Factor Model* with correlation matrix  $C$ , factor loadings  $\Lambda$ , and univariate margins  $F_j$ .

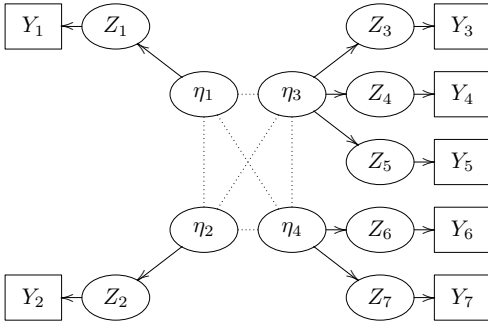


Figure 1: Gaussian copula factor model.

The model is also defined in [18], but the authors restrict the factors to be independent of each other while we allow for their interactions. An example of the model is shown in Figure 1. Our model is a combination of a Gaussian factor model (from  $\boldsymbol{\eta}$  to  $\mathbf{Z}$ ) and a Gaussian copula model (from  $\mathbf{Z}$  to  $\mathbf{Y}$ ). In the special case of a factor having a single response (thus a single observed variable), e.g.,  $\eta_1 \rightarrow Z_1 \rightarrow Y_1$ , it reduces to a Gaussian copula model where we set  $\lambda_{11} = 1$  and  $\epsilon_1 = 0$ , thus  $Y_1 = F_1^{-1}(\Phi[\eta_1])$ .

In the typical design for questionnaires, one tries to get a grip on a latent concept through a particular set of well-designed questions [16, 4], which implies that a factor (latent concept) in our model is connected to multiple indicators (questions) while an indicator is only used to measure a single factor, as shown in Figure 1. This kind of measurement model is called a pure measurement model (Definition 2 of [23]). Throughout this paper, we assume that all measurement models are given and pure, which makes that there is only a single non-zero entry in

each row of the factor loadings matrix  $\Lambda$ . This inductive bias about the sparsity pattern of  $\Lambda$  is fully motivated by the typical design of a measurement model.

In what follows, we transform the Gaussian copula factor model into an equivalent model, which we will use for inference in the next section. We consider an integrated random vector  $\mathbf{X} = (\mathbf{Z}^T, \boldsymbol{\eta}^T)^T$ , which is still multivariate Gaussian, and obtain its covariance matrix

$$\Sigma = \begin{bmatrix} \Lambda C \Lambda^T + D & \Lambda C \\ C \Lambda^T & C \end{bmatrix}, \quad (5)$$

and precision matrix

$$\Omega = \Sigma^{-1} = \begin{bmatrix} D^{-1} & -D^{-1}\Lambda \\ -\Lambda^T D^{-1} & C^{-1} + \Lambda^T D^{-1}\Lambda \end{bmatrix}. \quad (6)$$

Since  $D$  is diagonal and  $\Lambda$  only has one non-zero entry per row,  $\Omega$  contains many intrinsic zeros. The sparsity pattern of such  $\Omega = (\omega_{ij})$  can be represented by an undirected graph  $G = (\mathbf{V}, \mathbf{E})$ , where  $(i, j) \notin \mathbf{E}$  whenever  $\omega_{ij} = 0$  by construction. Then, a Gaussian copula factor model can be transformed into an equivalent model controlled by a single precision matrix  $\Omega$ , which in turn is constrained by  $G$ , i.e.,  $P(\mathbf{X}|C, \Lambda, D) = P(\mathbf{X}|\Omega_G)$ .

**Definition 2 ( $G$ -Wishart Distribution [22]).** Given an undirected graph  $G = (\mathbf{V}, \mathbf{E})$ , a zero-constrained random matrix  $\Omega$  has a  $G$ -Wishart distribution, if its density is

$$p(\Omega|G) = \frac{|\Omega|^{(\nu-2)/2}}{I_G(\nu, \Psi)} \exp \left[ -\frac{1}{2} \text{tr}(\Psi \Omega) \right] \mathbb{1}_{\Omega \in M^+(G)},$$

with  $M^+(G)$  the space of symmetric positive definite matrices with off-diagonal elements  $\omega_{ij} = 0$  whenever  $(i, j) \notin \mathbf{E}$ ,  $\nu$  the number of degrees of freedom,  $\Psi$  a scale matrix,  $I_G(\nu, \Psi)$  the normalizing constant, and  $\mathbb{1}$  the indicator function.

The  $G$ -Wishart distribution is the conjugate prior of precision matrices  $\Omega$  that are constrained by a graph  $G$  [22]. That is, given the  $G$ -Wishart prior, i.e.,  $P(\Omega|G) = \mathcal{W}_G(\nu_0, \Psi_0)$  and data  $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n)^T$  drawn from  $\mathcal{N}(0, \Omega^{-1})$ , the posterior for  $\Omega$  is another  $G$ -Wishart distribution:

$$P(\Omega|G, \mathbf{X}) = \mathcal{W}_G(\nu_0 + n, \Psi_0 + \mathbf{X}^T \mathbf{X}).$$

When the graph  $G$  is fully connected, the  $G$ -Wishart distribution reduces to a Wishart distribution [17]. Placing a  $G$ -Wishart prior on  $\Omega$  is equivalent to placing an inverse-Wishart on  $C$ , a product of multivariate normals on  $\Lambda$ , and an inverse-gamma on the diagonal elements of  $D$ . With a diagonal scale matrix  $\Psi_0$  and the number of degrees of freedom  $\nu_0$  equal to the number of factors plus one, the implied marginal densities between any pair of factors are uniformly distributed in the interval  $[-1, 1]$  [3].

## 4 METHODS

In this section, we propose a Bayesian inference method for Gaussian copula factor models, based on which we derive our ‘Copula Factor PC’ algorithm. Then, we introduce two alternative approaches.

### 4.1 INFERENCE FOR GAUSSIAN COPULA FACTOR MODEL

For a Gaussian copula model, Hoff [13] proposed a likelihood that only concerns the ranks among observations, which is derived as follows. Since the transformation  $Y_j = F_j^{-1}(\Phi[Z_j])$  is non-decreasing, observing  $\mathbf{y}_j = (y_{1,j}, \dots, y_{n,j})^T$  implies a partial ordering on  $\mathbf{z}_j = (z_{1,j}, \dots, z_{n,j})^T$ , namely,  $\mathbf{z}_j$  must lie in the space restricted by  $\mathbf{y}_j$ :

$$\mathcal{D}(\mathbf{y}_j) = \{\mathbf{z}_j \in \mathbb{R}^n : y_{i,j} < y_{k,j} \Rightarrow z_{i,j} < z_{k,j}\}.$$

Therefore, observing  $\mathbf{Y}$  suggests that  $\mathbf{Z}$  must be in

$$\mathcal{D}(\mathbf{Y}) = \{\mathbf{Z} \in \mathbb{R}^{n \times p} : \mathbf{z}_j \in \mathcal{D}(\mathbf{y}_j), \forall j = 1, \dots, p\}.$$

Taking the occurrence of this event as the data, one can compute the following likelihood

$$\begin{aligned} P(\mathbf{Z} \in \mathcal{D}(\mathbf{Y}) | S, F_1, \dots, F_p) &= \int_{\mathcal{D}(\mathbf{Y})} p(\mathbf{Z} | S) d\mathbf{Z} \\ &= P(\mathbf{Z} \in \mathcal{D}(\mathbf{Y}) | S), \end{aligned}$$

where  $S$  is the correlation matrix over  $\mathbf{Z}$ .

Following the same argumentation, the likelihood in our Gaussian copula factor model reads

$$P(\mathbf{Z} \in \mathcal{D}(\mathbf{Y}) | \boldsymbol{\eta}, \Omega, F_1, \dots, F_p) = P(\mathbf{Z} \in \mathcal{D}(\mathbf{Y}) | \boldsymbol{\eta}, \Omega),$$

which is independent of the margins  $F_j$ .

For the Gaussian copula factor model, inference for the precision matrix  $\Omega$  of the vector  $\mathbf{X} = (\mathbf{Z}^T, \boldsymbol{\eta}^T)^T$  can now proceed via construction of a Markov chain having its stationary distribution equal to  $P(\mathbf{Z}, \boldsymbol{\eta}, \Omega | \mathbf{Z} \in \mathcal{D}(\mathbf{Y}), G)$ , where we ignore the values for  $\boldsymbol{\eta}$  and  $\mathbf{Z}$  in our samples. The prior graph  $G$  is uniquely determined by the sparsity pattern of the loading matrix  $\Lambda = (\lambda_{ij})$  and the residual matrix  $D$  (see Equation 6), which in turn is uniquely decided by the pure measurement models. The Markov chain can be constructed by iterating the following three steps:

1. **Sample  $\mathbf{Z}$ :**  $\mathbf{Z} \sim P(\mathbf{Z} | \boldsymbol{\eta}, \mathbf{Z} \in \mathcal{D}(\mathbf{Y}), \Omega)$ ;  
Since each coordinate  $Z_j$  directly depends on only one factor, i.e.,  $\eta_q$  such that  $\lambda_{jq} \neq 0$ , we can sample each of them independently through  $Z_j \sim P(Z_j | \eta_q, \mathbf{z}_j \in \mathcal{D}(\mathbf{y}_j), \Omega)$ .

---

### Algorithm 1 Gibbs sampler for Gaussian copula factor model

---

**Require:** Measurement models (decide sparsity of  $\Lambda$  and thus  $G$ ), and indicator data  $\mathbf{Y}$ .

- 1: **Step 1:** sample  $\mathbf{Z} \sim P(\mathbf{Z} | \boldsymbol{\eta}, \mathbf{Z} \in \mathcal{D}(\mathbf{Y}), \Omega)$ .
  - 2: **for**  $j \in \{1, \dots, p\}$  **do**
  - 3:    $q =$  factor index of  $Z_j$
  - 4:    $a = \Sigma_{[j, q+p]} / \Sigma_{[q+p, q+p]}$
  - 5:    $\sigma_j^2 = \Sigma_{[j, j]} - a \Sigma_{[q+p, j]}$
  - 6:   **for**  $\mathbf{y} \in \text{unique}\{y_{1,j}, \dots, y_{n,j}\}$  **do**
  - 7:      $z_l = \max\{z_{i,j} : y_{i,j} < \mathbf{y}\}$
  - 8:      $z_u = \min\{z_{i,j} : \mathbf{y} < y_{i,j}\}$
  - 9:     **for**  $i$  such that  $y_{i,j} = \mathbf{y}$  **do**
  - 10:        $\mu_{i,j} = \boldsymbol{\eta}_{[i,q]} \times a$
  - 11:        $u_{i,j} \sim \mathcal{U}(\Phi[\frac{z_l - \mu_{i,j}}{\sigma_j}], \Phi[\frac{z_u - \mu_{i,j}}{\sigma_j}])$
  - 12:        $z_{i,j} = \mu_{i,j} + \sigma_j \times \Phi^{-1}(u_{i,j})$
  - 13:     **end for**
  - 14:   **end for**
  - 15: **end for**
  - 16: **Step 2:** sample  $\boldsymbol{\eta} \sim P(\boldsymbol{\eta} | \mathbf{Z}, \Omega)$ .
  - 17:  $A = \Sigma_{[\boldsymbol{\eta}, \mathbf{Z}]} \Sigma_{[\mathbf{Z}, \mathbf{Z}]}^{-1}$
  - 18:  $B = \Sigma_{[\boldsymbol{\eta}, \boldsymbol{\eta}]} - A \Sigma_{[\mathbf{Z}, \boldsymbol{\eta}]}$
  - 19: **for**  $i \in \{1, \dots, n\}$  **do**
  - 20:    $\boldsymbol{\mu}_i = (\mathbf{Z}_{[i,:]} A^T)^T$
  - 21:    $\boldsymbol{\eta}_{[i,:]} \sim \mathcal{N}(\boldsymbol{\mu}_i, B)$
  - 22: **end for**
  - 23:  $\boldsymbol{\eta}_{[:,j]} = \boldsymbol{\eta}_{[:,j]} \times \text{sign}(\text{Cov}[\boldsymbol{\eta}_{[:,j]}, \mathbf{Z}_{[:,f(j)}])$ ,  $\forall j$ , where  $f(j)$  is the index of the first indicator of  $\eta_j$ .
  - 24: **Step 3:** sample  $\Omega \sim P(\Omega | \mathbf{Z}, \boldsymbol{\eta}, G)$ .
  - 25:  $\mathbf{X} = (\mathbf{Z}, \boldsymbol{\eta})$
  - 26:  $\Omega \sim \mathcal{W}_G(\nu_0 + n, \Psi_0 + \mathbf{X}^T \mathbf{X})$
  - 27:  $\Sigma = \Omega^{-1}$
  - 28:  $\Sigma_{ij} = \Sigma_{ij} / \sqrt{\Sigma_{ii} \Sigma_{jj}}$ ,  $\forall i, j$
- 

2. **Sample  $\boldsymbol{\eta}$ :**  $\boldsymbol{\eta} \sim P(\boldsymbol{\eta} | \mathbf{Z}, \Omega)$ ;
3. **Sample  $\Omega$ :**  $\Omega \sim P(\Omega | \mathbf{Z}, \boldsymbol{\eta}, G)$ .

A Gibbs sampler that implements the Markov chain is summarized in Algorithm 1.

**Identifiability of  $C$ :** Without additional constraints, the correlation matrix  $C$  over factors is non-identifiable [2]. More precisely, given a decomposable covariance matrix  $S = \Lambda C \Lambda^T + D$ , we can always replace  $\Lambda$  with  $\Lambda U$  and  $C$  with  $U^{-1} C U^{-T}$  to obtain an equivalent decomposition  $S = (\Lambda U)(U^{-1} C U^{-T})(U^T \Lambda^T) + D$ , where  $U$  is a  $k \times k$  invertible matrix. Since  $\Lambda$  only has one non-zero entry per row in our model,  $U$  can only be diagonal to ensure that  $\Lambda U$  has the same sparsity pattern as  $\Lambda$  (see Lemma 3 in Supplement). Thus, from the same  $S$ , we get a class of solutions for  $C$ , i.e.,  $U^{-1} C U^{-1}$ , where

$U$  can be any invertible diagonal matrix. However, we find that all members in this class encode the same set of conditional independencies (see Lemma 4 in Supplement), and therefore imply the same causal structure [27]. Hence, any solution in this class is appropriate for finding the underlying causal structure among latent variables.

In order to get a unique solution for  $C$ , we impose two sufficient identifying conditions: 1) restrict  $C$  to be a correlation matrix; 2) force the first non-zero entry in each column of  $\Lambda$  to be positive (see Lemma 5 in Supplement). Condition 1 is implemented via line 28 in Algorithm 1. As for the second condition, we force the covariance between a factor and its first indicator to be positive (line 23), which is equivalent to Condition 2. One could also choose one’s favorite constraints for identifying  $C$ , as long as the unique solution belongs to the class  $U^{-1}CU^{-1}$ .

## 4.2 COPULA FACTOR PC ALGORITHM

By iterating the steps in Algorithm 1 and extracting the submatrix over  $\eta$ , we can draw samples of  $C$ , denoted by  $\{C^{(1)}, \dots, C^{(m)}\}$ . The mean over all the samples is a natural estimate of the underlying correlation matrix  $\hat{C}$ , i.e.,  $\hat{C} = \frac{1}{m} \sum_{i=1}^m C^{(i)}$ . As for the effective sample size  $\hat{n}$ , we build upon the idea in [9], that is, taking the posterior distribution’s degrees of freedom  $\nu$  as an approximation to  $\hat{n}$ . Theorem 1 (the proof is provided in the Supplement) suggests a procedure to estimate the degrees of freedom of a  $G$ -Wishart distribution.

**Theorem 1.** *Consider a random matrix  $\Omega$  following a  $G$ -Wishart distribution with graph  $G = (\mathbf{V}, \mathbf{E})$  as well as parameters  $\nu$  and  $\Psi$ , i.e.,  $\Omega \sim \mathcal{W}_G(\nu, \Psi)$ . Let  $\Sigma = \Omega^{-1}$  and  $\tilde{\Sigma}$  be the normalized matrix of  $\Sigma$ , i.e.,  $\tilde{\Sigma}_{ij} = \Sigma_{ij} / \sqrt{\Sigma_{ii}\Sigma_{jj}}$ . Then, for large  $\nu$ , we have*

$$\text{Var} [\tilde{\Sigma}_{ij}] \approx \frac{(1 - (\mathbb{E} [\tilde{\Sigma}_{ij}])^2)^2}{\nu}, \quad (7)$$

for off-diagonal elements  $\tilde{\Sigma}_{ij}$  whenever  $(i, j) \in \mathbf{E}$ .

From the theorem, we have that all off-diagonal elements of the latent correlation matrix satisfy Equation (7), because the prior subgraph over latent factors is fully connected. Therefore, we estimate  $\hat{n}$  as follows

$$\hat{n} = \frac{1}{k(k-1)} \sum_{i \neq j} \nu_{ij}, \quad \text{where } \nu_{ij} = \frac{(1 - (\mathbb{E} [C_{ij}])^2)^2}{\text{Var} [C_{ij}]}.$$

The ‘Copula Factor PC’ (CFPC) algorithm arises when taking the estimated correlation matrix  $\hat{C}$  and the effective sample size  $\hat{n}$  (to replace the  $n$  in Equation 1) as the

input to the standard PC algorithm.<sup>1</sup> The CFPC algorithm is consistent, as shown in Theorem 2 (see proof in the Supplement).

### Theorem 2 (Consistency of the CFPC algorithm).

Let  $\mathbf{Y}_n = (\mathbf{y}_1, \dots, \mathbf{y}_n)^T$  be independent observations drawn from a Gaussian copula factor model. If 1) the measurement model per factor is known and pure; and 2) the distribution over factors is faithful to a DAG  $\mathcal{G}$ , then

$$\lim_{n \rightarrow \infty} P(\hat{\mathcal{M}}_n(\mathcal{G}) = \mathcal{M}(\mathcal{G})) = 1,$$

where  $\hat{\mathcal{M}}_n(\mathcal{G})$  is the output of the CFPC algorithm and  $\mathcal{M}(\mathcal{G})$  is the Markov equivalent class of the true underlying DAG  $\mathcal{G}$ .

## 4.3 ALTERNATIVE APPROACHES

**The PC-MIMBuild algorithm** The original PC-MIMBuild algorithm only works for continuous data. Here, we extend it to mixed cases by learning the correlation matrix of response variables via the Gibbs sampler by [13] and taking it as input to the original PC-MIMBuild. We further generalize the PC-MIMBuild algorithm to handle latent factors with just a single indicator, by replacing the conditional independence testing method designed only for factors with at least two indicators (Theorem 19 in [24]) with a test based on partial correlation. See Supplement B for more details.

**A greedy step-wise approach** This approach first extracts the measurement model of a factor with multiple indicators, e.g., the subpart of Figure 1 consisting of the variables  $\{\eta_3, Z_3, Z_4, Z_5, Y_3, Y_4, Y_5\}$ . Then, it uses off-the-shelf techniques [10] to fit such a model and obtain pseudo-data of the factor (factor scores). Using pseudo-data for factors with multiple indicators together with real data for factors with a single indicator, the ‘Copula PC’ algorithm is next applied for causal discovery. We refer to this approach as the greedy step-wise PC algorithm, whose pseudo-code is written out step by step in the Supplement C. One disadvantage of this approach is that it can overestimate the effective sample size when treating the pseudo-data at the same footing as real data. This might incur many false positives, as we will indeed observe in the experiment section.

## 5 SIMULATION STUDY

In this section, we compare our ‘Copula Factor PC’ algorithm (CFPC) with the PC-MIMBuild algorithm (MBPC) and the greedy step-wise PC algorithm (GSPC)

<sup>1</sup>The R code is publicly available in <https://github.com/cuiruifei/CopulaFactorModel>.

on simulated data. Kalisch & Buhlmann [14] provide a procedure to generate random DAGs and simulate normally distributed samples that are faithful to them. It first generates a  $k \times k$  adjacency matrix  $A$  representing a random DAG: 1) generate a  $k \times k$  zero matrix, 2) randomly set entries in the lower-triangle area to be *one* with probability  $s$  (measuring the sparseness), 3) change the *ones* to be random weights in the interval  $[0.1, 1]$ . Given the adjacency matrix  $A$ , values of a random vector  $\boldsymbol{\eta}$  are drawn recursively via

$$\eta_i = \sum_{k < i} A_{ik} \eta_k + \epsilon_i,$$

with each  $\epsilon_i \sim \mathcal{N}(0, 1)$ . Following this procedure, we simulate the factors of a Gaussian copula factor model, i.e., the  $\boldsymbol{\eta}$  in Equation (2). Then, the edge weights from factors to response variables (non-zero elements of  $\Lambda$  in Equation 3) are uniformly drawn from the interval  $[0.1, 1]$ . We next generate response variables using Equation (3) together with standard Gaussian noise. After discretizing some response variables, we obtain data following a Gaussian copula factor distribution.

Three metrics are used to evaluate the algorithms: the true and false positive rate (TPR and FPR) for assessing the skeleton, and the structural Hamming distance (SHD), counting the number of edge insertions, deletions, and flips to transfer the estimated CPDAG into the correct CPDAG [28], for assessing the CPDAG. A higher TPR, a lower FPR, and a smaller SHD imply better performance. We set the significance level in the PC algorithm to  $\alpha = 0.01$  (experiments with other values done suggest the same conclusion) and the sparseness parameter in generating DAGs to  $s = 2/(k - 1)$ , such that the average neighbors of each node is 2 [14]. For the Gibbs sampler, the first 500 samples (burn-in) are discarded and the next 500 samples are stored. We test the algorithms for different numbers of factors  $k \in \{4, 10\}$ , and sample sizes  $n \in \{500, 1000, 2000\}$ .

**Evaluation on Gaussian data** We first consider the case where the observed data are Gaussian and all factors have multiple indicators, since this matches the assumptions of the original PC-MIMBuild algorithm. The number of indicators per factor is randomly chosen from 3 to 10, to mimic typical real-world datasets [25, 29].

Figure 2 shows the results, providing the mean of TPR, FPR, and SHD over 100 repeated experiments with errorbars representing 95% confidence intervals. First, we see that CFPC performs clearly better than MBPC w.r.t. TPR despite an indistinguishable performance w.r.t. FPR (CFPC is slightly better than MBPC for  $k = 4$  while the other way around for  $k = 10$ ). Therefore, w.r.t. the overall metric SHD, CFPC significantly outperforms MBPC.

Our analysis is that MBPC tests for conditional independencies between all pairs of indicators and claims a dependence between factors even if just one of the pairs fails the test. This multiple testing approach, although elegant in theory, is difficult to make robust for largely varying numbers of indicators and sizes of the conditioning set. Second, while CFPC and GSPC report similar TPR scores, CFPC shows a clear advantage over GSPC w.r.t. FPR (thus a better SHD than GSPC), which becomes more prominent for a larger sample size. This is because the correlations between factors are estimated indirectly through their indicators, which makes the correlations less reliable than those estimated directly through the observed data. The effective sample size used in CFPC naturally incorporates the reduced reliability, whereas GSPC that still uses the original sample size rejects the null hypothesis of conditional independence more easily, resulting in more false positives.

**Evaluation on mixed data** We now focus on mixed data, in which two cases are considered: 1) all factors have multiple indicators; 2) half of the factors have multiple indicators and half only have a single indicator. When a factor has multiple indicators, the number of indicators per factor is randomly chosen from 3 to 10, and all such indicators are discretized into ordinal variables where the number of levels per variable is randomly chosen from 2 to 5. For factors with a single indicator, we discretize half into ordinal variables (from 2 to 5 levels) and keep the other half continuous.

Figures 3 and 4 summarize the experimental results, providing the mean of TPR, FPR, and SHD over 100 repeated experiments with 95% confidence intervals. From Figure 3a, we first see that GSPC is slightly better than CFPC w.r.t. TPR while GSPC and CFPC show a clear advantage over MBPC. Second, MBPC is rather sensitive to sample sizes in cases with only multiple indicators, where a small sample size incurs a poor performance. Figure 3b shows that CFPC is significantly better than GSPC w.r.t. FPR, which becomes more prominent in cases with only multiple indicators and larger sample sizes. This is because the effective sample size in CFPC better than GSPC represents the uncertainty in the partial correlation estimates and then incurs less false positives. CFPC also shows clear advantages over MBPC w.r.t. FPR when the number of factors is 4 ( $k = 4$ ), whereas MBPC works slightly better than CFPC when  $k = 10$ . As for the overall metric SHD shown in Figure 4, CFPC and GSPC perform clearly better than MBPC in almost all situations because of the bad performance of MBPC w.r.t. TPR. Meanwhile, we can see that CFPC generates a more accurate CPDAG than GSPC, in particular for larger sample sizes. This is because our proposed infer-

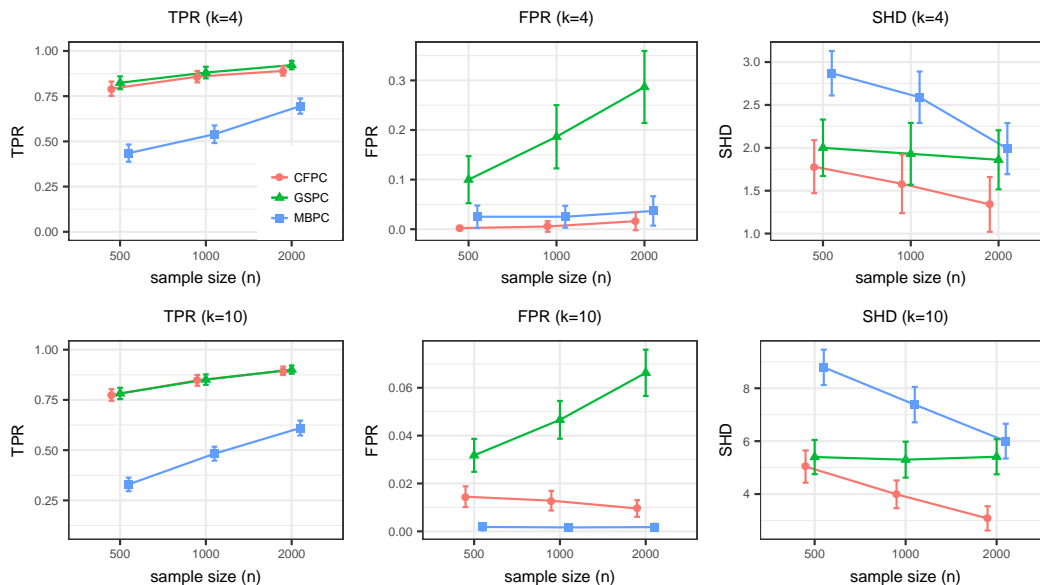


Figure 2: TPR, FPR, and SHD of CFPC, GSPC, and MBPC over different sample sizes when the data are fully Gaussian and all factors have multiple indicators, showing the mean over 100 experiments together with 95% confidence intervals. The two rows represent the results when the number of latent factors is 4 and 10 respectively.

ence procedure more accurately estimates the correlation matrix (not shown here) and, through the effective sample size, better represents the uncertainty in the correlation estimates than the greedy step-wise method. In a nutshell, our ‘Copula Factor PC’ algorithm, outperforms its two competitors in almost all situations.

## 6 REAL-WORLD APPLICATION

In this section, we give an illustration on a real-world dataset collected by [30] that includes 236 children with Attention Deficit Hyperactivity Disorder (ADHD) and 406 controls. We focus on 4 (explicit) variables that are related to ADHD symptoms: gender (Gen), Age, verbal IQ (VIQ), performance IQ (PIQ), as well as 18 questions that are designed to measure three latent concepts: inattention (Inatt), hyperactivity (Hyper), and impulsivity (Impul). The first 9 questions (Q1-Q9) are designed to measure ‘Inatt’, while the next 5 questions (Q10-Q14) and the last 4 questions (Q15-Q18) are used to measure ‘Hyper’ and ‘Impul’ respectively [29]. All the questions are ordinal with four levels: never (0), sometimes (1), frequently (2), and always (3).

Our task is to infer the causal structure among the 4 variables and 3 latent concepts from observations of the 4 variables and 18 questions. We run our ‘Copula Factor PC’ algorithm (using the order-independent version of the PC algorithm [7]) on this dataset and enforce the prior knowledge that no variables cause gender. The resulting

graph is shown in Figure 5, in which double arrows ‘ $\Rightarrow$ ’ represent the mapping from the three latent concepts to their corresponding questions (known) and other edges are those learned by our algorithm.

First, in the inferred model, we find that ‘Gen’ has a direct causal influence on ‘Inatt’. The finding is in the expected direction, namely males are at an increased risk of inattention, hyperactivity, and impulsivity problems. Meta-analyses in population-based samples suggested that males are 24 times more likely to meet full criteria for ADHD than females [31] and in clinically referred ADHD samples, the gender ratio was about 5:1 [19].

Second, the causal model implies that there is a significant causal path from inattention to hyperactivity (and subsequently to impulsivity), but not the other way around. It suggests that factors that cause inattention affect hyperactivity/impulsivity downstream of that, whereas those factors that lead to high hyperactivity/impulsivity do not necessarily lead to higher inattention. This causal path was previously observed in this sample and was also confirmed in two independent ADHD samples [26].

Third, the causal direction of the associations between verbal IQ and inattention as well as impulsivity is not clear from our model. Both interpretations seem reasonable. Previous studies suggest that ADHD is associated with lower (verbal) IQ, and particularly attention problems have been found to be strong predictors for lower

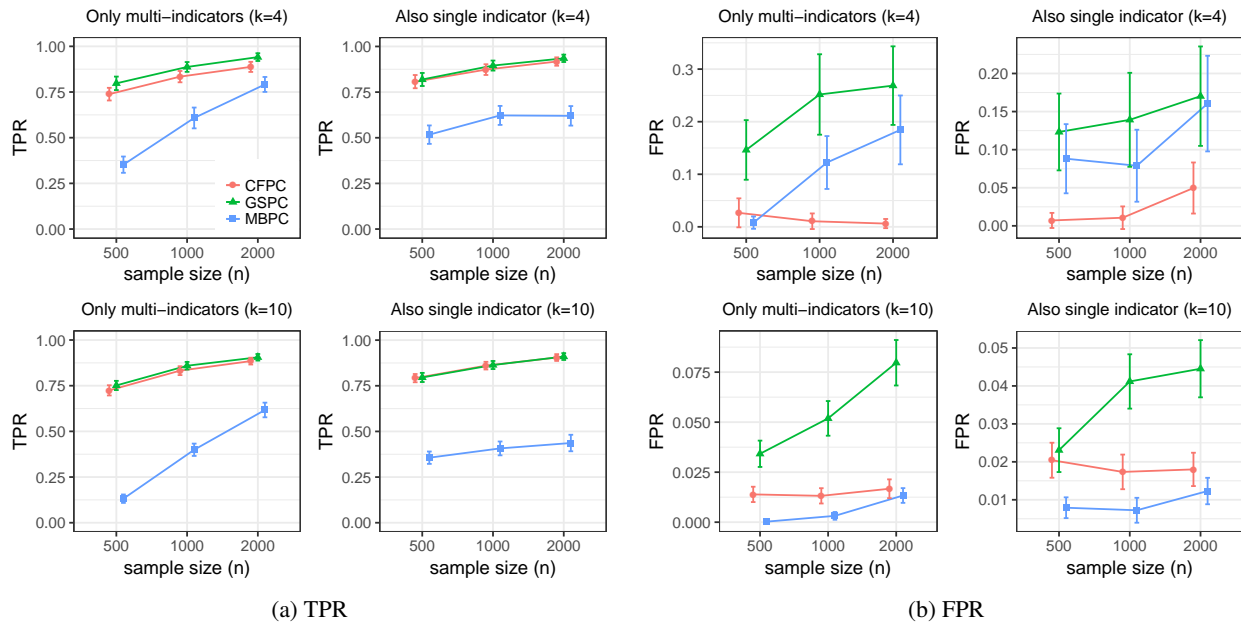


Figure 3: (a) TPR of CFPC, GSPC, and MBPC for the case where all factors have multiple indicators (left column) and the case where half of the factors have multiple indicators while the other half have a single indicator (right column), showing the mean over 100 experiments together with 95% confidence intervals. The two rows represent the results when the number of latent factors is 4 and 10 respectively. (b) FPR for the same experiments as in (a).

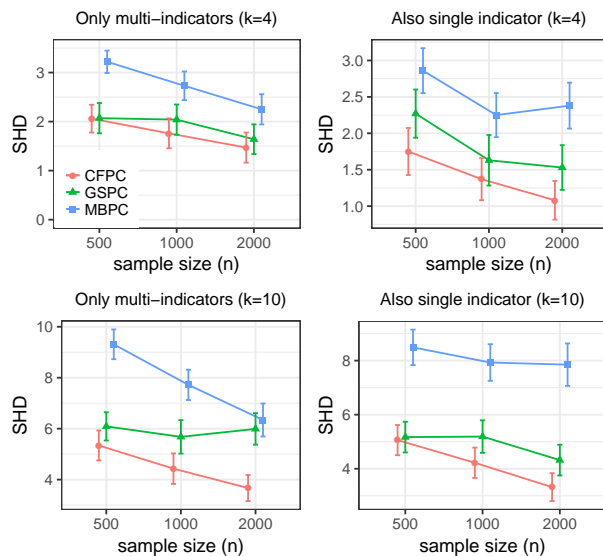


Figure 4: SHD of CFPC, GSPC, and MBPC, showing the mean over 100 experiments together with 95% confidence intervals, for the same experiments as in Figure 3.

IQ and poorer academic performance [12].

To conclude, using the Copula Factor PC algorithm in an ADHD sample allows us to infer causal relations between the different ADHD traits and generic factors

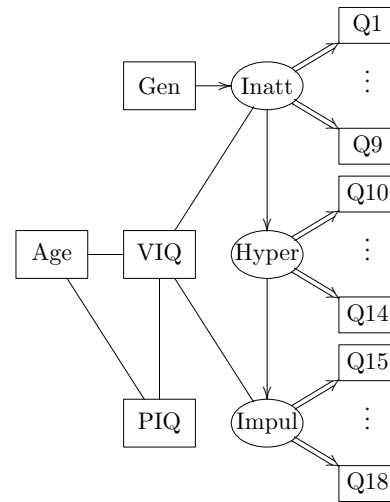


Figure 5: The resulting causal graph obtained by the ‘Copula Factor PC’ algorithm on the ADHD dataset, in which double arrows ‘ $\Rightarrow$ ’ represent the mapping from latent concepts to their corresponding questions (known) and other edges are those learned by our algorithm.

(age, gender, and IQ). This enhances knowledge of the causal structure of ADHD (e.g., by answering the question whether inattention is causing hyperactivity, or vice versa), which may have significant clinical implications, as it may inform therapeutic interventions.

## 7 CONCLUSION AND DISCUSSION

In this paper, we focused on learning causal relations between latent variables with pre-designed or pre-fitted measurement models. Our typical use case is that of psychological constructs that are linked to responses on questionnaire items. To the best of our knowledge, we are the first to propose a provably convergent algorithm that is able to recover the underlying causal structure between such factors and other observed variables, which can be both discrete and continuous.

In the experiments, our ‘Copula Factor PC’ algorithm clearly outperformed both the PC-MIMBuild algorithm and the greedy step-wise approach. PC-MIMBuild tests for conditional independencies between all pairs of indicators and concludes that the latent factors are dependent even if just one of the pairs fails the independence test. In our experience, this multiple testing approach, although elegant in theory, is difficult to make robust for largely varying numbers of indicators and sizes of the conditioning set. The ‘Copula Factor PC’ algorithm more naturally appears to find the right balance between true positives and false positives under varying conditions. It improves upon the greedy step-wise approach by estimating the full correlation matrix instead of individual sub-parts, which increases the power of the conditional independence tests.

Our approach extends earlier work, particularly [9] and [11], with various novel and essential ingredients needed to handle latent variables. Compared to [9], we replaced the Wishart prior with a  $G$ -Wishart distribution over factors and indicator variables, whose structure directly follows from the measurement model. The corresponding marginal prior on the factors is then still a Wishart distribution, which can be chosen such that the pairwise correlations are uniformly distributed. As in [11], but unlike [9], we can prove that our procedure is consistent. In the Supplement we show that, although the correlation matrix over factors itself is non-identifiable, all characteristics that relate to the identification of the correct causal structure can be consistently recovered.

While we considered the PC algorithm for inferring the underlying causal structure, one could plug in other standard algorithms like FCI [27], GES [5], or the recent improvements [6, 8, 32]. We further focused on so-called pure measurement models [24, 15], which is the major simplifying assumption of our procedure. We would argue that this is often satisfied, since it is the way in which questionnaires are typically designed by domain experts and that allows for a specific interpretation of the factors (e.g., a predefined set of items relates to the concept “hyperactivity”, another non-overlapping set of items to the

concept “inattention”). If the measurement models are not given, they can be learned using off-the-shelf algorithms, such as BPC [24] and FOFC [15], which output pure measurement models.

## Acknowledgements

We thank Anoeck Sluiter-Oerlemans for the profound analysis about experimental results on the ADHD dataset, and Ioan Gabriel Bucur for valuable discussions on the proof of Theorem 2. This research has been partially financed by the Netherlands Organisation for Scientific Research (NWO) under project 617.001.451.

## References

- [1] Anderson, Theodore Wilbur. *An introduction to multivariate statistical analysis*. John Wiley & Sons, 2003.
- [2] Anderson, Theodore Wilbur and Rubin, Herman. Statistical inference in factor analysis. In *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability, Volume 5: Contributions to Econometrics, Industrial Research, and Psychometry*, pp. 111–150, Berkeley, Calif., 1956. University of California Press.
- [3] Barnard, John, McCulloch, Robert, and Meng, Xiao-Li. Modeling covariance matrices in terms of standard deviations and correlations, with application to shrinkage. *Statistica Sinica*, pp. 1281–1311, 2000.
- [4] Byrne, Barbara M. *Structural equation modeling with EQS: Basic concepts, applications, and programming*. Routledge, 2013.
- [5] Chickering, David Maxwell. Optimal structure identification with greedy search. *Journal of machine learning research*, 3(Nov):507–554, 2002.
- [6] Claassen, Tom and Heskes, Tom. A Bayesian approach to constraint based causal inference. In *UAI*, pp. 207–216, 2012.
- [7] Colombo, Diego and Maathuis, Marloes H. Order-independent constraint-based causal structure learning. *Journal of Machine Learning Research*, 15(1):3741–3782, 2014.
- [8] Colombo, Diego, Maathuis, Marloes H, Kalisch, Markus, and Richardson, Thomas S. Learning high-dimensional directed acyclic graphs with latent and selection variables. *The Annals of Statistics*, 40(1):294–321, 2012.
- [9] Cui, Ruifei, Groot, Perry, and Heskes, Tom. Copula PC algorithm for causal discovery from mixed data. In *ECML PKDD*, pp. 377–392. Springer, 2016.



- [10] Finney, Sara J and DiStefano, Christine. Non-normal and categorical data in structural equation modeling. *Structural equation modeling: A second course*, pp. 269–314, 2006.
- [11] Harris, Naftali and Drton, Mathias. PC algorithm for nonparanormal graphical models. *The Journal of Machine Learning Research*, 14(Jan):3365–3383, 2013.
- [12] Heutink, Peter, Verhulst, Frank C, and Boomsma, Dorret I. A longitudinal twin study on IQ, executive functioning, and attention problems during childhood and early adolescence. *Acta neurol. belg*, 106: 191–207, 2006.
- [13] Hoff, Peter D. Extending the rank likelihood for semiparametric copula estimation. *The Annals of Applied Statistics*, pp. 265–283, 2007.
- [14] Kalisch, Markus and Bühlmann, Peter. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *The Journal of Machine Learning Research*, 8(Mar):613–636, 2007.
- [15] Kummerfeld, Erich and Ramsey, Joseph. Causal clustering for 1-factor measurement models. In *SIGKDD*, pp. 1655–1664. ACM, 2016.
- [16] Martínez-Torres, M Rocío. A procedure to design a structural and measurement model of intellectual capital: an exploratory study. *Information & Management*, 43(5):617–626, 2006.
- [17] Murphy, Kevin P. Conjugate Bayesian analysis of the Gaussian distribution. *def*, 1(2 $\sigma$ 2):16, 2007.
- [18] Murray, Jared S, Dunson, David B, Carin, Lawrence, and Lucas, Joseph E. Bayesian Gaussian copula factor models for mixed data. *Journal of the American Statistical Association*, 108(502): 656–665, 2013.
- [19] Nøvik, Torunn Stene, Hervas, Amaia, Ralston, Stephen J, Dalsgaard, Søren, Pereira, Rob Rodrigues, Lorenzo, Maria J, Group, ADORE Study, et al. Influence of gender on attention-deficit/hyperactivity disorder in Europe–ADORE. *European child & adolescent psychiatry*, 15(1): i15–i24, 2006.
- [20] Pearl, Judea. *Causality*. Cambridge university press, 2009.
- [21] Roverato, Alberto. Cholesky decomposition of a hyper inverse Wishart matrix. *Biometrika*, 87(1): 99–112, 2000.
- [22] Roverato, Alberto. Hyper inverse Wishart distribution for non-decomposable graphs and its application to Bayesian inference for Gaussian graphical models. *Scandinavian Journal of Statistics*, 29(3): 391–411, 2002.
- [23] Silva, Ricardo, Scheines, Richard, Glymour, Clark, and Spirtes, Peter. Learning measurement models for unobserved variables. In *UAI*, pp. 543–550, 2002.
- [24] Silva, Ricardo, Scheines, Richard, Glymour, Clark, and Spirtes, Peter. Learning the structure of linear latent variable models. *Journal of Machine Learning Research*, 7(Feb):191–246, 2006.
- [25] Skeem, Jennifer L and Cauffman, Elizabeth. Views of the downward extension: Comparing the youth version of the psychopathy checklist with the youth psychopathic traits inventory. *Behavioral sciences & the law*, 21(6):737–770, 2003.
- [26] Sokolova, Elena, Groot, Perry, Claassen, Tom, van Hulzen, Kimm J, Glennon, Jeffrey C, Franke, Barbara, Heskes, Tom, and Buitelaar, Jan. Statistical evidence suggests that inattention drives hyperactivity/impulsivity in attention deficit-hyperactivity disorder. *PLoS one*, 11(10):e0165120, 2016.
- [27] Spirtes, Peter, Glymour, Clark N, and Scheines, Richard. *Causation, prediction, and search*. MIT press, 2000.
- [28] Tsamardinos, Ioannis, Brown, Laura E, and Aliferis, Constantin F. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine learning*, 65(1):31–78, 2006.
- [29] Ullebø, Anne Karin, Breivik, Kyrre, Gillberg, Christopher, Lundervold, Astri J, and Posserud, Maj-Britt. The factor structure of ADHD in a general population of primary school children. *Journal of Child Psychology and Psychiatry*, 53(9):927–936, 2012.
- [30] van Steijn, Daphne J, Richards, Jennifer S, Oerlemans, Aniek M, de Ruiter, Saskia W, van Aken, Marcel AG, Franke, Barbara, Buitelaar, Jan, Rommelse, Nanda NJ, et al. The co-occurrence of autism spectrum disorder and attention-deficit/hyperactivity disorder symptoms in parents of children with ASD or ASD with ADHD. *Journal of Child Psychology and Psychiatry*, 53(9):954–963, 2012.
- [31] Willcutt, Erik G. The prevalence of DSM-IV attention-deficit/hyperactivity disorder: a meta-analytic review. *Neurotherapeutics*, 9(3):490–499, 2012.
- [32] Zhang, Kun, Zhang, Jiji, Huang, Biwei, Schölkopf, Bernhard, and Glymour, Clark. On the identifiability and estimation of functional causal models in the presence of outcome-dependent selection. In *UAI*, 2016.

---

# $f_{BGD}$ : Learning Embeddings From Positive Unlabeled Data with BGD

---

Fajie Yuan,<sup>1</sup> Xin Xin,<sup>1</sup> Xiangnan He,<sup>2</sup> Guibing Guo,<sup>3</sup> Weinan Zhang,<sup>4</sup>  
Tat-Seng Chua<sup>2</sup> and Joemon M. Jose<sup>1</sup>

University of Glasgow, UK<sup>1</sup>, National University of Singapore, Singapore<sup>2</sup>  
Northeastern University, China<sup>3</sup>, Shanghai Jiao Tong University, China<sup>4</sup>,

{f.yuan.1, x.xin.1, Joemon.Jose}@research.gla.ac.uk , xiangnanhe@gmail.com  
guogb@swc.neu.edu.cn, wnzhang@sjtu.edu.cn, chuats@comp.nus.edu.sg

## Abstract

Learning sparse features from only positive and unlabeled (PU) data is a fundamental task for problems of several domains, such as natural language processing (NLP), computer vision (CV), information retrieval (IR). Considering the numerous amount of unlabeled data, most prevalent methods rely on negative sampling (NS) to increase computational efficiency. However, sampling a fraction of unlabeled data as negative for training may ignore other important examples, and thus lead to non-optimal prediction performance. To address this, we present a fast and generic batch gradient descent optimizer ( $f_{BGD}$ ) to learn from *all* training examples without sampling. By leveraging sparsity in PU data, we accelerate  $f_{BGD}$  by several magnitudes, making its time complexity the same level as the NS-based stochastic gradient descent method. Meanwhile, we observe that the standard batch gradient method suffers from gradient instability issues due to the sparsity property. Driven by a theoretical analysis for this potential cause, an intuitive solution arises naturally. To verify its efficacy, we perform experiments on multiple tasks with PU data across domains, and show that  $f_{BGD}$  consistently outperforms NS-based models on all tasks with comparable efficiency.

## 1 INTRODUCTION

Learning from only positive and unlabeled (or non-observed) data, aka PU learning, occurs in numerous domains such as NLP, CV, IR. In these scenarios, the observed training data usually consists of positive data only. Moreover, the overall training data is typically very sparse, since

only a small fraction of positive examples are observed, and the non-observed negative examples are of a much larger scale.

To generalize well on such sparse data (He and Chua, 2017), embedding learning, such as word embedding in NLP (Mikolov et al., 2013b), image (category) embedding in CV (Weston et al., 2011), user (item) embedding in IR (Koren et al., 2009; Yuan et al., 2016a), and DNA k-mer embedding in genetic engineering (Ng, 2017), has become a common practice. However, learning embeddings from PU (or positive-only) data is computationally expensive, since each observed positive example needs to be paired with *all* non-observed negatives.

To learn from large-scale non-observed data, most recent embedding methods employ negative sampling (NS) and stochastic gradient descent (SGD) for efficient optimization (Mikolov et al., 2013b; Weston et al., 2012; Guo et al., 2018a,b; Yuan et al., 2016a, 2017). However, the training time and prediction accuracy are largely determined by the sampling distribution and size of negative samples. Sampling a fraction of non-observed data as negative for training may ignore other useful examples, or lead to insufficient training of them. This is our main motivation in this work. Another well-known difficulty is that the SGD optimizer performs frequent gradient updates with a high variance, which can cause the objective function to fluctuate heavily near the optimum (Ruder, 2016). By contrast, batch gradient descent (BGD) computes the gradient on all training data for updating a model parameter. As such, the learning process has the potential to converge to a better optimum. Unfortunately, the low efficiency caused by the full-batch gradient computation makes it less applicable to large-scale datasets.

To deal with these issues, we present a fast and generic batch gradient descent algorithm (called  $f_{BGD}$ ) for learning embeddings from positive-only data.  $f_{BGD}$  optimizes a commonly used square loss function that accounts for all

non-observed examples without any sampling. To ensure the learning efficiency, we accelerate  $f_{BGD}$  with rigorous mathematical reasoning. Notably, despite that  $f_{BGD}$  computes loss and gradients over all examples, its actual complexity is comparable with NS-based SGD methods that utilize only partial examples. Furthermore, we show that standard batch learning are prone to the gradient exploding and vanishing problem and stabilize it by an intuitive way.

To summarize, the main contributions of this paper are as follows:

- We propose a unified BGD approach to solve the sparse feature learning problem from PU data. For efficiency optimization, we accelerate it by a natural reformulation of the loss and rearrangement for the dot product operation. For generality, we identify the dot product structure for a variety of embedding models.
- We provide theoretical explanations that the standard batch gradient learning suffers from gradient instability issues when learning embedding models due to large batched summation of sparse features.
- We implement a general weighting scheme that suits well for unlabeled examples in various domains, which not only largely improves the prediction accuracy of  $f_{BGD}$ , but also makes efficiency optimization possible.
- $f_{BGD}$  achieves state-of-the-art performance in multiple research fields with comparable costs to NS-based SGD methods. Insightful comparisons for sampling based methods have been thoroughly studied. Another insightful observation is that many specific models used in one of these fields are promising to benefit others by minor (or no) changes. This opens a new direction of research to bridge these fields.
- We release the source code of  $f_{BGD}$  at: <https://github.com/fajieyuan/fBGD>.

## 2 PROBLEM FORMULATION

### 2.1 LEARNING FROM POSITIVE-ONLY DATA

Assuming we have two sets of examples that are available for training: the positive set  $P$  and an unlabeled set  $U$ , which is typically non-observed and contains both positive and negative samples. Each sample in  $P$  is an observed  $(x, y)$  pair, where  $x \in X$  and  $y \in Y$ .  $X$  and  $Y$  are the set of distinct  $x$  and  $y$  respectively. For a given  $x$ , we have a set of relevant  $y$  labelled, denoted by  $Y_x^+$ , the size of which is much smaller than that of the non-observed set  $Y_x^-$ . As shown in Figure 1 (a), we can use a matrix  $\mathbf{H} \in \mathbb{R}^{|X| \times |Y|}$  to denote the historical interactions between  $x$  and  $y$ . The goal of PU learning is to find a function  $\hat{r}_{xy}$  (parameterized by  $\Theta$ ) that explains a set of observed pairs  $(x, y)$ , such as “relevant-or-not” and “like-or-not”.

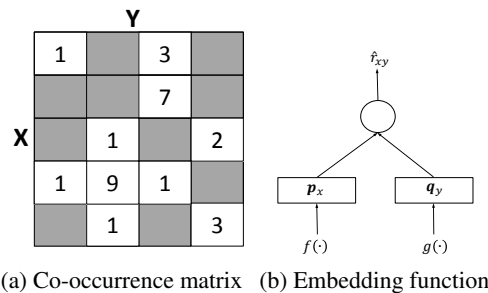


Figure 1: (a): PU data with  $(x, y)$  co-occurrence matrix  $\mathbf{H}$ . The grey cells denote no explicitly observed  $(x, y)$  examples. (b): Embedding function.  $f(\cdot)$  and  $g(\cdot)$  are functions to construct the embedding vectors  $\mathbf{p}_x$  and  $\mathbf{q}_y$  respectively.

### 2.2 EMBEDDING MODELS

Embedding models have been widely adopted in many specific PU learning tasks. In this work, we focus on optimizing the embedding functions that can be *explicitly* or *implicitly* expressed by a dot product structure, given below.

$$\hat{r}_{xy} = \langle \mathbf{p}_x, \mathbf{q}_y \rangle = \sum_{d=1}^g p_{x,d} q_{y,d} \quad (1)$$

where  $\mathbf{p}_x$  and  $\mathbf{q}_y$  are *compressed* embedding vectors with embedding dimension  $g$ . They can be obtained by directly projecting the ID of row/column into the embedding space (i.e., explicit structure as in Xin et al. (2018) and He et al. (2016b)), or projecting with other features of row/column (i.e., implicit structure as in Rendle and Freudenthaler (2014); Bayer et al. (2017)). The time complexity of evaluating this equation is  $O(g)$ . Note that the implicit dot product structure can describe a variety of multi-linear models, such as SVDFeature (Chen et al., 2012) and tensor models (Bailey and Aeron, 2017; Rendle and Schmidt-Thieme, 2010). Later, we will show how to construct this dot product structure for some state-of-the-art embedding models.

### 2.3 LOSS FUNCTION AND BGD OPTIMIZATION

We propose optimizing the standard regression loss, which can also be used for classification and ranking tasks. Unlike previous works (Pennington et al., 2014; Cer et al., 2017), the optimized loss function should explicitly account for all unlabeled samples.

$$J(\Theta) = \sum_{(x,y) \in P} \alpha_{xy}^+ (r^+ - \hat{r}_{xy})^2 + \underbrace{\sum_{(x,y) \in U} \alpha_{xy}^- (r^- - \hat{r}_{xy})^2}_{JM(\Theta)} \quad (2)$$

where  $JM(\Theta)$  denotes the errors of all unlabeled examples,  $\alpha_{xy}^+$  and  $\alpha_{xy}^-$  are the weight functions. Eq. (2) can be minimized by BGD, which computes the gradient of loss function w.r.t.  $\theta \in \Theta$  on the entire (positive and unlabeled) samples:

$$\theta \leftarrow \theta - \gamma \nabla_{\theta} J(\theta) \quad (3)$$

where  $\gamma$  is the learning rate, and  $\nabla_{\theta} J(\theta)$  is the gradient of  $J(\Theta)$  w.r.t.  $\theta$ , given below:

$$\begin{aligned} \nabla_{\theta} J(\theta) = & 2 \left( \underbrace{\sum_{(x,y) \in P} \alpha_{xy}^+ (r^+ - \hat{r}_{xy}) \nabla_{\theta} \hat{r}_{xy}}_{O(|Y_x^+|g)} \right. \\ & \left. + \underbrace{\sum_{(x,y) \in U} \alpha_{xy}^- (r^- - \hat{r}_{xy}) \nabla_{\theta} \hat{r}_{xy}}_{O(|Y_x^-|g)} \right) \end{aligned} \quad (4)$$

where  $O(|Y_x^+|g)$  and  $O(|Y_x^-|g)$  are the complexity of gradient computation on positive and unlabeled data.

## 2.4 EFFICIENCY ISSUES

As can be seen, the second term  $J_M(\Theta)$  in Eq.(2) dominates the computational complexity. This is because computing  $J_M(\Theta)$  in Eq.(2) has almost  $O(|X||Y|g)$  time because  $|P| \ll |U|$ . Similarly, updating a parameter (under the explicit dot product setting), e.g.,  $p_{x,d}$ , by Eq.(4) is  $O((|Y_x^-|)g)$ , or  $O(|Y|g)$ , because  $|Y_x^+| \ll |Y_x^-|$ . The total cost by iterating over all  $p_{x,d}$  in  $\Theta$  in each iteration becomes  $O(|X||Y|g)$ . Clearly, the straight-forward way to calculate gradients by BGD is generally infeasible, because  $|X||Y|$  can easily reach billion level or even higher.

## 3 FAST & GENERIC BGD For PU DATA

In this section, we first describe the derivation of  $f_{BGD}$  for the optimization of Eq.(1), and show how to generalize it to complex embedding models. Then, we design a general weighting scheme for the missing examples in  $f_{BGD}$ .

### 3.1 EFFICIENT $f_{BGD}$ LOSS

In the above learning setting, the dominant computation is the minimization of  $J_M(\Theta)$  in Eq.(2) since each  $x$  has its standalone unlabeled set of  $y$ , i.e.,  $Y_x^-$ . As such, the BGD algorithm basically needs to iterate through all elements in  $Y_x^-$ , and repeat the operation for all  $x \in X$ , which produces the main cost. To solve the problem, we reformulate the standard BGD loss according to the set theory<sup>1</sup>. Naturally, for any PU learning problem the loss of  $U$  (unlabeled) data can be expressed by the *residual* between the loss of *all* data and that of  $P$  (positive) data.

$$J_M(\Theta) = \sum_{x \in X} \left( \sum_{y \in Y} \alpha_{xy}^- (r^- - \hat{r}_{xy})^2 - \sum_{y \in Y_x^+} \alpha_{xy}^- (r^- - \hat{r}_{xy})^2 \right) \quad (5)$$

A new objective function can be achieved by substituting Eq.(5) in Eq.(2). We combine the two terms that associates with  $P$  data together into a single term (Note that  $r^+$ ,  $r^-$ ,

<sup>1</sup>The set relation was also applied in He et al. (2016b); Xin et al. (2018), which can be regarded as a special case of  $f_{BGD}$  as  $\hat{r}_{xy}$  is only limited to an explicit dot product function that deals with two features in a specific domain.

$\alpha_{xy}^+$ ,  $\alpha_{xy}^-$  are independent of  $\theta \in \Theta$ ).  $J(\Theta)$  is rewritten as

$$J(\Theta) = const + J_A(\Theta) + J_P(\Theta) \quad (6)$$

where

$$\begin{aligned} J_A(\Theta) &= \sum_{x \in X} \sum_{y \in Y} \alpha_{xy}^- (\hat{r}_{xy} - r^-)^2 \\ J_P(\Theta) &= \sum_{x \in X} \sum_{y \in Y_x^+} (\alpha_{xy}^+ - \alpha_{xy}^-) \left( \hat{r}_{xy} - \frac{\alpha_{xy}^+ r^+ - \alpha_{xy}^- r^-}{\alpha_{xy}^+ - \alpha_{xy}^-} \right)^2 \end{aligned} \quad (7)$$

where  $J_A(\Theta)$  and  $J_P(\Theta)$  denote the loss for *all* and  $P$  data respectively; *const* denotes a  $\Theta$ -invariant constant value. Clearly, the loss of  $U$  data has been eliminated. The new computation complexity is now in  $\tilde{J}_A(\Theta)$ , which is part of  $J_A(\Theta)$ , defined as:

$$\tilde{J}_A(\Theta) = \sum_{x \in X} \sum_{y \in Y} \alpha_{xy}^- \hat{r}_{xy}^2 - 2r^- \sum_{x \in X} \sum_{y \in Y} \alpha_{xy}^- \hat{r}_{xy} \quad (8)$$

So far, we have focused on the loss without considering the specific formulation of model prediction  $\hat{r}_{xy}$ . As described in Section 2.2, we focus on  $\hat{r}_{xy}$  that can be either *explicitly* or *implicitly* formalized as a dot product (i.e., Eq.(1)) structure based on embedding vectors of  $x$  and  $y$ . In the following, we first show the generalized transformation for a *compressed* dot product structure. Then, we show how to apply  $f_{BGD}$  to various complex embedding functions with more input features by constructing the similar structure.

$$\begin{aligned} \alpha_{xy}^- \hat{r}_{xy}^2 &= \alpha_{xy}^- \sum_{d=1}^g p_{x,d} q_{y,d} \sum_{d'=1}^g p_{x,d'} q_{y,d'} \\ &= \sum_{d=1}^g \sum_{d'=1}^g \alpha_{xy}^- (p_{x,d} p_{x,d'}) (q_{y,d} q_{y,d'}) \end{aligned} \quad (9)$$

where we observe that there exists a very nice structure in above equation — if  $\alpha_{xy}^-$  is a constant value or a value only associates with  $x$  or  $y$  but not  $(x, y)$  pair. Considering that there is no observed  $(x, y)$  interaction in unlabeled examples, it is reasonable to set  $\alpha_{xy}^-$  as  $\alpha_y^-$  or  $\alpha_x^-$ . The simplified weight design is a necessary condition for efficient optimization in the following. Here we continue to discuss the algorithm, assuming  $\alpha_{xy}^- = \alpha_y^-$ , and later show how to design a good weighting scheme. With this setting, the interaction between  $p_{x,d}$  and  $q_{y,d}$  can be safely separated. Thereby,  $\sum_{y \in Y} \alpha_y^- q_{y,d} q_{y,d'}$  can be independent of the optimization of  $x$ -related parameters. That is, we could achieve a significant speed-up by precomputing this term. Let caches  $S_{dd'}^q = \sum_{y \in Y} \alpha_y^- q_{y,d} q_{y,d'}$ , and  $S_d^q = \sum_{y \in Y} \alpha_y^- q_{y,d}$ ,  $\tilde{J}_A(\Theta)$  is derived as follows

$$\tilde{J}_A(\Theta) = \sum_{d=1}^g \sum_{d'=1}^g S_{dd'}^q \sum_{x \in X} p_{x,d} p_{x,d'} - 2r^- \sum_{d=1}^g S_d^q \sum_{x \in X} p_{x,d} \quad (10)$$

The rearrangement of nested sums in Eq.(10) is the key transformation that allows the fast optimization of  $f_{BGD}$ . The computation complexity has reduced from  $O(|X||Y|g)$  in Eq.(8) to  $O((|X| + |Y|)g^2)$  in Eq.(10). Optimization details regarding the gradient computation are given in Section 3.3.

### 3.2 IDENTIFYING THE DOT PRODUCT STRUCTURE

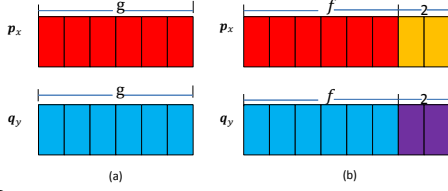


Figure 2: (a) denotes the explicit dot product structure, such as in AllVec (Xin et al., 2018), while (b) is the dot product that implicitly exists in SVDFeature. Each cell denotes a real value.

We notice that the dot product structure implicitly exists in a variety of embedding modes. Here we show the structure for a general embedding model, aka SVDFeature (Chen et al., 2012), which can be used in context-aware recommender systems (CARS), content-based image retrieval system (CBIR) and prior knowledge based word representation. We also identify the dot product structure for two tensor-based embedding models (Bailey and Aeron, 2017; Rendle and Schmidt-Thieme, 2010) in Appendix A. The model equation of SVDFeature is defined as

$$\hat{r}_{xy} = w_0 + \mathbf{wz}^T + \sum_{j=1}^{pX} \sum_{j'=1}^{pY} \langle \mathbf{v}_j^X, \mathbf{v}_{j'}^Y \rangle z_{x,j}^X z_{y,j'}^Y \quad (11)$$

where  $\mathbf{z}$  is the feature vector. E.g., in a context-aware music recommender system, it is defined as

$$\mathbf{z} = \underbrace{(0, \dots, 1, \dots, 0, 0, 0.1, \dots, 0.1, 0, 0, 1, \dots, 1, \dots, 0, 0, 1, \dots, 0, 0, 1, \dots, 0)}_{\mathbf{z}_x^X} \underbrace{, \dots, 1, \dots, 0, 0, 1, \dots, 0)}_{\mathbf{z}_y^Y}$$

$x$  and  $y$  are described by  $\mathbf{z}_x^X$  and  $\mathbf{z}_y^Y$  respectively.  $z_{x,j}^X$  is  $j$ -th element in  $\mathbf{z}_x^X$ , which is  $x$ -th row in  $\mathbf{Z}^X \in \mathbb{R}^{|X| \times pX}$ .  $pX$  is the number of features in  $\mathbf{z}_x^X$ .  $\mathbf{v}_j^X$  is the  $j$ -th row in  $\mathbf{V}^X \in \mathbb{R}^{pX \times f}$ , where  $f$  is the original embedding size. Inspired by Rendle and Freudenthaler (2014), we rewrite Eq.(11) as an implicit dot product structure (see Figure 2).

$$\hat{r}_{xy} = \sum_{d=1}^g p_{x,d} q_{y,d} \quad (12)$$

where  $g = f + 2$  and

$$\begin{aligned} p_{x,d} &= \sum_{j=1}^{pX} z_{x,j}^X v_{j,d}^X, & q_{y,d} &= \sum_{j=1}^{pY} z_{y,j}^Y v_{j,d}^Y, \\ p_{x,f+1} &= w_0 + \sum_{j=1}^{pX} w_j z_{x,j}^X, & q_{y,f+1} &= 1 \\ p_{x,f+2} &= 1, & q_{y,f+2} &= \sum_{j=1}^{pY} w_{(j+pX)} z_{y,j}^Y \end{aligned} \quad (13)$$

where  $v_{j,d}^X$  is the  $d$ -th element in  $\mathbf{v}_j^X$ . Next, we show the gradient computation for both explicit (i.e., Eq. (1)) and implicit dot product (e.g., Eq. (11)) structure.

### 3.3 EFFICIENT GRADIENTS

Following Section 3.1, the gradients of  $\tilde{J}_A(\theta)$  w.r.t.  $\theta^X \in \Theta^X$  is given by

$$\begin{aligned} \nabla_{\theta^X} \tilde{J}_A(\theta) &= 2 \sum_{d=1}^g \sum_{d'=1}^g S_{dd'}^q \sum_{x \in X} p_{x,d'} \nabla_{\theta} p_{x,d} \\ &\quad - 2r^- \sum_{d=1}^g S_d^q \sum_{x \in X} \nabla_{\theta} p_{x,d} \end{aligned} \quad (14)$$

The optimization process of  $\theta^Y \in \Theta^Y$  is almost symmetric to  $\theta^X$ , except that the weighting scheme  $\alpha_y^-$  is inside the sum of  $y \in Y$ . In what follows, we present the gradient computation for Eq.(14) with both explicit and implicit dot products.

#### 3.3.1 GRADIENT COMPUTATION WITH EQ.(1)

Assume Eq.(1) is a basic dot product, the gradient of  $p_{x,d}$  with respect to  $p_{x^*,d^*}$  is given by

$$\nabla_{p_{x^*,d^*}} p_{x,d} = \begin{cases} 1 & x = x^* \wedge d = d^* \\ 0 & \text{otherwise} \end{cases} \quad (15)$$

Thus, Eq.(14) simplifies to

$$\nabla_{p_{x^*,d^*}} \tilde{J}_A(\theta) = 2 \sum_{d=1}^g S_{d^*d}^q p_{x^*,d} - 2r^- S_{d^*}^q \quad (16)$$

The complexity of Eq.(16) is in  $O(g)$ , and correspondingly, updating all  $\theta^X \in \Theta^X$  is  $O(|X|g^2)$ . Overall, gradient computation for all  $\theta \in \Theta$  is  $O((|X| + |Y|)g^2 + |P|g)$ , where  $O(|P|g)$  is the complexity for the gradients of the positive loss. In contrast, the cost of NS-SGD is  $O((n+1)|P|g)$ , where  $n+1$  denotes  $n$  negative  $y$  and 1 positive  $y$ .

#### 3.3.2 GRADIENT COMPUTATION WITH EQ.(11)

The gradients of  $p_{x,d}$  with respect to  $w_{j^*}$  and  $v_{j^*,d^*}^X$  are given by

$$\nabla_{w_{j^*}} p_{x,d} = \begin{cases} z_{x,j^*}^X & d = f + 1 \\ 0 & \text{otherwise} \end{cases}, \quad \nabla_{v_{j^*,d^*}^X} p_{x,d} = \begin{cases} z_{x,j^*}^X & d \leq f \\ 0 & \text{otherwise} \end{cases} \quad (17)$$

Thus, Eq.(14) w.r.t. to  $w_{j^*}$  and  $v_{j^*,d^*}^X$  simplifies to

$$\nabla_{w_{j^*}} \tilde{J}_A(\theta) = 2 \sum_{d=1}^g S_{(f+1)d}^q \sum_{x \in X} p_{x,d} z_{x,j^*}^X - 2r^- S_{(f+1)}^q \sum_{x \in X} z_{x,j^*}^X \quad (18)$$

$$\nabla_{v_{j^*,d^*}^X} \tilde{J}_A(\theta) = 2 \sum_{d=1}^g S_{d^*d}^q \sum_{x \in X} p_{x,d} z_{x,j^*}^X - 2r^- S_{d^*}^q \sum_{x \in X} z_{x,j^*}^X \quad (19)$$

Note that the computation of sums over  $x \in X$  can be accelerated by only iterating over  $x$  where  $z_{x,j^*}^X \neq 0$ . Moreover,  $p_{x,d}$  is able to be precomputed to reduce the cost.

Although  $p_{x,d}$  changes when updating  $\theta^X$ , it can be updated in synchronization with the changes in  $\theta^X$ , denoted by  $\Delta\theta^X$ .

$$p_{x,d} \leftarrow p_{x,d} + z_{x,j}^X \Delta\theta^X = p_{x,d} - z_{x,j}^X \gamma \nabla_{\theta} J(\theta) \quad (20)$$

Analogously with Section 3.3.1, the total time complexity of  $\nabla_{\theta} J_A(\theta)$  (or  $\nabla_{\theta} \tilde{J}_A(\theta)$ ) in one iteration for all parameters is  $O(g^2(N(X) + N(Y)))$ , where  $N(X)$  and  $N(Y)$  are the number of non-zero elements in  $\mathbf{Z}^X$  and  $\mathbf{Z}^Y$ . Finally, the efficient computation for  $\theta$  is reasonably given as follows<sup>2</sup>

$$\theta \leftarrow \theta - \gamma(\nabla_{\theta} J_A(\theta) + \nabla_{\theta} J_p(\theta)) \quad (21)$$

The detailed implementation of  $f_{BGD}$  is in Appendix B.

### 3.4 WEIGHTING ON UNLABELED DATA

Now that the basic description of the speed-up process for  $f_{BGD}$  is completed, we proceed to discuss the weighting scheme in this section. First, in terms of  $\alpha_{xy}^+$ , any reasonable weighting scheme could be adopted and will not affect the analysed computation. For example, on the word embedding task (see Section 5) we set  $\alpha_{xy}^+$  the same as in GloVe (Pennington et al., 2014), while we set it as 1 for the other tasks considering that there is no available frequency information for positive  $(x, y)$  pairs.

As for  $\alpha_y^-$ , we design a non-uniform weighting scheme based on the property of  $y$ . Our weighting scheme is originally motivated by the frequency-based oversampling idea such as Skip-gram model (Mikolov et al., 2013b) and (Yuan et al., 2016a). However, both methods are tailored for the SGD or the mini-batch gradient descent (MGD) (He and Chua, 2017) optimization. Clearly, sampling techniques do not suit our model, because the focus of  $f_{BGD}$  is an all-sample based optimization method. Hence, a frequency-based weighting scheme is more suitable for our optimization setting. To effectively differentiate true negative and unknown examples, we assign a larger weight for the unlabeled data with high  $y$  frequency, and a smaller weight for the low-frequency  $y$ .

$$\alpha_y^- = \alpha_0 \frac{(e^{z_y} - 1)^\rho}{\sum_{y=1}^{|Y|} (e^{z_y} - 1)^\rho} \text{ where } z_y = \frac{p_y}{|P|} \quad (22)$$

where  $p_y$  denotes the frequency of  $y$ , given by the number of observations in  $P$ , and  $\alpha_0$  determinates the overall weight of unlabeled examples to solve the imbalanced-class problem. The exponent  $\rho$  controls weight distribution, which should be tuned based on the dataset.

## 4 IMPROVED $f_{BGD}$

So far, we have discussed the efficiency optimization of  $f_{BGD}$ . However, we observe unreliable results during evaluation especially for complex embedding models with

<sup>2</sup> Again,  $\nabla_{\theta} J_p(\theta)$  can be calculated by the standard way, which has the same time complexity with NS-SGD with the same ratio of negative and positive samples.

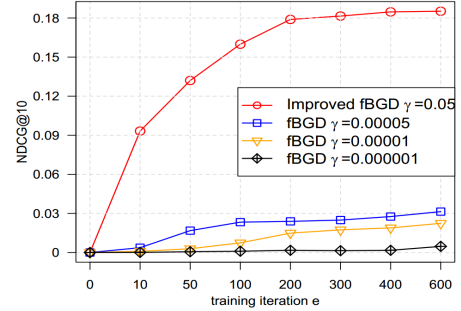


Figure 3: Performance of the improved  $f_{BGD}$  (Section 4.2) and standard  $f_{BGD}$  on Last.fm with four features. Note for the standard  $f_{BGD}$ , some gradients will be evaluated as infinite (NaN) when  $\gamma > 5 \times 10^{-5}$ . Clearly,  $f_{BGD}$  with vanishing gradient performs poorly on Last.fm even by fine tuning the learning rate.

more input features, as shown in Figure 3. A novel contribution here is to reveal why unstable gradient issues will occur for the standard BGD.

### 4.1 GRADIENT INSTABILITY OF $f_{BGD}$

While the unstable gradient problem, such as the gradient exploding and vanishing, has been observed when training deep neural networks (He et al., 2016a), the optimized models of  $f_{BGD}$  in this paper are mostly shallow embeddings. Therefore, the cause of the unstable gradient issue in our case is fundamentally different from that in the existing deep layer models, in the sense that in deep models unstable gradients occur mainly due to cumulative multiplying of small/big numbers from previous layers, whereas in  $f_{BGD}$  it is caused by the large batched summation of sparse features. We expect the following theoretical analysis and solution could provide practical guidelines for the future development of batch gradient optimization.

To understand the weird behavior of gradient instability, we need to revisit the form of gradients. We take the derivation of  $v_{j^*,d}^X$  ( $d \leq f$ ) in Eq. (11) w.r.t. the loss of positive data as an example.

$$\nabla_{v_{j^*,d}^X} J_P(\theta) = 2 \sum_{x \in X} \sum_{y \in Y_x^+} z_{x,j^*}^X (\alpha_{xy}^+ - \alpha_y^-) \left( \hat{r}_{xy} - \frac{\alpha_{xy}^+ \hat{r}_{xy}^+ - \alpha_y^- \hat{r}_{xy}^-}{\alpha_{xy}^+ - \alpha_y^-} \right) q_{y,d^*} \quad (23)$$

Due to the data sparsity, to compute  $\sum_{x \in X} \sum_{y \in Y_x^+} z_{x,j^*}^X$  we only need to consider  $x \in X$  that has a non-zero  $z_{x,j^*}^X$  (note that for a feature  $j$ , most  $x$  have  $z_{x,j^*}^X$  equal to zero which can be safely ignored). Let  $l_{j^*}$  be the number of non-zero elements in the  $j^*$ -th column of  $\mathbf{Z}^X$ .

In real-world data sets, the number of rows in  $\mathbf{Z}^X$ , i.e.,  $|X|$ , can easily scale to many millions or even billion level and, therefore, it is very likely that  $l_{j^*}$  has distinct magnitudes for a different column  $j^*$ . Moreover, in

Eq.(23) there is another summation  $\sum_{y \in Y_x^+}$ , which represents the size of observed  $y$  for  $x$ . The component value of  $\sum_{x \in X} \sum_{y \in Y_x^+} x_{x,j^*}^X$  in Eq.(23) varies from 1 to  $|X| \cdot |Y|$ , assuming  $\mathbf{Z}^X$  is a binary matrix. This indicates the value of Eq.(23) may be very unstable:  $\nabla_{v_{j^*,d}^X} J_P(\theta)$  can be too large for a denser feature  $j^*$  that is accompanied by a large  $\sum_{x \in X} \sum_{y \in Y_x^+} z_{x,j^*}^X$  (e.g.,  $= 10^6$ ), while it may be too small for a sparser feature with a small  $\sum_{x \in X} \sum_{y \in Y_x^+} z_{x,j^*}^X$  (e.g.,  $= 1$ ). Accordingly, the overall gradient  $\nabla_{\theta} J(\theta)$  in Eq.(21) has the same unstable problem. In this case, a uniform learning rate  $\gamma$  is no longer suitable because  $\nabla_{\theta} J(\theta)$  with a larger  $\sum_{x \in X} \sum_{y \in Y_x^+} z_{x,j^*}^X$  is likely to explode (i.e.  $\nabla_{\theta} J(\theta) = \text{NaN}$ ) if using a large  $\gamma$ , while  $\nabla_{\theta} J(\theta)$  with a smaller  $\sum_{x \in X} \sum_{y \in Y_x^+} z_{x,j^*}^X$  may vanish (i.e.  $\nabla_{\theta} J(\theta) \approx 0$ ) if using a small  $\gamma$ . Generally, it is hard or even impossible to find a medium learning rate that balances reasonably well in both conditions. To gain more insight into the performance of  $f_{BGD}$  with unstable gradients, we show results with different learning rates in Figure 3.

Interestingly, we empirically find that on many datasets with only two input features (or an explicit dot product structure), the gradient instability problem may be alleviated by carefully tuning  $\gamma$ . In other words, by many trials with different learning rates,  $f_{BGD}$  sometimes is able to offer reasonable results. However, on data sets with more feature variables (e.g., Last.fm), the outputs of  $f_{BGD}$  are prone to the NaN error. This is because in the pure dot product setting, the nested summation  $\sum_{x \in X}$  can be dropped. As such, although the gradient instability issue may still happen because of  $\sum_{y \in Y_x^+}$ , it is less severe as the value of  $|Y_x^+|$  is much smaller than that of  $l_j \cdot |Y_x^+|$ .

## 4.2 SOLVING THE UNSTABLE GRADIENT ISSUE

The above theoretical analysis for the gradient estimation over all data suggests that the same learning rate does not hold for all model parameters due to the large batched summation of sparse features. Analytically, by assigning a specific learning rate for each parameter update, we can control the unstable gradient to a certain extent. In other words,  $f_{BGD}$  should perform larger updates for small  $\nabla_{\theta} J(\theta)$ , and vice versa.

Based on the above analysis, an intuitive solution is to adapt the learning rate for each parameter, such as having done in Adagrad Duchi et al. (2011). While Adagrad is originally proposed for stochastic gradient method to accelerate convergence, here we show how to apply it on the full gradient method to address the gradient instability issue. Denoting  $\gamma_t$  as the learning rate for the  $t$ -th update, we then assign a personalized learning rate for each parameter  $\theta$ :

$$\gamma_t(\theta) = \frac{\gamma}{G_t(\theta)}, \quad G_t(\theta) = \begin{cases} \nabla_{\theta} J(\theta)_t + \epsilon & G_t(\theta) = 0 \\ \sqrt{\sum_{i=1}^t (\nabla_{\theta} J(\theta)_i)^2} & G_t(\theta) \neq 0 \end{cases} \quad (24)$$

Table 1: Dataset statistics ( $|U| = |X \times Y| - |P|$ ). ‘‘Open’’ is the OpenImages dataset. ‘‘K’’, ‘‘M’’ and ‘‘B’’ are short for thousand, million and billion.  $(x, y)$  denotes (word, context), (user, item) and (image, label) on the WE, CF and IC tasks respectively. Note that user and item in Lastfm contain user- and item-related variables. The density can be calculated by  $\frac{|P|}{|X \times Y|}$

Data	$ X $	$ Y $	$pX$	$pY$	$ P $	$ X \times Y $
NewsIR	83K	83K	83K	83K	150M	6.9B
Text8	71K	71K	71K	71K	47M	5.0B
Yahoo	200K	136K	200K	136K	76M	27.2B
Lastfm	63K	58K	65K	75K	1.3M	3.7B
Open	1.4M	7.5K	1.4M	7.5K	11.4M	10.5B

where  $\nabla_{\theta} J(\theta)_t$  is the gradient w.r.t.  $\theta$  for the  $t$ -th update,  $G_T(\theta)$  is the accumulation of the squared gradients, and  $\epsilon$  is a smoothing term to avoid division by zero, set as  $10^{-4}$ . The overall algorithm of improved  $f_{BGD}$  can be implemented by replacing  $\gamma$  in Eq.(21) and Eq.(20) with the new  $\gamma_t(\theta)$ .

## 5 EXPERIMENTS

$f_{BGD}$  is a generic PU learning model and can be applied in a wide range of tasks with PU data and sparse features. For evaluation purpose, we verify its performance in three fields — word embedding (WC) of NLP, collaborative filtering (CF) of IR, and image classification (IC) of CV.

### 5.1 EXPERIMENTAL SETUP

#### 5.1.1 Datasets

We use five large benchmark datasets for evaluation: NewsIR<sup>3</sup> and Text8<sup>4</sup> for WE, Yahoo music<sup>5</sup> and Lastfm<sup>6</sup> for CF, and OpenImages Krasin et al. (2017) for IC. For NewsIR, we preprocess them by a standard pipeline, i.e., removing non-textual elements, lowercasing and tokenization. For Yahoo, we use the ‘‘train\_0’’ file. For Lastfm, we follow Weston et al. (2012) by extracting the latest one-week actions per user via the timestamp, and consider two tracks played by the same user as ‘‘consecutive’’ if they are played within 90 minutes. It is used as a context-aware (or next-item) recommendation dataset, where each  $x$  contains a user and his previously played music tracks and each  $y$  contains a music track and its artist. For OpenImages, we randomly sample a number of (image, label) pairs from the original dataset. The statistics of datasets are summarized

<sup>3</sup><http://research.signalmedia.co/newsir16/signal-dataset.html>

<sup>4</sup><http://matthahoney.net/dc/text8.zip>

<sup>5</sup><http://webscope.sandbox.yahoo.com/catalog.php?datatype=r&did=2>

<sup>6</sup><http://www.dtic.upf.edu/~ocelma/MusicRecommendationDataset/>

Table 2: Comparison of well-known PU learning models. “SG” and “SVDF” is short for Skip-gram and SVDFeature respectively.

Model	Sampler	Ratio	Optimizer	Loss
SG×10	Static	1:10	SGD	LOG
GloVe	-	-	SGD	LS
SVDF×8	Uniform	1:8	SGD	LS
BPRFM	Uniform	1:1	SGD	L2R
λFM	Static	1:1	SGD	L2R
WARP	Dynamic	1:1	SGD	L2R
VSE-ens	Dynamic	1:1	SGD	L2R
$f_{BGD}$	-	-	BGD	LS

“Uniform”, “Static” and “Dynamic” are short for a uniform, static and dynamic sampler respectively. Static sampler means the sampling distribution of negative examples is defined before training and keeps unchanged during the whole optimization process. Dynamic sampler changes the sampling distribution of negative examples according to the current state of the learning algorithm. “Ratio” represents the positive-to-negative example ratio. “LS”, “LOG”, and “L2R” are short for the least square, logistic, and learning-to-rank loss function respectively. Note that we only evaluate λFM with the static sampler in this paper, considering the efficiency issues of the dynamic samplers.

in Table 1.

### 5.1.2 Baselines and Evaluation

For WE, we compare  $f_{BGD}$  with Skip-gram (Mikolov et al., 2013b) and GloVe (Pennington et al., 2014). For CF, we compare it with SVDFeature (Chen et al., 2012), BPRFM (Rendle, 2012; Rendle et al., 2009), and λFM (Yuan et al., 2016a). For IC, we compare it with SVDFeature, WARP (Weston et al., 2011) and VSE-ens (Guo et al., 2018b). For SVDFeature, we optimize it with the least square loss, and use the negative sampling strategy. The negative examples used for training are uniformly sampled from  $U$ . To show the impact of negative sampling, we vary the size of negative examples for each positive one. E.g., SVDFeature×8 means the positive-to-negative ratio is 1:8. Table 2 summarizes the characteristics of these baselines.

To assess the performance of  $f_{BGD}$  on the WE task, we use the analogical reasoning task introduced by Mikolov et al. (2013a). While to evaluate the CF and IC tasks, we regard them as a ranking or classification task. We report NDCG@10 (Normalized Discounted Cumulative Gain) and MRR@10 (Mean Reciprocal Rank) for CF and AUC (Area Under ROC Curve) for IC.

On the WE task, we evaluate the quality of the word vectors learned from the training datasets. For CF and IC, we adopt the leave-one-out evaluation protocol (Rendle et al., 2009).

### 5.1.3 Experimental Reproducibility

All reported results on each task use a fixed-size embedding dimension without special mention. Specifically, we set embedding dimension as 200, 20 and 100 for the WE, CF and IC tasks respectively. For  $f_{BGD}$ , we set the learning rate  $\gamma$  as 0.05 on all three tasks. Regarding  $r^+$ , we apply the PPMI (positive pointwise mutual information) on the WE task inspired by Levy and Goldberg (2014). For the

Table 3: Results on the word analogy task. “Sem”, “Syn” and “Tot” denote the semantic, syntactic and total accuracy [%]. The positive-to-negative example ratio in SG is 1 : 10 and 1 : 25 in NewsIR and Text8 respectively suggested by Mikolov et al. (2013b).

Model	NewsIR			Text8		
	Sem	Syn	Tot	Sem	Syn	Tot
SG	70.8	47.5	58.1	47.5	32.3	38.6
GloVe	78.8	41.6	58.5	45.1	26.9	34.5
$f_{BGD}$	77.0	46.1	<b>59.7</b>	56.5	30.4	<b>41.3</b>

Table 4: Results on the CF task. NDCG and MRR denote NDCG@10 and MRR@10 respectively. For each measure, the best results for SVDFeature (SVDF) and all models are indicated in bold. The results of SVDF, BPRFM, λFM and  $f_{BGD}$  are reported with all features in the Lastfm dataset.

Model	Yahoo		Lastfm	
	NDCG	MRR	NDCG	MRR
SVDF×1	0.0067	0.0044	0.0436	0.0285
SVDF×4	0.0133	0.009	<b>0.0565</b>	<b>0.0391</b>
SVDF×16	0.0186	0.0132	0.0535	0.0390
SVDF×64	<b>0.0197</b>	<b>0.0139</b>	0.0360	0.0263
SVDF×256	0.0193	0.0139	-	-
BPRFM	0.0178	0.0124	0.1056	0.0740
λFM	0.0200	0.0140	0.1312	0.0950
$f_{BGD}$	<b>0.0224</b>	<b>0.0161</b>	<b>0.1800</b>	<b>0.1371</b>

other two tasks, we simply set it as 1.  $r^-$  can be set as 0 or -0.5. Empirically, we report results of baseline models with optimal hyperparameters whereas for  $f_{BGD}$ , we only report results with above default settings.

## 5.2 ACCURACY AND DISCUSSION

### 5.2.1 Overall Results and Sampling Bias

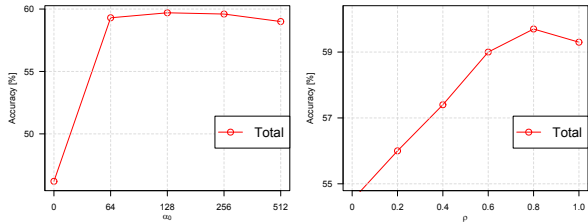
We report results of all models in Tables 3, 4 and 5 for the three tasks. Our first observation is that  $f_{BGD}$  achieves the best performance across all the evaluation metrics and all the datasets. For example,  $f_{BGD}$  outperforms Skip-gram and GloVe in the two text corpora w.r.t. the total accuracy.

Remarkably,  $f_{BGD}$  can easily outperform the strong baselines (e.g., λFM, WARP and VSE-ens) in the ranking and classification tasks, although it optimizes a regression loss which is typically suboptimal for ranking and classification. We attribute the advantage of  $f_{BGD}$  to two aspects: (1) the optimization of each model parameter in  $f_{BGD}$

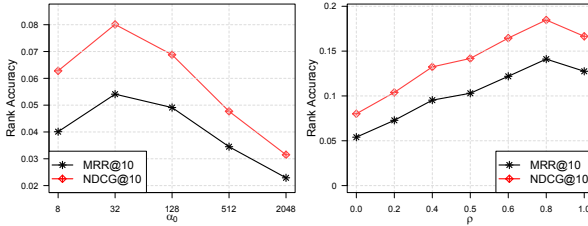
Table 5: Results on the IC task.

Metric	SVDF×1	SVDF×4	SVDF×16	
AUC	0.681	0.724	<b>0.747</b>	
Metric	SVDF×64	WARP	VSE-ens	$f_{BGD}$
AUC	0.663	0.696	0.723	<b>0.772</b>





(a) NewsIR: tune  $\alpha_0$  ( $\rho = 0.8$ ) (b) NewsIR: tune  $\rho$  ( $\alpha_0=128$ )



(c) Lastfm: tune  $\alpha_0$  ( $\rho = 0$ ) (d) Lastfm: tune  $\rho$  ( $\alpha_0=32$ )

Figure 4: Impact of  $\alpha_0$  and  $\rho$  on  $f_{BGD}$ .

Table 6: Accuracy evaluation of  $f_{BGD}$  by adding features.  $u$ ,  $p$ ,  $i$  and  $a$  denote user, last item (song), next item and artist respectively. All hyperparameters of  $f_{BGD}$  are fixed.

Metrics	$(u, i)$	$(u, p, i)$	$(u, p, i, a)$
NDCG@10	0.0416	0.1722	0.1800
MRR@10	0.0281	0.1301	0.1371

makes use of all unlabeled data, whereas the SGD models (including MGD) only use a fraction of sampled data. In other words, important negative examples may be ignored or under-trained; (2) the tailored weighting scheme can help BGD address the imbalanced-class problem in PU data (see Table 1), and assign fine-grained penalties for further improvement.

Our second key observation in the following also verifies the above analysis. As shown in Table 4 (Yahoo),  $SVDF \times 64 > SVDF \times 16 > SVDF \times 4 > SVDF \times 1$ , while  $SVDF \times 256 < SVDF \times 64$ . The results suggest that the performance of SGD models is sensitive to the sampling size of negative examples. To be more specific, one negative sample for a positive example is insufficient to achieve optimal performance; sampling more negative examples is beneficial but too many negative examples may also hurt the performance. In addition, although  $SVDF \times 64 > SVDF \times 1$ , the theoretical computation complexity of  $SVDF \times 64$  is about 32 times higher than  $SVDF \times 1$ . Still in Table 4,  $\lambda FM$  largely improves BPRFM, which demonstrates the impact of sampling distribution of negative examples (see Table 2). However, the true distribution of negative  $(x, y)$  pairs in unlabeled samples is unknown in practice. That is, regardless of what samplers are used, sampling based methods cannot converge to the same loss with all examples or true data.

Table 7: Time Complexity of various optimizers per iteration.  $|X||Y|g$  is much larger than  $(|X| + |Y|)g^2$  and  $|P|g$ . The size relation between  $|P|g$  and  $(|X| + |Y|)g^2$  depends on the sparsity of the data and the embedding dimension  $g$ .

Model	Time Complexity
$SGD \times n$	$O((n+1) P g)$
BGD	$O( X  Y g)$
$f_{BGD}$	$O(( X  +  Y )g^2 +  P g)$

Table 8: Comparison of runtime (second/minute/hour [s/m/h]). ‘‘S’’, ‘‘I’’ and ‘‘T’’ represents the training time for a single iteration, overall iterations and total time respectively. SGD denotes Skip-gram for NewsIR and SVDFeature for other datasets.  $n$  is set as the optimal value, i.e., 10, 4 and 16 for NewsIR, Lastfm and OpenImages respectively.

Model	NewsIR			Lastfm			OpenImages		
	S	I	T	S	I	T	S	I	T
$SGD \times n$	715s	15	179m	156s	50	130m	26m	100	43h
$f_{BGD}$	388s	75	485m	26s	200	87m	576s	200	32h

## 5.2.2 Impact of Weighting in $f_{BGD}$

In this section, we show the impact of the weighting function for  $f_{BGD}$ . We take the NewsIR and Lastfm datasets as an example, and omit similar results in other datasets. Figure 4 shows the prediction quality by tuning  $\alpha_0$  and  $\rho$  in the weight function. We first fix the value of  $\rho$  (e.g., 0 in CF and 0.8 in WE) to study the impact of  $\alpha_0$ . Then, we use the best value of  $\alpha_0$  to study  $\rho$ . As shown, the overall coefficient  $\alpha_0$  largely impacts the performance as the amount of positive and ‘‘negative’’ examples fed in  $f_{BGD}$  is highly imbalanced, the results of which are reflected in (a) and (c). We observe that a proper  $\rho$  can improve the performance, as shown in (b) and (d). The intuition behind the improvement is that high-frequent  $y$  (words or items) that are not observed in  $Y_x^+$  have a higher likelihood to be true negatives, and thus deserve more penalties.

## 5.2.3 Effectiveness in Modelling features

To show the generality of  $f_{BGD}$ , we have described how to apply it to complex embedding models, e.g., SVDFeature used in CARS. For example, we gradually add features for  $f_{BGD}$  on Lastfm and report results in Table 6. As expected,  $f_{BGD}$  performs largely better with  $(u, p, i)$  than  $(u, i)$  and that performance is further enhanced with  $(u, p, i, a)$ . That is,  $f_{BGD}$  yields the best prediction accuracy with all features, demonstrating its power on feature engineering.

## 5.2.4 Runtime

Table 7 summarizes the time complexity of the SGD, BGD and  $f_{BGD}$  algorithms in one iteration when optimizing the pure dot product function. As shown, the complexity of

$f_{BGD}$  is determined by the gradient computation of both positive and unlabeled data, rather than the unlabeled data only. In practice, the runtime is mainly affected by the data sparsity and embedding size. For example, on the WE task,  $O(|P|g)$  is larger than  $O((|X| + |Y|)g^2)$ , while on the IC task  $O((|X| + |Y|)g^2)$  is almost 10 times larger than  $O(|P|g)$  because the NewsIR and Text8 datasets are much denser than the OpenImages dataset (see Table 1). We have compared the overall training time<sup>7</sup> of  $f_{BGD}$  with the NS-based SGD methods in Table 8. It shows that  $f_{BGD}$  obtains comparable efficiency to the classic SGD-based algorithm. More detailed runtime results are shown in Appendix C.

## 6 RELATED WORK

Gradient methods are one of the most popular algorithms to perform optimization in the practice of machine learning. They have also been widely used for training embedding models, and have almost dominated the optimization field. So far the most commonly used gradient optimization method is SGD (Mikolov et al., 2013a,b; Pennington et al., 2014; Rendle et al., 2009; Weston et al., 2011) or a compromise MGD (mini-batch gradient descent) (Li et al., 2014; He et al., 2017), which attempts to approximate the true gradient by a single or a mini-batch of instances with sampling techniques. However, the balance between computing the expensive true gradient based on the whole batch and the immediate gradient based on a single or a fraction of instances could easily result in suboptimal performance. More importantly, on large-scale data the sampling size and distribution for SGD/MGD also significantly affect the convergence rate and prediction accuracy (Bengio and Senécal, 2008). In particular for PU data, it is non-trivial to sample from large and highly imbalanced unlabeled data. Most works deal with this issue by proposing a certain trade-off between efficiency and accuracy. For example, various negative sampling methods have been proposed in recent literature (Mikolov et al., 2013b; Pan et al., 2008; Weston et al., 2012; Yuan et al., 2016a, 2017; Wang et al., 2017; Guo et al., 2018a,b). The basic idea behind this is to select the most informative unlabeled instances as negative examples for an SGD/MGD trainer which, however, easily leads to bias itself. Moreover, all aforementioned works either expose efficiency issues with a dynamic sampler (Weston et al., 2012; Wang et al., 2017; Yuan et al., 2016a) or result in suboptimal training instances with a uniform (Rendle et al., 2009) or static (defined before optimization) sampler (Mikolov et al., 2013a,b; Yuan et al., 2016b, 2017) in practice. Our  $f_{BGD}$  in this work departs from all above studies by adopting BGD to optimize general embedding models with the entire batch of data.

It is worth mentioning that the  $f_{BGD}$  method is inspired

<sup>7</sup>For fairness, efficiency tests for all training models were running on Intel(R) Xeon(R) E5620 @ 2.40GHz CPU and 49G RAM. Note that on the WE task, we implemented all models using C++ with 8 threads in parallel, while on the other two tasks, we implemented the models using Java in a single-thread.

from our extensive empirical studies on previous works (He et al., 2016b; Bayer et al., 2017; Xin et al., 2018). The main difference is that these works are focused on a specific task, e.g., He et al. (2016b); Bayer et al. (2017) are only on recommendation and Xin et al. (2018) is on word representation. Specifically, He et al. (2016b); Xin et al. (2018) worked on the simple matrix factorization model, which cannot be used to incorporate other features, such as contextual variables associated with each observed example. While the alternating least squares (ALS) method proposed in Bayer et al. (2017) can be applied to any  $k$ -separable model<sup>8</sup>, it requires to estimate the second-order derivatives to apply the Newton update and only supports a constant weight on unlabeled examples; moreover, our empirical evidence shows that training with Newton update is (1) very sensitive to initialization point and the regularization term, and (2) highly unstable due to some gradient issues, especially for embedding models (e.g., FM and SVD-Feature) with many input features or large word corpus. By contrast, this work targets at solving the generic PU learning problem with generic embedding models. It leads to a unified solution that is applicable to a wide range of tasks, including but not limited to the ones demonstrated in this paper, with just simple changes on input features.

## 7 CONCLUSION

This work has several key contributions. First, we showed how to efficiently train a class of embedding models by batch gradient descent for positive unlabeled (PU) data. Second, we identified an unstable gradient issue in  $f_{BGD}$  due to the large batched summation of sparse features, and solve it by an intuitive way. To make the prediction accuracy of  $f_{BGD}$  comparable to the state-of-the-arts, we employed a general weighting scheme for unlabeled examples. Despite simple, the weighting scheme could address two challenges, namely imbalanced-class issue in PU data and the differentiation of true negative and unknown examples. We studied the performance of  $f_{BGD}$  in three sub-fields, and showed that  $f_{BGD}$  outperformed state-of-the-art baselines. Compared with the ranking or classification models,  $f_{BGD}$  is clearly a regression model, which means the real-valued scores estimated by it are more informative than those by ranking or classification algorithms. This will make our method highly attractive for practical usage. Moreover, the proposed  $f_{BGD}$  is not limited to the domains discussed in this paper. It potentially benefits many real-world applications with PU data, such as genes association studies (Asgari and Mofrad, 2015; Yang et al., 2014) and data stream mining (Li et al., 2009), etc.

<sup>8</sup>In essence, the concept of  $k$ -separable is to describe a model with a dot product structure of Equation (1).

## References

- E. Asgari and M. R. Mofrad. Continuous distributed representation of biological sequences for deep proteomics and genomics. *PLoS one*, 2015.
- E. Bailey and S. Aeron. Word embeddings via tensor factorization. *arXiv preprint arXiv:1704.02686*, 2017.
- I. Bayer, X. He, B. Kanagal, and S. Rendle. A generic coordinate descent framework for learning from implicit feedback. In *WWW. International World Wide Web Conferences Steering Committee*, 2017.
- Y. Bengio and J.-S. Senécal. Adaptive importance sampling to accelerate training of a neural probabilistic language model. *IEEE Transactions on Neural Networks*, 2008.
- D. Cer, M. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*, 2017.
- T. Chen, W. Zhang, Q. Lu, K. Chen, Z. Zheng, and Y. Yu. Svdfeature: a toolkit for feature-based collaborative filtering. *JMLR*, 2012.
- J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 2011.
- G. Guo, S. Ouyang, F. Yuan, and X. Wang. Approximating word ranking and negative sampling for word embedding. In *IJCAI*, 2018a.
- G. Guo, S. Zhai, F. Yuan, Y. Liu, and X. Wang. Vseens: Visual-semantic embeddings with efficient negative sampling. In *AAAI*, 2018b.
- K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016a.
- X. He and T.-S. Chua. Neural factorization machines for sparse predictive analytics. In *SIGIR*, 2017.
- X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua. Fast matrix factorization for online recommendation with implicit feedback. In *SIGIR*, 2016b.
- X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. Neural collaborative filtering. In *WWW*, 2017.
- Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 2009.
- I. Krasin, T. Duerig, N. Alldrin, and V. Ferrari. Open-images: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from <https://github.com/openimages>*, 2017.
- O. Levy and Y. Goldberg. Neural word embedding as implicit matrix factorization. In *NIPS*, 2014.
- M. Li, T. Zhang, Y. Chen, and A. J. Smola. Efficient mini-batch training for stochastic optimization. In *KDD*, 2014.
- X.-L. Li, P. S. Yu, B. Liu, and S.-K. Ng. Positive unlabeled learning for data stream classification. In *ICDM*, 2009.
- T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013a.
- T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013b.
- P. Ng. dna2vec: Consistent vector representations of variable-length k-mers. *arXiv preprint arXiv:1701.06279*, 2017.
- R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *ICDM*, 2008.
- J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.
- S. Rendle. Factorization machines with libFM. *ACM Trans. Intell. Syst. Technol.*, 2012.
- S. Rendle and C. Freudenthaler. Improving pairwise learning for item recommendation from implicit feedback. In *WSDM*, 2014.
- S. Rendle and L. Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In *WSDM*, 2010.
- S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *UAI*, 2009.
- S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.
- J. Wang, L. Yu, W. Zhang, Y. Gong, Y. Xu, B. Wang, P. Zhang, and D. Zhang. Irgan: A minimax game for unifying generative and discriminative information retrieval models. In *SIGIR*, 2017.
- J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *IJCAI*, 2011.
- J. Weston, C. Wang, R. Weiss, and A. Berenzweig. Latent collaborative retrieval. *ICML*, 2012.
- X. Xin, F. Yuan, X. He, and J. M. Jose. Batch is not heavy: Learning word representations from all samples. In *ACL*, 2018.
- P. Yang, X. Li, H.-N. Chua, C.-K. Kwoh, and S.-K. Ng. Ensemble positive unlabeled learning for disease gene identification. *PLoS one*, 2014.
- F. Yuan, G. Guo, J. M. Jose, L. Chen, H. Yu, and W. Zhang. Lambdafm: learning optimal ranking with factorization machines using lambda surrogates. In *CIKM*, 2016a.
- F. Yuan, J. M. Jose, G. Guo, L. Chen, H. Yu, and R. S. Alkhaldeh. Joint geo-spatial preference and pairwise ranking for point-of-interest recommendation. In *ICTAI. IEEE*, 2016b.
- F. Yuan, G. Guo, J. M. Jose, L. Chen, H. Yu, and W. Zhang. Boostfm: Boosted factorization machines for top-n feature-based recommendation. In *IUI*, 2017.

---

# Soft-Robust Actor-Critic Policy-Gradient

---

**Esther Derman**

Technion, Israel

estherderman@technion.ac.il

**Timothy A. Mann**

Google Deepmind, UK

timothymann@google.com

**Daniel J. Mankowitz**

Technion, Israel

danielm@campus.technion.ac.il

**Shie Mannor**

Technion, Israel

shie@ee.technion.ac.il

## Abstract

Robust Reinforcement Learning aims to derive an optimal behavior that accounts for model uncertainty in dynamical systems. However, previous studies have shown that by considering the worst case scenario, robust policies can be overly conservative. Our *soft-robust* framework is an attempt to overcome this issue. In this paper, we present a novel Soft-Robust Actor-Critic algorithm (SR-AC). It learns an optimal policy with respect to a distribution over an uncertainty set and stays robust to model uncertainty but avoids the conservativeness of robust strategies. We show the convergence of SR-AC and test the efficiency of our approach on different domains by comparing it against regular learning methods and their robust formulations.

## 1 INTRODUCTION

Markov Decision Processes (MDPs) are commonly used to model sequential decision making in stochastic environments. A strategy that maximizes the accumulated expected reward is then considered as optimal and can be learned from sampling. However, besides the uncertainty that results from stochasticity of the environment, model parameters are often estimated from noisy data or can change during testing [Mannor et al., 2007; Roy et al., 2017]. This second type of uncertainty can significantly degrade the performance of the optimal strategy from the model’s prediction.

Robust MDPs were proposed to address this problem [Iyengar, 2005; Nilim and El Ghaoui, 2005; Tamar et al., 2014]. In this framework, a transition model is assumed to belong to a known uncertainty set and an optimal strategy is learned under the worst parameter realizations. Although the robust approach is computationally efficient

when the uncertainty set is state-wise independent, compact and convex, it can lead to overly conservative results [Mannor et al., 2012, 2016; Xu and Mannor, 2012; Yu and Xu, 2016].

For example, consider a business scenario where an agent’s goal is to make as much money as possible. It can either create a startup which may make a fortune but may also result in bankruptcy. Alternatively, it can choose to live off school teaching and have almost no risk but low reward. By choosing the teaching strategy, the agent may be overly conservative and not account for opportunities to invest in his own promising projects. Our claim is that one could relax this conservativeness and construct a softer behavior that interpolates between being aggressive and robust. Ideally, the *soft-robust* agent should stay agnostic to outside financing uncertainty but still be able to take advantage of the startup experience.

This type of dilemma can be found in various domains. In the financial market, investors seek a good trade-off between low risk and high returns regarding portfolio management [Mitchell and Smetters, 2013]. In strategic management, product firms must choose the amount of resources they put into innovation. A conservative strategy would then consist of innovating only under necessary conditions [Miller and Friesen, 1982].

In this paper, we focus on learning a *soft-robust policy* (defined below) by incorporating soft-robustness into an online actor-critic algorithm and show its convergence properties. Existing works mitigate conservativeness of robust MDP either by introducing coupled uncertainties [Mannor et al., 2012, 2016] or by assuming prior information on the uncertainty set [Xu and Mannor, 2012; Yu and Xu, 2016]. They use dynamic programming techniques to estimate a robust policy. However, these methods present some limiting restrictions such as non-scalability and offline estimation. Besides being computationally more efficient than batch learning [Wiering and van Otterlo, 2012], the use of an online algorithm is of significant interest in

robust MDPs because it can detect non-adversarial state-action pairs along a trajectory and result in less conservative results, something which cannot be performed when solving the planning problem [Lim et al., 2016]. Other works have attempted to incorporate robustness into an online algorithm for policy optimization [Mankowitz et al., 2018; Tamar et al., 2015]. Although these approaches can deal with large domains, a sampling procedure is required for each critic estimate in Tamar et al. [2015], which differs from the strictly-speaking actor-critic. In Mankowitz et al. [2018], the authors introduce a robust version of actor-critic policy-gradient but its convergence results are only shown for the actor updates. Moreover, these works target the robust solution which may be too conservative. We review all existing methods in Section 7 and compare them to our approach.

To the best of our knowledge, our proposed work is the first attempt to incorporate a soft form of robustness into an online algorithm that has convergence guarantees besides being computationally scalable. We deal with the curse of dimensionality by using function approximation that parameterizes the expected value within a space of much smaller dimension than the state space. By fixing a distribution over the uncertainty set, the induced soft-robust actor-critic learns a locally optimal policy in an online manner. Under mild assumptions on the set of distributions and uncertainty set, we show that our novel Soft-Robust Actor-Critic (SR-AC) algorithm converges. We test the performance of soft-robustness on different domains, including a large state space with continuous actions. As far as we know, no other work has previously incorporated robustness into continuous action spaces.

Our specific contributions are: (1) A soft-robust derivation of the objective function for policy-gradient; (2) An SR-AC algorithm that uses stochastic approximation to learn a variant of distributionally robust policy in an online manner; (3) Convergence proofs of SR-AC; (4) An experiment of our framework to different domains that shows the efficiency of soft-robust behaviors in a continuous action space as well. All proofs can be found in the Appendix.

## 2 BACKGROUND

In this section, we introduce the background material related to our soft-robust approach.

**Robust MDP** A robust MDP is a tuple  $\langle \mathcal{X}, \mathcal{A}, r, \mathcal{P} \rangle$  where  $\mathcal{X}$  is a finite state-space,  $\mathcal{A}$  is a finite set of actions,  $r : \mathcal{X} \times \mathcal{A} \rightarrow \mathbb{R}$  is the immediate reward function which is deterministic and bounded and  $\mathcal{P}$  is a set of transition matrices. We assume that  $\mathcal{P}$  is structured as a cartesian product  $\bigotimes_{x \in \mathcal{X}} \mathcal{P}_x$ , which is known as the

rectangularity assumption [Nilim and El Ghaoui, 2005]. Given a state  $x \in \mathcal{X}$ , the uncertainty set  $\mathcal{P}_x$  is a family of transition models  $p_x \in \mathcal{P}_x$  we represent as vectors in which the transition probabilities of each action are arranged in the same block. For  $x, y \in \mathcal{X}$  and  $a \in \mathcal{A}$ , denote by  $p(x, a, y)$  the probability of getting from state  $x$  to state  $y$  given action  $a$ .

At timestep  $t$ , the agent is in state  $x_t$  and chooses an action  $a_t$  according to a stochastic policy  $\pi : \mathcal{X} \rightarrow \mathcal{M}(\mathcal{A})$  that maps each state to a probability distribution over the action space,  $\mathcal{M}(\mathcal{A})$  denoting the set of distributions over  $\mathcal{A}$ . It then gets a reward  $r_{t+1}$  and is brought to state  $x_{t+1}$  with probability  $p(x_t, a_t, x_{t+1})$ .

**Policy-Gradient** Policy-gradient methods are commonly used to learn an agent policy. A policy  $\pi$  is parametrized by  $\theta$  and estimated by optimizing an objective function using stochastic gradient descent. A typical objective to be considered is the average reward function

$$\begin{aligned} J_p(\pi) &= \lim_{T \rightarrow +\infty} \mathbb{E}^p \left[ \frac{1}{T} \sum_{t=0}^{T-1} r_{t+1} \mid \pi \right] \\ &= \sum_{x \in \mathcal{X}} d_p^\pi(x) \sum_{a \in \mathcal{A}} \pi(x, a) r(x, a) \end{aligned}$$

where  $r_t$  is the reward at time  $t$ ,  $p$  an aperiodic and irreducible transition model under which the agent operates and  $d_p^\pi$  is the stationary distribution of the Markov process induced by  $p$  under policy  $\pi$ . The gradient objective has previously been shown to be

$$\nabla_\theta J_p(\pi) = \sum_{x \in \mathcal{X}} d_p^\pi(x) \sum_{a \in \mathcal{A}} \nabla_\theta \pi(x, a) Q_p^\pi(x, a)$$

where  $Q_p^\pi(x, a)$  is the expected differential reward associated with state-action pair  $(x, a)$ . This gradient is then used to update the policy parameters according to:  $\theta_{t+1} = \theta_t + \beta_t \nabla_\theta J_p(\pi)$ , with  $\beta_t$  a positive step-size [Sutton et al., 2000].

**Actor-Critic Algorithm** Theoretical analysis and empirical experiments have shown that regular policy-gradient methods present a major issue namely high variance in the gradient estimates that results in slow convergence and inefficient sampling [Grondman et al., 2012]. First proposed by Barto et al. [1983], actor-critic methods attempt to reduce the variance by using a critic that estimates the value function. They borrow elements from both value function and policy-based methods. The value function estimate plays the role of a critic that helps evaluating the performance of the policy. As in policy-based methods, the actor then uses this signal to update policy parameters in the direction of a gradient estimate of a performance measure. Under appropriate conditions, the resulting algorithm is tractable and converges to a locally optimal policy [Bhatnagar et al., 2009; Konda and Tsitsiklis, 2000].

**Deep Q-networks** Deep Q-Networks (DQNs) have proven their capability of solving complex learning tasks such as Atari video games [Mnih et al., 2013]. The Q-learning of Watkins and Dayan [1992] typically learns a greedy or  $\epsilon$ -greedy policy by updating the Q-function based on a TD-error. In Deep Q-learning [Mnih et al., 2013, 2015], a non-linear function such as a neural network is used as an approximator of the Q-function. It is referred to as a Q-network. The agent is then trained by optimizing the induced TD loss function thanks to stochastic gradient descent. Like actor-critic, DQN is an online algorithm that aims at finding an optimal policy. The main difference with actor-critic is that it is *off-policy*: it learns a greedy strategy while following an arbitrary behavior [Mnih et al., 2013].

**Deep Deterministic Policy-Gradient** Since DQN acts greedily at each iteration, it can only handle small action spaces. The Deep Deterministic Policy-Gradient (DDPG) is an *off-policy* algorithm that can learn behaviors in continuous action spaces [Lillicrap et al., 2016]. It is based on an actor-critic architecture that follows the same baseline as in DQN. The critic estimates the current Q-value of the actor using a TD-error while the actor is updated according to the critic. This update is based on the chain rule principle which establishes equivalence between the stochastic and the deterministic policy gradient [Silver et al., 2014].

### 3 SOFT-ROBUSTNESS

#### 3.1 SOFT-ROBUST FRAMEWORK

Unlike robust MDPs that maximize the worst-case performance, we fix a prior on how transition models are distributed over the uncertainty set. A distribution over  $\mathcal{P}$  is denoted by  $\omega$  and is structured as a cartesian product  $\otimes_{x \in \mathcal{X}} \omega_x$ . We find the same structure in Xu and Mannor [2012]; Yu and Xu [2016]. Intuitively,  $\omega$  can be thought as the way the adversary distributes over different transition models. The product structure then means that this adversarial distribution only depends on the current state of the agent without taking into account its whole trajectory. This defines a probability distribution  $\omega_x$  over  $\mathcal{P}_x$  independently for each state.

We further assume that  $\omega$  is non-diffuse. This implies that the uncertainty set is non-trivial with respect to  $\omega$  in a sense that the distribution does not affect zero mass to all of the models.

#### 3.2 SOFT-ROBUST OBJECTIVE

Throughout this paper, we make the following assumption:

**Assumption 3.1.** *Under any policy  $\pi$ , the Markov chains resulting from any of the MDPs with transition laws  $p \in \mathcal{P}$  are irreducible and aperiodic.*

Define  $d_p^\pi$  as the stationary distribution of the Markov chain that results from following policy  $\pi$  under transition model  $p \in \mathcal{P}$ .

**Definition 3.1.** *We call soft-robust objective or soft-robust average reward the function  $\bar{J}(\pi) := \mathbb{E}_{p \sim \omega} [J_p(\pi)]$ .*

The distribution  $\omega$  introduces a softer form of robustness in the objective function because it averages over the uncertainty set instead of considering the worst-case scenario. It also gives flexibility over the level of robustness one would like to keep. A robust strategy would then consist of putting more mass on pessimistic transition models. Likewise, a distribution that puts all of its mass on one target model would lead to an aggressive behavior and result in model misspecification.

The *soft-robust differential reward* is given by  $\bar{Q}^\pi(x, a) := \mathbb{E}_{p \sim \omega} [Q_p^\pi(x, a)]$  where

$$Q_p^\pi(x, a) := \mathbb{E}^p \left[ \sum_{t=0}^{+\infty} r_{t+1} - J_p(\pi) \mid x_0 = x, a_0 = a, \pi \right].$$

Similarly, we introduce the quantity

$$\bar{V}^\pi(x) := \sum_{a \in \mathcal{A}} \pi(x, a) \bar{Q}^\pi(x, a) = \mathbb{E}_{p \sim \omega} [V_p^\pi(x)]$$

with  $V_p^\pi(x) := \sum_{a \in \mathcal{A}} \pi(x, a) Q_p^\pi(x, a)$ . We will interchangeably term it as *soft-robust expected differential reward* or *soft-robust value function*.

#### 3.3 SOFT-ROBUST STATIONARY DISTRIBUTION

The above performance objective  $\bar{J}(\pi)$  cannot as yet be written as an expectation of the reward over a stationary distribution because of the added measure  $\omega$  on transition models. Define the average transition model as  $\bar{p} := \mathbb{E}_{p \sim \omega} [p]$ . It corresponds to the transition probability that results from distributing all transition models according to  $\omega$ . In analogy to the transition probability that minimizes the reward for each given state and action in the robust transition function [Mankowitz et al., 2018], our average model rather selects the expected distribution over all the uncertainty set for each state and action. Under Assumption 3.1, we can show that the transition  $\bar{p}$  as defined is irreducible and aperiodic, which ensures the existence of a unique stationary law we will denote by  $\bar{d}^\pi$ .

**Proposition 3.1** (Stationary distribution in the average transition model). *Under Assumption 3.1, the average transition matrix  $\bar{p} := \mathbb{E}_{p \sim \omega}[p]$  is irreducible and aperiodic. In particular, it admits a unique stationary distribution.*

As in regular MDPs, the soft-robust average reward satisfies a Poisson equation, as it was first stated in the discounted reward case in Lemma 3.1 of Xu and Mannor [2012]. The following proposition reformulates this result for the average reward.

**Proposition 3.2** (Soft-Robust Poisson equation).

$$\begin{aligned} \bar{J}(\pi) + \bar{V}^\pi(x) \\ = \sum_{a \in \mathcal{A}} \pi(x, a) \left( r(x, a) + \sum_{x' \in \mathcal{X}} \bar{p}(x, a, x') \bar{V}^\pi(x') \right) \end{aligned}$$

This Poisson equation enables us to establish an equivalence between the expectation of the stationary distributions over the uncertainty set and the stationary distribution of the average transition model, naming  $\bar{d}^\pi(x) = \mathbb{E}_{p \sim \omega}[d_p^\pi(x)]$  with  $x \in \mathcal{X}$ . Indeed, we have the following:

**Corollary 3.1.** *Recall  $\bar{d}^\pi$  the stationary distribution for the average transition model  $\bar{p}$ . Then*

$$\bar{J}(\pi) = \sum_{x \in \mathcal{X}} \bar{d}^\pi(x) \sum_{a \in \mathcal{A}} \pi(x, a) r(x, a).$$

The goal is to learn a policy that maximizes the soft-robust average reward  $\bar{J}$ . We use a policy-gradient method for that purpose.

## 4 SOFT-ROBUST POLICY-GRADIENT

In policy-gradient methods, we consider a class of parametrized stochastic policies  $\pi_\theta : \mathcal{X} \rightarrow \mathcal{M}(\mathcal{A})$  with  $\theta \in \mathbb{R}^{d_\theta}$  and estimate the gradient of the objective function  $\bar{J}$  with respect to policy parameters in order to update the policy in the direction of the estimated gradient of  $\bar{J}$ . The optimal set of parameters thus obtained is denoted by

$$\theta^* := \arg \max_{\theta} \bar{J}(\pi_\theta).$$

When clear in the context, we will omit the subscript  $\theta$  in  $\pi_\theta$  for notation ease. We further make the following assumption, which is standard in policy-gradient literature:

**Assumption 4.1.** *For any  $(x, a) \in \mathcal{X} \times \mathcal{A}$ , the mapping  $\theta \mapsto \pi_\theta(x, a)$  is continuously differentiable with respect to  $\theta$ .*

Using the same method as in Sutton et al. [2000], we can derive the gradient of the soft-robust average reward thanks to the previous results.

**Theorem 4.1** (Soft-Robust Policy-Gradient). *For any MDP satisfying previous assumptions, we have*

$$\nabla_{\theta} \bar{J}(\pi) = \sum_{x \in \mathcal{X}} \bar{d}^\pi(x) \sum_{a \in \mathcal{A}} \nabla_{\theta} \pi(x, a) \bar{Q}^\pi(x, a).$$

In order to manage with large state spaces, we also introduce a linear approximation of  $\bar{Q}^\pi$  we define as  $f_w(x, a) := w^T \psi_{xa}$ . Sutton et al. [2000] showed that if the features  $\psi_{xa}$  satisfy a compatibility condition and the approximation is locally optimal, then we can use it in place of  $\bar{Q}^\pi$  and still point roughly in the direction of the true gradient.

In the case of soft-robust average reward, this defines a soft-robust gradient update that possesses the ability to incorporate function approximation, as stated in the following result. The main difference with that of Sutton et al. [2000] is that we combine the dynamics of the system with distributed transitions over the uncertainty set.

**Theorem 4.2** (Soft-Robust Policy-Gradient with Function Approximation). *Let  $f_w : \mathcal{X} \times \mathcal{A} \rightarrow \mathbb{R}$  be a linear approximator of the soft-robust differential reward  $\bar{Q}^\pi$ . If  $f_w$  minimizes the mean squared error*

$$\mathcal{E}^\pi(w) := \sum_{x \in \mathcal{X}} \bar{d}^\pi(x) \sum_{a \in \mathcal{A}} \pi(x, a) \left[ \bar{Q}^\pi(x, a) - f_w(x, a) \right]^2$$

*and is compatible in a sense that  $\nabla_w f_w(x, a) = \nabla_{\theta} \log \pi(x, a)$ , then*

$$\nabla_{\theta} \bar{J}(\pi) = \sum_{x \in \mathcal{X}} \bar{d}^\pi(x) \sum_{a \in \mathcal{A}} \nabla_{\theta} \pi(x, a) f_w(x, a)$$

We can further improve our gradient estimate by reducing its variance. One direct method to do so is to subtract a baseline  $b(x)$  from the previous gradient update. It is easy to show that this will not affect the gradient derivation. In particular, Bhatnagar et al. [2009] proved that the value function minimizes the variance. It is therefore a proper baseline to choose. We can thus write the following:

$$\begin{aligned} \nabla_{\theta} \bar{J}(\pi) &= \sum_{x \in \mathcal{X}} \bar{d}^\pi(x) \sum_{a \in \mathcal{A}} \nabla_{\theta} \pi(x, a) \left( \bar{Q}^\pi(x, a) - \bar{V}^\pi(x) \right) \\ &= \sum_{x \in \mathcal{X}} \bar{d}^\pi(x) \sum_{a \in \mathcal{A}} \pi(x, a) \psi_{xa} \bar{A}^\pi(x, a), \end{aligned} \tag{1}$$

where  $\bar{A}^\pi(x, a)$  is the *soft-robust advantage function* defined by  $\bar{A}^\pi(x, a) := \bar{Q}^\pi(x, a) - \bar{V}^\pi(x)$ .

## 5 SOFT-ROBUST ACTOR-CRITIC ALGORITHM

In this section, we present our SR-AC algorithm which is defined as Algorithm 1. This novel approach incorporates a variation of distributional robustness into an online algorithm that effectively learns an optimal policy in a scalable manner. Under mild assumptions, the resulting two-timescale stochastic approximation algorithm converges to a locally optimal policy.

### 5.1 SR-AC ALGORITHM

An uncertainty set and a nominal model without uncertainty are provided as inputs. In practice, the nominal model and the uncertainty set can respectively be an estimate of the transition model resulting from data sampling and its corresponding confidence interval. A distribution  $\omega$  over the uncertainty set is also provided. It corresponds to our prior information on the uncertainty set. The step-size sequences  $(\alpha_t, \beta_t, \xi_t; t \geq 0)$  consist of small non-negative numbers properly chosen by the user (see Appendix for more details).

At each iteration, samples are generated using the nominal model and the current policy. These are utilized to update the soft-robust average reward (Line 5) and the critic (Line 7) based on an estimate of a soft-robust TD-error we detail further. In our setting, the soft-robust value function plays the role of the critic according to which the actor parameters are updated. We then exploit the critic to improve our policy by updating the policy parameters in the direction of a gradient estimate for the soft-robust objective (Line 8). This process is repeated until convergence.

### 5.2 CONVERGENCE ANALYSIS

We establish convergence of SR-AC to a local maximum of the soft-robust objective function by following an ODE approach [Kushner and Yin, 1997].

Consider  $\hat{V}$  and  $\hat{J}$  as unbiased estimates of  $\bar{V}$  and  $\bar{J}$  respectively. Calculating  $\delta_t$  (Line 6 in Algorithm 1) requires an estimate of the soft-robust average-reward that can be obtained by averaging over samples given immediate reward  $r$  and distribution  $\omega$  (Line 5). In order to get an estimate of the soft-robust differential value  $\hat{V}$ , we use linear function approximation. Considering  $\varphi$  as a  $d_2$ -dimensional feature extractor over the state space  $\mathcal{X}$ , we may then approximate  $\bar{V}^\pi(x)$  as  $v^T \varphi_x$ , where  $v$  is a  $d_2$ -dimensional parameter vector that we tune using linear TD. This results in the following soft-robust TD-error:

$$\delta_t := r_{t+1} - \hat{J}_{t+1} + \sum_{x' \in \mathcal{X}} \bar{p}(x_t, a_t, x') v_t^T \varphi_{x'} - v_t^T \varphi_{x_t},$$

---

### Algorithm 1 SR-AC

---

- 1: **Input:**  $\mathcal{P}$  - An uncertainty set;  $\hat{p} \in \mathcal{P}$  - A nominal model;  $\omega$  - A distribution over  $\mathcal{P}$ ;  $f_x$  - A feature extractor for the SR value function;
  - 2: **Initialize:**  $\theta = \theta_0$  - An arbitrary policy parameter;  $v = v_0$  - An arbitrary set of value function parameters;  $\alpha_0, \beta_0, \xi_0$  - Initial learning-rates;  $x_0$  - Initial state
  - 3: **repeat**
  - 4:   Act under  $a_t \sim \pi_{\theta_t}(x_t, a_t)$   
    Observe next state  $x_{t+1}$  and reward  $r_{t+1}$
  - 5:   **SR Average Reward Update:**  
     $\hat{J}_{t+1} = (1 - \xi_t)\hat{J}_t + \xi_t r_{t+1}$
  - 6:   **SR TD-Error:**  
     $\delta_t = r_{t+1} - \hat{J}_{t+1} + \sum_{x' \in \mathcal{X}} \bar{p}(x_t, a_t, x') \hat{V}_{x'} - \hat{V}_{x_t}$
  - 7:   **Critic Update:**  $v_{t+1} = v_t + \alpha_t \delta_t \varphi_{x_t}$
  - 8:   **Actor Update:**  $\theta_{t+1} = \theta_t + \beta_t \delta_t \psi_{x_t a_t}$
  - 9: **until** convergence
  - 10: **Return:** SR policy parameters  $\theta$  and SR value-function parameters  $v$
- 

where  $v_t$  corresponds to the current estimate of the soft-robust value function parameter.

As in regular MDPs, when doing linear TD learning, the function approximation of the value function introduces a bias in the gradient estimate [Bhatnagar et al., 2009]. Denoting it as  $e^\pi$ , we have  $E[\widehat{\nabla_\theta J}(\pi) | \theta] = \nabla_\theta \bar{J}(\pi) + e^\pi$  (see Appendix). This bias term then needs to be small enough in order to ensure convergence.

Convergence of Algorithm 1 can be established by applying Theorem 2 from Bhatnagar et al. [2009] which exploits Borkar’s work on two-timescale algorithms [1997]. The convergence result is presented as Theorem 5.1.

**Theorem 5.1.** *Under all the previous assumptions, given  $\epsilon > 0$ , there exists  $\delta > 0$  such that for a parameter vector  $\theta_t, t \geq 0$  obtained using the algorithm, if  $\sup_{\pi_t} \|e^{\pi_t}\| < \delta$ , then the SR-AC algorithm converges almost surely to an  $\epsilon$ -neighborhood of a local maximum of  $\bar{J}$ .*

## 6 NUMERICAL EXPERIMENTS

We demonstrate the performance of soft-robustness on various domains of finite as well as continuous state and action spaces. We used the existing structure of OpenAI Gym environments to run our experiments [Brockman et al., 2016].

### 6.1 DOMAINS

**Single-step MDP** We consider a simplified formulation of the startup vs teaching dilemma described in Section 1.



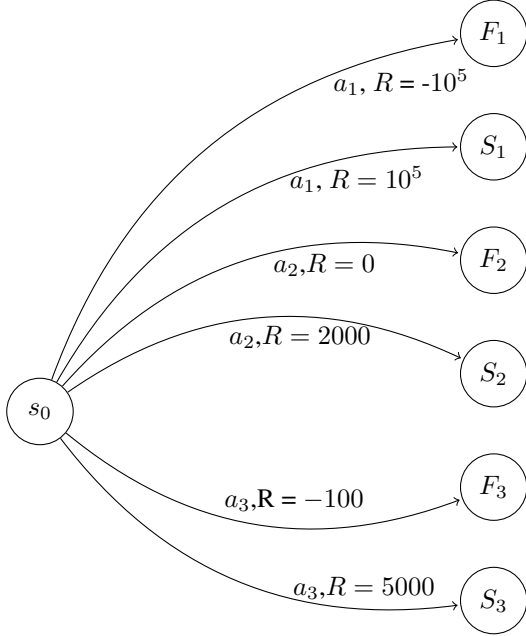


Figure 1: Illustration of the MDP with initial state  $s_0$ . States  $F_1, F_2, F_3$  correspond to failing scenarios for each action. The succeeding states are represented by states  $S_1, S_2, S_3$ .

The problem is modeled as a 7-state MDP in which one action corresponds to one strategy. An illustration of this construction is given in Figure 1. At the starting state  $s_0$ , the agent chooses one of three actions. Action  $a_1$  [corresponds to the startup adventure] may lead it to a very high reward in case of success but can be catastrophic in case of failure. Action  $a_2$  [corresponds to the teaching carrier] leads it to low positive reward in case of success with no possibility of negative reward. Action  $a_3$  [corresponds to an intermediate strategy] can lead to an intermediate positive reward with a slight risk of negative reward. Depending on the action it chose and if it succeeded or not, the agent is brought to one of the six right-hand states and receives the corresponding reward. It is brought back to  $s_0$  at the end of each episode. We assume the probability of success to be the same for all three actions.

**Cart-Pole** In the Cart-Pole system, the agent’s goal consists of balancing a pole atop a cart in a vertical position. It is modeled as a continuous MDP in which each state consists of a 4-tuple  $\langle x, \dot{x}, \theta, \dot{\theta} \rangle$  which represents the cart position, the cart speed, the pole angle with respect to the vertical and its angular speed respectively. The agent can make two possible actions: apply a constant force either to the right or to the left of the pole. It gets a positive reward of 1 if the pole has not fallen down and if it stayed in the boundary sides of the screen. If it terminates, the agent receives a reward of 0. Since each episode lasts for

200 timesteps, the maximal reward an agent can get is 200 over one episode.

**Pendulum** In the inverted pendulum problem, a pendulum starts in a random position and the goal is to swing it up so that it stabilizes upright. The state domain consists in a 2-tuple  $\langle \theta, \dot{\theta} \rangle$  which represents the pendulum angle with respect to the vertical and its angular velocity. At each timestep, the agent’s possible actions belong to a continuous interval  $[-a, a]$  which represents the force level being applied. Since there is no specified termination, we establish a maximal number of 200 steps for each episode.

## 6.2 UNCERTAINTY SETS

For each experiment, we generate an uncertainty set  $\mathcal{P}$  before training. In the single-step MDP, we sample from 5 different probabilities of success using a uniform distribution over  $[0, 1]$ . In Cart-Pole, we sample 5 different lengths from a normal distribution centered at the nominal length of the pole which we fix at 0.3. We proceed similarly for Pendulum by generating 10 different masses of pendulum around a nominal mass of 2. Each corresponding model thus generates a different transition function. We then sample the average model by fixing  $\omega$  as a realization of a Dirichlet distribution. A soft-robust update for the actor is applied by taking the optimal action according to this average transition function.

## 6.3 LEARNING ALGORITHMS

We trained the agent on the nominal model in each experiment. The soft-robust agent was learned using SR-AC in the single-step MDP. In Cart-Pole, we run a soft-robust version of a DQN algorithm. The soft-robust agent in Pendulum was trained using a soft-robust DDPG.

**Soft-Robust AC** We analyze the performance of SR-AC by training a soft-robust agent on the single-step MDP. We run a regular AC algorithm to derive an aggressive policy and learn a robust behavior by using a robust formulation of AC which consists in replacing the TD-error with a robust TD-error, as implemented in Mankowitz et al. [2018]. The derived soft-robust agent is then compared with the resulting aggressive and robust strategies respectively.

**Soft-Robust DQN** Robustness has already been incorporated in DQN [Di-Castro Shashua and Mannor, 2017]. The Q-network addressed there performs an online estimation of the Q-function by minimizing at each timestep

$t$  the following robust TD-error:

$$\begin{aligned} \delta_{dqn,t}^{rob} &:= r(x_t, a_t) - Q(x_t, a_t) \\ &+ \gamma \inf_{p \in \mathcal{P}} \sum_{x' \in \mathcal{X}} p(x_t, a_t, x') \max_{a' \in \mathcal{A}} Q(x', a'), \end{aligned}$$

where  $\gamma$  is a discount factor.

In our experiments, we incorporate a soft-robust TD-error inside a DQN that trains a soft-robust agent according to the induced loss function. The soft-robust TD-error for DQN is given by:

$$\begin{aligned} \delta_{dqn,t}^{srob} &:= r(x_t, a_t) - Q(x_t, a_t) \\ &+ \gamma \sum_{x' \in \mathcal{X}} \bar{p}(x_t, a_t, x') \max_{a' \in \mathcal{A}} Q(x', a') \end{aligned}$$

We use the Cart-Pole domain to compare the resulting policy with the aggressive and robust strategies that were obtained from a regular and a robust DQN respectively.

**Soft-Robust DDPG** Define  $\mu_t$  as the estimated deterministic policy at step  $t$ . We incorporate robustness in DDPG by updating the critic network according to the following robust TD-error:

$$\begin{aligned} \delta_{ddpg,t}^{rob} &:= r(x_t, a_t) - Q(x_t, a_t) \\ &+ \gamma \inf_{p \in \mathcal{P}} \sum_{x' \in \mathcal{X}} p(x_t, a_t, x') Q(x', \mu(x_t)), \end{aligned}$$

Similarly, we incorporate soft-robustness in DDPG by using the soft-robust TD-error:

$$\begin{aligned} \delta_{ddpg,t}^{srob} &:= r(x_t, a_t) - Q(x_t, a_t) \\ &+ \gamma \sum_{x' \in \mathcal{X}} \bar{p}(x_t, a_t, x') Q(x', \mu(x_t)) \end{aligned}$$

We compare the resulting soft-robust DDPG with its regular and robust formulations in the Pendulum domain.

## 6.4 IMPLEMENTATION

For each experiment, we train the agent on the nominal model but incorporate soft-robustness during learning. A soft-robust policy is learned thanks to SR-AC in the single-step MDP. We use a linear function approximation with 5 features to estimate the value function. For Cart-Pole, we run a DQN using a neural network of 3 fully-connected hidden layers with 128 weights per layer and ReLU activations. In Pendulum, a DDPG algorithm learns a policy based on two target networks: the actor and the critic network. Both have 2 fully-connected hidden layers with 400 and 300 units respectively. We use a tanh activation for the actor and a ReLU activation for the critic output. We chose the ADAM optimizer to minimize all the induced loss functions. We used constant

learning rates which worked well in practice. Each agent was trained over 3000 episodes for the single-step MDP and Cartpole and tested over 600 episodes per parameter setting. For Pendulum, the agents were trained over 5000 episodes evaluated over 800 episodes per parameter setting. Other hyper-parameter values can be found in the Appendix.

## 6.5 RESULTS

**Single-step MDP** Figure 2 shows the evolution of the performance for all three agents during training. It becomes more stable along training time, which confirms convergence of SR-AC. We see that the aggressive agent performs best due to the highest reward it can reach on the nominal model. The soft-robust agent gets rewards in between the aggressive and the robust agent which performs the worst due to its pessimistic learning method.



Figure 2: Comparison of robust, soft-robust and aggressive agents during training. One training epoch corresponds to 300 episodes.

The evaluation of each strategy is represented in Figure 3. As the probability of success gets low, the performance of the aggressive agent drops down below the robust and the soft-robust agents, although it performs best when the probability of success gets close to 1. The robust agent stays stable independently of the parameters but underperforms soft-robust agent which presents the best balance between high reward and low risk. We noticed that depending on the weighting distribution initially set, soft-robustness tends to being more or less aggressive (see Appendix). Incorporating a distribution over the uncertainty set thus gives significant flexibility on the level of aggressiveness to be assigned to the soft-robust agent.

**Cart-Pole** In Figure 4, we show the performance of all three strategies over different values of pole length during

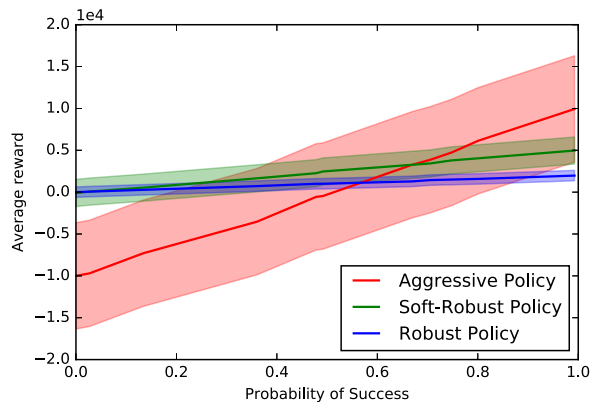


Figure 3: Average reward for AC, robust AC and SR-AC methods

testing. Similarly to our previous example, the non-robust agent performs well around the nominal model but its reward degrades on more extreme values of pole length. The robust agent keeps a stable reward under model uncertainty which is consistent with the results obtained in Di-Castro Shashua and Mannor [2017]; Mankowitz et al. [2018]. However, it is outperformed by the soft-robust agent around the nominal model. Furthermore, the soft-robust strategy shows an equilibrium between aggressiveness and robustness thus leading to better performance than the non-robust agent on larger pole lengths. We trained a soft-robust agent on other weighting distributions and noted that depending on its structure, soft-robustness interpolates between aggressive and robust behaviors (see Appendix).

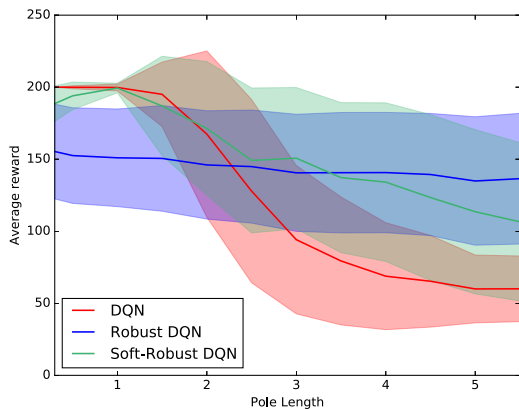


Figure 4: Average reward performance for DQN, robust DQN and soft-robust DQN

**Pendulum** Figure 5 shows the performance of all three

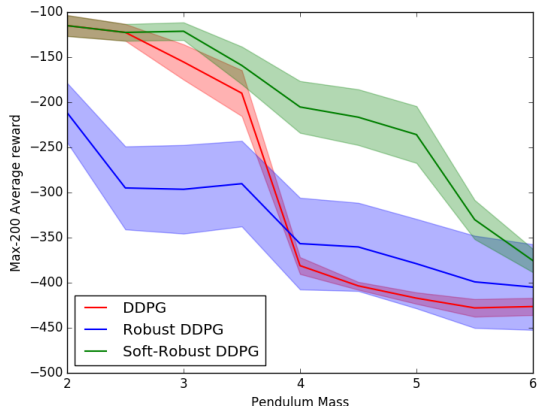


Figure 5: Max-200 episodes average performance for DDPG, robust DDPG and soft-robust DDPG

agents when evaluating them on different masses. Since the performance among different episodes is highly variable, we considered the best 200-episodes average reward as a performance measure. As seen in the figure, the robust strategy solves the task in a sub-optimal fashion, but is less affected by model misspecification due to its conservative strategy. The aggressive non-robust agent is more sensitive to model misspecification compared to the other methods as can be seen by its sudden dip in performance, below even that of the robust agent. The soft-robust solution strikes a nice balance between being less sensitive to model misspecification than the aggressive agent, and producing better performance compared to the robust solution.

## 7 RELATED WORK

This paper is related to several domains in RL such as robust and distributionally robust MDPs, actor-critic methods and online learning via stochastic approximation algorithms. Our work solves the problem of conservativeness encountered in robust MDPs by incorporating a variational form of distributional robustness. The SR-AC algorithm combines scalability to large scale state-spaces and online estimation of the optimal policy in an actor-critic algorithm. Table 1 compares our proposed algorithm with previous approaches.

Many solutions have been addressed to mitigate conservativeness of robust MDP. Mannor et al. [2012, 2016] relax the state-wise independence property of the uncertainty set and assume it to be coupled in a way such that the planning problem stays tractable. Another approach tends to assume *a priori* information on the parameter set. These

Table 1: Comparison of previous approaches with SR-AC

Reference	Scalable	Actor-Critic	Softly-Robust
SR-AC (this paper)	✓	✓	✓
Mankowitz et al. [2018]	✓	✗	✗
Lim et al. [2016]	✗	✗	✗
Yu and Xu [2016]	✗	✗	✓
Mannor et al. [2012, 2016]	✗	✗	✗
Tamar et al. [2015]	✓	✗	✗
Xu and Mannor [2012]	✗	✗	✓
Bhatnagar et al. [2009]	✓	✓	✗

methods include distributionally robust MDPs [Xu and Mannor, 2012; Yu and Xu, 2016] in which the optimal strategy maximizes the expected reward under the most adversarial distribution over the uncertainty set. For finite and known MDPs, under some structural assumptions on the considered set of distributions, this max-min problem reduces to classical robust MDPs and can be solved efficiently by dynamic programming [Puterman, 2009].

However, besides becoming untractable under large-sized MDPs, these methods use an offline learning approach which cannot adapt its level of protection against model uncertainty and may lead to overly conservative results. The work of Lim et al. [2016] solutions this issue and addresses an online algorithm that learns the transitions that are purely stochastic and those that are adversarial. Although it ensures less conservative results as well as low regret, this method sticks to the robust objective while strongly relying on the finite structure of the state-space. To alleviate the curse of dimensionality, we incorporate function approximation of the objective value and define it as a linear functional of features.

First introduced in Barto et al. [1983] and later addressed by Bhatnagar et al. [2009], actor-critic algorithms are online learning methods that aim at finding an optimal policy. We used the formulation of Bhatnagar et al. [2009] as a baseline for the algorithm we proposed. The key difference between their work and ours is that we incorporate soft-robustness. This relates in a sense to the Bayesian Actor-Critic setup in which the critic returns a complete posterior distribution of value functions using Bayes’ rule [Ghavamzadeh and Engel, 2007; Ghavamzadeh et al., 2015, 2016]. Our study keeps a frequentist approach, meaning that our algorithm updates return point estimates of the average value-function which prevents from tractability issues besides enabling the distribution to be more

flexible. Another major distinction is that the Bayesian approach incorporates a prior distribution on one model parameters whereas our method considers a prior on different transition models over an uncertainty set.

In Mankowitz et al. [2018]; Tamar et al. [2015], the authors incorporate robustness into policy-gradient methods. A sampling procedure is required for each critic estimate in Tamar et al. [2015], which differs from the strictly-speaking actor-critic. A robust version of actor-critic policy-gradient is introduced in Mankowitz et al. [2018] but its convergence guarantees are only shown for robust policy-gradient ascent. Both of these methods target the robust strategy whereas we seek a soft-robust policy that is less conservative while protecting itself against model uncertainty.

## 8 DISCUSSION

We have presented the SR-AC framework that is able to learn policies which keep a balance between aggressive and robust behaviors. SR-AC requires a stationary distribution under the average transition model and compatibility conditions for deriving a soft-robust policy-gradient. We have shown that this ensures convergence of SR-AC. This is the first work that has attempted to incorporate a soft form of robustness into an online actor-critic method. Our approach has been shown to be computationally scalable to large domains because of its low computational price. In our experiments, we have also shown that the soft-robust agent interpolates between aggressive and robust strategies without being overly conservative which leads it to outperform robust policies under model uncertainty even when the action space is continuous. Subsequent experiments should test the efficiency of soft-robustness on more complex domains.

The chosen weighting over the uncertainty set can be thought as the way the adversary distributes over different transition laws. In our current setting, this adversarial distribution stays constant without accounting for the rewards obtained by the agent. Future work should address the problem of learning the sequential game induced by an evolving adversarial distribution to derive an optimal soft-robust policy. Other extensions of our work may also consider non-linear objective functions such as higher order moments with respect to the adversarial distribution.

## Acknowledgements

This work was partially funded by the Israel Science Foundation under contract 1380/16 and by the European Community’s Seventh Framework Programme (FP7/2007-2013) under grant agreement 306638 (SUPREL).

## References

- Andrew G. Barto, Richard S. Sutton, and Charles W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(5):834–846, 1983.
- Shalabh Bhatnagar, Richard Sutton, Mohammad Ghavamzadeh, and Mark Lee. *Natural Actor-Critic Algorithms*. Automatica, elsevier edition, 2009.
- Vivek S. Borkar. Stochastic Approximation with Two Timescales. *Systems and Control Letters*, 29:291–294, 1997.
- Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym. arXiv:1606.01540v1, 2016.
- Shirli Di-Castro Shashua and Shie Mannor. Deep Robust Kalman Filter. *arXiv preprint arXiv:1703.02310v1*, 2017.
- Mohammad Ghavamzadeh and Yaakov Engel. Bayesian Actor-Critic Algorithms. *Proceedings of the 24th international conference on Machine learning*, pages 297–304, 2007.
- Mohammad Ghavamzadeh, Shie Mannor, Joelle Pineau, and Aviv Tamar. Bayesian Reinforcement Learning: A Survey. *Foundations and Trends in Machine Learning*, 8(5-6):359–492, 2015.
- Mohammad Ghavamzadeh, Yaakov Engel, and Michal Valko. Bayesian Policy Gradient and Actor-Critic Algorithms. *Journal of Machine Learning Research*, 17(66):1–53, 2016.
- Ivo Grondman, Lucian Busoniu, Gabriel A.D. Lopes, and Robert Babuska. A Survey of Actor-Critic Reinforcement Learning: Standard and Natural Policy Gradients. *IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews*, 42(1291-1307), 2012.
- Garud N. Iyengar. Robust Dynamic Programming. *Mathematics of Operations Research*, 30(2):257–280, 2005.
- Vijay R. Konda and John N. Tsitsiklis. Actor-Critic Algorithms. In *Advances in Neural Information Processing Systems*, volume 12, 2000.
- Harold J. Kushner and G. George Yin. *Stochastic Approximation Algorithms and Applications*. Springer Verlag, New York, 1997.
- Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous Control with Deep Reinforcement Learning. *arXiv:1509.02971*, US Patent App. 15/217,758, 2016.
- Shiau Hong Lim, Huan Xu, and Shie Mannor. Reinforcement Learning in Robust Markov Decision Processes. *Mathematics of Operations Research*, 41(4):1325–1353, 2016.
- Daniel J Mankowitz, Timothy A Mann, Shie Mannor, Doina Precup, and Pierre-Luc Bacon. Learning Robust Options. In *AAAI*, 2018.
- Shie Mannor, Duncan Simester, Peng Sun, and John N. Tsitsiklis. Bias and Variance Approximation in Value Function Estimates. *Management Science*, 53(2):308–322, 2007.
- Shie Mannor, Ofir Mebel, and Huan Xu. Lightning Does Not Strike Twice: Robust MDPs with Coupled Uncertainty. In *ICML*, 2012.
- Shie Mannor, Ofir Mebel, and Huan Xu. Robust MDPs with k-Rectangular Uncertainty. *Mathematics of Operations Research*, 41(4):1484–1509, 2016.
- Danny Miller and Peter H. Friesen. Innovation in conservative and entrepreneurial firms: Two models of strategic momentum. *Strategic Management Journal*, 1982.
- Olivia S. Mitchell and Kent Smetters. *The Market for Retirement Financial Advice*. Oxford University Press, first edition edition, 2013.
- Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing Atari with Deep Reinforcement Learning: Technical Report. *DeepMind Technologies*, arXiv:1312.5602, 2013.
- Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.
- Arnab Nilim and Laurent El Ghaoui. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):783–798, 2005.
- Martin L. Puterman. *Markov decision processes: Discrete stochastic dynamic programming*, volume 414. Wiley Series in Probability and Statistics, 2009.
- Aurko Roy, Huan Xu, and Sebastian Pokutta. Reinforcement learning under Model Mismatch. *31st Conference on Neural Information Processing Systems*, 2017.
- David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic Policy Gradient Algorithms. *ICML*, 2014.

- Richard S. Sutton, David McAllester, Satinger Singh, and Yishay Mansour. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems*, volume 12, pages 1057–1063, 2000.
- Aviv Tamar, Shie Mannor, and Huan Xu. Scaling up robust mdps using function approximation. *ICML*, 32: 1401–1415, 2014.
- Aviv Tamar, Yinlam Chow, Mohammad Ghavamzadeh, and Shie Mannor. Policy gradient for coherent risk measures. In *Advances in Neural Information Processing Systems*, pages 1468–1476, 2015.
- Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.
- Marco Wiering and Martijn van Otterlo. *Reinforcement Learning: State-of-the-Art*. 12. Springer-Verlag Berlin Heidelberg, 2012.
- Huan Xu and Shie Mannor. Distributionally Robust Markov Decision Processes. *Mathematics of Operations Research*, 37(2):288–300, 2012.
- Pengqian Yu and Huan Xu. Distributionally Robust Counterpart in Markov Decision Processes. *IEEE Transactions on Automatic Control*, 61(9):2538 – 2543, 2016.

---

# Constant Step Size Stochastic Gradient Descent for Probabilistic Modeling

---

**Dmitry Babichev**  
INRIA - ENS  
PSL Research University  
Paris, France  
dmitry.babichev@inria.fr

**Francis Bach**  
INRIA - ENS  
PSL Research University  
Paris, France  
francis.bach@inria.fr

## Abstract

Stochastic gradient methods enable learning probabilistic models from large amounts of data. While large step-sizes (learning rates) have shown to be best for least-squares (e.g., Gaussian noise) once combined with parameter averaging, these are not leading to convergent algorithms in general. In this paper, we consider generalized linear models, that is, conditional models based on exponential families. We propose averaging moment parameters instead of natural parameters for constant-step-size stochastic gradient descent. For finite-dimensional models, we show that this can sometimes (and surprisingly) lead to better predictions than the best linear model. For infinite-dimensional models, we show that it always converges to optimal predictions, while averaging natural parameters never does. We illustrate our findings with simulations on synthetic data and classical benchmarks with many observations.

## 1 INTRODUCTION

Faced with large amounts of data, efficient parameter estimation remains one of the key bottlenecks in the application of probabilistic models. Once cast as an optimization problem, for example through the maximum likelihood principle, difficulties may arise from the size of the model, the number of observations, or the potential non-convexity of the objective functions, and often all three (Koller and Friedman, 2009; Murphy, 2012).

In this paper we focus primarily on situations where the number of observations is large; in this context, stochastic gradient descent (SGD) methods which look at one sample at a time are usually favored for their cheap iter-

ation cost. However, finding the correct step-size (sometimes referred to as the learning rate) remains a practical and theoretical challenge, for probabilistic modeling but also in most other situations beyond maximum likelihood (Bottou et al., 2016).

In order to preserve convergence, the step size  $\gamma_n$  at the  $n$ -th iteration typically has to decay with the number of gradient steps (here equal to the number of data points which are processed), typically as  $C/n^\alpha$  for  $\alpha \in [1/2, 1]$  (see, e.g., Bach and Moulines, 2011; Bottou et al., 2016). However, these often leads to slow convergence and the choice of  $\alpha$  and  $C$  is difficult in practice. More recently, constant step-sizes have been advocated for their fast convergence towards a neighborhood of the optimal solution (Bach and Moulines, 2013), while it is a standard practice in many areas (Goodfellow et al., 2016). However, it is not convergent in general and thus small step-sizes are still needed to converge to a decent estimator.

Constant step-sizes can however be made to converge in one situation. When the functions to optimize are quadratic, like for least-squares regression, using a constant step-size combined with an averaging of all estimators along the algorithm can be shown to converge to the global solution with the optimal convergence rates (Bach and Moulines, 2013; Dieuleveut and Bach, 2016).

The goal of this paper is to explore the possibility of such global convergence with a constant step-size in the context of probabilistic modeling with exponential families, e.g., for logistic regression or Poisson regression (McCullagh, 1984). This would lead to the possibility of using probabilistic models (thus with a principled quantification of uncertainty) with rapidly converging algorithms. Our main novel idea is to replace the averaging of the *natural* parameters of the exponential family by the averaging of the *moment* parameters, which can also be formulated as averaging *predictions* instead of *estimators*. For example, in the context of predicting bi-

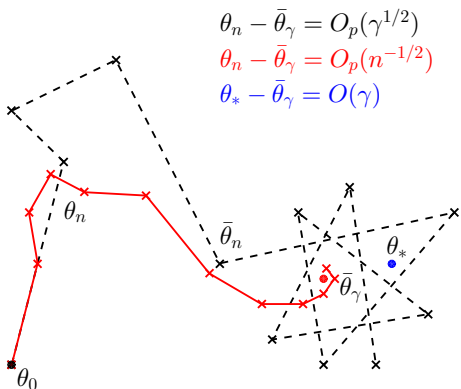


Figure 1: Convergence of iterates  $\theta_n$  and averaged iterates  $\bar{\theta}_n$  to the mean  $\bar{\theta}_\gamma$  under the stationary distribution  $\pi_\gamma$ .

nary outcomes in  $\{0, 1\}$  through a Bernoulli distribution, the moment parameter is the probability  $p \in [0, 1]$  that the variable is equal to one, while the natural parameter is the “log odds ratio”  $\log \frac{p}{1-p}$ , which is unconstrained. This lack of constraint is often seen as a benefit for optimization; it turns out that for stochastic gradient methods, the moment parameter is better suited to averaging. Note that for least-squares, which corresponds to modeling with the Gaussian distribution with fixed variance, moment and natural parameters are equal, so it does not make a difference.

More precisely, our main contributions are:

- For generalized linear models, we propose in Section 4 averaging moment parameters instead of natural parameters for constant-step-size stochastic gradient descent.
- For finite-dimensional models, we show in Section 5 that this can sometimes (and surprisingly) lead to better predictions than the best linear model.
- For infinite-dimensional models, we show in Section 6 that it always converges to optimal predictions, while averaging natural parameters never does.
- We illustrate our findings in Section 7 with simulations on synthetic data and classical benchmarks with many observations.

## 2 CONSTANT STEP SIZE STOCHASTIC GRADIENT DESCENT

In this section, we present the main intuitions behind stochastic gradient descent (SGD) with constant step-size. For more details, see Dieuleveut et al. (2017). We

consider a real-valued function  $F$  defined on the Euclidean space  $\mathbb{R}^d$  (this can be generalized to any Hilbert space, as done in Section 6 when considering Gaussian processes and positive-definite kernels), and a sequence of random functions  $(f_n)_{n \geq 1}$  which are independent and identically distributed and such that  $\mathbb{E}f_n(\theta) = F(\theta)$  for all  $\theta \in \mathbb{R}^d$ . Typically,  $F$  will be the expected negative log-likelihood on unseen data, while  $f_n$  will be the negative log-likelihood for a single observation. Since we require *independent* random functions, we assume that we make single pass over the data, and thus the number of iterations is equal to the number of observations.

Starting from an initial  $\theta_0 \in \mathbb{R}^d$ , then SGD will perform the following recursion, from  $n = 1$  to the total number of observations:

$$\theta_n = \theta_{n-1} - \gamma_n \nabla f_n(\theta_{n-1}). \quad (1)$$

Since the functions  $f_n$  are independent, the iterates  $(\theta_n)_n$  form a Markov chain. When the step-size  $\gamma_n$  is constant (equal to  $\gamma$ ) and the functions  $f_n$  are identically distributed, the Markov chain is *homogeneous*. Thus, under additional assumptions (see, e.g., Dieuleveut et al., 2017; Meyn and Tweedie, 1993), it converges in distribution to a stationary distribution, which we refer to as  $\pi_\gamma$ . These additional assumptions include that  $\gamma$  is not too large (otherwise the algorithm diverges) and in the traditional analysis of step-sizes for gradient descent techniques, we analyze the situation of small  $\gamma$ 's (and thus perform asymptotic expansions around  $\gamma = 0$ ).

The distribution  $\pi_\gamma$  is in general not equal to a Dirac mass, and thus, constant-step-size SGD is *not* convergent. However, averaging along the path of the Markov chain has some interesting properties. Indeed, several versions of the “ergodic theorem” (see, e.g., Meyn and Tweedie, 1993) show that for functions  $g$  from  $\mathbb{R}^d$  to any vector space, then the empirical average  $\frac{1}{n} \sum_{i=1}^n g(\theta_i)$  converges in probability to the expectation  $\int g(\theta) d\pi_\gamma(\theta)$  of  $g$  under the stationary distribution  $\pi_\gamma$ . This convergence can also be quantified by a central limit theorem with an error which tends to a normal distribution with variance proportional equal to a constant times  $1/n$ .

Thus, if denote  $\bar{\theta}_n = \frac{1}{n+1} \sum_{i=0}^n \theta_i$ , applying the previous result to the identity function  $g$ , we immediately obtain that  $\bar{\theta}_n$  converges to  $\bar{\theta}_\gamma = \int \theta d\pi_\gamma(\theta)$ , with a squared error converging in  $O(1/n)$ . The key question is the relationship between  $\bar{\theta}_\gamma$  and the global optimizer  $\theta_*$  of  $F$ , as this characterizes the performance of the algorithm with an infinite number of observations.

By taking expectations in Eq. (1), and taking a limit with  $n$  tending to infinity we obtain that

$$\int \nabla F(\theta) d\pi_\gamma(\theta) = 0, \quad (2)$$



that is, under the stationary distribution  $\pi_\gamma$ , the average gradient is zero. When the gradient is a linear function (like for a quadratic objective  $F$ ), this leads to  $\nabla F(\int \theta d\pi_\gamma(\theta)) = \nabla F(\bar{\theta}_\gamma) = 0$ , and thus  $\bar{\theta}_\gamma$  is a stationary point of  $F$  (and hence the global minimizer if  $F$  is convex). However this is not true in general.

As shown by Dieuleveut et al. (2017), the deviation  $\bar{\theta}_\gamma - \theta_*$  is of order  $\gamma$ , which is an improvement on the non-averaged recursion, which is at average distance  $O(\gamma^{1/2})$  (see an illustration in Figure 1); thus, small or decaying step-sizes are needed. In this paper, we explore alternatives which are not averaging the estimators  $\theta_1, \dots, \theta_n$ , and rely instead on the specific structure of our cost functions, namely negative log-likelihoods.

### 3 WARM-UP: EXPONENTIAL FAMILIES

In order to highlight the benefits of averaging moment parameters, we first consider unconditional exponential families. We thus consider the standard exponential family  $q(x|\theta) = h(x) \exp(\theta^\top T(x) - A(\theta))$ , where  $h(x)$  is the base measure,  $T(x) \in \mathbb{R}^d$  is the sufficient statistics and  $A$  the log-partition function. The function  $A$  is always convex (see, e.g., Koller and Friedman, 2009; Murphy, 2012). Note that we do not assume that the data distribution  $p(x)$  comes from this exponential family. The expected (with respect to the input distribution  $p(x)$ ) negative log-likelihood is equal to

$$\begin{aligned} F(\theta) &= -\mathbb{E}_{p(x)} \log q(x|\theta) \\ &= A(\theta) - \theta^\top \mathbb{E}_{p(x)} T(x) - \mathbb{E}_{p(x)} \log h(x). \end{aligned}$$

It is known to be minimized by  $\theta_*$  such that  $\nabla A(\theta_*) = \mathbb{E}_{p(x)} T(x)$ . Given i.i.d. data  $(x_n)_{n \geq 1}$  sampled from  $p(x)$ , then the SGD recursion from Eq. (1) becomes:

$$\theta_n = \theta_{n-1} - \gamma [\nabla A(\theta_{n-1}) - T(x_n)],$$

while the stationarity equation in Eq. (2) becomes

$$\int [\nabla A(\theta) - \mathbb{E}_{p(x)} T(x)] d\pi_\gamma(\theta) = 0,$$

which leads to

$$\int \nabla A(\theta) d\pi_\gamma(\theta) = \mathbb{E}_{p(x)} T(x) = \nabla A(\theta_*).$$

Thus, averaging  $\nabla A(\theta_n)$  will converge to  $\nabla A(\theta_*)$ , while averaging  $\theta_n$  will *not* converge to  $\theta_*$ . This simple observation is the basis of our work.

Note that in this context of unconditional models, a simpler estimator exists, that is, we can simply compute the

empirical average  $\frac{1}{n} \sum_{i=1}^n T(x_i)$  that will converge to  $\nabla A(\theta_*)$ . Nevertheless, this shows that averaging moment parameters  $\nabla A(\theta)$  rather than natural parameters  $\theta$  can bring convergence benefits. We now turn to conditional models, for which no closed-form solutions exist.

## 4 CONDITIONAL EXPONENTIAL FAMILIES

Now we consider the conditional exponential family  $q(y|x, \theta) = h(y) \exp(y \cdot \eta_\theta(x) - a(\eta_\theta(x)))$ . For simplicity we consider only one-dimensional families where  $y \in \mathbb{R}$ —but our framework would also extend to more complex models such as conditional random fields (Lafferty et al., 2001). We will also assume that  $h(y) = 1$  for all  $y$  to avoid carrying constant terms in log-likelihoods. We consider functions of the form  $\eta_\theta(x) = \theta^\top \Phi(x)$ , which are linear in a feature vector  $\Phi(x)$ , where  $\Phi : \mathcal{X} \rightarrow \mathbb{R}^d$  can be defined on an arbitrary input set  $\mathcal{X}$ . Calculating the negative log-likelihood, one obtains:

$$f_n(\theta) = -\log q(y_n|x_n, \theta) = -y_n \Phi(x_n)^\top \theta + a(\Phi(x_n) \theta),$$

and, for any distribution  $p(x, y)$ , for which  $p(y|x)$  may not be a member of the conditional exponential family,

$$\begin{aligned} F(\theta) &= \mathbb{E}_{p(x_n, y_n)} f_n(\theta) \\ &= \mathbb{E}_{p(x_n, y_n)} \left[ -y_n \Phi(x_n)^\top \theta + a(\Phi(x_n) \theta) \right]. \end{aligned}$$

The goal of estimation in such generalized linear models is to find an unknown parameter  $\theta$  given  $n$  observations  $(x_i, y_i)_{i=1, \dots, n}$ :

$$\theta_* = \arg \min_{\theta \in \mathbb{R}^d} F(\theta). \quad (3)$$

### 4.1 FROM ESTIMATORS TO PREDICTION FUNCTIONS

Another point of view is to consider that an estimator  $\theta \in \mathbb{R}^d$  in fact defines a function  $\eta : \mathcal{X} \rightarrow \mathbb{R}$ , with value a natural parameter for the exponential family  $q(y) = \exp(\eta y - a(\eta))$ . This particular choice of function  $\eta_\theta$  is linear in  $\Phi(x)$ , and we have, by decomposing the joint probability  $p(x_n, y_n)$  in two (and dropping the dependence on  $n$  since we have assumed i.i.d. data):

$$\begin{aligned} F(\theta) &= \mathbb{E}_{p(x)} \left( \mathbb{E}_{p(y|x)} \left[ -y \Phi(x)^\top \theta + a(\Phi(x)^\top \theta) \right] \right) \\ &= \mathbb{E}_{p(x)} \left( -\mathbb{E}_{p(y|x)} y \Phi(x)^\top \theta + a(\Phi(x)^\top \theta) \right) \\ &= \mathcal{F}(\eta_\theta), \end{aligned}$$

with  $\mathcal{F}(\eta) = \mathbb{E}_{p(x)} (-\mathbb{E}_{p(y|x)} y \cdot \eta(x) + a(\eta(x)))$  is the performance measure defined for a function  $\eta : \mathcal{X} \rightarrow \mathbb{R}$ . By definition  $F(\theta) = \mathcal{F}(\eta_\theta) = \mathcal{F}(\theta^\top \Phi(\cdot))$ .

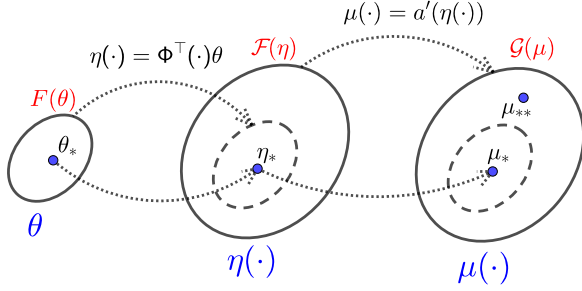


Figure 2: Graphical representation of reparametrization: firstly we expand the class of functions, replacing parameter  $\theta$  with function  $\eta(\cdot) = \Phi^\top(\cdot)\theta$  and then we do one more reparametrization:  $\mu(\cdot) = a'(\eta(\cdot))$ . Best linear prediction  $\mu_*$  is constructed using  $\theta_*$  and the global minimizer of  $\mathcal{G}$  is  $\mu_{**}$ . Model is well-specified if and only if  $\mu_* = \mu_{**}$ .

However, the global minimizer of  $\mathcal{F}(\eta)$  over all functions  $\eta : \mathcal{X} \rightarrow \mathbb{R}$  may not be attained at a linear function in  $\Phi(x)$  (this can only be the case if the linear model is well-specified or if the feature vector  $\Phi(x)$  is flexible enough). Indeed, the global minimizer of  $\mathcal{F}$  is the function  $\eta_{**} : x \mapsto (a')^{-1}(\mathbb{E}_{p(y|x)}y)$  (starting from  $\mathcal{F}(\eta) = \int [a(\eta(x)) - \mathbb{E}_{p(x|y)}y \cdot \eta(x)]p(x)dx$  and writing down the Euler - Lagrange equation:  $\frac{\partial \mathcal{F}}{\partial \eta} - \frac{d}{dx} \frac{\partial \mathcal{F}}{\partial \eta'} = 0 \Leftrightarrow [a'(\eta) - \mathbb{E}_{p(x|y)}y]p(x) = 0$  and finally  $\eta \mapsto (a')^{-1}(\mathbb{E}_{p(x|y)}y)$ ) and is typically not a linear function in  $\Phi(x)$  (note here that  $p(y|x)$  is the conditional data-generating distribution).

The function  $\eta$  corresponds to the *natural* parameter of the exponential family, and it is often more intuitive to consider the *moment* parameter, that is defining functions  $\mu : \mathcal{X} \rightarrow \mathbb{R}$  that now correspond to moments of outputs  $y$ ; we will refer to them as *prediction functions*. Going from natural to moment parameter is known to be done through the gradient of the log-partition function, and we thus consider for  $\eta$  a function from  $\mathcal{X}$  to  $\mathbb{R}$ ,  $\mu(\cdot) = a'(\eta(\cdot))$ , and this leads to the performance measure

$$\mathcal{G}(\mu) = \mathcal{F}((a')^{-1}(\mu(\cdot))).$$

Note now, that the global minimum of  $\mathcal{G}$  is reached at

$$\mu_{**}(x) = \mathbb{E}_{p(y|x)}y.$$

We introduce also the prediction function  $\mu_*(x)$  corresponding to the best  $\eta$  which is linear in  $\Phi(x)$ , that is:

$$\mu_*(x) = a'(\theta_*^\top \Phi(x)).$$

We say that the model is well-specified when  $\mu_* = \mu_{**}$ , and for these models,  $\inf_\theta F(\theta) = \inf_\mu \mathcal{G}(\mu)$ . However, in general, we only have  $\inf_\theta F(\theta) > \inf_\mu \mathcal{G}(\mu)$

and (very often) the inequality is strict (see examples in our simulations).

To make the further developments more concrete, we now present two classical examples: logistic regression and Poisson regression.

**Logistic regression.** The special case of conditional family is logistic regression, where  $y \in \{0, 1\}$ ,  $a(t) = \log(1 + e^{-t})$  and  $a'(t) = \sigma(t) = \frac{1}{1+e^{-t}}$  is the sigmoid function and the probability mass function is given by  $p(y|\eta) = \exp(\eta y - \log(1 + e^\eta))$ .

**Poisson regression.** One more special case is Poisson regression with  $y \in \mathbb{N}$ ,  $a(t) = \exp(t)$  and the response variable  $y$  has a Poisson distribution. The probability mass function is given by  $p(y|\eta) = \exp(\eta y - e^\eta - \log(y!))$ . Poisson regression may be appropriate when the dependent variable is a count, for example in genomics, network packet analysis, crime rate analysis, fluorescence microscopy, etc. (Hilbe, 2011).

## 4.2 AVERAGING PREDICTIONS

Recall from Section 2 that  $\pi_\gamma$  is the stationary distribution of  $\theta$ . Taking expectation of both parts of Eq. (1), we get, by using the fact that  $\pi_\gamma$  is the limiting distribution of  $\theta_n$  and  $\theta_{n-1}$ :

$$\begin{aligned} & \mathbb{E}_{\pi_\gamma(\theta_n)}\theta_n \\ &= \mathbb{E}_{\pi_\gamma(\theta_{n-1})}\theta_{n-1} - \gamma \mathbb{E}_{\pi_\gamma(\theta_{n-1})}\mathbb{E}_{p(x_n, y_n)}f'_n(\theta_{n-1}), \end{aligned}$$

which leads to  $\mathbb{E}_{\pi_\gamma(\theta)}\mathbb{E}_{p(x_n, y_n)}\nabla f_n(\theta) = 0$ , that is, now removing the dependence on  $n$  (data  $(x, y)$  are i.i.d.):

$$\mathbb{E}_{\pi_\gamma(\theta)}\mathbb{E}_{p(x, y)}\left[-y\Phi(x) + a'(\Phi(x)^\top \theta)\Phi(x)\right] = 0,$$

which finally leads to

$$\mathbb{E}_{p(x)}\left[\mathbb{E}_{\pi_\gamma(\theta)}a'(\Phi(x)^\top \theta) - \mu_{**}(x)\right]\Phi(x) = 0. \quad (4)$$

This is the core equation our method relies on. It does not imply that  $b(x) = \mathbb{E}_{\pi_\gamma(\theta)}a'(\Phi(x)^\top \theta) - \mu_{**}(x)$  is uniformly equal to zero (which we want), but only that  $\mathbb{E}_{p(x)}\Phi(x)b(x) = 0$ , i.e.,  $b(x)$  is uncorrelated with  $\Phi(x)$ .

If the feature vector  $\Phi(x)$  is “large enough” then this is equivalent to  $b = 0$ .<sup>1</sup> For example, when  $\Phi(x)$  is composed of an orthonormal basis of the space of integrable

<sup>1</sup>Let  $\Phi(x) = (\phi_1(x), \dots, \phi_n(x))^\top$  be an orthogonal basis and  $b(x) = \sum_{i=1}^n c_i \phi_i(x) + \varepsilon(x)$ , where  $\varepsilon(x)$  is small if the basis is big enough. Then  $\mathbb{E}_{p(x)}\Phi(x)b(x) = 0 \Leftrightarrow \mathbb{E}\phi_i(x)[\sum_{i=1}^n c_i \phi_i(x) + \varepsilon(x)] = 0$  for every  $i$ , and due to the orthogonality of the basis and the smallness of  $\varepsilon(x)$ :  $c_i \cdot \mathbb{E}_{p(x)}\phi_i^2(x) \approx 0$  and hence  $c_i \approx 0$  and thus  $b(x) \approx 0$ .

functions (like for kernels in Section 6), then this is exactly true. Thus, in this situation,

$$\mu_{**}(x) = \mathbb{E}_{\pi_\gamma(\theta)} a'(\Phi(x)^\top \theta), \quad (5)$$

and averaging predictions  $a'(\Phi(x)^\top \theta_n)$ , along the path  $(\theta_n)$  of the Markov chain should exactly converge to the optimal prediction.

This exact convergence is weaker (requires high-dimensional features) than for the unconditional family in Section 3 but it can still bring surprising benefits even when  $\Phi$  is not large enough, as we present in Section 5 and Section 6.

### 4.3 TWO TYPES OF AVERAGING

Now we can introduce two possible ways to estimate the prediction function  $\mu(x)$ .

**Averaging estimators.** The first one is the usual way: we first estimate parameter  $\theta$ , using Ruppert-Polyak averaging (Polyak and Juditsky, 1992):  $\bar{\theta}_n = \frac{1}{n+1} \sum_{i=0}^n \theta_i$  and then we denote

$$\bar{\mu}_n(x) = a'(\Phi(x)^\top \bar{\theta}_n) = a'(\Phi(x)^\top \frac{1}{n+1} \sum_{i=0}^n \theta_i)$$

the corresponding prediction. As discussed in Section 2 it converges to  $\bar{\mu}_\gamma : x \mapsto a'(\Phi(x)^\top \bar{\theta}_\gamma)$ , which is *not* equal to in general to  $a'(\Phi(x)^\top \theta_*)$ , where  $\theta_*$  is the optimal parameter in  $\mathbb{R}^d$ . Since, as presented at the end of Section 2,  $\bar{\theta}_\gamma - \theta_*$  is of order  $O(\gamma)$ ,  $F(\bar{\theta}_\gamma) - F(\theta_*)$  is of order  $O(\gamma^2)$  (because  $\nabla F(\theta_*) = 0$ ), and thus an error of  $O(\gamma^2)$  is added to the usual convergence rates in  $O(1/n)$ .

Note that we are limited here to prediction functions which corresponds to *linear functions* in  $\Phi(x)$  in the natural parameterization, and thus  $F(\theta_*) \geq \mathcal{G}(\mu_{**})$ , and the inequality is often strict.

**Averaging predictions.** We propose a new estimator

$$\bar{\bar{\mu}}_n(x) = \frac{1}{n+1} \sum_{i=0}^n a'(\theta_i^\top \Phi(x)).$$

In general  $\mathcal{G}(\bar{\bar{\mu}}_n) - \mathcal{G}(\mu_{**})$  does not converge to zero either (unless the feature vector  $\Phi$  is large enough and Eq. (5) is satisfied). Thus, on top of the usual convergence in  $O(1/n)$  with respect to the number of iterations, we have an extra term that depends only on  $\gamma$ , which we will study in Section 5 and Section 6.

We denote by  $\bar{\bar{\mu}}_\gamma(x)$  the limit when  $n \rightarrow \infty$ , that is, using properties of converging Markov chains,  $\bar{\bar{\mu}}_\gamma(x) = \mathbb{E}_{\pi_\gamma(\theta)} a'(\Phi(x)^\top \theta)$ .

Rewriting Eq. (4) using our new notations, we get:

$$\mathbb{E}[(\mu_{**}(x) - \bar{\bar{\mu}}_\gamma(x))\Phi(x_n)] = 0.$$

When  $\Phi : \mathbb{R} \rightarrow \mathbb{R}^d$  is high-dimensional, this leads to  $\mu_{**} = \bar{\bar{\mu}}_\gamma$  and in contrast to  $\bar{\mu}_\gamma$ , averaging predictions potentially converge to the optimal prediction.

**Computational complexity.** Usual averaging of estimators (Polyak and Juditsky, 1992) to compute  $\bar{\mu}_n(x) = a'(\Phi(x)^\top \bar{\theta}_n)$  is simple to implement as we can simply update the average  $\bar{\theta}_n$  with essentially no extra cost on top the complexity  $O(nd)$  of the SGD recursion. Given the number  $n$  of training data points and the number  $m$  of testing data points, the overall complexity is  $O(d(n+m))$ .

Averaging prediction functions is more challenging as we have to store all iterates  $\theta_i$ ,  $i = 1, \dots, n$ , and for each testing point  $x$ , compute  $\bar{\bar{\mu}}_n(x) = \frac{1}{n+1} \sum_{i=0}^n a'(\theta_i^\top \Phi(x))$ . Thus the overall complexity is  $O(dn + mnd)$ , which could be too costly with many test points (i.e.,  $m$  large).

There are several ways to alleviate this extra cost: (a) using sketching techniques (Woodruff et al., 2014), (b) using summary statistics like done in applications of MCMC (Gilks et al., 1995), or (c) leveraging the fact that all iterates  $\theta_i$  will end up being close to  $\bar{\theta}_\gamma$  and use a Taylor expansion of  $a'(\theta^\top \Phi(x))$  around  $\bar{\theta}_\gamma$ . This expansion is equal to:

$$\begin{aligned} & a'(\Phi(x)^\top \bar{\theta}_\gamma) + (\theta - \bar{\theta}_\gamma)^\top \Phi(x) \cdot a''(\Phi(x)^\top \bar{\theta}_\gamma) + \\ & + \frac{1}{2} ((\theta - \bar{\theta}_\gamma)^\top \Phi(x))^2 \cdot a'''(\Phi(x)^\top \bar{\theta}_\gamma) + O(\|\theta - \bar{\theta}_\gamma\|^3). \end{aligned}$$

Taking expectation in both sides above leads to:

$$\bar{\bar{\mu}}_\gamma(x) \approx \bar{\mu}_\gamma(x) + \frac{1}{2} \Phi(x)^\top \text{cov}(\theta) \cdot \Phi(x) \cdot a'''(\bar{\theta}_\gamma^\top \Phi(x)),$$

where  $\text{cov}(\theta)$  is the covariance matrix of  $\theta$  under  $\pi_\gamma$ . This provides a simple correction to  $\bar{\mu}_\gamma$ , and leads to an approximation of  $\bar{\bar{\mu}}_n(x)$  as

$$\bar{\bar{\mu}}_n(x) + \frac{1}{2} \Phi(x)^\top \text{cov}_n(\theta) \cdot \Phi(x) \cdot a'''(\bar{\theta}_n^\top \Phi(x)),$$

where  $\text{cov}_n(\theta)$  is the empirical covariance matrix of the iterates  $(\theta_i)$ .

The computational complexity now becomes  $O(nd^2 + md^2)$ , which is an improvement when the number of testing points  $m$  is large. In all of our experiments, we used this approximation.

## 5 FINITE-DIMENSIONAL MODELS

In this section we study the behavior of  $\bar{A}(\gamma) = \mathcal{G}(\bar{\mu}_\gamma) - \mathcal{G}(\mu_*)$  for finite-dimensional models, for which it is usually not equal to zero. We know that our estimators  $\bar{\mu}_n$  will converge to  $\bar{\mu}_\gamma$ , and our goal is to compare it to  $\bar{A}(\gamma) = \mathcal{G}(\bar{\mu}_\gamma) - \mathcal{G}(\mu_*) = F(\bar{\theta}_\gamma) - F(\theta_*)$  which is what averaging estimators tends to. We also consider for completeness the non-averaged performance  $A(\gamma) = \mathbb{E}_{\pi_\gamma(\theta)} [F(\theta) - F(\theta_*)]$ .

Note that we must have  $A(\gamma)$  and  $\bar{A}(\gamma)$  non-negative, because we compare the negative log-likelihood performances to the one of the best linear prediction (in the natural parameter), while  $\bar{A}(\gamma)$  could potentially be negative (it will in certain situations), because the corresponding natural parameters may not be linear in  $\Phi(x)$ .

We consider the same standard assumptions as Dieuleveut et al. (2017), namely smoothness of the negative log-likelihoods  $f_n(\theta)$  and strong convexity of the expected negative log-likelihood  $F(\theta)$ . We first recall the results from Dieuleveut et al. (2017). See detailed explicit formulas in the supplementary material.

### 5.1 EARLIER WORK

**Without averaging.** We have that  $A(\gamma) = \gamma B + O(\gamma^{3/2})$ , that is  $\gamma$  is *linear* in  $\gamma$ , with  $B$  non-negative.

**Averaging estimators.** We have that  $\bar{A}(\gamma) = \gamma^2 \bar{B} + O(\gamma^{5/2})$ , that is  $\bar{A}$  is *quadratic* in  $\gamma$ , with  $\bar{B}$  non-negative. Averaging does provably bring some benefits because the order in  $\gamma$  is higher (we assume  $\gamma$  small).

### 5.2 AVERAGING PREDICTIONS

We are now ready to analyze the behavior of our new framework of averaging predictions. The following result is shown in the supplementary material.

**Proposition 1** *Under the assumptions on the negative loglikelihoods  $f_n$  of each observation from Dieuleveut et al. (2017):*

- *In the case of well-specified data, that is, there exists  $\theta_*$  such that for all  $(x, y)$ ,  $p(y|x) = q(y|x, \theta_*)$ , then  $\bar{A} \sim \gamma^2 \bar{B}^{\text{well}}$ , where  $\bar{B}^{\text{well}}$  is a positive constant.*
- *In the general case of potentially mis-specified data,  $\bar{A} = \gamma \bar{B}^{\text{mis}} + O(\gamma^2)$ , where  $\bar{B}^{\text{mis}}$  is constant which may be positive or negative.*

Note, that in contrast to averaging parameters, the constant  $\bar{B}^{\text{mis}}$  can be negative. It means, that we obtain the

estimator better than the optimal linear estimator, which is the limit of capacity for averaging parameters. In our simulations, we show examples for which  $\bar{B}^{\text{mis}}$  is positive, and examples for which it is negative. Thus, in general, for low-dimensional models, averaging predictions can be worse or better than averaging parameters. However, as we show in the next section, for infinite-dimensional models, we always get convergence.

## 6 INFINITE-DIMENSIONAL MODELS

Recall, that we have the following objective function to minimize:

$$F(\theta) = \mathbb{E}_{x,y} \left[ -y \cdot \eta_\theta(x) + a(\eta_\theta(x)) \right], \quad (6)$$

where till this moment we consider unknown functions  $\eta_\theta(x)$  which were linear in  $\Phi(x)$  with  $\Phi(x) \in \mathbb{R}^d$ , leading to a complexity in  $O(dn)$ .

We now consider infinite-dimensional features, by considering that  $\Phi(x) \in \mathcal{H}$ , where  $\mathcal{H}$  is a Hilbert space. Note that this corresponds to modeling the function  $\eta_\theta$  as a Gaussian process (Rasmussen and Williams, 2006).

This is computationally feasible through the usual “kernel trick” (Scholkopf and Smola, 2001; Shawe-Taylor and Cristianini, 2004), where we assume that the kernel function  $k(x, y) = \langle \Phi(x), \Phi(y) \rangle$  is easy to compute. Indeed, following Bordes et al. (2005) and Dieuleveut and Bach (2016), starting from  $\theta_0$ , each iterate of constant-step-size SGD is of the form  $\theta_n = \sum_{t=1}^n \alpha_t \Phi(x_t)$ , and the gradient descent recursion  $\theta_n = \theta_{n-1} - \gamma [a'(\eta_{\theta_{n-1}}(x_n)) - y_n] \Phi(x_n)$  leads to the following recursion on  $\alpha_t$ 's:

$$\begin{aligned} \alpha_n &= -\gamma \left[ a' \left( \sum_{t=1}^{n-1} \alpha_t \langle \Phi(x_t), \Phi(x_n) \rangle \right) - y_n \right] \\ &= -\gamma \left[ a' \left( \sum_{t=1}^{n-1} \alpha_t k(x_t, x_n) \right) - y_n \right]. \end{aligned}$$

This leads to  $\eta_{\theta_n}(x) = \langle \Phi(x), \theta_n \rangle$  and  $\mu_{\theta_n}(x) = a'(\eta_{\theta_n}(x))$  with

$$\eta_{\theta_n}(x) = \sum_{t=1}^n \alpha_t \langle \Phi(x), \Phi(x_t) \rangle = \sum_{t=1}^n \alpha_t k(x, x_t),$$

and finally we can express  $\bar{\mu}_n(x)$  in kernel form as:

$$\bar{\mu}_n(x) = \frac{1}{n+1} \sum_{t=0}^n a' \left[ \sum_{l=1}^t \alpha_l \cdot k(x, x_l) \right].$$

There is also a straightforward estimator for averaging parameters, i.e.,  $\bar{\mu}_n(x) = a' \left( \frac{1}{n+1} \sum_{t=0}^n \sum_{l=1}^t \alpha_l k(x, x_l) \right)$ . If we assume that the kernel function is *universal*, that is,  $\mathcal{H}$

is dense in the space of squared integrable functions, then it is known that if  $\mathbb{E}_x b(x)\Phi(x) = 0$ , then  $b = 0$  (Sriperumbudur et al., 2008). This implies that we must have  $\bar{\mu}_\gamma = 0$  and thus averaging predictions does always converge to the global optimum (note that in this setting, we must have a well-specified model because we are in a non-parametric setting).

**Column sampling.** Because of the usual high running-time complexity of kernel method in  $O(n^2)$ , we consider a ‘‘column-sampling approximation’’ (Williams and Seeger, 2001). We thus choose a small subset  $I = (x_1, \dots, x_m)$  of samples and construct a new finite  $m$ -dimensional feature map  $\bar{\Phi}(x) = K(I, I)^{-1/2}K(I, x) \in \mathbb{R}^m$ , where  $K(I, I)$  is the  $m \times m$  kernel matrix of the  $m$  points and  $K(I, x)$  the vector composed of kernel evaluations  $k(x_i, x)$ . This allows a running-time complexity in  $O(m^2n)$ . In theory and practice,  $m$  can be chosen small (Bach, 2013; Rudi et al., 2017).

**Regularized learning with kernels.** Although we can use an unregularized recursion with good convergence properties (Dieuleveut and Bach, 2016), adding a regularisation by the squared Hilbertian norm is easier to analyze and more stable with limited amounts of data. We thus consider the recursion (in Hilbert space), with  $\lambda$  small:

$$\begin{aligned} \theta_n &= \theta_{n-1} - \gamma [f'_n(\theta_{n-1}) + \lambda \theta_{n-1}] \\ &= \theta_{n-1} + \gamma (y_n - a'(\langle \Phi(x_n), \theta \rangle)) \Phi(x_n) - \gamma \lambda \theta_{n-1}. \end{aligned}$$

This recursion can also be computed efficiently as above using the kernel trick and column sampling approximations.

In terms of convergence, the best we can hope for is to converge to the minimizer  $\theta_{*,\lambda}$  of the regularized expected negative log-likelihood  $F(\theta) + \frac{\lambda}{2} \|\theta\|^2$  (which we assume to exist). When  $\lambda$  tends to zero, then  $\theta_{*,\lambda}$  converges to  $\theta_*$ .

Averaging *parameters* will tend to a limit  $\bar{\theta}_{\gamma,\lambda}$  which is  $O(\gamma)$ -close to  $\theta_{*,\lambda}$ , thus leading to predictions which deviate from the optimal predictions for two reasons: because of regularization and because of the constant step-size. Since  $\lambda$  should decrease as we get more data, the first effect will vanish, while the second will not.

When averaging *predictions*, the two effects will vanish as  $\lambda$  tends to zero. Indeed, by taking limits of the gradient equation, and denoting by  $\bar{\mu}_{\gamma,\lambda}$  the limit of  $\bar{\mu}_n$ , we have

$$\mathbb{E}[(\mu_{**}(x) - \bar{\mu}_{\gamma,\lambda}(x))\Phi(x)] = \lambda \bar{\theta}_{\gamma,\lambda}. \quad (7)$$

Given that  $\bar{\theta}_{\gamma,\lambda}$  is  $O(\gamma)$ -away from  $\theta_*$ , if we assume that

$\theta_*$  corresponds to a sufficiently regular<sup>2</sup> element of the Hilbert space  $\mathcal{H}$ , then the  $L_2$ -norm of the deviation satisfies  $\|\mu_{**}(x) - \bar{\mu}_{\gamma,\lambda}\| = O(\lambda)$  and thus as the regularization parameter  $\lambda$  tends to zero, our predictions tend to the optimal one.

## 7 EXPERIMENTS

In this section, we compare the two types of averaging (estimators and predictions) on a variety of problems, both on synthetic data and on standard benchmarks. When averaging predictions, we always consider the Taylor expansion approximation presented at the end of Section 4.3.

### 7.1 SYNTHETIC DATA

**Finite-dimensional models.** we consider the following logistic regression model:

$$q(y|x, \theta) = \exp(y \cdot \eta_\theta(x) - a(\eta_\theta(x))),$$

where we consider a linear model  $\eta_\theta(x) = \theta^\top x$  in  $x$  (i.e.,  $\Phi(x) = x$ ), the link function  $a(t) = \log(1 + e^t)$  and  $a'(t) = \sigma(t)$  is the sigmoid function. Let  $x$  be distributed as a standard normal random variable in dimension  $d = 2$ ,  $y \in \{0, 1\}$  and  $\mathbb{P}(y = 1|x) = \mu_{**}(x) = \sigma(\eta_{**}(x))$ , where we consider two different settings:

- Model 1:  $\eta_{**}(x) = \sin x_1 + \sin x_2$ ,
- Model 2:  $\eta_{**}(x) = x_1^3 + x_2^3$ .

The global minimum  $\mathcal{F}_{**}$  of the corresponding optimization problem can be found as

$$\mathcal{F}_{**} = \mathbb{E}_{p(x)}[-\mu_{**}(x) \cdot \eta_{**}(x) + a(\eta_{**}(x))].$$

We also introduce the performance measure  $\mathcal{F}(\eta)$

$$\mathcal{F}(\eta) = \mathbb{E}_{p(x)}[-\mu_{**}(x) \cdot \eta(x) + a(\eta(x))], \quad (8)$$

which can be evaluated directly in the case of synthetic data. Note that in our situation, the model is misspecified because  $\eta_{**}(x)$  is not linear in  $\Phi(x) = x$ , and thus,  $\inf_\theta F(\theta) > \mathcal{F}_{**}$ , and thus our performance measures  $\mathcal{F}(\mu_n) - \mathcal{F}_{**}$  for various estimators  $\mu_n$  will not converge to zero.

The results of averaging 10 replications are shown in Fig. 3 and Fig. 4. We first observed that constant step-size SGD without averaging leads to a bad performance.

<sup>2</sup>While our reasoning is informal here, it can be made more precise by considering so-called ‘‘source conditions’’ commonly used in the analysis of kernel methods (Caponnetto and De Vito, 2007), but this is out of the scope of this paper.

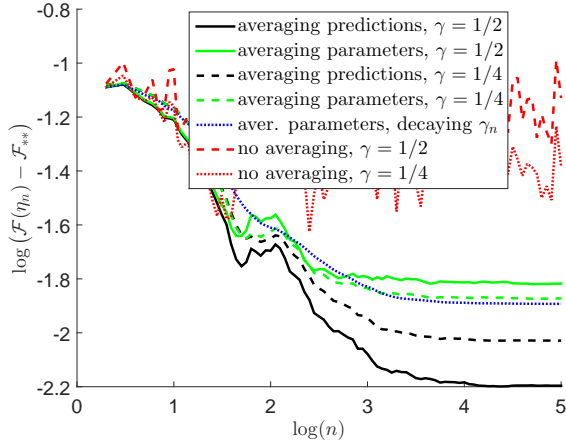


Figure 3: Synthetic data for linear model  $\eta_\theta(x) = \theta^\top x$  and  $\eta_{**}(x) = \sin x_1 + \sin x_2$ . Excess prediction performance vs. number of iterations (both in log-scale).

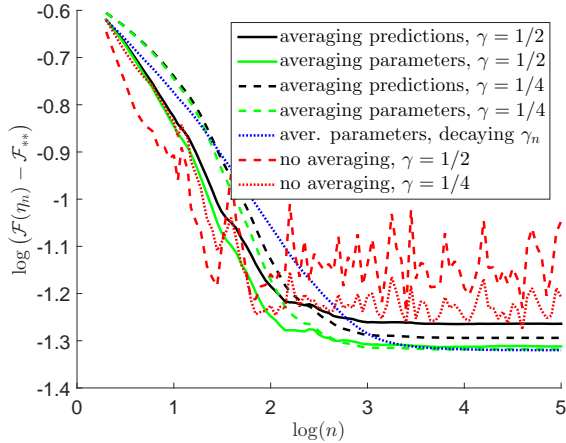


Figure 4: Synthetic data for linear model  $\eta_\theta(x) = \theta^\top x$  and  $\eta_{**}(x) = x_1^3 + x_2^3$ . Excess prediction performance vs. number of iterations (both in log-scale).

Moreover, we can see, that in the first case (Fig. 3) averaging predictions beats averaging parameters, and moreover beats the best linear model: if we use the best linear error  $\mathcal{F}_*$  instead of  $\mathcal{F}_{**}$ , at some moment  $\mathcal{F}(\eta_n) - \mathcal{F}_*$  becomes negative. However in the second case (Fig. 4), averaging predictions is not superior to averaging parameters. Moreover, by looking at the final differences between performances with different values of  $\gamma$ , we can see the dependency of the final performance in  $\gamma$  for averaging predictions, instead of  $\gamma^2$  for averaging parameters (as suggested by our theoretical results in Section 5). In particular in Fig. 3, we can observe the surprising behavior of a larger step-size leading to a better performance (note that we cannot increase too much otherwise the algorithm would diverge).

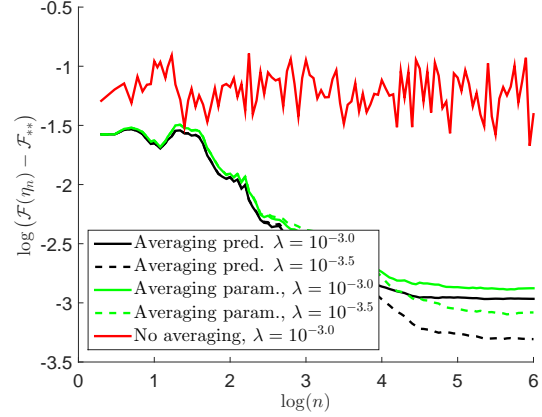


Figure 5: Synthetic data for Laplacian kernel for  $\eta_{**}(x) = \frac{5}{5+x^\top x}$ . Excess prediction performance vs. number of iterations (both in log-scale).

**Infinite-dimensional models** Here we consider the kernel setup described in Section 6. We consider Laplacian kernels  $k(s, t) = \exp\left(-\frac{\|s-t\|_1}{\sigma}\right)$  with  $\sigma = 50$ , dimension  $d = 5$  and generating log odds ratio  $\eta_{**}(x) = \frac{5}{5+x^\top x}$ . We also use a squared norm regularization with several values of  $\lambda$  and column sampling with  $m = 100$  points. We use the exact value of  $\mathcal{F}_{**}$  which we can compute directly for synthetic data. The results are shown in Fig. 5, where averaging predictions leads to a better performance than averaging estimators.

## 7.2 REAL DATA

Note, that in the case of real data, one does not have access to unknown  $\mu_{**}(x)$  and computing the performance measure in Eq. (8) is inapplicable. Instead of it we use its sampled version on held out data:

$$\hat{\mathcal{F}}(\eta) = - \sum_{i:y_i=1} \log(\mu(x_i)) - \sum_{j:y_j=0} \log(1 - \mu(x_j)).$$

We use two datasets, with  $d$  not too large, and  $n$  large, from (Lichman, 2013): the “MiniBooNE particle identification” dataset ( $d = 50$ ,  $n = 130\,064$ ), the “Covertypes” dataset ( $d = 54$ ,  $n = 581\,012$ ).

We use two different approaches for each of them: a linear model  $\eta_\theta(x) = \theta^\top x$  and a kernel approach with Laplacian kernel  $k(s, t) = \exp\left(-\frac{\|s-t\|_1}{\sigma}\right)$ , where  $\sigma = d$ . The results are shown in Figures 6 to 9. Note, that for linear models we use  $\hat{\mathcal{F}}_*$ —the estimator of the best performance among linear models (learned on the test set, and hence not reachable from learning on the training data), and for kernels we use  $\hat{\mathcal{F}}_{**}$  (same definition as  $\hat{\mathcal{F}}_*$  but with the kernelized model), that is why graphs are not comparable (but, as shown below, the value of  $\hat{\mathcal{F}}_{**}$  is

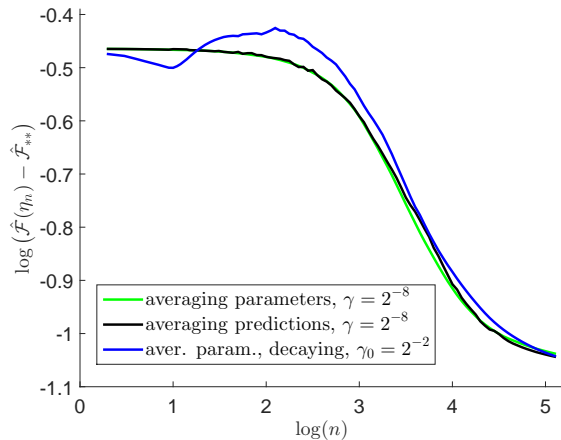


Figure 6: MiniBooNE dataset, dimension  $d = 50$ , linear model. Excess prediction performance vs. number of iterations (both in log-scale).

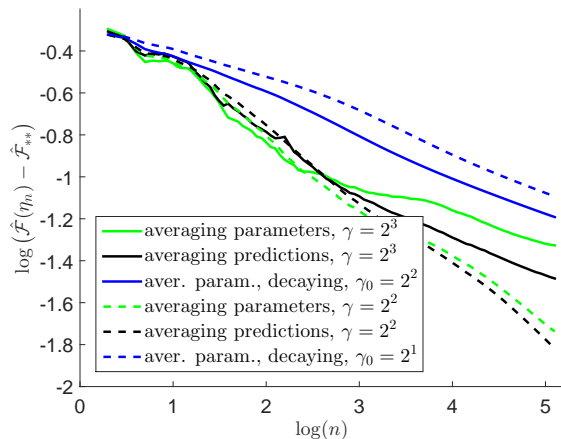


Figure 7: MiniBooNE dataset, dimension  $d = 50$ , kernel approach, column sampling  $m = 200$ . Excess prediction performance vs. number of iterations (both in log-scale).

lower than the value of  $\hat{\mathcal{F}}_*$  because using kernels correspond to a larger feature space).

For the “MiniBooNE particle identification” dataset  $\hat{\mathcal{F}}_* = 0.35$  and  $\hat{\mathcal{F}}_{**} = 0.21$ ; for the “Covertypes” dataset  $\hat{\mathcal{F}}_* = 0.46$  and  $\hat{\mathcal{F}}_{**} = 0.39$ . We can see from the four plots that, especially in the kernel setting, averaging predictions also shows better performance than averaging parameters.

## 8 CONCLUSION

In this paper, we have explored how averaging procedures in stochastic gradient descent, which are crucial for fast convergence, could be improved by looking at the specifics of probabilistic modeling. Namely, averaging

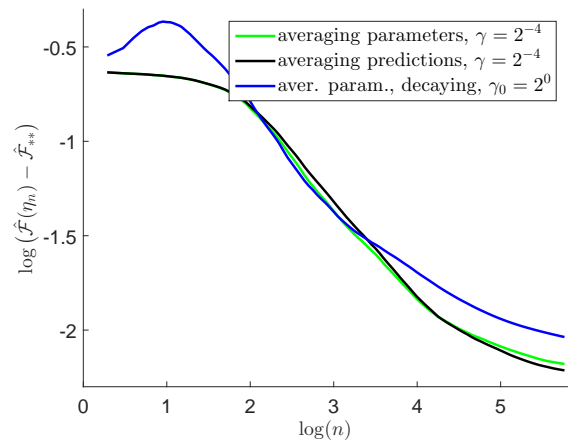


Figure 8: CoverType dataset, dimension  $d = 54$ , linear model. Excess prediction performance vs. number of iterations (both in log-scale).

in the moment parameterization can have better properties than averaging in the natural parameterization.

While we have provided some theoretical arguments (asymptotic expansion in the finite-dimensional case, convergence to optimal predictions in the infinite-dimensional case), a detailed theoretical analysis with explicit convergence rates would provide a better understanding of the benefits of averaging predictions.

## Acknowledgements

The research leading to these results has received funding from the European Union’s H2020 Framework Programme (H2020-MSCA-ITN-2014) under grant agreement n<sup>o</sup> 642685 MacSeNet, and from the European Research Council (grant SEQUOIA 724063).

## References

- F. Bach. Sharp analysis of low-rank kernel matrix approximations. In *Conference on Learning Theory*, pages 185–209, 2013.
- F. Bach and E. Moulines. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Adv. NIPS*, 2011.
- F. Bach and E. Moulines. Non-strongly-convex smooth stochastic approximation with convergence rate  $O(1/n)$ . *Advances in Neural Information Processing Systems (NIPS)*, 2013.
- A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research*, 6(Sep):1579–1619, 2005.

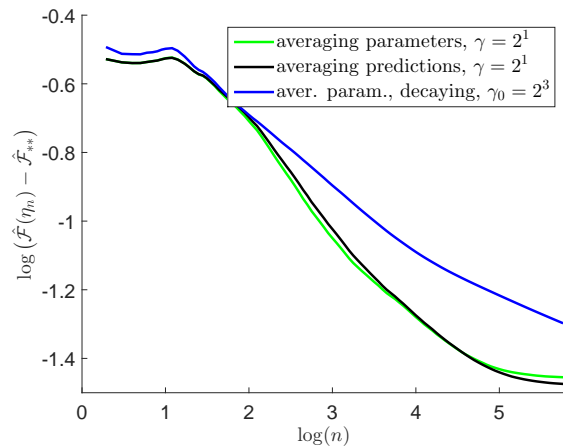


Figure 9: CoverType dataset, dimension  $d = 54$ , kernel approach, column sampling  $m = 200$ . Excess prediction performance vs. number of iterations (both in log-scale).

- L. Bottou, F. E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. Technical Report 1606.04838, arXiv, 2016.
- A. Caponnetto and E. De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368, 2007.
- A. Dieuleveut and F. Bach. Nonparametric stochastic approximation with large step-sizes. *Ann. Statist.*, 44(4):1363–1399, 08 2016. doi: 10.1214/15-AOS1391. URL <http://dx.doi.org/10.1214/15-AOS1391>.
- A. Dieuleveut, A. Durmus, and F. Bach. Bridging the gap between constant step size stochastic gradient descent and markov chains. Technical Report arXiv:1707.06386, arXiv, 2017.
- W. R. Gilks, S. Richardson, and D. Spiegelhalter. *Markov chain Monte Carlo in practice*. CRC press, 1995.
- I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- J. M. Hilbe. *Negative binomial regression*. Cambridge University Press, 2011.
- D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009. ISBN 0262013193, 9780262013192.
- J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. ICML*, 2001.
- M. Lichman. UCI machine learning repository, 2013. URL <http://archive.ics.uci.edu/ml>.
- P. McCullagh. Generalized linear models. *European Journal of Operational Research*, 16(3):285–292, 1984.
- S. P. Meyn and R. L. Tweedie. *Markov chains and stochastic stability*. Springer-Verlag Inc, Berlin; New York, 1993.
- K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. ISBN 0262018020, 9780262018029.
- B. T. Polyak and A. B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.
- C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.
- A. Rudi, L. Carratino, and L. Rosasco. Falkon: An optimal large scale kernel method. In *Advances in Neural Information Processing Systems*, pages 3891–3901, 2017.
- B. Scholkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and beyond*. MIT press, 2001.
- J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge university press, 2004.
- B. K. Sriperumbudur, A. Gretton, K. Fukumizu, G. Lanckriet, and B. Schölkopf. Injective hilbert space embeddings of probability measures. In *Proc. COLT*, 2008.
- C. K.I. Williams and M. Seeger. Using the nystrom method to speed up kernel machines. In *Advances in neural information processing systems*, pages 682–688, 2001.
- D. P. Woodruff et al. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1–2):1–157, 2014.



---

# Discrete Sampling using Semigradient-based Product Mixtures

---

**Alkis Gotovos**  
ETH Zurich  
alkisg@inf.ethz.ch

**Hamed Hassani**  
University of Pennsylvania  
hassani@seas.upenn.edu

**Andreas Krause**  
ETH Zurich  
krausea@ethz.ch

**Stefanie Jegelka**  
MIT  
stefje@mit.edu

## Abstract

We consider the problem of inference in discrete probabilistic models, that is, distributions over subsets of a finite ground set. These encompass a range of well-known models in machine learning, such as determinantal point processes and Ising models. Locally-moving Markov chain Monte Carlo algorithms, such as the Gibbs sampler, are commonly used for inference in such models, but their convergence is, at times, prohibitively slow. This is often caused by state-space bottlenecks that greatly hinder the movement of such samplers. We propose a novel sampling strategy that uses a specific mixture of product distributions to propose global moves and, thus, accelerate convergence. Furthermore, we show how to construct such a mixture using semigradient information. We illustrate the effectiveness of combining our sampler with existing ones, both theoretically on an example model, as well as practically on three models learned from real-world data sets.

## 1 INTRODUCTION

Discrete probabilistic models have played a fundamental role in machine learning. Examples range from classic graphical models, such as Ising and Potts models (Koller and Friedman, 2009), which have long been used in computer vision applications (Boykov et al., 2001), to determinantal point processes (Kulesza and Taskar, 2012) used in video summarization (Gong et al., 2014), and facility location diversity models used for product recommendation (Tschitschek et al., 2016). Recently, there has been increased interest in general distributions over subsets of a finite ground set  $V$ ; that is, given a

set function  $F : 2^V \rightarrow \mathbb{R}$ , distributions of the form  $\pi(S) \propto \exp(F(S))$ , for all  $S \subseteq V$ . These can be equivalently seen as distributions over binary random vectors, if  $S$  is replaced by the indicator function of the corresponding vector. All the aforementioned examples can be expressed in this form for a suitable choice of  $F$ .

While exact inference in such models is known to be intractable in general (Jerrum and Sinclair, 1993), there has been recent work on analyzing approximate inference techniques, such as variational methods (Djolonga and Krause, 2014; Djolonga et al., 2016b), and Markov chain Monte Carlo (MCMC) sampling (Gotovos et al., 2015; Rebeschini and Karbasi, 2015). The sampling analyses, in particular, focus on the Gibbs sampler, and derive sufficient conditions under which it mixes—converges toward the target distribution—sufficiently fast.

Unfortunately, oftentimes in practice these conditions do not hold and the Gibbs sampler mixes prohibitively slowly. A fundamental reason for this slow mixing behavior is the existence of bottlenecks in the state space of the Markov chain. Conceptually, one can think about the state-space graph containing several isolated components that are poorly connected to each other, thus making it hard for the Gibbs sampler to move between them.

In this work, we propose a novel sampling strategy that allows for global moves in the state space, thereby avoiding bottlenecks, and, thus, accelerating mixing. Our sampler is based on using a proposal distribution that approximates the target  $\pi$  by a mixture of product distributions. We further propose an algorithm for constructing such a mixture using discrete semigradient information of the associated function  $F$ . This idea makes a step towards bridging optimization and sampling, a theme that has been successful in continuous spaces. Our sampler is readily combined with other existing samplers, and we show provable theoretical, as well as empirical examples of speedups.

**Contributions.** The main contributions of this paper are as follows.

- We propose the  $M^3$  sampler, which makes global moves according to a specific mixture of product distributions.
- We theoretically analyze mixing times on an illustrative family of Ising models, and prove that adding the  $M^3$  sampler results in an exponential improvement over the Gibbs sampler.
- We demonstrate the effectiveness of combining the  $M^3$  and Gibbs samplers in practice on three models learned from real-world data.

**Related work.** Recent work on analyzing the mixing time of MCMC samplers for discrete probabilistic models includes deriving general conditions on  $F$  to achieve fast mixing (Gotovos et al., 2015; Rebeschini and Karbasi, 2015; Li et al., 2016), as well as looking at specific subclasses, such as strongly Rayleigh distributions (Li et al., 2016; Anari et al., 2016).

There has also been work on mapping discrete inference to continuous domains (Zhang et al., 2012; Pakman and Paninski, 2013; Dinh et al., 2017; Nishimura et al., 2018) to enable the use of well-established continuous samplers, such as Hamiltonian Monte Carlo (Neal, 2012; Betancourt, 2017). It is worth pointing out that, while these methods usually outperform simple Gibbs or Metropolis samplers, they still tend to suffer from considerable slowdowns in multimodal distributions (Neal, 2012). Our work is orthogonal to these methods, in the sense that our proposed sampler can be combined with any of the existing ones to provide a principled way for performing global moves that can lead to improved mixing.

Both darting Monte Carlo (Sminchisescu and Welling, 2007; Ahn et al., 2013) and variational MCMC (de Freitas et al., 2001) share the high-level concept of combining two chains, one making global moves between high-probability regions, and another making local moves around those regions. However, their proposed global samplers for continuous spaces are generally not applicable to the class of discrete distributions we consider.

There are several well-known results on mixing of the Gibbs sampler for the Ising model on different graph structures (Jerrum and Sinclair, 1993; Berger et al., 2005; Levin et al., 2008a;b). Other (non-MCMC) approaches to discrete sampling include Perturb-and-MAP (Papan-dreou and Yuille, 2011; Hazan et al., 2013), and random projections (Zhu and Ermon, 2015). Semigradients of submodular set functions have recently been exploited

for optimization (Iyer et al., 2013; Jegelka and Bilmes, 2011) and variational inference (Djolonga et al., 2016a), but, to our knowledge, no prior work has used them for sampling.

## 2 BACKGROUND

We consider set functions  $F : 2^V \rightarrow \mathbb{R}$ , where  $V$  is a finite ground set of size  $n$  that can be assumed to be  $V = \{1, \dots, n\}$  without loss of generality. In this paper, we focus on distributions over  $\Omega := 2^V$  of the form

$$\pi(S) = \frac{1}{Z} \exp(F(S)), \quad (1)$$

for all  $S \in \Omega$ . The partition function  $Z := \sum_{S \in \Omega} \exp(F(S))$  serves as the normalizer of the distribution. Alternatively, we can describe distributions of the above form via binary vectors  $X \in \{0, 1\}^n$ . If we define  $V(X) := \{v \in V \mid X_v = 1\}$ , then the distribution  $p_X(X) \propto \exp(F(V(X)))$  over binary vectors is isomorphic to the distribution (1) over sets.

Perhaps the simplest family of such models are log-modular distributions, which describe a collection of independent binary random variables. Equivalently, they are distributions of the form (1) where  $F$  is a modular function, that is, a function of the form  $F(S) = c + \sum_{v \in S} m_v$ , where  $c, m_v \in \mathbb{R}$ , for all  $v \in V$ . The partition function of a log-modular distribution can be derived in closed form as  $Z_m = \exp(c) \prod_{v \in V} (1 + \exp(m_v))$ . Consequently, the corresponding log-modular distribution is

$$\pi_m(S) = \frac{\exp(\sum_{v \in S} m_v)}{\prod_{v \in V} (1 + \exp(m_v))}.$$

**Inference and sampling.** Performing exact inference in models of the form (1), that is, computing conditional probabilities such as  $\pi(A \subseteq S \subseteq B \mid C \subseteq S \subseteq D)$ , is known to be in general #P-hard (Jerrum and Sinclair, 1993). As a result, we have to resort to approximate inference algorithms, such as Markov chain Monte Carlo sampling (Levin et al., 2008b), which is the primary focus of this paper. An MCMC algorithm for distribution  $\pi$  simulates a Markov chain in state space  $\Omega$  in such a way that the sequence of visited states  $(X_0, X_1, \dots) \in \Omega^{\mathbb{N}}$  ultimately converges to  $\pi$ .

**Gibbs sampler.** One of the most commonly used chains is the (single-site) Gibbs sampler, which adds or removes a single element at a time. It first selects uniformly at random an element  $v \in V$ ; subsequently, it adds or removes  $v$  to the current state  $X_t$  according to the probability of the resulting state. We denote by  $P : \Omega \times$

$\Omega \rightarrow \mathbb{R}$  the transition matrix of a Markov chain, that is, for all  $S, R \in \Omega$ ,  $P(S, R) := \mathbb{P}[X_{t+1} = R \mid X_t = S]$ . Then, if we define

$$p_{S \rightarrow R} = \frac{\exp(F(R))}{\exp(F(R)) + \exp(F(S))},$$

and denote by  $S \sim R$  states that differ by exactly one element (i.e.,  $\|R\| - \|S\| = 1$ ), the transition matrix  $P^G$  of the Gibbs sampler is

$$P^G(S, R) = \begin{cases} \frac{1}{n} p_{S \rightarrow R}, & \text{if } R \sim S \\ 1 - \sum_{T \sim S} \frac{1}{n} p_{S \rightarrow T}, & \text{if } R = S \\ 0, & \text{otherwise} \end{cases}.$$

**Mixing.** The efficiency of a Markov chain in approximating its target distribution depends largely on the speed of convergence of the chain, which is quantified by the chain's mixing time. Most commonly, distance from stationarity is measured by the maximum total variation distance, over all starting states, between  $X_t$  and the target distribution  $\pi$ , that is,  $d(t) := \max_{X_0 \in \Omega} d_{\text{TV}}(P^t(X_0, \cdot), \pi)$ . Then, the mixing time denotes the minimum number of iterations required to get  $\epsilon$ -close to stationarity,  $t_{\text{mix}}(\epsilon) := \min\{t \mid d(t) \leq \epsilon\}$ .

A common way to obtain an upper bound on the mixing time of a chain is by lower bounding its spectral gap, defined as  $\gamma := 1 - \lambda_2$ , where  $\lambda_2$  is the second largest eigenvalue of the transition matrix  $P$ . The following well-known theorem connects the spectral gap to mixing time.

**Theorem 1** (cf. Theorems 12.3, 12.4 in (Levin et al., 2008b)). *Let  $P$  be the transition matrix of a lazy, irreducible, and reversible Markov chain, and let  $\gamma$  be its spectral gap, and  $\pi_{\min} := \min_{S \in \Omega} \pi(S)$ . Then,*

$$\left(\frac{1}{\gamma} - 1\right) \log\left(\frac{1}{2\epsilon}\right) \leq t_{\text{mix}}(\epsilon) \leq \frac{1}{\gamma} \log\left(\frac{1}{\epsilon \pi_{\min}}\right).$$

### 3 THE MIXTURE CHAIN

Despite the simplicity and computational efficiency of the Gibbs sampler, the fact that it is constrained to performing local moves makes it susceptible to state-space bottlenecks, which hinder the movement of the chain around the state space. Intuitively, the state space may contain several high-probability regions arranged in such a way that moving from one to another using only single-element additions and deletions requires passing through states of very low probability. As a result, the Gibbs sampler may mix extremely slowly on the whole state space, despite the fact that it can move sufficiently fast within each of the high-probability regions.

To alleviate this shortcoming, it is natural to ask whether it is possible to bypass such bottlenecks by using a chain that performs larger moves. In this paper, we introduce a novel approach that uses a Metropolis chain based on a specific mixture of log-modular distributions, which we call the  $M^3$  chain, to perform global moves in state space. Concretely, we define a proposal distribution

$$\begin{aligned} q(S, R) &= q(R) = \frac{1}{Z_q} \sum_{i=1}^r \exp(F_i(R)) \\ &= \frac{1}{Z_q} \sum_{i=1}^r w_i \exp(m_i(R)), \end{aligned} \quad (2)$$

where each  $F_i(R) = c_i + \sum_{v \in R} m_{iv}$  is a modular function, while each  $m_i(R) = \sum_{v \in R} m_{iv}$  is a normalized modular function ( $m_i(\emptyset) = 0$ ), and  $w_i = \exp(c_i) > 0$ . If we denote by  $Z_i$  the normalizer of  $m_i$ , then the normalizer of the mixture can be written in closed form as

$$\begin{aligned} Z_q &= \sum_{R \in \Omega} q(R) = \sum_{R \in \Omega} \sum_{i=1}^r w_i \exp(m_i(R)) \\ &= \sum_{i=1}^r w_i \sum_{R \in \Omega} \exp(m_i(R)) \\ &= \sum_{i=1}^r w_i Z_i. \end{aligned}$$

We define the  $M^3$  chain as a Metropolis chain (Levin et al., 2008b) using  $q$  as a proposal distribution; its transition matrix  $P^M : \Omega \times \Omega \rightarrow \mathbb{R}$  is given by

$$P^M(S, R) = \begin{cases} q(R) p_a(S, R), & \text{if } R \neq S \\ 1 - \sum_{T \neq S} q(T) p_a(S, T), & \text{otherwise} \end{cases},$$

where

$$p_a(S, R) := \min \left\{ 1, \frac{\pi(R) q(S)}{\pi(S) q(R)} \right\}.$$

Note that, contrary to usual practice, the proposal  $q$  only depends on the proposed state, but not on the current state of the chain. As a result, the chain is not constrained to local moves, but rather can potentially jump to any part of the state space. In practice,  $M^3$  sampling proceeds in two steps: first, a candidate set  $R$  is sampled according to  $q$ ; then, the move to  $R$  is accepted with probability  $p_a$ . Sampling from  $q$  can be done in  $\mathcal{O}(n)$  time—first, sample a log-modular component, then sample a set from that component. Computing  $p_a$  requires  $\mathcal{O}(r)$  time for the sum in (2), and it can be straightforwardly improved by parallelizing this computation. All in all, the total time for one step of  $M^3$  is  $\mathcal{O}(n + r)$ .

As is always the case with Metropolis chains, the mixing time of the  $M^3$  sampler will depend on how well the proposal  $q$  approximates the target distribution  $\pi$ . The following observation shows that, in theory, we can approximate any distribution of the form (1) by a mixture of the form (2).

**Proposition 1.** *For any  $\pi$  on  $\Omega$  as in (1), and any  $\epsilon > 0$ , there are positive constants  $w_i = w_i(\epsilon) > 0$ , and normalized modular functions  $m_i = m_i(\epsilon)$ , such that, if we define  $q(S) := \sum_{i=1}^r w_i \exp(m_i(S))$ , for all  $S \in \Omega$ , then  $d_{TV}(\pi, q) \leq \epsilon$ .*

Conceptually, the proof relies on having one log-modular term per set in  $\Omega$ .<sup>1</sup> Therefore, while the above result shows that mixtures of log-modulars are expressive enough, the constructed mixture of exponential size in  $n$  is not useful for practical purposes. On the other hand, it is not necessary for us to have  $q$  be an accurate approximation of  $\pi$  everywhere, as long as the corresponding  $M^3$  chain is able to bypass state-space bottlenecks. With this in mind, we suggest combining the  $M^3$  and Gibbs chains, so that each of them serve complementary purposes in the final chain; the role of  $M^3$  is to make global moves and avoid bottlenecks, while the role of Gibbs is to move fast within well-connected regions of the state space. To make this happen, we define the transition matrix  $P^C : \Omega \times \Omega \rightarrow \mathbb{R}$  of the combined chain as

$$P^C(S, R) = \alpha P^G(S, R) + (1 - \alpha) P^M(S, R), \quad (3)$$

where  $0 < \alpha < 1$ . It is easy to see that  $P^C$  is reversible, and has stationary distribution  $\pi$ .

We next illustrate how combining the two chains works on a simple example, where a mixture of only a few log-modular distributions can dramatically improve mixing compared to running the vanilla Gibbs chain. Then we propose an algorithm for automatically creating such a mixture.

### 3.1 EXAMPLE: ISING MODEL ON THE COMPLETE GRAPH

We consider the Ising model on a finite complete graph (Levin et al., 2008a), also known as the Curie-Weiss model in statistical physics, which can be written in the form of (1) as follows:

$$\pi_\beta(S) = \frac{1}{Z(\beta)} \exp\left(-\frac{2\beta}{n} |S|(n - |S|)\right). \quad (\text{ISING}_\beta)$$

<sup>1</sup>Detailed proofs of all our results can be found in the appendix.

In particular, we focus on the case where  $\beta = \ln(n)$ , that is,

$$\pi(S) = \frac{1}{Z} \exp\left(-\frac{2 \ln(n)}{n} |S|(n - |S|)\right). \quad (\text{ISING})$$

In this case, if we define  $d_n := 2 \ln(n)/n$ , then  $F(S) = -d_n |S|(n - |S|)$ .

The Gibbs sampler is known to experience poor mixing in this model; the following is an immediate corollary of Theorem 15.3 in (Levin et al., 2008b).

**Corollary 1** (cf. Theorem 15.3 in (Levin et al., 2008b)). *For  $n \geq 3$ , the Gibbs sampler on ISING has spectral gap  $\gamma^G = \mathcal{O}(e^{-cn})$ , where  $c > 0$  is a constant.*

From Theorem 1 it follows that the mixing time of Gibbs is  $t_{\text{mix}}(\epsilon) = \Omega((e^{cn} - 1) \log(1/(2\epsilon)))$ . Yet, it has been shown that the only reason for this is a single bottleneck in the state space (Levin et al., 2008a). To make this statement more formal, let us define a decomposition of  $\Omega$  into two disjoint sets,  $\Omega_0 := \{S \in \Omega \mid |S| < n/2\}$ , and  $\Omega_1 := \{S \in \Omega \mid |S| > n/2\}$  (Jerrum et al., 2004). To keep things simple, we will assume for the remainder of this section that  $n$  is odd; the analysis when  $n$  is even follows from the same arguments with only a minor technical adjustment. Our goal is to separately examine two characteristics of the sampler: (i) its movement between the two sets  $\Omega_0$ ,  $\Omega_1$ , and (ii) its movement when restricted to stay within each of these sets.

For analyzing the “between-sets” behavior, we define the projection  $\bar{\pi} : \{0, 1\} \rightarrow \mathbb{R}$  of  $\pi$  as

$$\bar{\pi}(i) := \sum_{S \in \Omega_i} \pi(S),$$

and, for any reversible chain  $P$ , we define its projection chain  $\bar{P} : \{0, 1\} \times \{0, 1\} \rightarrow \mathbb{R}$  as

$$\bar{P}(i, j) := \frac{1}{\bar{\pi}(i)} \sum_{S \in \Omega_i, R \in \Omega_j} \pi(S) P(S, R).$$

It is easy to see that  $\bar{P}$  is also reversible and has stationary distribution  $\bar{\pi}$ . For analyzing the “within-set” behavior, we define the restrictions  $\pi_i : \Omega_i \rightarrow \mathbb{R}$  of  $\pi$  as

$$\pi_i(S) := \frac{\pi(S)}{\bar{\pi}(i)},$$

and the two restriction chains  $P_i : \Omega_i \times \Omega_i \rightarrow \mathbb{R}$  of  $P$  as

$$P_i(S, R) := \begin{cases} P(S, R), & \text{if } S \neq R \\ 1 - \sum_{T \in \Omega_i: T \neq S} P(S, T), & \text{otherwise} \end{cases}.$$

Again, it is easy to see that each of the  $P_i$  is also reversible and has stationary distribution  $\pi_i$ .

Coming back to the Gibbs sampler, if we could show that it mixes fast within each of  $\Omega_0$  and  $\Omega_1$ , then we could deduce that the only reason for the slow mixing on  $\Omega$  is the bottleneck between these two sets. Indeed, the following corollary of a theorem by [Ding et al. \(2009\)](#) shows exactly that.

**Corollary 2** (cf. Theorem 2 in ([Ding et al., 2009](#))). *For all  $n \geq 3$ , the restriction chains of the Gibbs sampler  $P_i^G$ ,  $i = 0, 1$ , on ISING have spectral gap  $\gamma_i^G = \Theta\left(\frac{2\ln(n)-1}{n}\right)$ .*

To improve mixing we want to create an  $M^3$  chain that is able to bypass the aforementioned bottleneck. For this purpose, we use a mixture of two log-modular distributions, the first of which puts most of its mass on  $\Omega_0$ , and the second on  $\Omega_1$ . We define the mixture of the form (4) by

$$m_1(S) = \sum_{v \in S} -d_n(n-1) = -d_n(n-1)|S|,$$

$$m_2(S) = \sum_{v \in S} d_n(n-1) = d_n(n-1)|S|.$$

We also use  $w_1 = 1/Z_1$  and  $w_2 = 1/Z_2$ , where  $Z_1$  and  $Z_2$  are the normalizers of  $m_1$  and  $m_2$  respectively. It follows that  $Z_q = 1/2$ , and, furthermore, the mixture  $q$  is symmetric, that is,  $q(S) = q(V \setminus S)$ . Since the proposal  $q$  is symmetric and state independent, we would expect the  $M^3$  chain to jump between  $\Omega_0$  and  $\Omega_1$  without being hindered by the bottleneck described previously. We verify this intuition by proving the following lemma.

**Lemma 1.** *For all  $n \geq 10$ , the projection chain  $\bar{P}^M$  of the  $M^3$  sampler on ISING has spectral gap  $\bar{\gamma}^M = \Omega(1)$ .*

Putting everything together we show the following result about the combined chain  $P^C$ .

**Theorem 2.** *For all  $n \geq 10$ , the combined chain  $P^C$  on ISING has spectral gap*

$$\gamma^C = \Omega\left(\frac{2\ln(n)-1}{2n}\right).$$

The proof consists of two steps. In the first step we make a comparison argument ([Diaconis and Saloff-Coste, 1993](#); [Levin et al., 2008b](#)) to show that the spectral gaps of the projection and restriction chains of the combined sampler are smaller by at most a constant factor in  $\alpha$  compared to those of Gibbs and  $M^3$ . In particular, we use the  $M^3$  bound ([Lemma 1](#)) for the projection chain, and the Gibbs bound ([Theorem 2](#)) for the restriction chains. The second step, then, combines the projection and restriction bounds to establish a bound on the spectral gap of the combined chain. To accomplish this we use a result by [Jerrum et al. \(2004\)](#), which, roughly

---

### Algorithm 1 Iterative semigradient-based mixture construction

---

**Input:** Set function  $F$ , mixture size  $r$

- 1: **for**  $i = 1$  **to**  $r$  **do**
- 2:    $\sigma \leftarrow \text{GREEDY}(F, \{m_1, \dots, m_{i-1}\})$
- 3:    $m_i \leftarrow \text{SEMIGRAIDENT}(F, \sigma)$
- 4: **return**  $\{m_1, \dots, m_r\}$

---

speaking, states that the spectral gap of the whole chain cannot be much smaller than the smallest of the projection and restriction spectral gaps.

Finally, using [Theorem 1](#), and noting that, in this case,  $\pi_{\min} = \mathcal{O}(e^{-n})$  (cf. proof of [Lemma 1](#)), we get a mixing time of  $t_{\text{mix}}(\epsilon) = \mathcal{O}(n^2 \log(1/\epsilon))$  for the combined chain. This shows that the addition of the  $M^3$  sampler results in an exponential improvement in mixing time over the Gibbs sampler by itself.

## 4 CONSTRUCTING THE MIXTURE

Having seen the positive effect of the  $M^3$  sampler, we now turn to the issue of how to choose the proposal  $q$ . While a manual construction like the one we just presented for the Ising model may be feasible in some cases, it is often more practical to have an automated way of obtaining the mixture.

Let us assume, as is usually the case, that we have access to a function oracle for  $F$ , and we want to create a mixture of size  $r$ . Ideally, we would like to construct a proposal  $q$  that is as close to  $\pi$  as possible, that is, minimize an objective such as the following,

$$E_1(q) := \min_q \|\pi - q\|$$

$$= \min_q \left\| \frac{\exp(F(\cdot))}{Z} - \frac{1}{Z_q} \sum_{i=1}^r w_i \exp(m_i(\cdot)) \right\|,$$

where  $\|\cdot\|$  could be, for example, total variation distance or the maximum norm. Unfortunately, this problem is hard: both computing the partition function  $Z$ , and jointly optimizing over all  $w_i, m_i$  are infeasible in practice. To make the problem easier, we could try to get rid of the normalizers and weights  $w_i$ , and iteratively minimize over each  $m_i$  individually:

$$E_2^{(i)}(m_i) := \min_{m_i} \left\| \exp(F(\cdot)) - \sum_{j=1}^{i-1} \exp(m_j(\cdot)) \right\|,$$

for  $i \in \{1, \dots, r\}$ . This problem is still hard, since optimizing  $\|\exp(F(\cdot))\|$  is by itself infeasible in general.

To arrive at a practical algorithm, we approximate the above objective using the two-step procedure described in [Algorithm 1](#). In the first step, we generate a permutation  $\sigma$  of the ground set  $V$  by running

---

**Algorithm 2** Greedy difference maximization

---

**Input:** Set function  $F$ , modular functions  $\{m_1, \dots, m_{i-1}\}$

- 1:  $D_i(S) \leftarrow F(S) - \log \sum_{j=1}^{i-1} \exp(m_j(S))$ , for all  $S \in \Omega$
- 2:  $\sigma \leftarrow (1, \dots, n)$
- 3:  $A \leftarrow \emptyset$
- 4: **for**  $i = 1$  **to**  $n$  **do**
- 5:    $v^* \leftarrow \operatorname{argmax}_{v \in V} (D_i(A \cup \{v\}) - D_i(A))$
- 6:    $\sigma_i \leftarrow v^*$
- 7:    $A \leftarrow A \cup \{v^*\}$
- 8: **return**  $\sigma$

---

the greedy algorithm on function  $D_i(S) := F(S) - \log \sum_{j=1}^{i-1} \exp(m_j(S))$ , as shown in [Algorithm 2](#). Intuitively, the sets that are formed by elements near the beginning of  $\sigma$  are those on which  $F$  and the current mixture disagree by the most. Therefore, in the second step, we would like to add to the mixture a modular function  $m_i$  that is a good approximation for  $F$  on  $\{\sigma_1, \dots, \sigma_k\}$ , for a choice of  $1 \leq k \leq n$ . To accomplish this, we propose using discrete semigradients.

Semigradients are modular functions that provide lower (subgradient) or upper (supergradient) approximations of a set function  $F$  ([Fujishige, 2005](#); [Iyer et al., 2013](#)). More concretely, given a set  $S \in \Omega$ , a modular function  $m$  is a subgradient of  $F$  at  $S$ , if, for all  $R \in \Omega$ ,  $F(R) \geq F(S) + m(R) - m(S)$ . Similarly,  $m$  is a supergradient if the inequality is reversed. Although, in general, a function is not guaranteed to have sub- or supergradients at each  $S \in \Omega$ , it has been shown that this is true when  $F$  is submodular or supermodular ([Fujishige, 2005](#); [Jegelka and Bilmes, 2011](#); [Iyer and Bilmes, 2012](#)).

Submodularity expresses a notion of diminishing returns; that is, adding an element to a larger set provides less benefit than adding that same element to a smaller set. More formally,  $F$  is submodular if, for any  $S \subseteq R \subseteq V$ , and any  $v \in V \setminus R$ , it holds that  $F(R \cup \{v\}) - F(R) \leq F(S \cup \{v\}) - F(S)$ . Supermodularity is defined in a similar way by reversing the sign of this inequality. The resulting models of the form (1) are referred to as log-submodular and log-supermodular respectively. Many commonly used models fall under these categories; Ising and Potts models, including our example in the previous section, are log-supermodular, while determinantal point processes and facility location diversity models are log-submodular.

Coming back to the second step of [Algorithm 1](#), to create a subgradient of  $F$  given permutation  $\sigma$  we just need to define a modular function via marginal gains according to the permutation order ([Iyer et al., 2013](#)), as shown in [Algorithm 3](#). Moreover, this is a subgradient of  $F$  at  $\{\sigma_1, \dots, \sigma_k\}$ , for all  $1 \leq k \leq n$ . On the other hand,

---

**Algorithm 3** Subgradient computation

---

**Input:** Set function  $F$ , permutation  $\sigma$

- 1:  $A \leftarrow \emptyset$
- 2:  $f \leftarrow F(\emptyset)$
- 3: **for**  $v = 1$  **to**  $n$  **do**
- 4:    $m_v \leftarrow F(A \cup \{\sigma_v\}) - F(A)$
- 5:    $A \leftarrow A \cup \sigma_v$
- 6: **return**  $m(S) := \sum_{v \in S} m_v$ , for all  $S \in \Omega$

---



---

**Algorithm 4** Supergradient computation

---

**Input:** Set function  $F$ , permutation  $\sigma$

- 1:  $k \leftarrow \text{DRAWUNIFORM}(1, n)$
- 2: **for**  $v = 1$  **to**  $k$  **do**
- 3:    $m_v \leftarrow F(V) - F(V \setminus \{v\})$
- 4: **for**  $v = k + 1$  **to**  $n$  **do**
- 5:    $m_v \leftarrow F(\{v\})$
- 6: **return**  $m(S) := \sum_{v \in S} m_v$ , for all  $S \in \Omega$

---

[Algorithm 4](#) creates a supergradient of  $F$  at  $\{\sigma_1, \dots, \sigma_k\}$  for a randomly chosen  $k$ . (This type of supergradient is denoted by  $\bar{g}_Y$  by [Iyer et al. \(2013\)](#).) In fact, the modular functions  $m_1, m_2$  that we used in analyzing the Ising model in the previous section were supergradients of  $F$  at sets  $S_1 = \emptyset$ , and  $S_2 = V$  respectively.

In practice, we can use [Algorithm 1](#) regardless of whether  $F$  is sub- or supermodular. We have, however, noticed that subgradients give better results when  $F$  is submodular, and the same goes for supergradients and supermodular functions.

## 5 EXPERIMENTS

We now evaluate the performance of our proposed sampler on the Ising model we analyzed earlier, as well as the following three models learned from real-world data sets.

**WATER.** A (log-submodular) facility location model, which was used in a problem of sensor placement in a water distribution network ([Krause et al., 2008](#)). The function  $F$  is of the form

$$F(S) = \sum_{j=1}^L \max_{i \in S} c_{ij}.$$

We randomly subsample the original facility location matrix  $C = (c_{ij})$ , so that  $n = 50$ , and  $L = 500$ .

**SENSOR.** A (log-submodular) determinantal point process ([Kulesza and Taskar, 2012](#)), which was used in a problem of sensor placement for indoor temperature

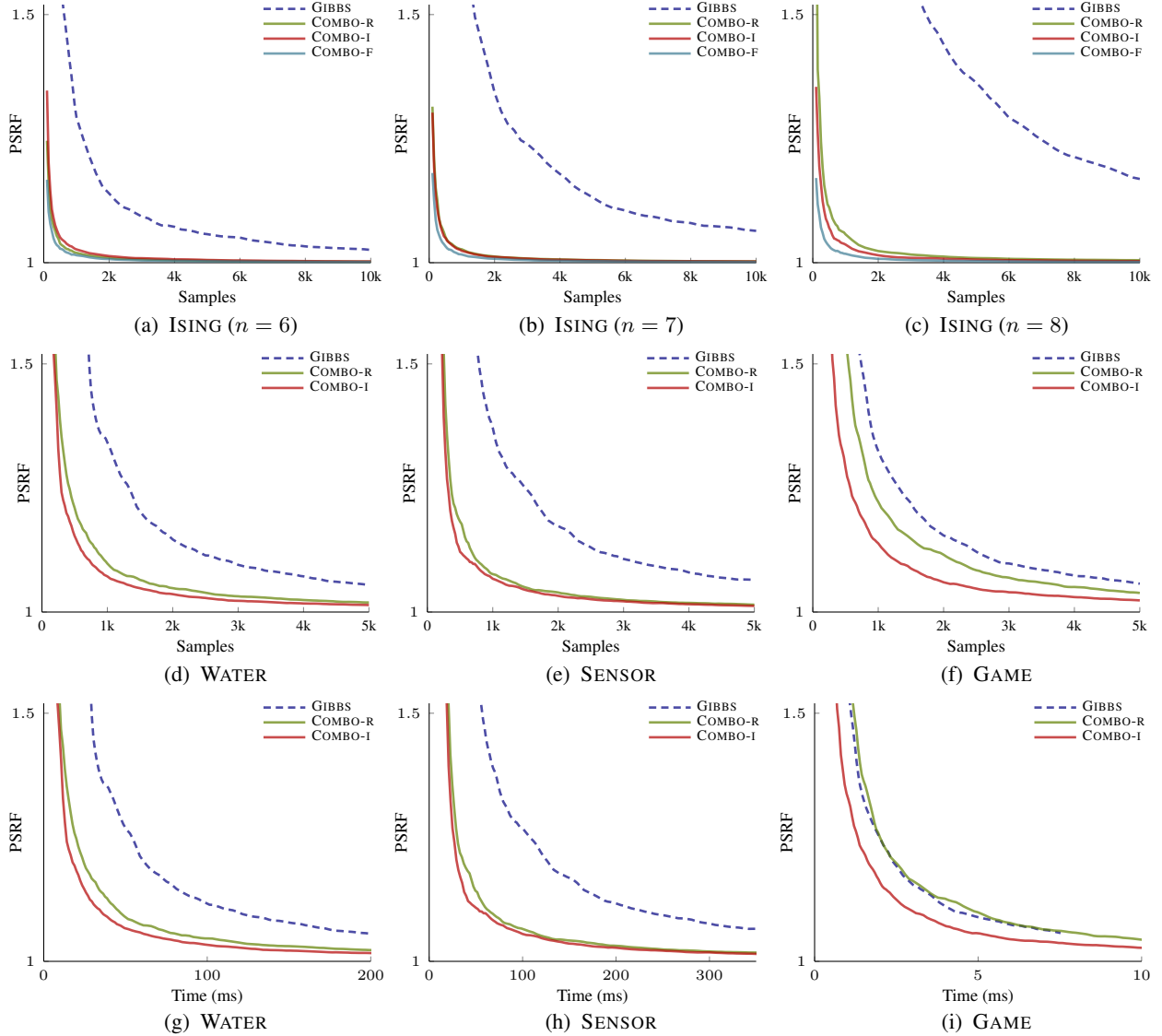


Figure 1: (a)-(c) Ising model results for increasing  $n$ . Note how the Gibbs sampler gets worse significantly faster than the combined ones. (d)-(f) Potential scale reduction factor (PSRF) as a function of sampling iterations. (g)-(i) PSRF as a function of wall-clock time in milliseconds. The combined sampler outperforms Gibbs both in terms of samples required, as well as actual runtime.

monitoring (Guestrin et al., 2005). The function  $F$  is of the form

$$F(S) = \log |K + \sigma^2 I|,$$

where  $K$  is a kernel matrix, and  $\sigma$  is a noise parameter. The size of the ground set is  $n = 46$ .

**GAME.** A (log-submodular) facility location diversity model (Tschitschek et al., 2016), which represents the characters that are chosen by players in the popular online game “Heroes of the Storm”. We learned the model from an online data set of approximately 8,000 teams of

5 characters<sup>2</sup> using noise-contrastive estimation, as described by Tschitschek et al. (2016). The function  $F$  is of the form

$$F(S) = \sum_{v \in S} w_v + \sum_{j=1}^L \max_{i \in S} c_{ij},$$

with  $n = 48$ , and  $L = 10$ . In practice, we would only be interested in sampling sets of fixed size  $\ell = 5$ . The Gibbs sampler can be easily modified to sample under a cardinality constraint by using moves that swap an element in

<sup>2</sup><https://www.hotlogs.com>

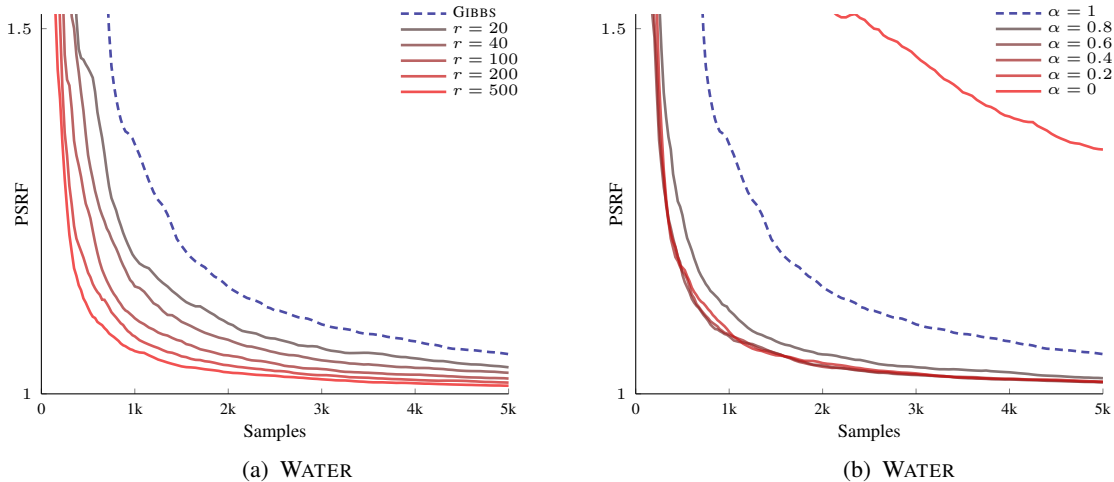


Figure 2: (a) Increasing the number of mixture components improves performance. (b) The combination of Gibbs and  $M^3$  performs better than either of them does individually.

the current set  $X_t$  with an element in  $V \setminus X_t$ . Extending the  $M^3$  chain to sample from cardinality-constrained models is also straightforward. In fact, the only additional ingredient required is a procedure to sample a set of size  $\ell$  from a log-modular distribution, which can be easily done, as before, in  $\mathcal{O}(n)$  time.

In what follows, we compare the performance of the Gibbs sampler (GIBBS) against our proposed combined sampler using a proposal mixture  $q$  constructed by **Algorithm 1** (COMBO-I). We also compare against a variation where we substitute the greedy procedure in **line 2** of **Algorithm 1** with picking a permutation  $\sigma$  of the ground set uniformly at random (COMBO-R).

To assess convergence we use the potential scale reduction factor (PSRF) (Brooks et al., 2011) using 20 parallel chains. We compute the PSRF using single-element marginal probabilities averaged over 50 repetitions of each simulation.

In **Figures 1a–1c** we show the results for the Ising model ( $n = 6, 7, 8$ ) with the additional COMBO-F line denoting the combined sampler with two mixture components described in **Section 3.1**. The other two combined samplers use mixtures of size  $r = 20$ . Note that Gibbs mixes dramatically slower than the combined sampler, even for such small  $n$ .

In **Figures 1d–1f** we show the results on the three log-submodular models described before using mixtures of size  $r = 200$ . It is interesting to see that even random permutations are enough to significantly improve over the performance of Gibbs. Similar observations can be made with respect to computation time, as shown in **Figures 1g–1i**, which measure wall-clock time on the  $x$ -axis.

In **Figure 2a** we show how mixture size affects performance; as expected, adding more components to the mixture results in a proposal that approximates the target distribution better, and, therefore, mixes faster. Finally, in **Figure 2b** we see that both Gibbs ( $\alpha = 1$ ) and  $M^3$  ( $\alpha = 0, r = 200$ ) perform poorly by themselves, but combining them results in much improved performance. This highlights again the complementary nature of the two chains (local vs. global moves) we discussed earlier.

## 6 CONCLUSION

We considered the problem of sampling from general discrete probabilistic models, and presented the  $M^3$  sampler that proposes global moves using a mixture of log-modular distributions. We theoretically analyzed the effect of combining our sampler with the Gibbs sampler on a class of Ising models, and proved an exponential improvement in mixing time. We also demonstrated notable improvements when combining the two samplers on three models of practical interest. We believe that our work represents a step towards moving beyond local samplers, and incorporating ideas from optimization, such as semigradients, into probabilistic inference.

### Acknowledgements

This work was partially supported by ERC Starting Grant 307036, NSF CAREER award 1553284, and the Simons Institute for the Theory of Computing. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.



## References

- S. Ahn, Y. Chen, and M. Welling. Distributed and adaptive darting monte carlo through regenerations. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2013.
- N. Anari, S. O. Gharan, and A. Rezaei. Monte Carlo Markov chain algorithms for sampling strongly Rayleigh distributions and determinantal point processes. In *Conference on Learning Theory (COLT)*, 2016.
- N. Berger, C. Kenyon, E. Mossel, and Y. Peres. Glauber dynamics on trees and hyperbolic graphs. *Probability Theory and Related Fields*, 2005.
- M. Betancourt. A conceptual introduction to hamiltonian monte carlo. *arXiv*, 2017.
- Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2001.
- S. Brooks, A. Gelman, G. Jones, and X.-L. Meng. In *Handbook of Markov Chain Monte Carlo*. CRC Press, 2011.
- N. de Freitas, P. Højen-Sørensen, M. I. Jordan, and S. Russell. Variational mcmc. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2001.
- P. Diaconis and L. Saloff-Coste. Comparison techniques for random walk on finite groups. *Annals of Probability*, 1993.
- J. Ding, E. Lubetzky, and Y. Peres. Censored Glauber dynamics for the mean field Ising model. *Journal of Statistical Physics*, 2009.
- V. Dinh, A. Bilge, C. Zhang, and F. A. M. IV. Probabilistic path hamiltonian monte carlo. In *International Conference on Machine Learning (ICML)*, 2017.
- J. Djolonga and A. Krause. From MAP to marginals: Variational inference in bayesian submodular models. In *Neural Information Processing Systems*, 2014.
- J. Djolonga, S. Jegelka, S. Tschitschek, and A. Krause. Cooperative graphical models. In *Neural Information Processing Systems (NIPS)*, 2016a.
- J. Djolonga, S. Tschitschek, and A. Krause. Variational inference in mixed probabilistic submodular models. In *Neural Information Processing Systems (NIPS)*, 2016b.
- S. Fujishige. *Submodular Functions and Optimization*. Elsevier Science, 2005.
- B. Gong, W.-L. Chao, K. Grauman, and F. Sha. Diverse sequential subset selection for supervised video summarization. In *Neural Information Processing Systems (NIPS)*, 2014.
- A. Gotovos, H. S. Hassani, and A. Krause. Sampling from probabilistic submodular models. In *Neural Information Processing Systems (NIPS)*, 2015.
- C. Guestrin, A. Krause, and A. P. Singh. Near-optimal sensor placements in gaussian processes. In *International Conference on Machine Learning (ICML)*, 2005.
- T. Hazan, S. Maji, and T. Jaakkola. On sampling from the Gibbs distribution with random maximum a-posteriori perturbations. In *Neural Information Processing Systems (NIPS)*, 2013.
- R. Iyer and J. Bilmes. The submodular Bregman and Lovász-Bregman divergences with applications. In *Neural Information Processing Systems (NIPS)*, 2012.
- R. Iyer, S. Jegelka, and J. Bilmes. Fast semidifferential-based submodular function optimization. In *International Conference on Machine Learning (ICML)*, 2013.
- S. Jegelka and J. Bilmes. Submodularity beyond submodular energies: Coupling edges in graph cuts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.
- M. Jerrum and A. Sinclair. Polynomial-time approximation algorithms for the Ising model. *SIAM Journal on Computing*, 1993.
- M. Jerrum, J.-B. Son, P. Tetali, and E. Vigoda. Elementary bounds on Poincaré and log-Sobolev constants for decomposable Markov chains. *Annals of Applied Probability*, 2004.
- D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, 2009.
- A. Krause, J. Leskovec, C. Guestrin, J. Vanbriesen, and C. Faloutsos. Efficient sensor placement optimization for securing large water distribution networks. *Journal of Water Resources Planning and Management*, 2008.
- A. Kulesza and B. Taskar. Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 2012.
- D. A. Levin, M. J. Luczak, and Y. Peres. Glauber dynamics for the mean-field Ising model: cut-off, critical power law, and metastability. *Probability Theory and Related Fields*, 2008a.
- D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society, 2008b.
- C. Li, S. Jegelka, and S. Sra. Fast mixing markov chains for strongly Rayleigh measures, DPPs, and constrained sampling. In *Neural Information Processing Systems (NIPS)*, 2016.
- R. M. Neal. Mcmc using hamiltonian dynamics. *arXiv*, 2012.
- A. Nishimura, D. Dunson, and J. Lu. Discontinuous hamiltonian monte carlo for models with discrete parameters and discontinuous likelihoods. *arXiv*, 2018.
- A. Pakman and L. Paninski. Auxiliary-variable exact hamiltonian monte carlo samplers for binary distributions. In *Neural Information Processing Systems (NIPS)*, 2013.
- G. Papandreou and A. L. Yuille. Perturb-and-MAP random fields: Using discrete optimization to learn and sample from energy models. In *International Conference on Computer Vision (ICCV)*, 2011.
- P. Rebeschini and A. Karbasi. Fast mixing for discrete point processes. In *Conference on Learning Theory*, 2015.
- C. Sminchisescu and M. Welling. Generalized darting monte carlo. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2007.
- S. Tschitschek, J. Djolonga, and A. Krause. Learning probabilistic submodular diversity models via noise contrastive estimation. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2016.
- Y. Zhang, C. Sutton, A. Storkey, and Z. Ghahramani. Continuous relaxations for discrete hamiltonian monte carlo. In *Neural Information Processing Systems (NIPS)*, 2012.
- M. H. Zhu and S. Ermon. A hybrid approach for probabilistic inference using random projections. In *International Conference on Machine Learning (ICML)*, 2015.

---

# Combining Knowledge and Reasoning through Probabilistic Soft Logic for Image Puzzle Solving

---

**Somak Aditya, Yezhou Yang, Chitta Baral**

School of Computing, Informatics and Decision Systems Engineering  
Arizona State University  
{saditya1,yz.yang,chitta}@asu.edu

**Yiannis Aloimonos**

UMIACS, Computer Science  
University of Maryland, College Park  
yiannis@cs.umd.edu

## Abstract

The uncertainty associated with human perception is often reduced by one’s extensive prior experience and knowledge. Current datasets and systems do not emphasize the necessity and benefit of using such knowledge. In this work, we propose the task of solving a genre of image-puzzles (“image riddles”) that require both capabilities involving visual detection (including object, activity recognition) and, knowledge-based or commonsense reasoning. Each puzzle involves a set of images and the question “what word connects these images?”. We compile a dataset of over 3k riddles where each riddle consists of 4 images and a groundtruth answer. The annotations are validated using crowd-sourced evaluation. We also define an automatic evaluation metric to track future progress. Our task bears similarity with the commonly known IQ tasks such as analogy solving, sequence filling that are often used to test intelligence. We develop a Probabilistic Reasoning-based approach that utilizes commonsense knowledge about words and phrases to answer these riddles with a reasonable accuracy. Our approach achieves some promising results for these riddles and provides a strong baseline for future attempts. We make the entire dataset and related materials publicly available to the community ([bit.ly/22f9Ala](http://bit.ly/22f9Ala)).

## 1 INTRODUCTION

Human visual perception is greatly aided by the human’s knowledge and reasoning (with that knowledge) about the domain (what it is looking at) and purpose (what it is looking for and why) (Lake et al., 2016). This knowledge



Figure 1: An Image Riddle Example. Question: “What word connects these images?” .

greatly helps in overcoming the uncertainty often associated with perception. Most work in computer vision do not take into account the vast body of knowledge that humans use in their visual perception. Several researchers<sup>1</sup> have recently pointed out the necessity and the potential benefit of using such knowledge, as well as the lack of it in current systems. This absence is also reflected in the various popular data sets and benchmarks. Our goal in this paper is to present a new task, a corresponding new data set, and our approach to them that highlights the importance of using knowledge and reasoning in visual perception. This necessitates considering issues such as what kind of knowledge is needed, where and how to get them, and what kind of reasoning mechanism to adopt for such knowledge.

The new task we propose in this paper is referred to as Image Riddles which requires deep conceptual understanding of images. In this task a set of images are provided and one needs to find a common concept that is invoked by all the images. Often the common concept is not something that even a human can observe in the first glance; but after some thought about the images, he/she can come up with it. Hence the word “riddle” in the phrase “image riddles”. Figure 1 shows an example

---

<sup>1</sup>Lake et al. (2016) quotes a reviewer: “Human learners - unlike DQN and many other deep Learning systems - approach new problems armed with extensive prior experience.”. The authors also ask “How do we bring to bear rich prior knowledge to learn new tasks and solve new problems?”. In “A Path to AI”, Prof. Yann Lecun recognizes the absence of common-sense to be an obstacle to AI.

of an image riddle. The images individually connect to multiple concepts such as: *outdoors, nature, trees, road, forest, rainfall, waterfall, statue, rope, mosque* etc. On further thought, the common concept that emerges for this example is “fall”. Here, the first image represents the fall season (*concept*). There is a “waterfall” (*region*) in the second image. In the third image, it shows “rain-fall” (*concept*) and the fourth image depicts that a statue is “fall”ing (*action/event*). The word “fall” is invoked by all the images as it shows logical connections to objects, regions, actions or concepts specific to each image. Additionally, the answer also connects the most salient aspects of the images. Other possible answers like “nature” or “outdoors” do not demonstrate such properties. They are too abstract. In essence, answering Image Riddles is a challenging task that not only tests an intelligent agent’s ability to detect visual concepts, but also tests its (ontological) knowledge and its ability to think and reason.

Image Riddles can also be thought of as a visual counterpart to IQ tests such as sequence filling ( $x_1, x_2, x_3, ?$ ) and analogy solving ( $x_1 : y_1 :: x_2 : ?$ )<sup>2</sup>, where one needs to find commonalities between items. It is worth to note that this task is different from traditional Visual Question-Answering (VQA), as in VQA the queries provide some clues regarding what to look for in the image. Most Image Riddles require both superior detection and reasoning capabilities, whereas a large percentage of questions from the VQA dataset tests mainly the system’s detection capabilities. Moreover, answering Image Riddles differs from both VQA and Captioning in that it requires analysis of multiple seemingly different images.

Hence, this task of answering Image Riddles is simple to explain; shares similarities with well-known and pre-defined types of IQ questions and it requires a combination of vision and reasoning capabilities. In this paper, we introduce a novel benchmark for Image Riddles and put forward a promising approach to tackle it. In our approach, we first use the state-of-the-art Image Classification techniques (Sood (2015) and He et al. (2016)) to get the top identified class-labels from each image. Given these detections, we use ontological and commonsense relations of these words to infer a set of most probable concepts. We adopt ConceptNet 5 (Liu and Singh, 2004) as the source of commonsense and background knowledge that encodes the relations between words and short phrases through a structured graph. Note, the possible range of candidates are **the entire vocabulary** of ConceptNet 5 (roughly 0.2 million), which is fundamentally different from supervised end-to-end models. For representation and reasoning with this huge probabilis-

<sup>2</sup>Examples are: word analogy tasks (male : female :: king : ?); numeric sequence filling tasks: (1, 2, 3, 5, ?).

tic knowledge one needs a powerful reasoning engine. Here, we adopt the Probabilistic Soft Logic (PSL) (Kim-mig et al., 2012; Bach et al., 2013) framework. Given the inferred concepts of each image, we adopt a second stage inference to output the final answer.

Our **contributions** are threefold: i) we introduce the 3K Image Riddles Dataset; ii) we present a probabilistic reasoning approach to solve the riddles with reasonable accuracy; iii) our reasoning module inputs detected words (a closed set of class-labels) and *logically* infers all relevant concepts (belonging to a much larger vocabulary), using background knowledge about words.

## 2 RELATED WORK

The problem of Image Riddles has some similarities to the genre of topic modeling (Blei, 2012) and Zero-shot Learning (Larochelle et al., 2008). However, this dataset imposes a few unique challenges: i) the possible set of target labels is the entire natural language vocabulary; ii) each image, when grouped with different sets of images can map to a different label; iii) almost all the target labels in the dataset are unique (3k examples with 3k class-labels). These challenges make it hard to simply adopt topic model-based or Zero-shot learning-based approaches.

Our work is also related to the field of **Visual Question Answering** (VQA). Very recently, researchers spent a significant amount of efforts on both creating datasets and proposing new models (Antol et al., 2015; Malinowski et al., 2015; Gao et al., 2015; Ma et al., 2016) for VQA. Interestingly both (Antol et al., 2015; Goyal et al., 2017) and Gao et al. (2015) adapted MS-COCO (Lin et al., 2014) images and created an open domain dataset with human generated questions and answers. Both Malinowski et al. (2015) and Gao et al. (2015) use recurrent networks to encode the sentence and output the answer.

Even though some questions from Antol et al. (2015) and Gao et al. (2015) are very challenging, and actually require logical reasoning in order to answer correctly, popular approaches still aim to learn the direct signal-to-signal mapping from image and question to its answer, given a large enough annotated data. The necessity of common-sense reasoning is often neglected. Here we introduce the new Image Riddle problem to serve as the testbed for vision and reasoning research.

## 3 KNOWLEDGE AND REASONING MECHANISM

In this Section, we briefly introduce the kind of knowledge that is useful for solving Image Riddles and the

kind of reasoning needed. The primary types of knowledge needed are the distributional and relational similarities between words and concepts. We obtain them from analyzing the ConceptNet knowledge base and using Word2Vec. Both the knowledge sources are considered because ConceptNet embodies commonsense knowledge and Word2vec encodes word-meanings.

**ConceptNet** (Speer and Havasi, 2012), is a multilingual Knowledge Graph, that encodes commonsense knowledge about the world and is built primarily to assist systems that attempts to understand natural language text. The knowledge in ConceptNet is semi-curated. The nodes (called concepts) in the graph are words or short phrases written in natural language. The nodes are connected by edges which are labeled with meaningful relations. For example: (*reptile, IsA, animal*), (*reptile, HasProperty, cold blood*) are some edges. Each edge has an associated confidence score. Compared to other knowledge-bases such as WordNet, YAGO, NELL (Suchanek et al., 2007; Mitchell et al., 2015), ConceptNet has a more extensive coverage of English language words and phrases. These properties make this Knowledge Graph a perfect source for the required probabilistic commonsense knowledge. We use different methods on ConceptNet, elaborated in the next section, to define similarity between different types of words and concepts.

**Word2vec** uses the theory of distributional semantics to capture word meanings and produce word embeddings (vectors). The pre-trained word-embeddings have been successfully used in numerous Natural Language Processing applications and the induced vector-space is known to capture the graded similarities between words with reasonable accuracy (Mikolov et al., 2013). Throughout the paper, for word2vec-based similarities, we use the 3 Million word-vectors trained on Google-News corpus (Mikolov et al., 2013).

The similarity between words  $w_i$  and  $w_j$  with a similarity score  $w_{ij}$  is expressed as propositional formulas of the form:  $w_i \Rightarrow w_j : w_{ij}$ . (The exact formulas, and when they are bidirectional and when they are not are elaborated in the next section.) To reason with such knowledge we explored various reasoning formalisms and found Probabilistic Soft Logic (PSL) (Kimmig et al., 2012; Bach et al., 2013) to be the most suitable, as it can not only handle relational structure, inconsistencies and uncertainty, thus allowing one to express rich probabilistic graphical models (such as Hinge-loss Markov random fields), but it also seems to scale up better than its alternatives such as Markov Logic Networks (Richardson and Domingos, 2006). In this work, we also use different weights for different groundings of the same rule. Even though some work has been done along this line

for MLNs (Mittal et al., 2015), implementing those ideas in MLNs to define weights using word2vec and ConceptNet is not straightforward. Learning grounding-specific weights is also difficult as that will require augmentation of MLN syntax and learning.

### 3.1 PROBABILISTIC SOFT LOGIC (PSL)

Probabilistic soft logic (PSL) differs from most other probabilistic formalisms in that its ground atoms have continuous truth values in the interval  $[0,1]$ , instead of having binary truth values. The syntactic structure of rules and the characterization of the logical operations have been chosen judiciously so that the space of interpretations with nonzero density forms a convex polytope. This makes inference in PSL a convex optimization problem in continuous space, which in turn allows efficient inference. We now give a brief overview of PSL.

A PSL model is defined using a set of weighted if-then rules in first-order logic. Let  $\mathcal{C} = (C_1, \dots, C_m)$  be such a collection where each  $C_j$  is a disjunction of literals, where each literal is a variable  $y_i$  or its negation  $\neg y_i$ , where  $y_i \in \mathbf{y}$ . Let  $I_j^+$  (resp.  $I_j^-$ ) be the set of indices of the variables that are not negated (resp. negated) in  $C_j$ . Each  $C_j$  is:

$$w_j : \bigwedge_{i \in I_j^-} \neg y_i \rightarrow \bigvee_{i \in I_j^+} y_i, \quad (1)$$

or equivalently,  $w_j : \bigvee_{i \in I_j^-} (\neg y_i) \vee \bigvee_{i \in I_j^+} y_i$ . Each rule  $C_j$  is associated with a non-negative weight  $w_j$ . PSL relaxes the boolean truth values of each ground atom  $a$  (constant term or predicate with all variables replaced by constants) to the interval  $[0, 1]$ , denoted as  $I(a)$ . To compute soft truth values, Lukasiewicz’s relaxation (Klir and Yuan, 1995) of conjunctions ( $\wedge$ ), disjunctions ( $\vee$ ) and negations ( $\neg$ ) is used:

$$\begin{aligned} I(l_1 \wedge l_2) &= \max\{0, I(l_1) + I(l_2) - 1\} \\ I(l_1 \vee l_2) &= \min\{1, I(l_1) + I(l_2)\} \\ I(\neg l_1) &= 1 - I(l_1). \end{aligned} \quad (2)$$

In PSL, the ground atoms are considered as random variables and the distribution is modeled using **Hinge-Loss Markov Random Field**, which is defined as follows: Let  $\mathbf{y}$  and  $\mathbf{x}$  be two vectors of  $n$  and  $n'$  random variables respectively, over the domain  $D = [0, 1]^{n+n'}$ . The feasible set  $\tilde{D}$  is a subset of  $D$ , which satisfies a set of inequality constraints over the random variables. A *Hinge-Loss Markov Random Field*  $\mathbb{P}$  is a probability density, defined as: if  $(\mathbf{y}, \mathbf{x}) \notin \tilde{D}$ , then  $\mathbb{P}(\mathbf{y}|\mathbf{x}) = 0$ ; if  $(\mathbf{y}, \mathbf{x}) \in \tilde{D}$ , then:

$$\mathbb{P}(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{w}, \mathbf{x})} \exp(-f_{\mathbf{w}}(\mathbf{y}, \mathbf{x})), \quad (3)$$

where  $Z(\mathbf{w}, \mathbf{x}) = \int_{\mathbf{y} | (\mathbf{y}, \mathbf{x}) \in \tilde{D}} \exp(-f_{\mathbf{w}}(\mathbf{y}, \mathbf{x})) d\mathbf{y}$ .

Here, the hinge-loss energy function  $f_{\mathbf{w}}$  is defined as:  $f_{\mathbf{w}}(\mathbf{y}, \mathbf{x}) = \sum_{j=1}^m w_j (\max\{l_j(\mathbf{y}, \mathbf{x}), 0\})^{p_j}$ , where  $w_j$ ’s

are non-negative free parameters and  $l_j(\mathbf{y}, \mathbf{x})$  are linear constraints over  $\mathbf{y}, \mathbf{x}$  and  $p_j \in \{1, 2\}$ . As we are interested in finding the maximum probable solution given the evidence, the inference objective of HL-MRF becomes:

$$\mathbb{P}(\mathbf{y}|\mathbf{x}) \equiv \arg \min_{\mathbf{y} \in [0,1]^n} \sum_{j=1}^m w_j (\max\{l_j(\mathbf{y}, \mathbf{x}), 0\})^{p_j}. \quad (4)$$

In PSL, each logical rule  $C_j$  in the database  $\mathcal{C}$  is used to define  $l_j(\mathbf{y}, \mathbf{x})$ , i.e. the linear constraints over  $(\mathbf{y}, \mathbf{x})$ . Given a set of weighted logical formulas, PSL builds a graphical model defining a probability distribution over the continuous value space of the random variables in the model.

More precisely,  $l_j(\cdot)$  is defined in terms of “distance to satisfaction”. For each rule  $C_j \in \mathcal{C}$  this “distance to satisfaction” is measured using the term  $w_j \times \max\{1 - \sum_{i \in I_j^+} y_i - \sum_{i \in I_j^-} (1 - y_i), 0\}$ . This encodes the penalty if a rule is not satisfied. Then, the right hand side of the Eq. 4 becomes:

$$\arg \min_{\mathbf{y} \in [0,1]^n} \sum_{C_j \in \mathcal{C}} w_j \max\{1 - \sum_{i \in I_j^+} y_i - \sum_{i \in I_j^-} (1 - y_i), 0\}, \quad (5)$$

which is used to estimate  $\mathbb{P}(\mathbf{y}|\mathbf{x})$  efficiently.

## 4 APPROACH

Given a set of images  $(\{\mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3, \mathcal{I}_4\})$ , our objective is to determine a set of ranked words ( $T$ ) based on how well they semantically connect the images. In this work, we present an approach that uses the previously introduced Probabilistic Reasoning framework on top of a probabilistic Knowledge Base (ConceptNet). It also uses additional semantic knowledge from Word2vec. Using these knowledge sources, we predict the answers to the riddles. Although our approach consists of multiple resources and stages, it can be easily modularized, pipelined and reproduced. It is also worth to mention that the PSL engine is a general tool. It could be used for further research along the conjunction of vision, language and reasoning.

### 4.1 OUTLINE OF OUR FRAMEWORK

As outlined in algorithm 1, for each image  $\mathcal{I}_k$  (here,  $k \in \{1, \dots, 4\}$ ), we follow three steps to infer related words and phrases: i) Image Classification: we get top class labels and the confidence from Image Classifier ( $\mathcal{S}_k, \tilde{P}(\mathcal{S}_k|\mathcal{I}_k)$ ), ii) Rank and Retrieve: using these labels and confidence scores, we rank and retrieve top related words ( $T_k$ ) from ConceptNet ( $\mathcal{K}_{cnet}$ ), iii) Probabilistic Reasoning and Inference (Stage I): using the labels ( $\mathcal{S}_k$ ) and the top related words ( $T_k$ ), we design an inference model to logically infer final set of words ( $\hat{T}_k$ )

### Algorithm 1: Solving Image Riddles

```

1: procedure UNRIDDLER( $\mathcal{I} = \{\mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3, \mathcal{I}_4\}, \mathcal{K}_{cnet}$ )
2:   for  $\mathcal{I}_k \in \mathcal{I}$  do
3:      $\tilde{P}(\mathcal{S}_k|\mathcal{I}_k) = \text{getClassLabelsNeuralNetwork}(\mathcal{I}_k)$ .
4:     for  $s \in \mathcal{S}_k$  do
5:        $T_s, W_m(s, T_s) = \text{retrieveTargets}(s, \mathcal{K}_{cnet})$ ;
6:        $W_m(s, t_j) = \text{sim}(s, t_j) \forall t_j \in T_s$ .
7:     end for
8:      $T_k = \text{rankTopTargets}(\tilde{P}(\mathcal{S}_k|\mathcal{I}_k), T_{\mathcal{S}_k}, W_m)$ ;
9:      $I(\hat{T}_k) = \text{inferConfidenceStageI}(T_k, \tilde{P}(\mathcal{S}_k|\mathcal{I}_k))$ .
10:    end for
11:     $I(T) = \text{inferConfidenceStageII}([\hat{T}_k]_{k=1}^4, [\tilde{P}(\mathcal{S}_k|\mathcal{I}_k)]_{k=1}^4)$ .
12: end procedure

```

for each image. Lastly, we use another probabilistic reasoning model (Stage II) on the combined set of inferred words (*targets*) from all images in a riddle. This model assigns the final confidence scores on the combined set of targets ( $T$ ). We depict the pipeline with an example in Figure 2.

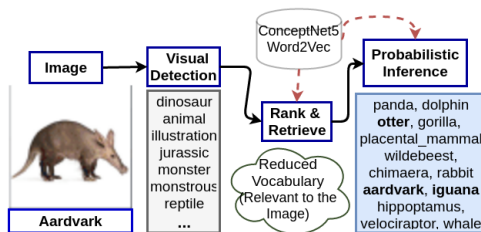


Figure 2: An overview of the framework followed for each Image; demonstrated using an example image of an *aardvark* (resembles animals such as tapir, ant-eater). As shown, the uncertainty in detecting concepts is reduced after considering additional knowledge. We run a similar pipeline for each image and then infer final results using a final Probabilistic Inference Stage (Stage II).

### 4.2 IMAGE CLASSIFICATION

Neural Networks trained on ample source of images and numerous image classes has been very effective. Studies have found that convolutional neural networks (CNN) can produce near human level image classification accuracy (Krizhevsky et al., 2012), and related work has been used in various visual recognition tasks such as scene labeling (Farabet et al., 2013) and object recognition (Girshick et al., 2014). To exploit these advances, we use the state-of-the-art class detections provided by the Clarifai API (Sood, 2015) and the Deep Residual Network Architecture by (He et al., 2016) (using the trained ResNet-200 model). For each image ( $\mathcal{I}_k$ ) we use top 20 detections ( $\mathcal{S}_k$ ) (*seeds*). Figure 2 provides an example. Each detection is accompanied with the classifier’s confidence score ( $\tilde{P}(\mathcal{S}_k|\mathcal{I}_k)$ ).

### 4.3 RETRIEVE AND RANK RELATED WORDS

Our goal is to logically infer words or phrases that represent (higher or lower-level) concepts that can best explain the co-existence of the detected *seeds* in a scene. For examples, for “hand” and “care”, implied words could be “massage”, “ill”, “ache” etc. For “transportation” and “sit”, implied words/phrases could be “sit in bus” and “sit in plane”. The reader might be inclined to infer other concepts. However, to “infer” is to derive “logical” conclusions. Hence, we prefer the concepts which shares strong explainable connections (i.e. relational similarity) with the *seeds*.

A logical choice would be traversing a knowledge-graph like ConceptNet and find the common reachable nodes from these *seeds*. As this is computationally infeasible, we use the association-space matrix representation of ConceptNet, where the words are represented as vectors. The similarity between two words approximately embodies the strength of the connection over all paths connecting the two words in the graph. We get the top similar words for each *seed*, approximating the reachable nodes.

#### 4.3.1 Retrieve Related Words For a Seed

We observe that, for objects, the ConceptNet-similarity gives a poor result (See Table 1). So, we define a metric called **visual similarity**. Let us call the similar words as *targets*. In this metric, we represent the seed and the target as vectors. To define the dimensions, for each *seed*, we use the relations (HasA, HasProperty, PartOf and MemberOf). We query ConceptNet to get the related words ( $W1, W2, W3, \dots$ ) under such relations for the seed-word and its superclasses. Each of these relation-word pairs (i.e.  $HasA-W1, HasA-W2, PartOf-W3, \dots$ ) becomes a separate dimension. The values for the seed-vector are the weights assigned to the assertions. For each *target*, we query ConceptNet and populate the target-vector using the edge-weights for the dimensions defined by the seed-vector.

To get the top words using visual similarity, we use the cosine similarity of the seed-vector and the target-vector to re-rank the top 10000 retrieved similar target-words. For abstract *seeds*, we do not get any such relations and thus use the ConceptNet similarity directly.

Table 1 shows the top similar words using ConceptNet, word2vec and visual-similarity for the word “men”.

**Formulation:** For each seed ( $s$ ), we get the top words ( $T_s$ ) from ConceptNet using the visual similarity metric and the similarity vector  $W_m(s, T_s)$ . Together for an image, these constitute  $T_{S_k}$  and the matrix  $W_m$ , where

ConceptNet	Visual Similarity	Word2vec
man, merby, misandrous, philandry, male_human, dirty_pig, mantyhose, date_woman, guyliner, manslut	priest, uncle, guy, geezer, bloke, pope, bouncer, ecologist, cupid, fella	women, men, males, mens, boys, man, female, teenagers, girls, ladies

Table 1: Top 10 similar Words for “Men”. The ranked list based on visual-similarity ranks *boy, chap, husband, godfather, male person, male* in the ranks 16 to 22. See appendix for more.

$$W_m(s_i, t_j) = sim_{vis}(s_i, t_j) \forall s_i \in S_k, t_j \in T_{S_k}.$$

A large percentage of the error from Image Classifiers are due to visually similar objects or objects from the same category (Hoiem et al., 2012). In such cases, we use this visual similarity metric to predict the possible visually similar objects and then use an inference model to infer the actual object.

#### 4.3.2 Rank Targets

We use the classifier confidence scores  $\tilde{P}(S_k | I_k)$  as an approximate vector representation for an image, in which the *seeds* are the dimensions. The columns of  $W_m$  provides vector representations for the target words ( $t \in T_{S_k}$ ) in the space. We calculate cosine similarities for each target with such a image-vector and then re-rank the targets. We denote the top  $\theta_{\#t}$  targets as  $T_k$  (see Table. 2).

### 4.4 PROBABILISTIC REASONING AND INFERENCE

#### 4.4.1 PSL Inference Stage I

Given a set of candidate *targets*  $T_k$  and a set of weighted *seeds* ( $S_k, \tilde{P}(S_k | I_k)$ ), we build an inference model to infer a set of most probable *targets* ( $\hat{T}_k$ ). We model the joint distribution using PSL as this formalism adopts Markov Random Field which obeys the properties of Gibbs Distribution. In addition, a PSL model is declared using rules. Given the final answer, the set of satisfied rules show the logical connections between the detected words and the final answer. The PSL model can be best explained as an Undirected Graphical Model involving *seeds* (observed) and *targets* (unobserved). We define the seed-target and target-target potentials using PSL rules. We connect each seed to each target and the potential depends on their similarity and the target’s popularity bias. We connect each target to  $\theta_{t-t}$  (1 or 2) maximally similar targets. The potential depends on their similarity.

**Formulation:** Using PSL, we add two sets of rules: i) to define seed-target potentials, we add rules of the form  $wt_{ij} : s_{ik} \rightarrow t_{jk}$  for each word  $s_{ik} \in S_k$  and target  $t_{jk} \in T_k$ ; ii) to define target-target potentials, for each target  $t_{jk}$ , we take the most similar  $\theta_{t-t}$  targets ( $T_j^{max}$ ).

For each target  $t_{jk}$  and each  $t_{mk} \in T_j^{max}$ , we add two rules  $wt_{jm} : t_{jk} \rightarrow t_{mk}$  and  $wt_{jm} : t_{mk} \rightarrow t_{jk}$ . Next, we describe the choices in detail.

i) From the perspective of optimization, the rule  $wt_{ij} : s_{ik} \rightarrow t_{jk}$  adds the term  $wt_{ij} * \max\{I(s_{ik}) - I(t_{jk}), 0\}$  to the objective. This means that if confidence score of the target  $t_{jk}$  is not greater than  $I(s_{ik})$  (i.e.  $\tilde{P}(\mathbf{S}_k | \mathcal{I}_k)$ ), then the rule is not satisfied and we penalize the model by  $wt_{ij}$  times the difference between the confidence scores. We add the above rule for seeds and targets for which the combined similarity ( $wt_{ij}$ ) exceeds certain threshold  $\theta_{sim,psl1}$ . We encode the commonsense knowledge of words and phrases obtained from different knowledge sources into the weights of these rules  $wt_{ij}$ . It is also important that the inference model is not biased towards more popular targets (i.e. abstract words or words too commonly used/detected in corpus). We compute eigenvector centrality score ( $\mathbb{C}(\cdot)$ ) for each word in the context of ConceptNet. Higher  $\mathbb{C}(\cdot)$  indicates higher connectivity of a word in the graph. This yields a higher similarity score to many words and might give an unfair bias to this *target* in the inference model. Hence, the higher the  $\mathbb{C}(\cdot)$ , the word provides less specific information for an image. Hence, the weight becomes

$$wt_{ij} = \theta_{\alpha_1} * sim_{cn}(s_{ik}, t_{jk}) + \theta_{\alpha_2} * sim_{w2v}(s_{ik}, t_{jk}) + 1/\mathbb{C}(t_{jk}), \quad (6)$$

where  $sim_{cn}(\cdot, \cdot)$  is the normalized ConceptNet-based similarity.  $sim_{w2v}(\cdot, \cdot)$  is the normalized word2vec similarity of two words and  $\mathbb{C}(\cdot)$  is the eigenvector-centrality score of the argument in the ConceptNet matrix.

ii) To model dependencies among the targets, we observe that if two concepts  $t_1$  and  $t_2$  are very similar in meaning, then a system that infer  $t_1$  should infer  $t_2$  too, given the same set of observed words. Therefore, the two rules  $wt_{jm} : t_{jk} \rightarrow t_{mk}$  and  $wt_{jm} : t_{mk} \rightarrow t_{jk}$  are designed to force the confidence values of  $t_{jk}$  and  $t_{mk}$  to be as close to each other as possible.  $wt_{jm}$  is the same as Equation 6 without the penalty for popularity.

Using Equation 5, the PSL inference objective becomes:

$$\begin{aligned} \arg \min_{I(\mathbf{T}_k) \in [0,1]^{|\mathcal{T}_k|}} & \sum_{s_{ik} \in \mathbf{S}_k} \sum_{t_{jk} \in \mathcal{T}_k} wt_{ij} \max\{I(s_{ik}) - I(t_{jk}), 0\} + \\ & \sum_{t_{jk} \in \mathcal{T}_k} \sum_{t_{mk} \in T_j^{max}} wt_{jm} \left\{ \max\{I(t_{mk}) - I(t_{jk}), 0\} + \right. \\ & \left. \max\{I(t_{jk}) - I(t_{mk}), 0\} \right\}. \end{aligned}$$

To let the targets compete against each other, we add one more constraint on the sum of the confidence scores of the targets i.e.  $\sum_{j:t_{jk} \in \mathcal{T}_k} I(t_{jk}) \leq \theta_{sum1}$ . Here  $\theta_{sum1} \in \{1, 2\}$  and  $I(t_{jk}) \in [0, 1]$ . The above optimizer provides us  $\mathbb{P}(\mathbf{T}_k | \mathbf{S}_k)$  and thus the top set of targets  $[\hat{\mathbf{T}}_k]_{k=1}^4$ .

## 4.4.2 PSL Inference Stage II

To learn the most probable common targets jointly, we consider the *targets* and the *seeds* from all images together. Assume that the *seeds* and the *targets* are nodes in a knowledge-graph. Then, the most appropriate target-nodes should observe similar properties as an appropriate answer to the riddle: i) a target-node should be connected to the high-weight seeds in an image i.e. should relate to the important aspects of the image; and ii) a target-node should be connected to seeds from all images.

**Formulation:** Here, we use the rules  $wt_{ij} : s_{ik} \rightarrow t_{jk}$  for each word  $s_{ik} \in \mathbf{S}_k$  and target  $t_{jk} \in \hat{\mathbf{T}}_k$  for all  $k \in \{1, 2, \dots, 4\}$ . To let the set of targets compete against each other, we add the constraint  $\sum_{k=1}^4 \sum_{j:t_{jk} \in \hat{\mathbf{T}}_k} I(t_{jk}) \leq \theta_{s2}$ . Here  $\theta_{s2} = 1$  and  $I(t_{jk}) \in [0, 1]$ . The second inference stage provides us  $\mathbb{P}([\hat{\mathbf{T}}_k]_{k=1}^4 | \mathbf{S}_1, \mathbf{S}_2, \mathbf{S}_3, \mathbf{S}_4)$  and thus the top targets that constitutes the final answers. To minimize the penalty for each rule, the optimal solution maximizes the confidence score of  $t_{jk}$ . To minimize the overall penalty, it should maximize the confidence scores of these targets which satisfy most of the rules. As the summation of confidence scores is bounded, only a few top inferred targets should have non-zero confidence.

## 5 EXPERIMENTS AND RESULTS

### 5.1 DATASET VALIDATION AND ANALYSIS

We have collected a set of 3333 riddles from the Internet (puzzle websites). Each riddle has 4 images and a groundtruth answer associated with it. To make it more challenging to computer systems, we include both photographic and non-photographic images in the dataset.

To verify the groundtruth answers, we define the metrics: i) ‘‘correctness’’ - how correct and appropriate the answers are, and ii) ‘‘difficulty’’ - how difficult are the riddles. We conduct an Amazon Mechanical Turk (AMT)-based evaluation for dataset validation. We ask them to rate the correctness from 1-6<sup>3</sup>. The ‘‘difficulty’’ is rated from 1-7<sup>4</sup>. We provide the Turkers with examples to calibrate our evaluation. According to the Turkers, the mean correctness rating is 4.4 (with Standard Deviation

<sup>3</sup>1: Completely gibberish, incorrect, 2: relates to one image, 3 and 4: connects two and three images respectively, 5: connects all 4 images, but could be a better answer, 6: connects all images and an appropriate answer.

<sup>4</sup>These gradings are adopted from VQA AMT instructions (Antol et al., 2015). 1: A toddler can solve it (ages:3-4), 2: A younger child can solve it (ages:5-8), 3: A older child can solve it (ages:9-12), 4: A teenager can solve it (ages:13-17), 5: An adult can solve it (ages:18+), 6: Only a Linguist (one who has above-average knowledge about English words and the language in general) can solve it, 7: No-one can solve it.

1.5). The “difficulty” ratings show the following distribution: toddler (0.27%), younger child (8.96%), older child (30.3%), teenager (36.7%), adult (19%), linguist (3.6%), no-one (0.64%). In short, the average age to answer the riddles is closer to **13-17yrs**. Also, few of these (4.2%) riddles seem to be incredibly hard. Interestingly, the average age perceived reported for the recently proposed VQA dataset (Antol et al., 2015) is **8.92 yrs**. Although, this experiment measures “the turkers’ perception of the required age”, one can conclude with statistical significance that the riddles are comparably harder.

## 5.2 SYSTEMS EVALUATION

The presented approach suggests the following hypotheses that requires empirical tests: I) the proposed approach (and their variants) attain reasonable accuracy in solving the riddles; II) the individual stages of the framework improves the final inference accuracy of the answers. In addition, we also experiment to observe the effect of using commercial classification methods like Clarifai against a published state-of-the-art Image Classification method.

### 5.2.1 Systems

We propose several variations of the proposed approach and compare them with simple vision-only baselines. We introduce an additional Bias-Correction stage after the Image Classification, which aims to re-weight the detected seeds using additional information from other images. The variations then are created to test the effects of varying the Bias-Correction stage and the effects of the individual stages of the framework on the final accuracy (hypothesis II). We also vary the initial Image Classification Methods (Clarifai, Deep Residual Network).

**Bias-Correction:** We experimented with two variations: i) greedy bias-correction and ii) no bias-correction. We follow the intuition that the re-weighting of the seeds of one image can be influenced by the others<sup>5</sup>. To this end, we develop the “GreedyUnRiddler” (**GUR**) approach. In this approach, we consider all of the images together to dictate the new weight of each seed. Take image  $\mathcal{I}_k$  for example. To re-weight seeds in  $\mathcal{S}_k$ , we calculate the weights using the following equation:  $\tilde{W}(s_k) = \frac{\sum_{j \in 1, \dots, 4} \text{sim}_{\cosine}(V_{s_k, j}, V_j)}{4.0}$ .  $V_j$  is vector of the weights assigned  $\tilde{P}(\mathcal{S}_j | \mathcal{I}_j)$  i.e. confidence scores of each seed in the image. Each element of  $V_{s_k, j}[i]$  is the ConceptNet-similarity score between the seed  $s_k$  and  $s_{i, j}$  i.e. the  $i^{th}$  seed of the  $j^{th}$  image. The re-weighted seeds ( $\mathcal{S}_k, \tilde{W}(\mathcal{S}_k)$ ) of an image are then passed through

<sup>5</sup>A person often skims through all the images at one go and will try to come up with the aspects that needs more attention.

the rest of the pipeline to infer the final answers.

In the original pipeline (“UnRiddler”, in short **UR**), we just normalize the weights of the seeds and pass on to the next stage. We experiment with another variation (called BiasedUnRiddler or **BUR**), the results of which are included in appendix, as **GUR** achieves the best results.

**Effect of Stages:** We observe the accuracy after each stage in the pipeline (**VB**: Up to Bias Correction, **RR**: Up to Rank and Retrieve stage, **All**: The entire Pipeline). For **VB**, we use the normalized weighted seeds, get the weighted centroid vector over the word2vec embeddings of the seeds for each image. Then we obtain the mean vector over these centroids. The top similar words from the word2vec vocabulary to this mean vector, constitutes the final answers. For **RR**, we get the mean vector over the top predicted targets for all images. Again, the most similar words from the word2vec vocabulary constitutes the answers.

**Baseline (VQA+VB+UR):** For the sake of completion, we experiment with a pre-trained Visual Question Answering system (from Lu et al. (2016)). For each image, we take top 20 answers for the question “What is the image about”, and, then we follow the above procedure (**VB+UR**) to calculate the mean. We get the closest word using the mean vector, from the Word2vec vocabulary. We observe that, the detected words are primarily top frequent answers and do not contain any specific information. Therefore, subsequent stages hardly improve the results. We provide one detailed example in appendix.

**Baseline (Clarifai+VB+UR and ResNet+VB+UR):** We create a strong baseline by directly going from seeds to target using word2vec-based similarities. We use the class-labels and the confidence scores predicted using the state-of-the-art classifiers. For each image, we calculate the weighted centroid of the word2vec embeddings of these labels and the mean of these centroids for the 4 images. For the automatic evaluation we use top  $K$  (10) similar words and for human evaluation, we use the most similar word to this vector, from the word2vec vocabulary. The Baseline performances are listed in Table 3.

**Human Baseline:** In an independent AMT study, we ask the turkers to answer each riddle without any hint towards the answer. We ask them to input maximum 5 words (comma-separated) that can connect all four of the images. In cases, where the riddles are difficult we instruct them to find words that connect at least three images. These answers constitute our human baseline.

### 5.2.2 Experiment I: Automatic Evaluation

We evaluate the performance of the proposed approach on the Image Riddles dataset using both automatic and



Amazon Mechanical Turker (AMT)-based evaluations. An answer to a riddle may have several semantically similar answers. Hence, as evaluation metrics, we use both word2vec and WordNet-based similarity measures. For each riddle, we calculate the maximum similarity between the groundtruth with the top 10 detections, and report the average of such maximum similarities in percentage form:  $S = \frac{1}{n} \sum_{i=1}^n \max_{1 \leq l \leq 10} sim(GT_i, T_l)$ . To calculate phrase similarities, i) we use `n_similarity` method of the `gensim.models.word2vec` package; or, ii) average of WordNet-based word pair similarities that is calculated as a product of `length` (of the shortest path between sysnsets of the words), and `depth` (the depth of the subsumer in the hierarchical semantic net) (Li et al., 2006)<sup>6</sup>.

Number of Targets: $\theta_{\#t}$ (2500), ConceptNet-similarity Weight: $\theta_{\alpha_1}$ (1), word2vec-similarity weight: $\theta_{\alpha_2}$ (4), Number of maximum similar Targets: $\theta_{t-l}$ (1) Seed-target similarity Threshold: $\theta_{sim\_psl1}$ (0.8), Sum of confidence scores in Stage I: $\theta_{sum1}$ (2)
--

Table 2: A List of parameters  $\theta$  used in the approach

			3.3k		2.8k	
			W2V	WN	W2V	WN
Human	-	-	74.6	68.9	74.56	67.8
VQA	VB	UR †	59.6	15.7	59.7	15.6
		GUR	62.59	17.7	62.5	17.7
Clarifai	VB	UR †	65	26.2	65.3	26.4
		GUR	65.3	26.2	65.36	26.2
	RR	UR	65.9	34.9	65.7	34.8
		GUR	65.9	36.6	65.73	36.4
	All	UR	68.5	<b>40.3</b>	68.57	<b>40.4*</b>
		GUR	<b>68.8*</b>	<b>40.3</b>	<b>68.7</b>	<b>40.4*</b>
Resnet	VB	UR †	68.3	35	68	33.5
		GUR	66.8	33.1	66.4	32.6
	RR	UR	66.7	38.5	66.7	38.2
		GUR	66.3	38.1	66.2	37.6
	All	UR	68.53	39.9	68.2	<b>40.2</b>
		GUR	68.2	39.5	68.2	39.6

Table 3: Accuracy (in percentage) on the Image Riddle Dataset. Pipeline variants (VB, RR and All) are combined with Bias-Correction stage variants (GUR, UR). We show both word2vec and WordNet-based (WN) accuracies. (\*- Best, † - Baselines).

To select the parameters in the parameter vector  $\theta$ , We employed a random search on the parameter-space over first 500 riddles over 500 combinations. The final set of parameters used and their values are tabulated in Table 2.

The accuracies after different stages of the pipeline (VB, RR and All) combined with variations of the initial Bias-Correction stage (GUR and UR), are listed in Table 3<sup>7</sup>. We provide our experimental results on this 3333 riddles

<sup>6</sup>The groundtruth is a single word. Code: [bit.ly/2gqmnwEe](http://bit.ly/2gqmnwEe).

<sup>7</sup>For ablation study on varying top  $K$ , check appendix.

and 2833 riddles (barring 500 riddles as validation set for the parameter search).

### 5.2.3 Experiment II: Human Evaluation

We conduct an AMT-based comparative evaluation of the results of the proposed approach (GUR+All using Clarifai) and two vision-only baselines. We define two metrics: i) “correctness” and ii) “intelligence”. Turkers are presented with the instructions: *We have three separate robots that attempted to answer this riddle. You have to rate the answer based on the correctness and the degree of intelligence (explainability)*. The correctness is defined as before. In addition, turkers are asked to rate intelligence in a scale of 1-4<sup>8</sup>. Figure 3 plots the percentage of total riddles per each value of correctness and intelligence. In these histograms plots, we expect an increase in the rightmost buckets for the more “correct” and “intelligent” systems.

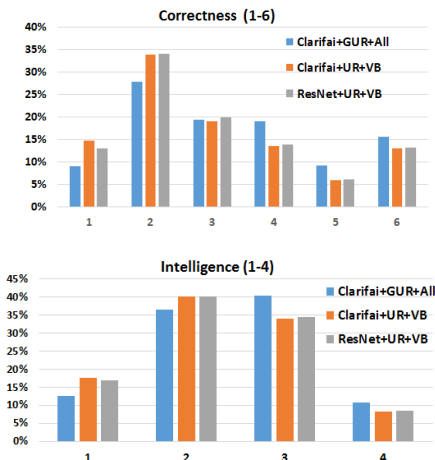


Figure 3: AMT Results of The Clarifai+GUR+All (our), Clarifai+UR+VB (baseline 1) and ResNet+UR+VB (baseline 2) approaches. Correctness Means are:  $2.6 \pm 1.4$ ,  $2.4 \pm 1.45$ ,  $2.3 \pm 1.4$ . For Intelligence:  $2.2 \pm 0.87$ ,  $2 \pm 0.87$ ,  $1.8 \pm 0.8$

### 5.2.4 Analysis

Experiment I shows that the GUR variant (Clarifai+GUR+All in Table 3) achieves the best results in terms of word2vec-based accuracy. The WordNet-based metric gives clear evidence of improvement by the stages of our pipeline (a sharp 14% increase over Clarifai and 6% increase over ResNet baselines). Improvement from the final reasoning stage is also evident from the result. The increase in accuracy after reasoning shows how knowledge helped in decreasing overall uncertainty in perception. Similar trend is reflected in the AMT-based evaluations (Figure 3).

<sup>8</sup>{1: Not, 2: Moderately, 3: Normal, 4: Very} intelligent



Figure 4: Positive and Negative (in red) results of the “GUR” approach (**Clarifai+GUR+All**) on some of the riddles. The groundtruth labels, closest label among top 10 from GUR and the Clarifai+VB+UR baseline are provided for all images. For more results, check Appendix.

Our system has increased the percentage of puzzles for the rightmost bins i.e. produces more “correct” and “intelligent” answers for more number of puzzles. The word2vec-based accuracy puts the performance of ResNet baseline close to that of the GUR variant. However, as evident from the WordNet-based metric and the AMT evaluation of the correctness (Figure 3), the GUR variant clearly predicts more meaningful answers than the ResNet baseline. Experiment II also includes what the turkers think about the intelligence of the systems that tried to solve the puzzles. This also puts the GUR variant at the top. The above two experiments empirically show that our approach achieves a reasonable accuracy in solving the riddles (Hypothesis I). In table 3, we observe how the accuracy varies after each stage of the pipeline (hypothesis II). The table shows a jump in the (WN) accuracy after the RR stage, which leads us to believe the primary improvement of our approach is attributed to the Probabilistic Reasoning model. We also provide our detailed results for the “GUR” approach using a few riddles in Figure 4.

**Difficulty of Riddles:** From our AMT study (**Human** baseline), we observe that the riddles are quite difficult for (untrained) human mechanical turkers. There are around 500 riddles which were labeled as “blank”, another 500 riddles were labeled as “not found”. Lastly, 457 riddles (391 with wordnet similarity higher than 0.9 and 66 higher than 0.8) were predicted perfectly, which leads us to believe that these easy riddles mostly show visual similarities (object-level) whereas others mostly

show conceptual similarity.

**Running Time:** Our implementation of PSL solves each riddle in nearly 20s in an Intel core i7 2.0 GHz processor, with 4 parallel threads. Solving each riddle boils down to solving 5 optimization problems (1 for each image and 1 joint). This eventually means our engine takes nearly 4 sec. to solve an inference problem with approximately  $20 \times 2500$  i.e. 50k rules.

**Reason to use a Probabilistic Logic:** We stated our reasons for choosing PSL over other available Probabilistic Logics. However, the simplicity of the used rules can leave the reader wondering about the reason for choosing a complex probabilistic logic in the first place. Each riddle requires an answer which is “logically” connected to each image. To show such logical connection, we need ontological knowledge graphs such as ConceptNet which shows connections between the answer and words detected from the images. To integrate ConceptNet’s knowledge seamlessly into the reasoning mechanism, we use a probabilistic logic such as PSL.

## 6 CONCLUSION

In this work, we presented a Probabilistic Reasoning based approach that uses background knowledge to solve a new class of image puzzles, called “Image Riddles”. We have collected over 3k such riddles. Crowd-sourced evaluation of the dataset demonstrates the validity of the annotations and the nature of the difficulty of the riddles. We empirically show that our approach improves on vision-only baselines and provides a stronger baseline for future attempts. The task of “Image Riddles” is equivalent to conventional IQ test questions such as analogy solving, sequence filling; which are often used to test human intelligence. This task of “Image Riddles” is also in line with the current trend of VQA datasets which require visual recognition and reasoning capabilities. However, it focuses more on the combination of both vision and reasoning capabilities. In addition to the task, the proposed approach introduces a novel inference model to infer related words (from a large vocabulary) given class labels (from a smaller set), using semantic knowledge of words. This method is general in terms of its applications. Systems such as (Wu et al., 2016), which use a collection of high-level concepts to boost VQA performance; can benefit from this approach.

## 7 ACKNOWLEDGMENTS

The support of the National Science Foundation under the Robust Intelligence Program (1816039 and 1750082) is gratefully acknowledged.

## References

- Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *International Conference on Computer Vision (ICCV)*, 2015. 2, 6, 7
- Stephen Bach, Bert Huang, Ben London, and Lise Getoor. Hinge-loss markov random fields: Convex inference for structured prediction. *arXiv preprint arXiv:1309.6813*, 2013. 2, 3
- David M. Blei. Probabilistic topic models. *Commun. ACM*, 55(4):77–84, April 2012. ISSN 0001-0782. doi: 10.1145/2133806.2133826. URL <http://doi.acm.org/10.1145/2133806.2133826>. 2
- Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):1915–1929, 2013. 4
- Haoyuan Gao, Junhua Mao, Jie Zhou, Zhiheng Huang, Lei Wang, and Wei Xu. Are you talking to a machine? dataset and methods for multilingual image question answering. In *NIPS*, 2015. 2
- Ross Girshick, Jeff Donahue, Trevor Darrell, and Jaganath Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 580–587. IEEE, 2014. 4
- Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90. URL <http://dx.doi.org/10.1109/CVPR.2016.90>. 2, 4
- Derek Hoiem, Yodsawalai Chodpathumwan, and Qieyun Dai. Diagnosing error in object detectors. In *European conference on computer vision*, pages 340–353. Springer, 2012. 5
- Angelika Kimmig, Stephen Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. A short introduction to probabilistic soft logic. In *Proceedings of the NIPS Workshop on Probabilistic Programming: Foundations and Applications*, pages 1–4, 2012. 2, 3
- GJ Klir and B Yuan. Fuzzy sets and fuzzy logic: theory and applications. 1995. 3
- Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 4
- Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, pages 1–101, 2016. 1
- Hugo Larochelle, Dumitru Erhan, Yoshua Bengio, Universit De Montral, and Montral Qubec. Zero-data learning of new tasks. In *In AAAI*, 2008. 2
- Yuhua Li, David McLean, Zuhair A Bandar, James D O’shea, and Keeley Crockett. Sentence similarity based on semantic nets and corpus statistics. *IEEE transactions on knowledge and data engineering*, 18(8):1138–1150, 2006. 8
- Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014*, pages 740–755. Springer, 2014. 2
- H. Liu and P. Singh. Conceptnet - a practical common-sense reasoning tool-kit. *BT Technology Journal*, 22(4):211–226, October 2004. ISSN 1358-3948. 2
- Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Hierarchical question-image co-attention for visual question answering. In *Advances In Neural Information Processing Systems*, pages 289–297, 2016. 7
- Lin Ma, Zhengdong Lu, and Hang Li. Learning to answer questions from image using convolutional neural network. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, pages 3567–3573, 2016. URL <http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11745>. 2
- Mateusz Malinowski, Marcus Rohrbach, and Mario Fritz. Ask your neurons: A neural-based approach to answering questions about images. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1–9, 2015. 2
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. 3
- T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, J. Beteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saporov, M. Greaves, and J. Welling. Never-ending

- learning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI-15)*, 2015. 3
- Happy Mittal, Shubhankar Suman Singh, Vibhav Gogate, and Parag Singla. Fine grained weight learning in markov logic networks. 2015. 3
- Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine learning*, 62(1-2):107–136, 2006. 3
- Gaurav Sood. *clarifai: R Client for the Clarifai API*, 2015. R package version 0.2. 2, 4
- Robert Speer and Catherine Havasi. Representing general relational knowledge in conceptnet 5. 2012. 3
- Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web, WWW '07*, pages 697–706, New York, NY, USA, 2007. ACM. 3
- Qi Wu, Chunhua Shen, Lingqiao Liu, Anthony Dick, and Anton van den Hengel. What value do explicit high level concepts have in vision to language problems? In *CVPR*, June 2016. 9

---

# Nesting Probabilistic Programs

---

**Tom Rainforth**

Department of Statistics  
University of Oxford

rainforth@stats.ox.ac.uk

## Abstract

We formalize the notion of nesting probabilistic programming queries and investigate the resulting statistical implications. We demonstrate that while query nesting allows the definition of models which could not otherwise be expressed, such as those involving agents reasoning about other agents, existing systems take approaches which lead to inconsistent estimates. We show how to correct this by delineating possible ways one might want to nest queries and asserting the respective conditions required for convergence. We further introduce a new *on-line* nested Monte Carlo estimator that makes it substantially easier to ensure these conditions are met, thereby providing a simple framework for designing statistically correct inference engines. We prove the correctness of this online estimator and show that, when using the recommended setup, its asymptotic variance is always better than that of the equivalent fixed estimator, while its bias is always within a factor of two.

## 1 INTRODUCTION

Probabilistic programming systems (PPSs) allow probabilistic models to be represented in the form of a generative model and statements for conditioning on data (Goodman et al., 2008; Gordon et al., 2014). Informally, one can think of the generative model as the definition of a prior, the conditioning statements as the definition of a likelihood, and the output of the program as samples from a posterior distribution. Their core philosophy is to decouple model specification and inference, the former corresponding to the user-specified program code and the latter to an inference engine capable of operating on arbitrary programs. Removing the need for users to write inference algorithms significantly reduces the burden of developing new models and makes effective statistical methods accessible to non-experts.

Some, so-called universal, systems (Goodman et al., 2008; Goodman and Stuhlmüller, 2014; Mansinghka et al., 2014; Wood et al., 2014) further allow the definition of models that would be hard, or even impossible, to convey using conventional frameworks such as graphical models. One enticing manner they do this is by allowing arbitrary nesting of models, known in the probabilistic programming literature as queries (Goodman et al., 2008), such that it is easy to define and run problems that fall outside the standard inference framework (Goodman et al., 2008; Mantadelis and Janssens, 2011; Stuhlmüller and Goodman, 2014; Le et al., 2016). This allows the definition of models that could not be encoded without nesting, such as experimental design problems (Ouyang et al., 2016) and various models for theory-of-mind (Stuhlmüller and Goodman, 2014). In particular, models that involve agents reasoning about other agents require, in general, some form of nesting. For example, one might use such nesting to model a poker player reasoning about another player as shown in Section 3.1. As machine learning increasingly starts to try and tackle problem domains that require interaction with humans or other external systems, such as the need for self-driving cars to account for the behavior of pedestrians, we believe that such nested problems are likely to become increasingly common and that PPSs will form a powerful tool for encoding them.

However, previous work has, in general, implicitly, and incorrectly, assumed that the convergence results from standard inference schemes carry over directly to the nested setting. In truth, inference for nested queries falls outside the scope of conventional proofs and so additional work is required to prove the consistency of PPS inference engines for nested queries. Such problems constitute special cases of *nested estimation*. In particular, the use of Monte Carlo (MC) methods by most PPSs mean they form particular instances of *nested Monte Carlo* (NMC) estimation (Hong and Juneja, 2009). Recent work (Rainforth et al., 2016a, 2018; Fort et al., 2017) has demonstrated that NMC is consistent for a general class of models, but

also that it entails a convergence rate in the total computational cost which decreases exponentially with the depth of the nesting. Furthermore, additional assumptions are required to achieve this convergence, most noticeably that, except in a few special cases, one needs to drive not only the total number of samples used to infinity, but also the number of samples used at each layer of the estimator, a requirement generally flaunted by existing PPSs.

The aim of this work is to formalize the notion of query nesting and use these recent NMC results to investigate the statistical correctness of the resulting procedures carried out by PPS inference engines. To do this, we postulate that there are three distinct ways one might nest one query within another: sampling from the conditional distribution of another query (which we refer to as *nested inference*), factoring the trace probability of one query with the partition function estimate of another (which we refer to as *nested conditioning*), and using expectation estimates calculated using one query as *first class variables* in another. We use the aforementioned NMC results to assess the relative correctness of each of these categories of nesting. In the interest of exposition, we will mostly focus on the PPS *Anglican* (Tolpin et al., 2016; Wood et al., 2014) (and also occasionally Church (Goodman et al., 2008)) as a basis for our discussion, but note that our results apply more generally. For example, our nested inference case covers the problem of sampling from cut distributions in OpenBugs (Plummer, 2015).

We find that nested inference is statistically challenging and incorrectly handled by existing systems, while nested conditioning is statistically straightforward and done correctly. Using estimates as variables turns out to be exactly equivalent to generic NMC estimation and must thus be dealt with on a case-by-case basis. Consequently, we will focus more on nested inference than the other cases.

To assist in the development of consistent approaches, we further introduce a new *online* NMC (ONMC) scheme that obviates the need to revisit previous samples when refining estimates, thereby simplifying the process of writing consistent online nested estimation schemes, as required by most PPSs. We show that ONMC’s convergence rate only varies by a small constant factor relative to conventional NMC: given some weak assumptions and the use of recommended parameter settings, its asymptotic variance is always better than the equivalent NMC estimator with matched total sample budget, while its asymptotic bias is always within a factor of two.

## 2 BACKGROUND

### 2.1 NESTED MONTE CARLO

We start by providing a brief introduction to NMC, using similar notation to that of Rainforth et al. (2018). Conventional MC estimation approximates an intractable

expectation  $\gamma_0$  of a function  $\lambda$  using

$$\gamma_0 = \mathbb{E} \left[ \lambda(y^{(0)}) \right] \approx I_0 = \frac{1}{N_0} \sum_{n=1}^{N_0} \lambda(y_n^{(0)}) \quad (1)$$

where  $y_n^{(0)} \stackrel{i.i.d.}{\sim} p(y^{(0)})$ , resulting in a mean squared error (MSE) that decreases at a rate  $O(1/N_0)$ . For nested estimation problems,  $\lambda(y^{(0)})$  is itself intractable, corresponding to a nonlinear mapping of a (nested) estimation. Thus in the single nesting case,  $\lambda(y^{(0)}) = f_0(y^{(0)}, \mathbb{E}[f_1(y^{(0)}, y^{(1)})|y^{(0)}])$  giving

$$\begin{aligned} \gamma_0 &= \mathbb{E} \left[ f_0 \left( y^{(0)}, \mathbb{E} \left[ f_1 \left( y^{(0)}, y^{(1)} \right) \middle| y^{(0)} \right] \right) \right] \\ &\approx I_0 = \frac{1}{N_0} \sum_{n=1}^{N_0} f_0 \left( y_n^{(0)}, \frac{1}{N_1} \sum_{m=1}^{N_1} f_1 \left( y_n^{(0)}, y_{n,m}^{(1)} \right) \right) \end{aligned}$$

where each  $y_{n,m}^{(1)} \sim p(y^{(1)}|y_n^{(0)})$  is drawn independently and  $I_0$  is now a NMC estimate using  $T = N_0 N_1$  samples.

More generally, one may have multiple layers of nesting. To notate this, we first presume some fixed integral depth  $D \geq 0$  (with  $D = 0$  corresponding to conventional estimation), and real-valued functions  $f_0, \dots, f_D$ . We then recursively define

$$\begin{aligned} \gamma_D \left( y^{(0:D-1)} \right) &= \mathbb{E} \left[ f_D \left( y^{(0:D)} \right) \middle| y^{(0:D-1)} \right], \quad \text{and} \\ \gamma_k \left( y^{(0:k-1)} \right) &= \mathbb{E} \left[ f_k \left( y^{(0:k)}, \gamma_{k+1} \left( y^{(0:k)} \right) \right) \middle| y^{(0:k-1)} \right] \end{aligned}$$

for  $0 \leq k < D$ . Our goal is to estimate  $\gamma_0 = \mathbb{E}[f_0(y^{(0)}, \gamma_1(y^{(0)}))]$ , for which the NMC estimate is  $I_0$  defined recursively using

$$\begin{aligned} I_D \left( y^{(0:D-1)} \right) &= \frac{1}{N_D} \sum_{n_D=1}^{N_D} f_D \left( y^{(0:D-1)}, y_{n_D}^{(D)} \right) \quad \text{and} \\ I_k \left( y^{(0:k-1)} \right) &= \frac{1}{N_k} \sum_{n_k=1}^{N_k} f_k \left( y^{(0:k-1)}, y_{n_k}^{(k)}, I_{k+1} \left( y^{(0:k-1)}, y_{n_k}^{(k)} \right) \right) \end{aligned} \quad (2)$$

for  $0 \leq k < D$ , where each  $y_{n_k}^{(k)} \sim p(y^{(k)}|y^{(0:k-1)})$  is drawn independently. Note that there are multiple values of  $y^{(k)}$  for each associated  $y^{(0:k-1)}$  and that  $I_k(y^{(0:k-1)})$  is still a random variable given  $y^{(0:k-1)}$ .

As shown by (Rainforth et al., 2018, Theorem 3), if each  $f_k$  is continuously differentiable and

$$\begin{aligned} \varsigma_k^2 &= \mathbb{E} \left[ \left( f_k \left( y^{(0:k)}, \gamma_{k+1} \left( y^{(0:k)} \right) \right) - \gamma_k \left( y^{(0:k-1)} \right) \right)^2 \right] \\ &< \infty \quad \forall k \in 0, \dots, D, \end{aligned}$$

then the MSE converges at rate

$$\begin{aligned} \mathbb{E} \left[ (I_0 - \gamma_0)^2 \right] &\leq \frac{\varsigma_0^2}{N_0} + \\ &\left( \frac{C_0 \varsigma_1^2}{2N_1} + \sum_{k=0}^{D-2} \left( \prod_{d=0}^k K_d \right) \frac{C_{k+1} \varsigma_{k+2}^2}{2N_{k+2}} \right)^2 + O(\epsilon) \end{aligned} \quad (3)$$

where  $K_k$  and  $C_k$  are respectively bounds on the magni-

tude of the first and second derivatives of  $f_k$ , and  $O(\epsilon)$  represents asymptotically dominated terms – a convention we will use throughout. Note that the dominant terms in the bound correspond respectively to the variance and the bias squared. Theorem 2 of Rainforth et al. (2018) further shows that the continuously differentiable assumption must hold almost surely, rather than absolutely, for convergence more generally, such that functions with measure-zero discontinuities still converge in general.

We see from (3) that if any of the  $N_k$  remain fixed, there is a minimum error that can be achieved: convergence requires each  $N_k \rightarrow \infty$ . As we will later show, many of the shortfalls in dealing with nested queries by existing PPSs revolve around implicitly fixing  $N_k \forall k \geq 1$ .

For a given total sample budget  $T = N_0 N_1 \dots N_D$ , the bound is tightest when  $\sqrt{N_0} \propto N_1 \propto \dots \propto N_D$  giving a convergence rate of  $O(1/T^{\frac{2}{D+2}})$ . The intuition behind this potentially surprising optimum setting is that the variance is mostly dictated by  $N_0$  and bias by the other  $N_k$ . We see that the convergence rate diminishes exponentially with  $D$ . However, this optimal setting of the  $N_k$  still gives a substantially faster rate than the  $O(1/T^{\frac{1}{D+1}})$  from naively setting  $N_0 \propto N_1 \propto \dots \propto N_D$ .

## 2.2 THE ANGLICAN PPS

Anglican is a universal probabilistic programming language integrated into *Clojure* (Hickey, 2008), a dialect of Lisp. There are two important ideas to understand for reading Clojure: almost everything is a function and parentheses cause evaluation. For example,  $a + b$  is coded as `(+ a b)` where `+` is a function taking two arguments and the parentheses cause the function to evaluate.

Anglican inherits most of the syntax of Clojure, but extends it with the key special forms `sample` and `observe` (Wood et al., 2014; Tolpin et al., 2015, 2016), between which the distribution of the query is defined. Informally, `sample` specifies terms in the prior and `observe` terms in the likelihood. More precisely, `sample` is used to make random draws from a provided distribution and `observe` is used to apply conditioning, factoring the probability density of a program trace by a provided density evaluated at an “observed” point.

The syntax of `sample` is to take a *distribution object* as its only input and return a sample. `observe` instead takes a distribution object and an observation and returns `nil`, while changing the program trace probability in Anglican’s back-end. Anglican provides a number of *elementary random procedures*, i.e. distribution object constructors for common sampling distributions, but also allows users to define their own distribution object constructors using the `defdist` macro. Distribution objects are generated by calling a class constructor with the required parameters, e.g. `(normal 0 1)`.

Anglican queries are written using the macro `defquery`. This allows users to define a model using a mixture of `sample` and `observe` statements and deterministic code, and bind that model to a variable. As a simple example,

```
(defquery my-query [data]
  (let [μ (sample (normal 0 1))
        σ (sample (gamma 2 2))
        lik (normal μ σ)]
    (map (fn [obs] (observe lik obs)) data)
        [μ σ]))
```

corresponds to a model where we are trying to infer the mean and standard deviation of a Gaussian given some data. The syntax of `defquery` is `(defquery name [args] body)` such that we are binding the query to `my-query` here. The query starts by sampling  $\mu \sim \mathcal{N}(0, 1)$  and  $\sigma \sim \Gamma(2, 2)$ , before constructing a distribution object `lik` to use for the observations. It then maps over each datapoint and observes it under the distribution `lik`. After the observations are made,  $\mu$  and  $\sigma$  are returned from the variable-binding `let` block and then by proxy the query itself. Denoting the data as  $y_{1:S}$  this particular query defines the joint distribution

$$p(\mu, \sigma, y_{1:S}) = \mathcal{N}(\mu; 0, 1) \Gamma(\sigma; 2, 2) \prod_{s=1}^S \mathcal{N}(y_s; \mu, \sigma).$$

Inference on a query is performed using the macro `doquery`, which produces a lazy infinite sequence of approximate samples from the conditional distribution and, for appropriate inference algorithms, an estimate of the partition function. Its calling syntax is `(doquery inf-alg model inputs & options)`.

Key to our purposes is Anglican’s ability to *nest* queries within one another. In particular, the special form `conditional` takes a query and returns a distribution object constructor, the outputs of which ostensibly corresponds to the conditional distribution defined by the query, with the inputs to the query becoming its parameters. However, as we will show in the next section, the true behavior of `conditional` deviates from this, thereby leading to inconsistent nested inference schemes.

## 3 NESTED INFERENCE

One of the clearest ways one might want to nest queries is by sampling from the conditional distribution of one query inside another. A number of examples of this are provided for Church in (Stuhlmüller and Goodman, 2014).<sup>1</sup> Such *nested inference* problems fall under a more general framework of inference for so-called doubly (or multiply) intractable distributions (Murray et al., 2006). The key feature of these problems is that they include terms with unknown, *parameter dependent*, normalization constants. For nested probabilistic programming queries, this manifests through *conditional normalization*.

<sup>1</sup>Though their nesting happens within the conditioning predicate, Church’s semantics means they constitute nested inference.

Consider the following unnested model using the Anglican function declaration `defm`

```
(defm inner [y D]
  (let [z (sample (gamma y 1))]
    (observe (normal y z) D)
    z))

(defquery outer [D]
  (let [y (sample (beta 2 3))
        z (inner y D)]
    (* y z)))
```

Here `inner` is simply an Anglican function: it takes in inputs `y` and `D`, effects the trace probability through its `observe` statement, and returns the random variable `z` as output. The unnormalized distribution for this model is thus straightforwardly given by

$$\begin{aligned}\pi_u(y, z, D) &= p(y)p(z|y)p(D|y, z) \\ &= \text{BETA}(y; 2, 3) \Gamma(z; y, 1) \mathcal{N}(D; y, z^2),\end{aligned}$$

for which we can use conventional inference schemes.

We can convert this model to a nested inference problem by using `defquery` and `conditional` as follows

```
(defquery inner [y D]
  (let [z (sample (gamma y 1))]
    (observe (normal y z) D)
    z))

(defquery outer [D]
  (let [y (sample (beta 2 3))
        dist (conditional inner)
        z (sample (dist y D))]
    (* y z)))
```

This is now a nested query: a separate inference procedure is invoked for each call of `(sample (dist y D))`, returning an approximate sample from the conditional distribution defined by `inner` when input with the current values of `y` and `D`. Mathematically, `conditional` applies a conditional normalization. Specifically, the component of  $\pi_u$  from the previous example corresponding to `inner` was  $p(z|y)p(D|y, z)$  and `conditional` locally normalizes this to the probability distribution  $p(z|D, y)$ . The distribution now defined by `outer` is thus given by

$$\begin{aligned}\pi_n(y, z, D) &= p(y)p(z|y, D) = \frac{p(y)p(z|y)p(D|y, z)}{\int p(z|y)p(D|y, z)dz} \\ &= p(y) \frac{p(z|y)p(D|y, z)}{p(D|y)} \neq \pi_u(z, y, D).\end{aligned}$$

Critically, the partial normalization constant  $p(D|y)$  depends on `y` and so the conditional distribution is doubly intractable: we cannot evaluate  $\pi_n(y, z, D)$  exactly.

Another way of looking at this is that wrapping `inner` in `conditional` has “protected” `y` from the conditioning in `inner` (noting  $\pi_u(y, z, D) \propto p(y|D)p(z|y, D)$ ), such that its `observe` statement only affects the probability of `z` given `y` and not the marginal probability of `y`. This is why, when there is only a single layer of nesting, nested inference is equivalent to the notion of sampling from “cut

distributions” (Plummer, 2015), whereby the sampling of certain subsets of the variables in a model are made with factors of the overall likelihood omitted.

It is important to note that if we had observed the *output* of the inner query, rather than sampling from it, this would still constitute a nested inference problem. The key to the nesting is the conditional normalization applied by `conditional`, not the exact usage of the generated distribution object `dist`. However, as discussed in Appendix B, actually observing a nested query requires numerous additional computational issues to be overcome, which are beyond the scope of this paper. We thus focus on the nested sampling scenario.

### 3.1 MOTIVATING EXAMPLE

Before jumping into a full formalization of nested inference, we first consider the motivating example of modeling a poker player who reasons about another player. Here each player has access to information the other does not, namely the cards in their hand, and they must perform their own inference to deal with the resulting uncertainty.

Imagine that the first player is deciding whether or not to bet. She could naïvely just make this decision based on the strength of her hand, but more advanced play requires her to reason about actions the other player might take given her own action, e.g. by considering whether a bluff is likely to be successful. She can carry out such reasoning by constructing a model for the other player to try and predict their action given her action and their hand. Again this nested model could just simply be based on a naïve simulation, but we can refine it by adding another layer of meta-reasoning: the other player will themselves try to infer the first player’s hand to inform their own decision.

These layers of meta-reasoning create a nesting: for the first player to choose an action, they must run multiple simulations for what the other player will do given that action and their hand, each of which requires inference to be carried out. Here adding more levels of meta-reasoning can produce smarter models, but also requires additional layers of nesting. We expand on this example to give a concrete nested inference problem in Appendix E.

### 3.2 FORMALIZATION

To formalize the nested inference problem more generally, let `y` and `x` denote all the random variables of an outer query that are respectively passed or not to the inner query. Further, let `z` denote all random variables generated in the inner query – for simplicity, we will assume, without loss of generality, that these are all returned to the outer query, but that some may not be used. The unnormalized density for the outer query can now be written in the form

$$\pi_o(x, y, z) = \psi(x, y, z)p_i(z|y) \quad (4)$$



where  $p_i(z|y)$  is the normalized density of the outputs of the inner query and  $\psi(x, y, z)$  encapsulates all other terms influencing the trace probability of the outer query. Now the inner query defines an unnormalized density  $\pi_i(y, z)$  that can be evaluated pointwise and we have

$$p_i(z|y) = \frac{\pi_i(y, z)}{\int \pi_i(y, z') dz'} \quad \text{giving} \quad (5)$$

$$p_o(x, y, z) \propto \pi_o(x, y, z) = \frac{\psi(x, y, z)\pi_i(y, z)}{\int \pi_i(y, z') dz'} \quad (6)$$

where  $p_o(x, y, z)$  is our target distribution, for which we can directly evaluate the numerator, but the denominator is intractable and must be evaluated separately for each possible value of  $y$ . Our previous example is achieved by fixing  $\psi(x, y, z) = p(y)$  and  $\pi_i(y, z) = p(z|y)p(D|y, z)$ . We can further straightforwardly extend to the multiple layers of nesting setting by recursively defining  $\pi_i(y, z)$  in the same way as  $\pi_o(x, y, z)$ .

### 3.3 RELATIONSHIP TO NESTED ESTIMATION

To relate the nested inference problem back to the nested estimation formulation from Section 2.1, we consider using a proposal  $q(x, y, z) = q(x, y)q(z|y)$  to calculate the expectation of some arbitrary function  $g(x, y, z)$  under  $p_o(x, y, z)$  as per self-normalized importance sampling

$$\begin{aligned} \mathbb{E}_{p_o(x, y, z)} [g(x, y, z)] &= \frac{\mathbb{E}_{q(x, y, z)} \left[ \frac{g(x, y, z)\pi_o(x, y, z)}{q(x, y, z)} \right]}{\mathbb{E}_{q(x, y, z)} \left[ \frac{\pi_o(x, y, z)}{q(x, y, z)} \right]} \\ &= \frac{\mathbb{E}_{q(x, y, z)} \left[ \frac{g(x, y, z)\psi(x, y, z)\pi_i(y, z)}{q(x, y, z)\mathbb{E}_{z' \sim q(z|y)} [\pi_i(y, z')/q(z'|y)]} \right]}{\mathbb{E}_{q(x, y, z)} \left[ \frac{\psi(x, y, z)\pi_i(y, z)}{q(x, y, z)\mathbb{E}_{z' \sim q(z|y)} [\pi_i(y, z')/q(z'|y)]} \right]}. \end{aligned} \quad (7)$$

Here both the denominator and numerator are nested expectations with a nonlinearity coming from the fact that we are using the reciprocal of an expectation. A similar reformulation could also be applied in cases with multiple layers of nesting, i.e. where `inner` itself makes use of another query. The formalization can also be directly extended to the sequential MC (SMC) setting by invoking extended space arguments (Andrieu et al., 2010).

Typically  $g(x, y, z)$  is not known upfront and we instead return an empirical measure from the program in the form of weighted samples which can later be used to estimate an expectation. That is, if we sample  $(x_n, y_n) \sim q(x, y)$  and  $z_{n,m} \sim q(z|y_n)$  and return all samples  $(x_n, y_n, z_{n,m})$  (such that each  $(x_n, y_n)$  is duplicated  $N_1$  times in the sample set) then our unnormalized weights are given by

$$w_{n,m} = \frac{\psi(x_n, y_n, z_{n,m})\pi_i(y_n, z_{n,m})}{q(x_n, y_n, z_{n,m}) \frac{1}{N_1} \sum_{\ell=1}^{N_1} \frac{\pi_i(y_n, z_{n,\ell})}{q(z_{n,\ell}|y_n)}}. \quad (8)$$

This, in turn, gives us the empirical measure

$$\hat{p}(\cdot) = \frac{\sum_{n=1}^{N_0} \sum_{m=1}^{N_1} w_{n,m} \delta_{(x_n, y_n, z_{n,m})}(\cdot)}{\sum_{n=1}^{N_0} \sum_{m=1}^{N_1} w_{n,m}} \quad (9)$$

where  $\delta_{(x_n, y_n, z_{n,m})}(\cdot)$  is a delta function centered on  $(x_n, y_n, z_{n,m})$ . By definition, the convergence of this empirical measure to the target requires that expectation estimates calculated using it converge in probability for any integrable  $g(x, y, z)$  (presuming our proposal is valid). We thus see that the convergence of the ratio of nested expectations in (7) for any arbitrary  $g(x, y, z)$ , is equivalent to the produced samples converging to the distribution defined by the program. Informally, the NMC results then tell us this will happen in the limit  $N_0, N_1 \rightarrow \infty$  provided that  $\int \pi_i(y, z) dz$  is strictly positive for all possible  $y$  (as otherwise the problem becomes ill-defined). More formally we have the following result. Its proof, along with all others, is given in Appendix A.

**Theorem 1.** *Let  $g(x, y, z)$  be an integrable function, let  $\gamma_0 = \mathbb{E}_{p_o(x, y, z)}[g(x, y, z)]$ , and let  $I_0$  be a self-normalized MC estimate for  $\gamma_0$  calculated using  $\hat{p}(\cdot)$  as per (9). Assuming that  $q(x, y, z)$  forms a valid importance sampling proposal distribution for  $p_o(x, y, z)$ , then*

$$\mathbb{E} \left[ (I_0 - \gamma_0)^2 \right] = \frac{\sigma^2}{N_0} + \frac{\delta^2}{N_1^2} + O(\epsilon) \quad (10)$$

where  $\sigma$  and  $\delta$  are constants derived in the proof and, as before,  $O(\epsilon)$  represents asymptotically dominated terms.

Note that rather than simply being a bound, this result is an equality and thus provides the exact asymptotic rate. Using the arguments of (Rainforth et al., 2018, Theorem 3), it can be straightforwardly extended to cases of multiple nesting (giving a rate analogous to (3)), though characterizing  $\sigma$  and  $\delta$  becomes more challenging.

### 3.4 CONVERGENCE REQUIREMENTS

We have demonstrated that the problem of nested inference is a particular case of nested estimation. This problem equivalence will hold whether we elect to use the aforementioned nested importance sampling based approach or not, while we see that our finite sample estimates must be biased for non-trivial  $g$  by the convexity of  $f_0$  and Theorem 4 of Rainforth et al. (2018). Presuming we cannot produce exact samples from the inner query and that the set of possible inputs to the inner query is not finite (these are respectively considered in Appendix D and Appendix C), we thus see that there is no ‘‘silver bullet’’ that can reduce the problem to a standard estimation.

We now ask, what behavior do we need for Anglican’s **conditional**, and nested inference more generally, to ensure convergence? At a high level, the NMC results show us that we need the computational budget of each call of a nested query to become arbitrarily large, such that we use an infinite number of samples at each layer of

the estimator: we require each  $N_k \rightarrow \infty$ .

We have formally demonstrated convergence when this requirement is satisfied and the previously introduced nested importance sampling approach is used. Another possible approach would be to, instead of drawing samples to estimate (7) directly, importance sample  $N_1$  times for each call of the inner query and then return a single sample from these, drawn in proportion to the inner query importance weights. We can think of this as drawing the same raw samples, but then constructing the estimator as

$$\hat{p}^*(\cdot) = \frac{\sum_{n=1}^{N_0} w_n^* \delta_{(x_n, y_n, z_n, m^*(n))}(\cdot)}{\sum_{n=1}^{N_0} w_n^*} \quad (11)$$

where  $w_n^* = \frac{\psi(x_n, y_n, z_n, m^*(n))}{q(x_n, y_n)}$  and  $(12)$

$$m^*(n) \sim \text{DISCRETE} \left( \frac{\pi_i(y_n, z_n, m) / q(z_n, m | y_n)}{\sum_{\ell=1}^{N_1} \pi_i(y_n, z_n, \ell) / q(z_n, \ell | y_n)} \right)$$

As demonstrated formally in Appendix A, this approach also converges. However, if we Rao Blackwellize (Casella and Robert, 1996) the sampling of  $m^*(n)$ , we find that this recovers (9). Consequently, this is a strictly inferior estimator (it has an increased variance relative to (9)). Nonetheless, it may often be a convenient setup from the perspective of the PPS semantics and it will typically have substantially reduced memory requirements: we need only store the single returned sample from the inner query to construct our empirical measure, rather than all of the samples generated within the inner query.

Though one can use the results of Fort et al. (2017) to show the correctness of instead using an MCMC estimator for the outer query, the correctness of using MCMC methods for the inner queries is not explicitly covered by existing results. Here we find that we need to start a new Markov chain for each call of the inner query because each value of  $y$  defines a different local inference problem. One would intuitively expect the NMC results to carry over – as  $N_1 \rightarrow \infty$  all the inner queries will run their Markov chains for an infinitely long time, thereby in principle returning exact samples – but we leave formal proof of this case to future work. We note that such an approach effectively equates to what is referred to as *multiple imputation* by Plummer (2015).

### 3.5 SHORTFALLS OF EXISTING SYSTEMS

Using the empirical measure (9) provides one possible manner of producing a consistent estimate of our target by taking  $N_0, N_1 \rightarrow \infty$  and so we can use this as a gold-standard reference approach (with a large value of  $N_1$ ) to assess whether Anglican returns samples for the correct target distribution. To this end, we ran Anglican’s importance sampling inference engine on the simple model introduced earlier and compared its output to the reference approach using  $N_0 = 5 \times 10^6$  and  $N_1 = 10^3$ . As

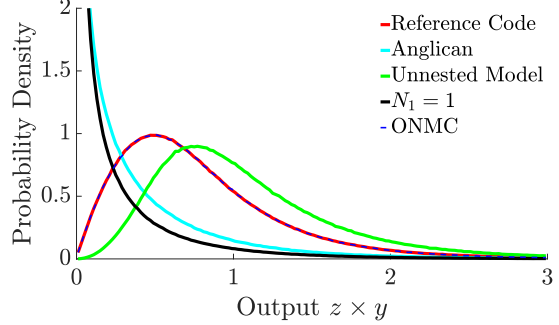


Figure 1: Empirical densities produced by running the nested Anglican queries given in the text, a reference NMC estimate, the unnested model, a naïve estimation scheme where  $N_1 = 1$ , and the ONMC approach introduced in Section 6, with the same computational budget of  $T = 5 \times 10^9$  and  $\tau_1(n_0) = \min(500, \sqrt{n_0})$ . Note that the results for ONMC and the reference approach overlap.

shown in Figure 1, the samples produced by Anglican are substantially different to the reference code, demonstrating that the outputs do not match their semantically intended distribution. For reference, we also considered the distribution induced by the aforementioned unnested model and a naïve estimation scheme where a sample budget of  $N_1 = 1$  is used for each call to `inner`, effectively corresponding to ignoring the `observe` statement by directly returning the first draw of  $z$ .

We see that the unnested model defines a noticeably different distribution, while the behavior of Anglican is similar, but distinct, to ignoring the `observe` statement in the inner query. Further investigation shows that the default behavior of `conditional` in a query nesting context is equivalent to using (11) but with  $N_1$  held fixed to at  $N_1 = 2$ , inducing a substantial bias. More generally, the Anglican source code shows that `conditional` defines a Markov chain generated by equalizing the output of the weighted samples generated by running inference on the query. When used to nest queries, this Markov chain is only ever run for a finite length of time, specifically one accept-reject step is carried out, and so does not produce samples from the true conditional distribution.

Plummer (2015) noticed that WinBugs and OpenBugs (Spiegelhalter et al., 1996) similarly do not provide valid inference when using their cut function primitives, which effectively allow the definition of nested inference problems. However, they do not notice the equivalence to the NMC formulation and instead propose a heuristic for reducing the bias that itself has no theoretical guarantees.

## 4 NESTED CONDITIONING

An alternative way one might wish to nest queries is to use the partition function estimate of one query to factor the trace probability of another. We refer to this as *nested conditioning*. In its simplest form, we can think about

conditioning on the values input to the inner query. In Anglican we can carry this out by using the following custom distribution object constructor

```
(defdist nest [inner inputs inf-alg M] []
  (sample [this] nil)
  (observe [this] _)
  (log-marginal (take M
    (doquery inf-alg inner inputs))))
```

When the resulting distribution object is observed, this will now generate, and factor the trace probability by, a partition function estimate for `inner` with inputs `inputs`, constructed using `M` samples of the inference algorithm `inf-alg`. For example, if we were to use the query

```
(defquery outer [D]
  (let [y (sample (beta 2 3))]
    (observe (nest inner [y D] :smc 100) nil
      y)))
```

with `inner` from the nested inference example, then this would form a pseudo marginal sampler (Andrieu and Roberts, 2009) for the unnormalized target distribution

$$\pi_c(y, D) = \text{BETA}(y; 2, 3) \int \Gamma(z; y, 1) \mathcal{N}(D; y, z^2) dz.$$

Unlike the nested inference case, nested conditioning turns out to be valid even if our budget is held fixed, provided that the partition function estimate is unbiased, as is satisfied by, for example, importance sampling and SMC. In fact, it is important to hold the budget fixed to achieve a MC convergence rate. In general, we can define our target density as

$$p_o(x, y) \propto \pi_o(x, y) = \psi(x, y) p_i(y), \quad (13)$$

where  $\psi(x, y)$  is as before (except that we no longer have returned variables from the inner query) and  $p_i(y)$  is the true partition function of the inner query when given input  $y$ . In practice, we cannot evaluate  $p_i(y)$  exactly, but instead produce unbiased estimates  $\hat{p}_i(y)$ . Using an analogous self-normalized importance sampling to the nested inference case leads to the weights

$$w_n = \psi(x_n, y_n) \hat{p}_i(y_n) / q(x_n, y_n) \quad (14)$$

and corresponding empirical measure

$$\hat{p}(\cdot) = \frac{1}{\sum_{n=1}^{N_0} w_n} \sum_{n=1}^{N_0} w_n \delta_{(x_n, y_n)}(\cdot) \quad (15)$$

such that we are conducting conventional MC estimation, but our weights are now themselves random variables for a given  $(x_n, y_n)$  due to the  $\hat{p}_i(y_n)$  term. However, the weights are unbiased estimates of the “true weights”  $\psi(x_n, y_n) p_i(y_n) / q(x_n, y_n)$  such that we have proper weighting (Naesseth et al., 2015) and thus convergence at the standard MC rate, provided the budget of the inner query remains fixed. This result also follows directly from Theorem 6 of Rainforth et al. (2018), which further ensures no complications arise when conditioning on multiple queries if the corresponding partition function estimates are generated independently. These results

further trivially extend to the repeated nesting case by recursion, while using the idea of pseudo-marginal methods (Andrieu and Roberts, 2009), the results also extend to using MCMC based inference for the outermost query.

Rather than just fixing the inputs to the nested query, one can also consider conditioning on the internally sampled variables in the program taking on certain values. Such a nested conditioning approach has been implicitly carried out by Rainforth et al. (2016b); Zinkov and Shan (2017); Scibior and Ghahramani (2016); Ge et al. (2018), each of which manipulate the original program in some fashion to construct a partition function estimator that is used within a greater inference scheme, e.g. a PMMH estimator (Andrieu et al., 2010).

## 5 ESTIMATES AS VARIABLES

Our final case is that one might wish to use estimates as first class variables in another query. In other words, a variable in an outer query is assigned to a MC expectation estimate calculated from the outputs of running inference on another, nested, query. By comparison, the nested inference case (without Rao-Blackwellization) can be thought of as assigning a variable in the outer query to a single approximate sample from the conditional distribution of the inner query, rather than an MC expectation estimate constructed by averaging over multiple samples.

Whereas nested inference can only encode a certain class of nested estimation problems – because the only nonlinearity originates from taking the reciprocal of the partition function – using estimates as variables allows, in principle, the encoding of any nested estimation. This is because using the estimate as a first class variable allows arbitrary nonlinear mappings to be applied by the outer query.

An example of this approach is shown in Appendix G, where we construct a generic estimator for Bayesian experimental design problems. Here a partition function estimate is constructed for an inner query and is then used in an outer query. The output of the outer query depends on the logarithm of this estimate, thereby creating the nonlinearity required to form a nested expectation.

Because using estimates as variables allows the encoding of any nested estimation problem, the validity of doing so is equivalent to that of NMC more generally and must thus satisfy the requirements set out in (Rainforth et al., 2018). In particular, one needs to ensure that the budgets used for the inner estimates increase as more samples of the outermost query are taken.

## 6 ONLINE NESTED MONTE CARLO

NMC will be highly inconvenient to actually implement in a PPS whenever one desires to provide online estimates; for example, a lazy sequence of samples that converges to the target distribution. Suppose that we have already

calculated an NMC estimate, but now desire to refine it further. In general, this will require an increase to all  $N_k$  for each sample of the outermost estimator. Consequently, the previous samples of the outermost query must be revisited to refine their estimates. This significantly complicates practical implementation, necessitating additional communication between queries, introducing computational overhead, and potentially substantially increasing the memory requirements.

To highlight these shortfalls concretely, consider the nested inference class of problems and, in particular, constructing the un–Rao–Blackwellized estimator (11) in an online fashion. Increasing  $N_1$  requires  $m^*(n)$  to be redrawn for each  $n$ , which in turn necessitates storage of previous samples and weights.<sup>2</sup> This leads to an overhead cost from the extra computation carried out for revisitation and a memory overhead from having to store information about each call of the inner query.

Perhaps even more problematically, the need to revisit old samples when drawing new samples can cause substantial complications for implementation. Consider implementing such an approach in Anglican. Anglican is designed to return a lazy infinite sequence of samples converging to the target distribution. Once samples are taken from this sequence, they become external to Anglican and cannot be posthumously updated when further samples are requested. Even when all the output samples remain internal, revisiting samples remains difficult: one either needs to implement some form of memory for nested queries so they can be run further, or, if all information is instead stored at the outermost level, additional non-trivial code is necessary to apply post-processing and to revisit queries with previously tested inputs. The latter of these is likely to necessitate inference–algorithm–specific changes, particularly when there are multiple levels of nesting, thereby hampering the entire language construction.

To alleviate these issues, we propose to only increase the computational budget of *new* calls to nested queries, such that earlier calls use fewer samples than later calls. This simple adjustment removes the need for communication between different calls and requires only the storage of the number of times the outermost query has previously been sampled to make updates to the overall estimate. We refer to this approach as *online* NMC (ONMC), which, to the best of our knowledge, has not been previously considered in the literature. As we now show, ONMC only leads to small changes in the convergence rate of the resultant estimator compared to NMC: using recommended parameter settings, the asymptotic root mean squared error

<sup>2</sup>Note that not all previous samples and weights need storing – when making the update we can sample whether to change  $m^*(n)$  or not based on combined weights from all the old samples compared to all the new samples.

for ONMC is never more than twice that of NMC for a matched sample budget and can even be smaller.

Let  $\tau_k(n_0) \in \mathbb{N}^+$ ,  $k = 1, \dots, D$  be monotonically increasing functions dictating the number of samples used by ONMC at depth  $k$  for the  $n_0$ -th iteration of the outermost estimator. The ONMC estimator is defined as

$$J_0 = \frac{1}{N_0} \sum_{n_0=1}^{N_0} f_0 \left( y_{n_0}^{(0)}, I_1 \left( y_{n_0}^{(0)}, \tau_{1:D}(n_0) \right) \right) \quad (16)$$

where  $I_1(y_{n_0}^{(0)}, \tau_{1:D}(n_0))$  is calculated using  $I_1$  in (2), setting  $y^{(0)} = y_{n_0}^{(0)}$  and  $N_k = \tau_k(n_0)$ ,  $\forall k \in 1, \dots, D$ . For reference, the NMC estimator,  $I_0$ , is as per (16), except for replacing  $\tau_{1:D}(n_0)$  with  $\tau_{1:D}(N_0)$ . Algorithmically, we have that the ONMC approach is defined as follows.

---

**Algorithm 1** Online Nested Monte Carlo

---

- 1:  $n_0 \leftarrow 0, \quad J_0 \leftarrow 0$
  - 2: **while** true **do**
  - 3:    $n_0 \leftarrow n_0 + 1, \quad y_{n_0}^{(0)} \sim p(y^{(0)})$
  - 4:   Construct  $I_1 \left( y_{n_0}^{(0)}, \tau_{1:D}(n_0) \right)$  using  $N_k = \tau_k(n_0) \forall k$
  - 5:    $J_0 \leftarrow \frac{n_0-1}{n_0} J_0 + f_0 \left( y_{n_0}^{(0)}, I_1 \left( y_{n_0}^{(0)}, \tau_{1:D}(n_0) \right) \right)$
- 

We see that OMMC uses fewer samples at inner layers for earlier samples of the outermost level, and that each of resulting inner estimates is calculated as per an NMC estimator with a reduced sample budget. We now show the consistency of the ONMC estimator.

**Theorem 2.** *If each  $\tau_k(n_0) \geq A(\log(n_0))^\alpha$ ,  $\forall n_0 > B$  for some constants  $A, B, \alpha > 0$  and each  $f_k$  is continuously differentiable, then the mean squared error of  $J_0$  as an estimator for  $\gamma_0$  converges to zero as  $N_0 \rightarrow \infty$ .*

In other words, ONMC converges for any realistic choice of  $\tau_k(n_0)$  provided  $\lim_{n_0 \rightarrow \infty} \tau_k(n_0) = \infty$ : the requirements on  $\tau_k(n_0)$  are, for example, much weaker than requiring a logarithmic or faster rate of growth, which would already be an impractically slow rate of increase.

In the case where  $\tau_k(n_0)$  increases at a polynomial rate, we can further quantify the rate of convergence, along with the relative variance and bias compared to NMC:

**Theorem 3.** *If each  $\tau_k(n_0) \geq An_0^\alpha$ ,  $\forall n_0 > B$  for some constants  $A, B, \alpha > 0$  and each  $f_k$  is continuously differentiable, then*

$$\mathbb{E} \left[ (J_0 - \gamma_0)^2 \right] \leq \frac{s_0^2}{N_0} + \left( \frac{\beta g(\alpha, N_0)}{AN_0^\alpha} \right)^2 + O(\epsilon), \quad (17)$$

$$\text{where } g(\alpha, N_0) = \begin{cases} 1/(1-\alpha), & \alpha < 1 \\ \log(N_0) + \eta, & \alpha = 1; \\ \zeta(\alpha)N_0^{\alpha-1}, & \alpha > 1 \end{cases} \quad (18)$$

$$\beta = \frac{C_0 s_1^2}{2} + \sum_{k=0}^{D-2} \left( \prod_{d=0}^k K_d \right) \frac{C_{k+1} s_{k+2}^2}{2}; \quad (19)$$

$\eta \approx 0.577$  is the Euler–Mascheroni constant;  $\zeta$  is the

Riemann–zeta function; and  $C_k$ ,  $K_k$ , and  $\varsigma_k$  are constants defined as per the corresponding NMC bound given in (3).

**Corollary 1.** Let  $J_0$  be an ONMC estimator setup as per Theorem 3 with  $N_0$  outermost samples and let  $I_0$  be an NMC estimator with a matched overall sample budget. Defining  $c = (1 + \alpha D)^{-1/(1+\alpha D)}$ , then

$$\text{Var}[J_0] \rightarrow c \text{Var}[I_0] \quad \text{as } N_0 \rightarrow \infty.$$

Further, if the NMC bias decreases at a rate proportional to that implied by the bound given in (3), namely

$$|\mathbb{E}[I_0 - \gamma_0]| = \frac{b}{M_0^\alpha} + O(\epsilon) \quad (20)$$

for some constant  $b > 0$ , where  $M_0$  is the number of outermost samples used by the NMC sampler, then

$$|\mathbb{E}[J_0 - \gamma_0]| \leq c^\alpha g(\alpha, N_0) |\mathbb{E}[I_0 - \gamma_0]| + O(\epsilon).$$

We expect the assumption that the bias scales as  $1/M_0^\alpha$  to be satisfied in the vast majority of scenarios, but there may be edge cases, e.g. when an  $f_k$  gives a constant output, for which faster rates are observed. Critically, the assumption holds for all nested inference problems because the rate given in (10) is an equality.

We see that if  $\alpha < 1$ , which will generally be the case in practice for sensible setups, then the convergence rates for ONMC and NMC vary only by a constant factor. Specifically, for a fixed value of  $N_0$ , they have the same asymptotic variance and ONMC has a factor of  $1/(1-\alpha)$  higher bias. However, the cost of ONMC is (asymptotically) only  $c < 1$  times that of NMC, so for a fixed overall sample budget it has lower variance.

As the bound varies only in constant factors for  $\alpha < 1$ , the asymptotically optimal value for  $\alpha$  for ONMC is the same as that for NMC, namely  $\alpha = 0.5$  (Rainforth et al., 2018). For this setup, we have  $c \in \{0.763, 0.707, 0.693, 0.693, 0.699, 1\}$  respectively for  $D \in \{1, 2, 3, 4, 5, \infty\}$ . Consequently, when  $\alpha = 0.5$ , the fixed budget variance of ONMC is always better than NMC, while the bias is no more than 1.75 times larger if  $D \leq 13$  and no more than 2 times large more generally.

## 6.1 EMPIRICAL CONFIRMATION

To test ONMC empirically, we consider the simple analytic model given in Appendix F, setting  $\tau_1(n_0) = \max(25, \sqrt{n_0})$ . The rationale for setting a minimum value of  $N_1$  is to minimize the burn-in effect of ONMC – earlier samples will have larger bias than later samples and we can mitigate this by ensuring a minimum value for  $N_1$ . More generally, we recommend setting (in the absence of other information)  $\tau_1(n_0) = \tau_2(n_0) = \dots = \tau_D(n_0) = \max(T_{\min}^{1/3}, \sqrt{n_0})$ , where  $T_{\min}$  is the minimum overall budget we expect to spend. In Figure 2, we have chosen to set  $T_{\min}$  deliberately low so as to emphasize the differences between NMC and ONMC. Given our value for  $T_{\min}$ , the ONMC approach is identical to fix-

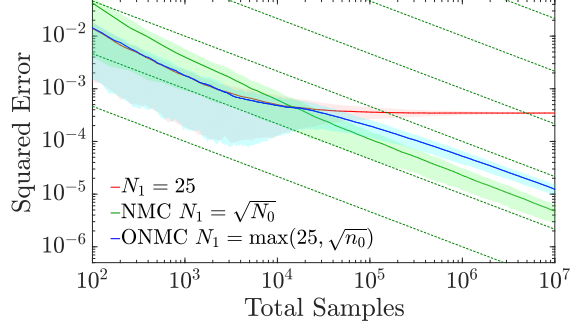


Figure 2: Convergence of ONMC, NMC, and fixed  $N_1$ . Results are averaged over 1000 runs, with solid lines showing the mean and shading the 25-75% quantiles. The theoretical rates for NMC are shown by the dashed lines.

ing  $N_1 = 25$  for  $T < 25^3 = 15625$ , but unlike fixing  $N_1$ , it continues to improve beyond this because it is not limited by asymptotic bias. Instead, we see an inflection point-like behavior around  $T_{\min}$ , with the rate recovering to effectively match that of the NMC estimator.

## 6.2 USING ONMC IN PPSs

Using ONMC based estimation schemes to ensure consistent estimation for nested inference in PPSs is straightforward – the number of iterations the outermost query has been run for is stored and used to set the number of iterations used for the inner queries. In fact, even this minimal level of communication is not necessary –  $n_0$  can be inferred from the number of times we have previously run inference on the current query, the current depth  $k$ , and  $\tau_1(\cdot), \dots, \tau_{k-1}(\cdot)$ .

As with NMC, for nested inference problems ONMC can either return a single sample from each call of a nested query, or Rao–Blackwellize the drawing of this sample when possible. Each respectively produces an estimator analogous to (11) and (9) respectively, except that  $N_1$  in the definition of the inner weights is now a function of  $n$ . Returning to Figure 1, we see that using ONMC with nested importance sampling and only returning a single sample corrects the previous issues with how Anglican deals with nested inference, producing samples indistinguishable from the reference code.

## 7 CONCLUSIONS

We have formalized the notion of nesting probabilistic program queries and investigated the statistical validity of different categories of nesting. We have found that current systems tend to use methods that lead to asymptotic bias for nested inference problems, but that they are consistent for nested conditioning. We have shown how to carry out the former in a consistent manner and developed a new *online* estimator that simplifies the construction algorithms that satisfy the conditions required for convergence.

## References

- C. Andrieu and G. O. Roberts. The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, pages 697–725, 2009.
- C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2010.
- G. Casella and C. P. Robert. Rao-Blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94, 1996.
- K. Chaloner and I. Verdinelli. Bayesian experimental design: A review. *Statistical Science*, 1995.
- R. Cornish, F. Wood, and H. Yang. Efficient exact inference in discrete Anglican programs. 2017.
- K. Csilléry, M. G. Blum, O. E. Gaggiotti, and O. François. Approximate Bayesian Computation (ABC) in practice. *Trends in Ecology & Evolution*, 25(7):410–418, 2010.
- M. F. Cusumano-Towner and V. K. Mansinghka. Using probabilistic programs as proposals. *arXiv preprint arXiv:1801.03612*, 2018.
- G. Fort, E. Gobet, and E. Moulines. MCMC design-based non-parametric regression for rare-event. application to nested risk computations. *Monte Carlo Methods Appl*, 2017.
- H. Ge, K. Xu, and Z. Ghahramani. Turing: a language for composable probabilistic inference. In *AISTATS*, 2018.
- N. Goodman, V. Mansinghka, D. M. Roy, K. Bonawitz, and J. B. Tenenbaum. Church: a language for generative models. *UAI*, 2008.
- N. D. Goodman and A. Stuhlmüller. *The Design and Implementation of Probabilistic Programming Languages*. 2014.
- A. D. Gordon, T. A. Henzinger, A. V. Nori, and S. K. Rajamani. Probabilistic programming. In *Proceedings of the on Future of Software Engineering*. ACM, 2014.
- R. Hickey. The Clojure programming language. In *Proceedings of the 2008 symposium on Dynamic languages*, page 1. ACM, 2008.
- L. J. Hong and S. Juneja. Estimating the mean of a non-linear function of conditional expectation. In *Winter Simulation Conference*, 2009.
- T. A. Le, A. G. Baydin, and F. Wood. Nested compiled inference for hierarchical reinforcement learning. In *NIPS Workshop on Bayesian Deep Learning*, 2016.
- V. Mansinghka, D. Selsam, and Y. Perov. Venture: a higher-order probabilistic programming platform with programmable inference. *arXiv preprint arXiv:1404.0099*, 2014.
- T. Mantadelis and G. Janssens. Nesting probabilistic inference. *arXiv preprint arXiv:1112.3785*, 2011.
- I. Murray, Z. Ghahramani, and D. J. MacKay. MCMC for doubly-intractable distributions. In *UAI*, 2006.
- C. A. Naesseth, F. Lindsten, and T. B. Schön. Nested sequential Monte Carlo methods. In *ICML*, 2015.
- L. Ouyang, M. H. Tessler, D. Ly, and N. Goodman. Practical optimal experiment design with probabilistic programs. *arXiv preprint arXiv:1608.05046*, 2016.
- M. Plummer. Cuts in Bayesian graphical models. *Statistics and Computing*, 25(1):37–43, 2015.
- J. G. Propp and D. B. Wilson. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random structures and Algorithms*, 9(1-2): 223–252, 1996.
- T. Rainforth. *Automating Inference, Learning, and Design using Probabilistic Programming*. PhD thesis, 2017.
- T. Rainforth, R. Cornish, H. Yang, and F. Wood. On the pitfalls of nested Monte Carlo. *NIPS Workshop on Advances in Approximate Bayesian Inference*, 2016a.
- T. Rainforth, T. A. Le, J.-W. van de Meent, M. A. Osborne, and F. Wood. Bayesian optimization for probabilistic programs. In *NIPS*, pages 280–288, 2016b.
- T. Rainforth, R. Cornish, H. Yang, A. Warrington, and F. Wood. On nesting Monte Carlo estimators. In *ICML*, 2018.
- A. Scibior and Z. Ghahramani. Modular construction of Bayesian inference algorithms. In *NIPS Workshop on Advances in Approximate Bayesian Inference*, 2016.
- D. Spiegelhalter, A. Thomas, N. Best, and W. Gilks. BUGS 0.5: Bayesian inference using Gibbs sampling manual (version ii). *MRC Biostatistics Unit, Cambridge*, 1996.
- A. Stuhlmüller and N. D. Goodman. A dynamic programming algorithm for inference in recursive probabilistic programs. In *Second Statistical Relational AI workshop at UAI 2012 (StaRAI-12)*, 2012.
- A. Stuhlmüller and N. D. Goodman. Reasoning about reasoning by nested conditioning: Modeling theory of mind with probabilistic programs. *Cognitive Systems Research*, 28:80–99, 2014.
- D. Tolpin, J.-W. van de Meent, and F. Wood. Probabilistic programming in Anglican. Springer, 2015.
- D. Tolpin, J.-W. van de Meent, H. Yang, and F. Wood. Design and implementation of probabilistic programming language Anglican. In *Proceedings of the 28th Symposium on the Implementation and Application of Functional Programming Languages*. ACM, 2016.
- F. Wood, J. W. van de Meent, and V. Mansinghka. A new approach to probabilistic programming inference. In *AISTATS*, pages 2–46, 2014.
- R. Zinkov and C.-C. Shan. Composing inference algorithms as program transformations. In *UAI*, 2017.

---

# Scalable Algorithms for Learning High-Dimensional Linear Mixed Models

---

**Zilong Tan**  
Duke University  
ztan@cs.duke.edu

**Kimberly Roche**  
Duke University  
kimberly.roche@duke.edu

**Xiang Zhou**  
University of Michigan  
xzhoushp@umich.edu

**Sayan Mukherjee**  
Duke University  
sayan@stat.duke.edu

## Abstract

Linear mixed models (LMMs) are used extensively to model observations that are not independent. Parameter estimation for LMMs can be computationally prohibitive on big data. State-of-the-art learning algorithms require computational complexity which depends at least linearly on the dimension  $p$  of the covariates, and often use heuristics that do not offer theoretical guarantees. We present scalable algorithms for learning high-dimensional LMMs with sublinear computational complexity dependence on  $p$ . Key to our approach are novel dual estimators which use only kernel functions of the data, and fast computational techniques based on the subsampled randomized Hadamard transform. We provide theoretical guarantees for our learning algorithms, demonstrating the robustness of parameter estimation. Finally, we complement the theory with experiments on large synthetic and real data.

## 1 INTRODUCTION

Linear mixed models (LMMs) are widely used in many real world applications ranging from longitudinal data analysis (Laird and Ware, 1982; Demidenko, 2013) and genome wide association studies (Kang et al., 2008; Lippert et al., 2011; Zhou, 2017) to recommender systems (Zhang et al., 2016). LMMs provide a flexible framework for modeling a wide range of data types, including clustered, longitudinal, and spatial data. Parameter estimation for LMMs is computationally prohibitive for big data, both for large sample size  $n$  (Zhou and Stephens, 2014; Darnell et al., 2017; Perry, 2017) and for high-dimensional covariates  $p$  (Schelldorfer et al., 2011). The

main computational bottlenecks for parameter estimation arise from the non-convexity of the optimization problem (Kang et al., 2008; Perry, 2017) as well as the computational cost of matrix inversions (Zhou, 2017; Laird et al., 1987; Lindstrom and Bates, 1988; Bates et al., 2015). State-of-the-art methods for parameter estimation in LMMs require computational complexity that depends at least linearly on  $p$ : (i)  $O(nkp)$  for the setting  $n > p$  with a rank  $k$  covariance matrix (Zhou, 2017; Darnell et al., 2017); and (ii)  $O(n^2p)$  per iteration for  $p \gg n$  (Schelldorfer et al., 2011, 2014; Jakubik, 2015). In this paper, we present scalable algorithms with sublinear computational complexity in  $p$ , making the proposed approach useful for high-dimensional LMMs. In addition, we provide a theoretical analysis for our approach that states provable error guarantees between the estimated and ground-truth parameters.

Two sets of parameters are estimated in LMMs, the fixed-effects coefficients and the variances for the unobservable random effects and noise. The random-effects variance is generally assumed to have a certain structure, such as a block-diagonal matrix (Laird and Ware, 1982; Demidenko, 2013). To estimate both sets of parameters, an expectation maximization (EM) algorithm is typically used (Laird et al., 1987; Bates et al., 2015) to handle the latent random-effect variable. The M-step in the EM algorithm incurs high computational costs due to matrix inversions. Newton-Raphson has been used to reduce the number of iterations required for parameter estimates to converge (Lindstrom and Bates, 1988); however, each iteration is still costly due to matrix inversions. A recent research focus is to avoid matrix inversions at each iteration. For instance, when  $n > p$  a spectral algorithm is available (Kang et al., 2008; Lippert et al., 2011; Patterson and Thompson, 1971). The state-of-the-art algorithm (Darnell et al., 2017) further improved the computational complexity of the spectral algorithm using randomized singular value decomposition (Darnell et al., 2017).

While approximate learning algorithms (Zhou, 2017;

Darnell et al., 2017) are efficient, few provide provable guarantees in terms of estimation accuracy. Recently, a guaranteed non-iterative algorithm was proposed in (Perry, 2017), which runs in  $O(n(p+d)^4)$  time for  $d$  random effects. Inference with guarantees for high-dimensional LMMs, i.e.,  $p \gg n$ , typically incurs greater computational complexity due to the regularization required to address high-dimensional data (Schell-dorfer et al., 2011, 2014). In the high-dimensional setting, most algorithms perform block coordinate descent with an  $O(n^2p)$  per-iteration cost (Schell-dorfer et al., 2011, 2014). In this paper, we show that efficiency and provable guarantees can be achieved simultaneously for learning high-dimensional LMMs.

There are two key ideas we use in our efficient algorithms. The first idea is to propose an approximate estimator that relies on an  $n \times n$  kernel matrix (§ 3) which can be computed efficiently using the subsampled randomized Hadamard transform (SRHT) (Tropp, 2011). This reduces the linear complexity dependence on  $p$ . Unlike some other approximation algorithms (Lu et al., 2013), the proposed estimator also has the advantage of recovering the fixed-effects coefficients for all  $p$  dimensions as opposed to the reduced dimensions. This allows us to provide effect sizes in terms of the original covariates, a requirement in many applications. The second idea is the introduction of *approximate variance components* (AVCs) to replace variance components when estimating the fixed-effects coefficients. These AVCs have a closed-form expression and are fast to compute.

We apply our novel approach to LMMs with a both general covariances as well as a block-diagonal covariances for the random effects. The former can be viewed as a special case of the latter with a single block, and has been adopted in genome-wide association studies (Kang et al., 2008; Lippert et al., 2011; Zhou, 2017). LMMs with a block-diagonal covariance structure have been widely used for modeling repeated measures data (Laird and Ware, 1982). We propose a non-iterative algorithm for the general covariance setting and a fast EM variant for the block-diagonal setting.

**Contribution** Our main contribution is providing a class of approximation algorithms for parameter inference in high-dimensional LMMs with provable guarantees. In Table 1, we state the computational complexity for several standard and state-of-the-art parameter inference algorithms. In the table and in this paper,  $n$  is the sample size,  $p$  is the number of covariates,  $k$  is the rank of the covariance matrix,  $s$  are the number of subsamples, and  $\epsilon$  is the approximation error. Our method is the only one that is sublinear in  $p$ , and can be a  $n/\log p$  magnitude

Table 1: Computational complexity for parameter inference. † denotes that the estimator has provable guarantees.

REML (LIPPERT ET AL., 2011)	$O(n^2p)$
†MOMENTS (PERRY, 2017)	$O(n(p+q)^4)$
SUBSAMPLING (ZHOU, 2017)	$O(ps^2)$
RSVD (DARNELL ET AL., 2017)	$O(pnk)$
†THIS WORK	$O\left(\frac{n^2(k+\log p)\log k}{\epsilon^2}\right)$

faster than the others (discussed in § 4.1). In addition to theoretical advantages, we demonstrate the empirical accuracy and speed of our method on both synthetic and real data in § 6.

**Notation** We denote the maximum and minimum eigenvalues of a matrix  $\mathbf{A}$  by  $\lambda_{\max}(\mathbf{A})$  and  $\lambda_{\min}(\mathbf{A})$ , respectively. Similarly, we denote the maximum and minimum singular values respectively by  $\sigma_{\max}(\mathbf{A})$  and  $\sigma_{\min}(\mathbf{A})$ .  $\mathbf{A}^\dagger$  represents the Moore–Penrose pseudoinverse of  $\mathbf{A}$ , and  $\kappa(\mathbf{A})$  denotes the condition number of  $\mathbf{A}$ . The superscripted notation  $\mathbf{y}^{(i)}$  refers to the copy of  $\mathbf{y}$  for group  $i$ . We write the spectral norm of a matrix as  $\|\cdot\|_2$ , the Frobenius norm as  $\|\cdot\|_F$ , and the Ky Fan  $k$ -norm (the sum of the  $k$  largest singular values) as  $\|\cdot\|_k$ .

**Organization** Section 2 provides the background on standard LMMs. In section 3, we formulate the  $L_2$ -regularized LMMs and present approximate estimators based on a kernel matrix. Section 4 describes fast computational techniques for the approximate estimators. In section 5, we provide theoretical guarantees for our estimators. Section 6 reports empirical evidence of the speed and accuracy of our methods, and section 7 concludes this paper.

## 2 LINEAR MIXED MODELS

Consider a regression problem with  $n$  observations, where  $\mathbf{y} \in \mathbb{R}^n$  denotes the response vector and  $\mathbf{X} \in \mathbb{R}^{n \times p}$  represents the covariate matrix with  $p$  covariates. The standard LMM is given by

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + c\mathbf{1} + \mathbf{e} \quad \text{with} \quad (1)$$

$$\begin{bmatrix} \boldsymbol{\gamma} \\ \mathbf{e} \end{bmatrix} \sim \text{MVN}\left(\mathbf{0}, \begin{bmatrix} \boldsymbol{\Lambda} & \mathbf{0} \\ \mathbf{0} & \sigma^2\mathbf{I} \end{bmatrix}\right),$$

where  $\boldsymbol{\beta} \in \mathbb{R}^p$  is the fixed-effect coefficient vector,  $\mathbf{Z} \in \mathbb{R}^{n \times q}$  is a full-rank random-effects design matrix,  $\boldsymbol{\gamma} \in \mathbb{R}^q$  is the random-effect coefficient vector,  $c$  is the intercept, and  $\mathbf{e} \in \mathbb{R}^n$  is the noise vector. The parameters to be estimated are the fixed-effects coefficients  $\boldsymbol{\beta}$ , and variance components  $\boldsymbol{\Lambda}$  and  $\sigma^2$ .



In general, the variables  $\mathbf{X}$ ,  $\mathbf{y}$ ,  $\boldsymbol{\gamma}$ , and  $\mathbf{e}$  in (1) correspond to observations from  $m$  classes, and are grouped by the following structure (Laird and Ware, 1982):

$$\begin{bmatrix} \mathbf{X}^{(1)} \\ \mathbf{X}^{(2)} \\ \vdots \\ \mathbf{X}^{(m)} \end{bmatrix}, \begin{bmatrix} \mathbf{y}^{(1)} \\ \mathbf{y}^{(2)} \\ \vdots \\ \mathbf{y}^{(m)} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\gamma}^{(1)} \\ \boldsymbol{\gamma}^{(2)} \\ \vdots \\ \boldsymbol{\gamma}^{(m)} \end{bmatrix}, \begin{bmatrix} \mathbf{e}^{(1)} \\ \mathbf{e}^{(2)} \\ \vdots \\ \mathbf{e}^{(m)} \end{bmatrix},$$

where  $\cdot^{(i)}$  denote the variables specific to group  $i$ , whose dimensions are  $\mathbf{X}^{(i)} \in \mathbb{R}^{n_i \times p}$ ,  $\boldsymbol{\gamma}^{(i)} \in \mathbb{R}^d$ , and  $\mathbf{y}^{(i)}, \mathbf{e}^{(i)} \in \mathbb{R}^{n_i}$ ,  $\sum_{i=1}^m n_i = n$ . The LMM assumes that  $\boldsymbol{\gamma}^{(i)}$  corresponding to distinct classes are independent. In particular, the random-effects design matrix  $\mathbf{Z}$  and the random-effects covariance are block-diagonal

$$\mathbf{Z} = \begin{bmatrix} \mathbf{Z}^{(1)} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{Z}^{(m)} \end{bmatrix}, \quad \boldsymbol{\Lambda} = \begin{bmatrix} \mathbf{H} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{H} \end{bmatrix}$$

with  $\mathbf{Z}^{(i)} \in \mathbb{R}^{n_i \times d}$ ,  $\mathbf{H} \in \mathbb{R}^{d \times d}$ , and  $q = md$ .

**Computational challenges** Parameter inference in LMMs aims to accurately recover  $\mathcal{P} := \{\boldsymbol{\beta}, \boldsymbol{\Lambda}, \sigma^2\}$  from  $\{\mathbf{X}, \mathbf{y}, \mathbf{Z}\}$ . This is straightforward if  $\boldsymbol{\Lambda}$  is given. When  $\boldsymbol{\Lambda}$  is unknown, inference can be computationally challenging even in the standard setting where  $n > p$  (Laird and Ware, 1982; Lippert et al., 2011; Zhou, 2017; Laird et al., 1987; Lindstrom and Bates, 1988; Patterson and Thompson, 1971; Zhang et al., 2011).

First, parameter estimation problem is non-convex for both maximum likelihood and restricted maximum likelihood (REML) (Laird et al., 1987; Patterson and Thompson, 1971; Harville, 1974). For instance, the methods using REML (Kang et al., 2008; Lippert et al., 2011) project the data onto two uncorrelated parts, and then estimate the fixed-effects and variance components separately on each part. This has the advantage of giving unbiased estimates of the variance components. However, the REML likelihood function is a non-convex function which involves the eigenvalues of the variance of the projected data (Patterson and Thompson, 1971).

Second, regularization is typically required to support the high-dimensional setting, which adds further computational overheads (Lippert et al., 2011; Zhou, 2017; Schelldorfer et al., 2011, 2014; Jakubík, 2015). To address these challenges, we develop novel approximate estimators that are efficient to compute (§ 4), and have provable accuracy guarantees (§ 5).

### 3 APPROXIMATE ESTIMATORS FOR HIGH-DIMENSIONAL LMMS

In this section, we consider an  $L_2$ -regularized LMM to support the high-dimensional setting  $p > n$ , and develop efficient approximate estimators for the parameters.

Standard parameter estimation algorithms for LMMs such as (Kang et al., 2008; Laird et al., 1987; Bates et al., 2015) do not support the high-dimensional setting  $p > n$ . We consider introducing the  $L_2$  regularization on the fixed-effects coefficients, which can be viewed as adding the prior  $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Phi})$ . The  $L_2$ -regularized LMM has the following log-likelihood

$$\begin{aligned} & \log p(\mathbf{y}, \boldsymbol{\beta} \mid \mathbf{X}; \mathbf{V}) \\ & \propto -\frac{1}{2} \boldsymbol{\beta}^\top \boldsymbol{\Phi}^{-1} \boldsymbol{\beta} - \frac{1}{2} \log \det \mathbf{V} \\ & \quad - \frac{1}{2} (\mathbf{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{c}\mathbf{1}) \mathbf{V}^{-1} (\mathbf{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{c}\mathbf{1}) \end{aligned} \quad (2)$$

with the marginal variance  $\mathbf{V} := \mathbf{Z}\boldsymbol{\Lambda}\mathbf{Z}^\top + \sigma^2\mathbf{I}$ .

Parameter estimation of an LMM is typically iterative and computationally prohibitive, especially in the high-dimensional setting (Darnell et al., 2017; Perry, 2017; Schelldorfer et al., 2011). To improve the computational efficiency, we propose dual as well as approximate estimators. These estimators are non-iterative and have reduced computational complexity, as we will show in § 4.

#### 3.1 FIXED-EFFECT COEFFICIENTS

We first derive the estimators for the fixed-effects coefficients  $\hat{\boldsymbol{\beta}}$  and  $\hat{\mathbf{c}}$ , which are the maximizers of the log-likelihood (2). A dual estimator of  $\boldsymbol{\beta}$  is then given for use in the high-dimensional setting. Using the partial derivatives, it is straightforward to show

$$(\mathbf{X}^\top \mathbf{V}^{-1} \mathbf{X} + \boldsymbol{\Phi}^{-1}) \hat{\boldsymbol{\beta}} = \mathbf{X}^\top \mathbf{V}^{-1} (\mathbf{y} - \hat{\mathbf{c}}\mathbf{1}) \quad (3)$$

$$\hat{\mathbf{c}} = \frac{\mathbf{1}^\top \mathbf{V}^{-1} \mathbf{y} - \mathbf{1}^\top \mathbf{V}^{-1} \mathbf{X} \hat{\boldsymbol{\beta}}}{\mathbf{1}^\top \mathbf{V}^{-1} \mathbf{1}}. \quad (4)$$

Let  $\mathbf{L} = \mathbf{I} - \mathbf{1}\mathbf{1}^\top \mathbf{V}^{-1} (\mathbf{1}^\top \mathbf{V}^{-1} \mathbf{1})^{-1}$ , we obtain

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top \mathbf{V}^{-1} \mathbf{L} \mathbf{X} + \boldsymbol{\Phi}^{-1})^{-1} \mathbf{X}^\top \mathbf{V}^{-1} \mathbf{L} \mathbf{y}. \quad (5)$$

The dual estimator using  $\mathbf{X}\boldsymbol{\Phi}\mathbf{X}^\top$  was proposed in (Saunders et al., 1998) where the authors used Lagrange multipliers to obtain the following estimator for ridge regression:

$$\hat{\boldsymbol{\beta}}_{\text{Dual}} = \boldsymbol{\Phi} \mathbf{X}^\top (\mathbf{V} + \mathbf{X}\boldsymbol{\Phi}\mathbf{X}^\top)^{-1} \mathbf{y}.$$

Here,  $\boldsymbol{\Phi}$  is set to be diagonal, and the above estimator (6) can be evaluated in  $O(n^2p)$  time, a significant improvement when  $p \gg n$ . However, the computational bottleneck becomes evaluating the *kernel matrix*  $\mathbf{X}\boldsymbol{\Phi}\mathbf{X}^\top$ .

For the zero intercept case  $\hat{c} = 0$ , the dual estimator (6) is equivalent to (5) from the following variant of the Woodbury identity  $(\mathbf{U}^{-1} + \mathbf{A}^\top \mathbf{V}^{-1} \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{V}^{-1} = \mathbf{U} \mathbf{A}^\top (\mathbf{A} \mathbf{U} \mathbf{A}^\top + \mathbf{V})^{-1}$  for invertible matrices  $\mathbf{U}$  and  $\mathbf{V}$ . The dual estimator can be generalized to any intercept,

$$\hat{\beta} = \Phi \mathbf{X}^\top \mathbf{V}^{-1} \mathbf{L} (\mathbf{X} \Phi \mathbf{X}^\top \mathbf{V}^{-1} \mathbf{L} + \mathbf{I})^{-1} \mathbf{y}. \quad (6)$$

Computing the dual estimator (6) takes  $O(n^2 p)$  time as opposed to  $O(p^3)$  time required by (5). This complexity will be further improved in § 4 for the setting  $p \gg n$ .

### 3.2 APPROXIMATE VARIANCE COMPONENTS

The variance components  $\Lambda$  and  $\sigma^2$  are typically estimated using an iterative EM algorithm with a per-iteration cost  $O(p^3)$  (Laird et al., 1987; Lindstrom and Bates, 1988) or an exhaustive grid search for the solution of a system of eigenvalue equations (Kang et al., 2008; Lippert et al., 2011). We consider an approximate non-iterative estimator based on the key observation that the optimization of the (2) has a simple closed-form solution if carried out with respect to  $\mathbf{M} = \mathbf{V} + \mathbf{X} \Phi \mathbf{X}^\top$ . We will estimate  $\mathbf{M}$  and use it as a proxy for estimating  $\Lambda$  as well as  $\sigma^2$ . The variance components inferred using  $\mathbf{M}$  are referred to as the *approximate variance components* (AVCs). While AVCs may be used as variance components estimates under certain circumstances, the main purpose is to serve as fast replacements in estimating fixed-effects coefficients.

**Proxy component estimation** To perform the REML estimation of the variance components in terms of  $\mathbf{M}$ , we first rewrite the log-likelihood (2) as

$$\begin{aligned} l(\beta, \mathbf{V}) &= -\frac{1}{2} \log \det \mathbf{V} \\ &\quad - \frac{1}{2} (\mathbf{y} - \hat{c} \mathbf{1})^\top \mathbf{M}^{-1} (\mathbf{y} - \hat{c} \mathbf{1}) \\ &\quad - \frac{1}{2} (\beta - \hat{\beta}(\mathbf{V}))^\top \mathbf{Q} (\beta - \hat{\beta}(\mathbf{V})) \end{aligned} \quad (7)$$

where  $\mathbf{Q} = \mathbf{X}^\top \mathbf{V}^{-1} \mathbf{X} + \Phi^{-1}$  and  $\hat{\beta}(\mathbf{V}) = (\mathbf{X}^\top \mathbf{V}^{-1} \mathbf{X} + \Phi^{-1})^{-1} \mathbf{X}^\top \mathbf{V}^{-1} (\mathbf{y} - \hat{c} \mathbf{1})$ . Here, the estimate  $\hat{\beta}$  depends on  $\mathbf{V}$ , and is consistent with the estimate given by (5). The  $\hat{c}$  in (7) can be set to the mean response, or estimated based on a prior distribution as in (Zhou et al., 2013).

Then, the REML estimator for the variance components is based on marginalizing the fixed effects  $\beta$  (Harville,

1974). It follows that

$$\begin{aligned} l_p(\mathbf{V}) &\propto \log \int_{\mathbb{R}^p} \exp(l(\beta, \mathbf{V})) d\beta \\ &\propto -\frac{1}{2} \log \det \mathbf{V} - \frac{1}{2} \log \det \mathbf{Q} \\ &\quad - \frac{1}{2} (\mathbf{y} - \hat{c} \mathbf{1})^\top \mathbf{M}^{-1} (\mathbf{y} - \hat{c} \mathbf{1}). \end{aligned}$$

From Sylvester's determinant theorem, one observes that  $\det(\mathbf{M}) = \det(\Phi) \det(\mathbf{V}) \det(\mathbf{Q})$ . Thus, we arrive at

$$\begin{aligned} l_p(\mathbf{V}) &\propto -\frac{1}{2} \log \det \mathbf{M} \\ &\quad - \frac{1}{2} (\mathbf{y} - \hat{c} \mathbf{1})^\top \mathbf{M}^{-1} (\mathbf{y} - \hat{c} \mathbf{1}). \end{aligned} \quad (8)$$

Now, what we have achieved through (8) is a simple closed-form REML estimate of  $\hat{\mathbf{V}}$ , rather than the non-convex or iterative updates for  $\hat{\Lambda}$  and  $\hat{\sigma}^2$  in state-of-the-art LMM parameter estimation algorithms. Unconstrained maximization of (8) with respect to  $\mathbf{M}$  results in the closed-form equality

$$\mathbf{Z} \hat{\Lambda} \mathbf{Z}^\top + \hat{\sigma}^2 \mathbf{I} = (\mathbf{y} - \hat{c} \mathbf{1}) (\mathbf{y} - \hat{c} \mathbf{1})^\top - \mathbf{X} \Phi \mathbf{X}^\top, \quad (9)$$

for an optimal  $\mathbf{M}$ . Note that  $\mathbf{Z} \hat{\Lambda} \mathbf{Z}^\top$  is positive semidefinite, whereas the right hand side has at most one positive eigenvalue. Thus, this optimal  $\mathbf{M}$  may not be achievable and the unbiased estimate of  $\Lambda$  may possibly have negative eigenvalues. The issue of negative variance estimates in linear mixed models is an open problem (Demidenko, 2013) and beyond the scope of this paper. One resolution is to introduce a Gamma prior on  $\Lambda$  (Chung et al., 2013). For unbiased estimation, we allow  $\Lambda$  to have negative eigenvalues, and intuitively we refer to the variance estimators obtained this way as *approximate variance components*.

**Approximate variance estimators** Assume that  $\mathbf{Z}$  has full column rank and let  $\mathbf{S} = (\mathbf{y} - \hat{c} \mathbf{1}) (\mathbf{y} - \hat{c} \mathbf{1})^\top - \mathbf{X} \Phi \mathbf{X}^\top$ . The approximate variance components  $\hat{\Lambda}_{\text{AVC}}$  and  $\hat{\sigma}_{\text{AVC}}^2$  can be obtained via

$$\arg \min_{\Lambda, \sigma^2} \|\mathbf{Z} \Lambda \mathbf{Z}^\top - \mathbf{S} + \sigma^2 \mathbf{I}\|_F^2. \quad (10)$$

Optimizing with respect to  $\Lambda$  yields

$$\Lambda_\star = \mathbf{Z}^\dagger (\mathbf{S} - \sigma^2 \mathbf{I}) \mathbf{Z}^{\dagger \top}, \quad (11)$$

where  $\mathbf{Z}^\dagger := (\mathbf{Z}^\top \mathbf{Z})^{-1} \mathbf{Z}^\top$ . The estimators are computed by substituting  $\Lambda_\star$  into (10) and optimizing with respect to  $\sigma^2$ :

$$\begin{aligned} \hat{\sigma}_{\text{AVC}}^2 &= \frac{\text{tr}[\mathbf{S} (\mathbf{I} - \mathbf{Z} \mathbf{Z}^\dagger)]}{n - q} \\ \hat{\Lambda}_{\text{AVC}} &= \mathbf{Z}^\dagger \mathbf{S} \mathbf{Z}^{\dagger \top} - \hat{\sigma}_{\text{AVC}}^2 (\mathbf{Z}^\top \mathbf{Z})^{-1}. \end{aligned} \quad (12)$$

Consider the parameterization  $\Lambda = \theta D$  in (Kang et al., 2008; Lippert et al., 2011) with a fixed symmetric positive semi-definite  $D$ , the solution to (10) is written as

$$\Lambda_* = \frac{\text{tr}(\mathbf{G}(\mathbf{S} - \sigma^2 \mathbf{I}))}{\text{tr}(\mathbf{G}^2)} \mathbf{D} \quad (13)$$

with  $\mathbf{G} = \mathbf{Z}\mathbf{D}\mathbf{Z}^\top$ . Substituting into (10), we obtain

$$\hat{\sigma}_{\text{AVC}}^2 = \frac{1}{n - \alpha} \left[ \text{tr}(\mathbf{S}) - \frac{\text{tr}(\mathbf{G}\mathbf{S})}{\text{tr}(\mathbf{G}^2)} \right], \quad (14)$$

where  $\alpha = \text{tr}(\mathbf{G})^2 / \text{tr}(\mathbf{G}^2)$ . Combined with (13), we arrive at

$$\hat{\Lambda}_{\text{AVC}} = \frac{\text{tr}(\mathbf{G}(\mathbf{S} - \hat{\sigma}_{\text{AVC}}^2 \mathbf{I}))}{\text{tr}(\mathbf{G}^2)} \mathbf{D}. \quad (15)$$

While AVCs may be used as variance components estimates under certain circumstances, the main purpose is to speed up estimating the fixed-effect coefficients. The complexity for computing the AVCs is  $O(n^3)$ , if  $\mathbf{S}$  is given. Like the dual fixed-effects estimator (6), the computational bottleneck of AVCs also lies in evaluating  $\mathbf{X}\Phi\mathbf{X}^\top$ .

## 4 FAST COMPUTATIONAL ALGORITHMS

In this section, we further improve the computational complexity  $O(n^2p)$  of the proposed approximate estimators in the high-dimensional setting  $p \gg n$ , where the computation bottleneck lies in evaluating the kernel  $\mathbf{X}\Phi\mathbf{X}^\top$ . We adopt the subsampled randomized Hadamard transform (SRHT) (Tropp, 2011) to compute the kernel matrix efficiently. In particular, the high-dimensional data is first projected into lower dimensions using SRHT, and the parameters of the LMM are then estimated using the projected data. However, there are two main challenges involved: 1) the estimated parameter  $\hat{\beta}$  now corresponds to the projected data of reduced dimensions, whereas the coefficients of the full original covariates are desired; and 2) the impact of applying the SRHT on the accuracy of parameter estimation needs to be justified. The techniques developed in this section recovers the coefficients to the full covariates from the SRHT projected data with high accuracy, as will be shown in § 5.

### 4.1 NON-ITERATIVE ALGORITHM FOR GENERAL LMMS

In this subsection, we provide a fast algorithm for parameter estimation in case of a general covariance matrix. Algorithm 1 takes as input the matrices  $\mathbf{X}$  and  $\Phi$

---

#### Algorithm 1 Approximate kernel matrix computation.

---

**Require:**  $\mathbf{X}$ ,  $\Phi$ , and error tolerance  $\epsilon$ .

- 1: Let  $p' = 2^{\lceil \log_2 p \rceil}$ , append  $p' - p$  all zero columns to  $\mathbf{X}$ , and  $p' - p$  all zero rows and columns to  $\Phi$ . Compute a diagonal matrix  $\mathbf{D}$  of dimension  $p'$  with Rademacher random diagonal elements.
- 2: Denote the fast Walsh-Hadamard transform by

$$\mathbf{W}_{p'} = \begin{bmatrix} \mathbf{W}_{p'/2} & \mathbf{W}_{p'/2} \\ \mathbf{W}_{p'/2} & -\mathbf{W}_{p'/2} \end{bmatrix} \quad \text{with } \mathbf{W}_1 = 1.$$

Let  $r$  be the rank of  $\mathbf{X}$  or  $r = n$  for unknown rank, then define

$$s_\epsilon := \frac{6 \left[ \sqrt{r} + \sqrt{8 \log(rp')} \right]^2 \log r}{\epsilon^2}.$$

Sample without replacement  $m$  rows of  $\mathbf{W}_{p'}\mathbf{D}/\sqrt{s_\epsilon}$  to obtain the SRHT  $\Pi$ . Compute  $\mathbf{A} = \mathbf{X}\sqrt{\Phi}\Pi^\top$ .

- 3: **return** the approximate kernel  $\mathbf{A}\mathbf{A}^\top$ ,  $\mathbf{A}$ , and  $\Pi$ .
- 

(which will be typically diagonal) and an approximation error  $\epsilon$  described in § 5. Both an approximation to the kernel matrix  $\mathbf{X}\Phi\mathbf{X}^\top$  and the SRHT matrix  $\Pi$  are computed. The computational efficiency of the algorithm is a result of replacing  $\mathbf{X}$  with the smaller transform  $\mathbf{A}$  in subsequent operations. Additionally, the structure of the SRHT allows for a divide-and-conquer scheme to compute  $\mathbf{A} = \mathbf{X}\sqrt{\Phi}\Pi^\top$  in  $O(np \log p)$  time. Note that the matrix  $\mathbf{W}_{p'}$  is not formed explicitly. The computation  $\mathbf{A}\mathbf{A}^\top$  requires  $O(n^2 s_\epsilon)$  time, which becomes dominant setting  $\epsilon \leq Cn\sqrt{\frac{\log n}{p \log p}}$  for some universal constant  $C$ . Thus, the overall runtime for the algorithm is  $O\left(\frac{n^3 \log n}{\epsilon^2}\right)$  for dense full-rank  $\mathbf{X}$ , and will be faster if  $\mathbf{X}$  is of low rank. The quality of the approximation depends on  $\epsilon$ , which will be discussed in § 5.

Given the approximate kernel, it is straight forward to compute the AVCs  $\Lambda_{\text{AVC}}$  and  $\sigma_{\text{AVC}}^2$  via (12). The coefficients for the fixed-effects can also be computed efficiently using the following estimator

$$\hat{\beta} = \sqrt{\Phi}\Pi^\top \mathbf{A}^\top \hat{\mathbf{V}}^{-1} \mathbf{L} \left( \mathbf{I} + \mathbf{A}\mathbf{A}^\top \hat{\mathbf{V}}^{-1} \mathbf{L} \right)^{-1} \mathbf{y}. \quad (16)$$

Given the approximate kernel matrix and  $\mathbf{A}$ , computing  $\mathbf{A}^\top \hat{\mathbf{V}}^{-1} \mathbf{L} \left( \mathbf{I} + \mathbf{A}\mathbf{A}^\top \hat{\mathbf{V}}^{-1} \mathbf{L} \right)^{-1} \mathbf{y}$  takes time  $O(\max\{n^2 s_\epsilon, n^3\})$  and multiplication of this vector by  $\sqrt{\Phi}\Pi^\top$  is  $O(p \log p)$  due to the structure of the SRHT matrix as well as the fact that  $\sqrt{\Phi}$  is diagonal. The resulting complexity in computing (16) is  $O(\max\{n^2 s_\epsilon, n^3, p \log p\})$ .

Approximating the kernel matrix using the SRHT was proposed for ridge regression in (Lu et al., 2013), a special case of our setting. A method for estimating the full set of fixed-effects coefficients was not provided in (Lu et al., 2013). Instead, a reduced set of  $m$  fixed-effects coefficients corresponding to the transformed covariate matrix  $\mathbf{X}\mathbf{\Pi}^\top$  was reported. For many applications, a major point of using an LMM is to estimate the effect-size of the fixed-effect coefficients, so computing  $\widehat{\boldsymbol{\beta}}$  is essential to the problem.

## 4.2 FAST EM FOR MULTI-GROUP LMMS

For efficient parameter estimation in  $L_2$ -regularized LMMS with repeated measurements, we extend the EM algorithm for the low-dimensional setting  $n \geq p$  (Laird et al., 1987) by combing the kernel estimators and Algorithm 1. While this high-dimensional EM variant is iterative, we show that the per-iteration computational cost is scalable in  $p$ .

The log-likelihood of the  $L_2$ -regularized LMM (17) can be rewritten in terms of class-specific variables as

$$\begin{aligned} & \log p(\mathbf{y}, \boldsymbol{\gamma}, \boldsymbol{\beta} \mid \mathbf{X}; \sigma^2, \boldsymbol{\Lambda}) \\ & \propto -\frac{1}{2} \boldsymbol{\beta}^\top \boldsymbol{\Phi}^{-1} \boldsymbol{\beta} - \frac{n}{2} \log \sigma^2 - \frac{m}{2} \log \det \mathbf{H} \\ & \quad - \frac{1}{2} \sum_{i=1}^m \boldsymbol{\gamma}^{(i)\top} \mathbf{H}^{-1} \boldsymbol{\gamma}^{(i)} - \frac{\mathbf{e}^\top \mathbf{e}}{2\sigma^2}, \end{aligned} \quad (17)$$

where  $\mathbf{e} = \mathbf{y} - c\mathbf{1} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\boldsymbol{\gamma}$ .

From the above log-likelihood, the posterior distribution of  $\boldsymbol{\beta}$  conditioned on the data and parameter estimates  $\widehat{\mathcal{P}} := \{\widehat{c}, \widehat{\sigma}^2, \widehat{\mathbf{H}}\}$  is multivariate normal with mean  $\boldsymbol{\Phi}\mathbf{X}^\top \widehat{\mathbf{M}}^{-1}(\mathbf{y} - \widehat{c}\mathbf{1})$  and covariance  $\boldsymbol{\Phi} - \boldsymbol{\Phi}\mathbf{X}^\top \widehat{\mathbf{M}}^{-1} \mathbf{X}\boldsymbol{\Phi}$ . Similarly, the posterior distribution of the vector of latent variables  $\boldsymbol{\gamma}$  is multivariate normal with mean  $\widehat{\boldsymbol{\Lambda}}\mathbf{Z}^\top \widehat{\mathbf{M}}^{-1}(\mathbf{y} - \widehat{c}\mathbf{1})$  and covariance  $\widehat{\boldsymbol{\Lambda}} - \widehat{\boldsymbol{\Lambda}}\mathbf{Z}^\top \widehat{\mathbf{M}}^{-1} \mathbf{Z}\widehat{\boldsymbol{\Lambda}}$ . Denote by  $\widehat{\boldsymbol{\gamma}}$  the mean of the posterior distribution of  $\boldsymbol{\gamma}$ , we also obtain the following posterior distributions of class-specific latent variable  $\boldsymbol{\gamma}^{(i)}$ :

$$\mathcal{N}\left(\widehat{\boldsymbol{\gamma}}^{(i)}, \widehat{\mathbf{H}} - \widehat{\mathbf{H}}\mathbf{Z}^{(i)\top} \left(\widehat{\mathbf{M}}^{-1}\right)^{(i)} \mathbf{Z}^{(i)} \widehat{\mathbf{H}}\right). \quad (18)$$

Note that  $\cdot^{(i)}$  represents the block matrix corresponding to group  $i$ . These posteriors are used in the E-step, discussed next.

**E-step** In the E-step, we derive the expectation of the log-likelihood (17) with respect to the aforementioned posterior distribution of  $\boldsymbol{\beta}$  and  $\boldsymbol{\gamma}^{(i)}$ :

$$\mathbb{E}_{\boldsymbol{\beta}, \boldsymbol{\gamma} \mid \mathbf{y}, \widehat{\mathcal{P}}} \left[ \log p(\mathbf{y}, \boldsymbol{\gamma}, \boldsymbol{\beta} \mid \mathbf{X}; \sigma^2, \mathbf{H}) \right].$$

We only need to consider terms in the expectation that involve  $c$ ,  $\sigma^2$ , and  $\mathbf{H}$ . Denote by  $\widehat{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}^{(i)}}$  the variance of (18), the following holds  $\mathbb{E}_{\boldsymbol{\beta}, \boldsymbol{\gamma} \mid \mathbf{y}, \widehat{\mathcal{P}}} (\boldsymbol{\gamma}^{(i)\top} \mathbf{H}^{-1} \boldsymbol{\gamma}^{(i)}) = \widehat{\boldsymbol{\gamma}}^{(i)\top} \mathbf{H}^{-1} \widehat{\boldsymbol{\gamma}}^{(i)} + \text{tr}(\widehat{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}^{(i)}} \mathbf{H}^{-1})$ . Using the previously derived posterior distributions, we get

$$\begin{aligned} \mathbb{E}(\mathbf{e} \mid \mathbf{y}, \widehat{\mathcal{P}}) &= \mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}} - \mathbf{Z}\widehat{\boldsymbol{\gamma}} - \widehat{c}\mathbf{1} \\ \text{cov}(\mathbf{e} \mid \mathbf{y}, \widehat{\mathcal{P}}) &= \widehat{\sigma}^2 \mathbf{I} - \widehat{\sigma}^4 \widehat{\mathbf{M}}^{-1}. \end{aligned}$$

Thus, we arrive at  $\mathbb{E}_{\boldsymbol{\beta}, \boldsymbol{\gamma} \mid \mathbf{y}, \widehat{\mathcal{P}}} (\mathbf{e}^\top \mathbf{e}) = \widehat{\mathbf{e}}^\top \widehat{\mathbf{e}} + \widehat{\sigma}^2 \mathbf{I} - \widehat{\sigma}^4 \widehat{\mathbf{M}}^{-1}$ , where  $\widehat{\mathbf{e}} := \mathbb{E}(\mathbf{e} \mid \mathbf{y}, \widehat{\mathcal{P}})$ .

**M-step** We now update the parameter estimates with the maximizers of the expectation from the E-step. First, observe that the  $\boldsymbol{\beta}$  estimate from the posterior distribution is the same as the the dual estimator developed in § 3. To maximize the expectation with respect to  $\mathbf{H}$  and  $\sigma^2$ , we take the partial derivatives with respect to  $\mathbf{H}^{-1}$  and  $\sigma^{-2}$ , and set them to zero. This gives the following M-step updates:

$$\begin{aligned} \widehat{\mathbf{H}} &\leftarrow \frac{1}{m} \sum_{i=1}^m \left( \widehat{\boldsymbol{\gamma}}^{(i)} \widehat{\boldsymbol{\gamma}}^{(i)\top} + \widehat{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}^{(i)}} \right) \\ \widehat{\sigma}^2 &\leftarrow \widehat{\sigma}^2 + \frac{1}{n} \left[ \widehat{\mathbf{e}}^\top \widehat{\mathbf{e}} - \widehat{\sigma}^4 \text{tr}(\widehat{\mathbf{M}}^{-1}) \right]. \end{aligned} \quad (19)$$

The fast version of the above EM variant uses Algorithm 1 for computing the kernel. Note that the original  $\mathbf{X}$  is no longer needed after the SRHT projection. This provides additional space advantages as data  $\mathbf{X}$  can be preprocessed, and the Hadamard transform in Step 1 requires a small constant amount of memory. Overall, the per-iteration computational complexity of the EM algorithm is  $O(\max\{n^2 s_\epsilon, n^3\})$ .

## 5 THEORETICAL GUARANTEES

In this section, we provide an analysis of the difference in the parameters estimated via the approximate algorithms versus minimizing the  $L_2$ -regularized LMM. We are not proving consistency of our estimator—convergence of the parameter estimates to the population quantity. Consistency results for LMMS and-regularized LMMS were provided in (Schelldorfer et al., 2011; Cui et al., 2004; Hall and Yao, 2003). See the supplementary materials for proofs of the theorems in this section.

**Theorem 1** (Fixed-effect norm error). *Let  $\widehat{\boldsymbol{\beta}}$  be the fixed-effect coefficients estimated by (5) and  $\widehat{\boldsymbol{\beta}}'$  be the fixed-effect coefficients estimated by the approximate proce-*

ture in (16). Then, with probability at least  $1 - 3/n$

$$\frac{\|\widehat{\boldsymbol{\beta}} - \widehat{\boldsymbol{\beta}}'\|}{\|\widehat{\boldsymbol{\beta}}\|} \leq \frac{\epsilon}{1 - \epsilon} \frac{\|\boldsymbol{\Phi}^{-1}\|_2 \kappa(\boldsymbol{\Gamma})}{\frac{\|\boldsymbol{\Phi}\|_2^{-1}}{1 + \sqrt{2/3}\epsilon} + \lambda_{\min}(\mathbf{X}^\top \mathbf{V}^{-1} \mathbf{X})}$$

with  $\boldsymbol{\Gamma} := \boldsymbol{\Phi}^{-1} + \mathbf{X}^\top \mathbf{V}^{-1} \mathbf{X}$ , or loosely

$$\frac{\|\widehat{\boldsymbol{\beta}} - \widehat{\boldsymbol{\beta}}'\|}{\|\widehat{\boldsymbol{\beta}}\|} \leq \frac{\epsilon(1 + \sqrt{2/3}\epsilon)}{1 - \epsilon} \kappa(\boldsymbol{\Phi}) \kappa(\boldsymbol{\Gamma})$$

for all  $0 \leq \epsilon < 1$ .

An intuitive interpretation of the theorem is that the fixed-effects coefficients estimator (16) has better accuracy when the predefined  $\boldsymbol{\Phi}$  is better conditioned and has smaller spectral norm. One can certainly improve the accuracy by setting a smaller  $\epsilon$ , which in turn uses more samples in Algorithm 1.

**Theorem 2** (AVC approximation errors). *Let  $\sigma_{AVC}^2$  and  $\widehat{\boldsymbol{\Lambda}}_{AVC}$  be computed using (12). Let  $\widehat{\sigma}_{AVC}'^2$  and  $\widehat{\boldsymbol{\Lambda}}'_{AVC}$  be computed using the same equations but with approximate kernel from Algorithm 1. Then, the following two statements hold jointly with probability at least  $1 - 3/n$ :*

$$\begin{aligned} |\widehat{\sigma}_{AVC}^2 - \widehat{\sigma}_{AVC}'^2| &\leq \epsilon \cdot \frac{\|\|\mathbf{X} \boldsymbol{\Phi} \mathbf{X}^\top\|\|_{n-q}}{n-q} \quad \text{and} \\ \|\widehat{\boldsymbol{\Lambda}}_{AVC} - \widehat{\boldsymbol{\Lambda}}'_{AVC}\|_2 &\leq \frac{\epsilon}{\widehat{\sigma}_{\min}(\mathbf{Z})^2} \left( \|\|\mathbf{X} \boldsymbol{\Phi} \mathbf{X}^\top\|\|_2 + \frac{\|\|\mathbf{X} \boldsymbol{\Phi} \mathbf{X}^\top\|\|_{n-q}}{n-q} \right). \end{aligned}$$

Note that the fraction of the Ky Fan norm does not exceed the spectral norm. A looser but more convenient bounds are  $|\widehat{\sigma}_{AVC}^2 - \widehat{\sigma}_{AVC}'^2| \leq \epsilon \|\|\mathbf{X} \boldsymbol{\Phi} \mathbf{X}^\top\|\|_2$  and  $\|\widehat{\boldsymbol{\Lambda}}_{AVC} - \widehat{\boldsymbol{\Lambda}}'_{AVC}\|_2 \leq 2\epsilon \sigma_{\min}(\mathbf{Z})^{-2} \|\|\mathbf{X} \boldsymbol{\Phi} \mathbf{X}^\top\|\|_2$ .

## 6 EXPERIMENTS

In this section, we conduct a simulation study as well as numerical experiments on real data. The simulation study demonstrates the accuracy of parameter estimation using the proposed Approximate Ridge LMM (arLMM) methods. We also examined the results on a real data example from the Wellcome Trust Case Control Consortium (WTCCC) study (The Wellcome Trust Case Control Consortium, 2007), which include about 14,000 cases from seven common diseases and a total of about 450,000 SNPs.

The main finding of the experiments is that the proposed approximate inference algorithms enjoy similar

predictive accuracy as state-of-the-art methods at a significantly reduced computation cost in practice. In particular, our Matlab prototype implementation is 6x faster than the optimized C implementation of the state-of-the-art BSLMM method for genome-wide association studies.

### 6.1 SIMULATION STUDIES

To evaluate parameter estimation, we consider two performance metrics. The first one is the correlation between the estimated and ground-truth fixed-effect coefficients. The second metric is the Negative Log Likelihood (NLL) of the standard LMM, which meaningfully reflects the quality of variance estimation.

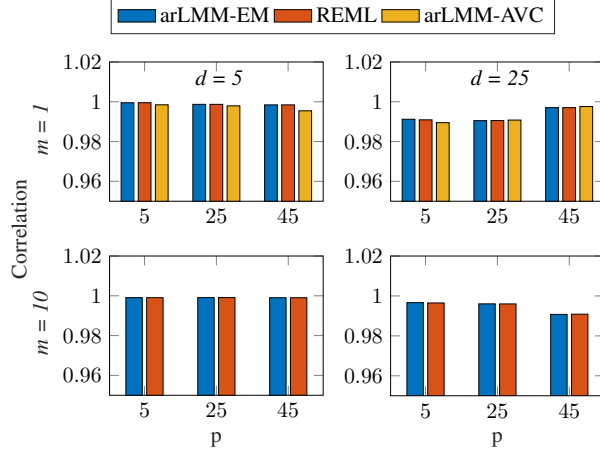
For the simulation, we compare the performance of our non-iterative algorithm arLMM-AVC based on (16) and (12), the proposed multi-group variant arLMM-EM based on (19), the standard REML (Bates et al., 2015),  $L_1$ -regularized LMM lmmlasso (Schelldorfer et al., 2011), and CovexLasso using both  $L_1$ - and  $L_2$ -regularization (Jakubik, 2015).

**Synthetic data generation** The simulation is based on synthetic training and validation sets sampled from a fixed LMM distribution. The design matrices as well as the parameters for the fixed LMM are randomly generated. Specifically,

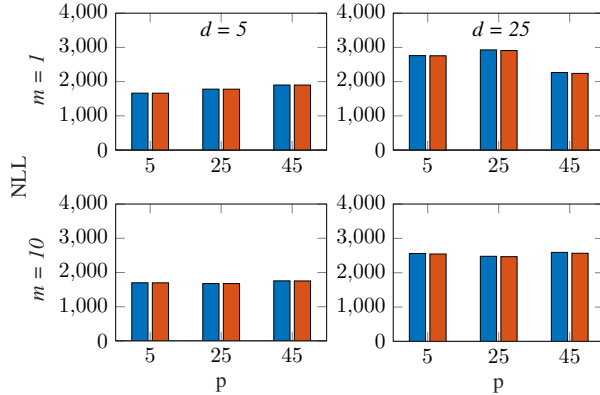
$$\begin{aligned} X_{ij} &\stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1) & Z_{ij}^{(k)} &\stackrel{\text{i.i.d.}}{\sim} \mathcal{U}(0, 1) \\ \boldsymbol{\gamma}^{(k)} &\stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{K}^\top \mathbf{K}) & K_{ij} &\stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1) \\ \boldsymbol{\beta} &\sim \mathcal{N}(\mathbf{0}, \mathbf{I}) & \sigma^2 &\sim \mathcal{U}(0, d). \end{aligned}$$

Note that there are  $d$  random-effect variables with covariance  $\mathbf{K}^\top \mathbf{K}$ . Thus, the random-effect design matrix  $\mathbf{Z} \in \mathbb{R}^{n \times q}$ ,  $q = md$ , will be block-diagonal with diagonal blocks  $\mathbf{Z}^{(k)}$ . Given the number of observations  $n$ , we randomly sample  $n_k$  observations for each group  $k$ , where the fractions  $n_k/n$  are specified by the Dirichlet distribution with the concentration parameters  $(1, 1, \dots)^\top$ .

**Overdetermined settings** Let us first consider the standard setting  $n > p$ , which are supported by many parameter estimation algorithms of LMMs. We evaluate the performance of arLMM-AVC and arLMM-EM in a variety of  $p$ ,  $d$ , and  $m$  settings. The parameter estimates obtained using the proposed methods are compared with the estimates given by the standard REML (see e.g., (Kang et al., 2008; Lippert et al., 2011; Laird et al., 1987; Bates et al., 2015)) which is known to produce unbiased estimates.



(a) Correlation between estimated and true  $\beta$ 's.



(b) NLL at convergence.

Figure 1: Comparing the performance of parameter estimation on synthetic data with  $n = 1,000$  observations. Note that arLMM-AVC is only applicable to the single group setting. This figure shows that the arLMM-EM and arLMM-AVC achieve comparable estimation performance as REML.

Figure 1 shows the error for the fitted parameters using 1,000 observations sampled from the underlying LMM. The average results are reported over 10 runs on independently generated datasets. These generated datasets have the same number of observations  $n = 1,000$  but different settings of  $p$ ,  $d$ , and  $m$ .

As shown in Figure 1, arLMM-EM and arLMM-AVC exhibit comparable estimation accuracy as the standard REML. Note that arLMM-AVC is applicable only when  $m = 1$  (the first row of Figure 1). Since arLMM-AVC is based on non-iterative approximation to the variance components, the error is slighted higher than the others as expected.

**High-dimensional (underdetermined) setting** We also examined the performance of our model in the high-

dimensional setting where we are interested in variable selection based on the fixed-effects coefficients. In Table 2, we specify the three regimes for which we generate simulated data: an overdetermined LMM, a moderate-dimensional LMM, and a high-dimensional LMM. Each regime is characterized by  $n$ ,  $p$ ,  $d$ , and  $m$ , and an extra parameter  $s$ , the number of non-zeros in the ground-truth  $\beta_{\text{True}}$ . Since  $m > 1$  we did not apply arLMM-AVC.

Table 2: Regimes of data.

	$(n, p, d, m, s)$
LOW	$(100, 1000, 5, 3, 10)$
MOD	$(200, 10^4, 5, 3, 10)$
HIGH	$(10^4, 10^6, 10, 100, 100)$

Figure 2 reports variable selection results for arLMM-EM, lmmlasso (Schelldorfer et al., 2011), and ConvexLasso. All the settings in Table 2 have sparse ground-truth  $\beta_{\text{True}}$ . Figure 2 shows the fraction of the signal (non-zeros in  $\beta_{\text{True}}$ ) recovered in the estimate  $\hat{\beta}$ . We varied the regularization parameters to obtain  $\hat{\beta}$  with different sparsity  $\|\hat{\beta}\|_0$ . The entries with the largest magnitude of  $\hat{\beta}$  is considered the signal in these evaluations. As can be seen, arLMM-EM delivers a competitive signal recovery ratio for  $p = 10^3, 10^4$ , and scales to considerably large dimensions  $n = 10^4$  and  $p = 10^6$ , which the other two methods cannot handle.

## 6.2 GENOME WIDE ASSOCIATION STUDIES

LMMs have been used extensively for mapping traits in statistical genetics. The problem formulation is that of regressing a quantitative or categorical trait onto a high-dimensional vector of 450,000 single nucleotide polymorphisms (SNPs), or locations of discrete genetic variation, for each subject included in the study. The random effects are driven by population structure or the pairwise similarity or relatedness between individuals.

We compare our approximate estimator to the performance of a state-of-the-art estimator called BSLMM (Bayesian sparse linear mixed model) (Zhou et al., 2013). Specifically, we run BSLMM in its ridge-regression with mixed models setting, the fastest setting of the package for a fair comparison. In this setting BSLMM is computing the maximum a posteriori estimate of the regularized LMM. We compare performance on the Wellcome Trust Case Control Consortium (WTCCC) dataset of 14,000 cases of 7 diseases - bipolar disorder (BD), coronary artery disease (CAD), Crohn's disease (CD), hypertension (HT), rheumatoid arthritis (RA), type 1 diabetes (T1D), and type 2 diabetes (T2D) - and 3,000 shared controls. This dataset characterizes over 450,000 single nu-

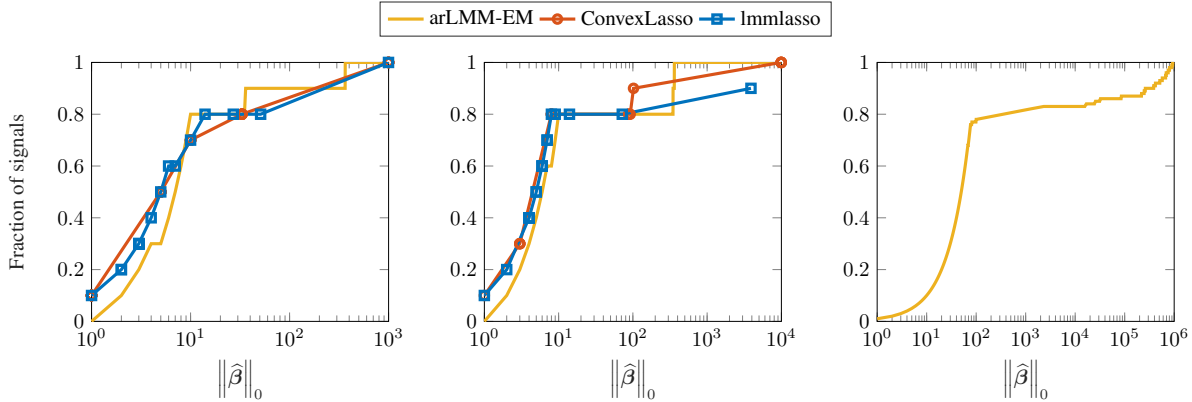


Figure 2: Fraction of signals captured by  $\hat{\beta}$ . From left to right, the configurations are respectively LOW, MOD, and HIGH in Table 2. It shows that arLMM-EM performs competitively in variable selection.

Table 3: Comparing the prediction performance as well as the runtime of BSLMM and arLMM-AVC on the WTCCC dataset.  $\text{Corr}(\hat{\beta}_{\text{BSLMM}}, \hat{\beta}_{\text{arLMM-AVC}})$  denotes the correlation between the fixed-effect coefficient estimates given by BSLMM and arLMM-AVC.

DISEASE	TIME (MIN)		AUC		CORR ( $\hat{\beta}_{\text{BSLMM}}, \hat{\beta}_{\text{arLMM-AVC}}$ )
	BSLMM	arLMM-AVC	BSLMM	arLMM-AVC	
BD	115.8	25.1	0.6520	0.6461	0.9898
CAD	161.0	26.1	0.5899	0.5937	0.9776
CD	110.3	25.4	0.6260	0.6328	0.9862
HT	120.6	19.4	0.5956	0.6010	0.9766
RA	147.4	19.9	0.6173	0.6206	0.9834
T1D	120.0	20.4	0.6846	0.6840	0.9939
T2D	155.3	18.9	0.6003	0.5993	0.9783

cleotide polymorphisms (SNPs), or locations of discrete genetic variation, for each subject included in the study. Disease status is indicated as a binary response (1 for disease case,  $-1$  for control). Each of the datasets had roughly equal numbers of cases and controls.

For this experiment, we adopted the same random-effect covariance parameterization used to control for population structure  $\theta \mathbf{X} \mathbf{X}^T / p$  as BSLMM, and used arLMM-AVC with AVCs (15) and (14). arLMM-AVC and BSLMM were run under identical conditions on each of the seven approximately 5,000-subject  $\times$  450,000-SNP datasets. This was the same experimental setup used to validate BSLMM in (Zhou et al., 2013).

Observed runtimes for each of the seven datasets are reported in Table 3. Correlation between the  $\hat{\beta}$  reported by arLMM-AVC and BSLMM in all cases was very high, 0.977 or greater.

We also compared disease status prediction by splitting each dataset into a training set comprised of 80% of subjects and a test set of the remaining 20%, selected at random. arLMM-AVC and BSLMM each estimated  $\hat{\beta}$  from the training set and attempted to predict disease status on the held-out set. We repeated this 20 times for each of the

seven datasets and evaluated performance of prediction on the held-out set by area under the ROC curve (AUC). These results are also given in Table 3. Predictive performance by arLMM-AVC and BSLMM was almost identical. Predicting disease status from genetic markers is hard and it is well known that the effect sizes of genetic variants are individually small and that a great deal of variance in the response will also be driven by environmental factors.

## 7 CONCLUSIONS

State-of-the-art parameter inference in LMMs requires computational complexity which depends at least linearly on the number of covariates  $p$  and generally relies on heuristics. In this paper, we presented scalable learning algorithms which have sublinear computational complexity in  $p$  and provide theoretical guarantees for the accuracy of parameter estimation. Our approach combines novel approximate estimators that use a kernel matrix of the observations and the subsampled randomized Hadamard transform. Experiments on synthetic and real data corroborate the theory.

## References

- D. Bates, M. Mächler, B. Bolker, and S. Walker. Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, 67(1):1–48, 2015.
- Y. Chung, S. Rabe-Hesketh, V. Dorie, A. Gelman, and J. Liu. A nondegenerate penalized likelihood estimator for variance parameters in multilevel models. *Psychometrika*, 78(4):685–709, 2013.
- H. Cui, K. W. Ng, and L. Zhu. Estimation in mixed effects model with errors in variables. *Journal of Multivariate Analysis*, 91(1):53–73, 2004.
- G. Darnell, S. Georgiev, S. Mukherjee, and B. E. Engelhardt. Adaptive randomized dimension reduction on massive data. *JMLR*, Apr. 2017.
- E. Demidenko. *Mixed Models: Theory and applications with R*. Wiley, 2nd edition, 2013.
- P. Hall and Q. Yao. Inference in components of variance models with low replication. *Annals of Statistics*, 31(2):414–441, 2003.
- D. A. Harville. Bayesian inference for variance components using only error contrasts. *Biometrika*, 61:383–385, 1974.
- J. Jakubík. Convex method for variable selection in high-dimensional linear mixed models. In *Proceedings of the 10th International Conference on Measurement*, pages 55–58, 2015.
- H. M. Kang, N. A. Zaitlen, C. M. Wade, A. Kirby, D. Heckerman, M. J. Daly, and E. Eskin. Efficient control of population structure in model organism association mapping. *Genetics*, 178:1709–23, Mar 2008.
- N. Laird, N. Lange, and D. Stram. Maximum likelihood computations with repeated measures: Application of the EM algorithm. *Journal of the American Statistical Association*, 82:97–105, 1987.
- N. M. Laird and J. H. Ware. Random-effects models for longitudinal data. *Biometrics*, 38(4):963–974, Dec. 1982.
- M. J. Lindstrom and D. M. Bates. Newton-Raphson and EM algorithms for linear mixed-effects models for repeated-measures data. *Journal of the American Statistical Association*, 83:1014–1022, 1988.
- C. Lippert, J. Listgarten, Y. Liu, C. Kadie, R. Davidson, and D. Heckerman. Fast linear mixed models for genome-wide association studies. *Nature Methods*, 8(10):833–835, Oct. 2011.
- Y. Lu, P. Dhillon, D. P. Foster, and L. Ungar. Faster ridge regression via the subsampled randomized Hadamard transform. In *NIPS*, pages 369–377. 2013.
- H. D. Patterson and R. Thompson. Recovery of inter-block information when block sizes are unequal. *Biometrika*, 58:545–554, 1971.
- P. O. Perry. Fast moment-based estimation for hierarchical models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(1):267–291, 2017.
- C. Saunders, A. Gammerman, and V. Vovk. Ridge regression learning algorithm in dual variables. In *ICML*, pages 515–521, 1998.
- J. Schelldorfer, P. Bühlmann, and S. V. De Geer. Estimation for high-dimensional linear mixed-effects models using  $\ell_1$ -penalization. *Scandinavian Journal of Statistics*, 38(2):197–214, 2011.
- J. Schelldorfer, L. Meier, and P. Bühlmann. GLMM-Lasso: An algorithm for high-dimensional generalized linear mixed models using  $\ell_1$ -penalization. *Journal of Computational and Graphical Statistics*, 23(2):460–477, 2014.
- The Wellcome Trust Case Control Consortium. Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls. *Nature*, 447(7145):661–678, June 2007.
- J. A. Tropp. Improved analysis of the subsampled randomized Hadamard transform. *Advances in Adaptive Data Analysis*, 3(1-2):115–126, 2011.
- X. Zhang, Y. Zhou, Y. Ma, B. Chen, L. Zhang, and D. Agarwal. GLMix: Generalized linear mixed models for large-scale response prediction. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 363–372, 2016.
- Z. Zhang, G. Dai, and M. I. Jordan. Bayesian generalized kernel mixed models. *JMLR*, 12:111–139, Feb. 2011.
- X. Zhou. A unified framework for variance component estimation with summary statistics in genome-wide association studies. *The Annals of Applied Statistics*, 11(4):2027–2051, 2017.
- X. Zhou and M. Stephens. Efficient multivariate linear mixed model algorithms for genome-wide association studies. *Nature Methods*, 2014.
- X. Zhou, P. Carbonetto, and M. Stephens. Polygenic modeling with Bayesian sparse linear mixed models. *PLOS Genetics*, 9(2):1–14, 02 2013.



---

# Constraint-based Causal Discovery for Non-Linear Structural Causal Models with Cycles and Latent Confounders

---

**Patrick Forré**  
Informatics Institute  
University of Amsterdam  
The Netherlands  
p.d.forre@uva.nl

**Joris M. Mooij**  
Informatics Institute  
University of Amsterdam  
The Netherlands  
j.m.mooij@uva.nl

## Abstract

We address the problem of causal discovery from data, making use of the recently proposed causal modeling framework of *modular structural causal models (mSCM)* to handle cycles, latent confounders and non-linearities. We introduce  $\sigma$ -connection graphs ( $\sigma$ -CG), a new class of mixed graphs (containing undirected, bidirected and directed edges) with additional structure, and extend the concept of  $\sigma$ -separation, the appropriate generalization of the well-known notion of d-separation in this setting, to apply to  $\sigma$ -CGs. We prove the closedness of  $\sigma$ -separation under marginalisation and conditioning and exploit this to implement a test of  $\sigma$ -separation on a  $\sigma$ -CG. This then leads us to the first causal discovery algorithm that can handle non-linear functional relations, latent confounders, cyclic causal relationships, and data from different (stochastic) perfect interventions. As a proof of concept, we show on synthetic data how well the algorithm recovers features of the causal graph of modular structural causal models.

## 1 INTRODUCTION

Correlation does not imply causation. To go beyond spurious probabilistic associations and infer the asymmetric causal relations we need sufficiently powerful models. Structural causal models (SCMs), also known as structural equation models (SEMs), provide a popular modeling framework (see [12, 25, 26, 32]) that is up to this task. Still, the problem of causal discovery from data is notoriously hard. Theory and algorithms need to address several challenges like probabilistic settings, stability under interventions, combining observational and

interventional data, latent confounders and marginalisation, selection bias and conditioning, faithfulness violations, cyclic causation like feedback loops and pairwise interactions, and non-linear functional relations in order to go beyond artificial simulation settings and become successful on real-world data.

Several algorithms for causal discovery have been introduced over the years. For the acyclic case without latent confounders, numerous constraint-based [25,32] and score-based approaches [6, 14, 17] exist. More sophisticated constraint-based [5, 25, 32, 34] and score-based approaches [4, 7, 8, 10, 11] can deal with latent confounders in the acyclic case. For the linear cyclic case, most algorithms assume no latent confounders [18, 27, 29], though some of the more recent ones allow for those [19, 30]. To the best of our knowledge, no algorithms have yet been proposed for the general non-linear cyclic case.

In this work we present a novel conditional independence constraint-based causal discovery algorithm that—up to the knowledge of the authors—is the first causal discovery algorithm that addresses most of the previously mentioned problems at once, notably non-linearities, cycles, latent confounders, and multiple interventional data sets, only excluding selection bias and faithfulness violations.

For this to work we build upon the theory of *modular structural causal models (mSCM)* introduced in [12]. mSCMs form a general and convenient class of structural causal models that can deal with cycles and latent confounders. The measure-theoretically rigorous presentation opens the door for general non-linear measurable functions and any kind of probability distributions (e.g. mixtures of discrete or continuous ones). mSCM are provably closed under any combination of perfect interventions and marginalisations (see [12]).

Unfortunately, it is known that the direct generalization of the *d-separation criterion* (also called m- or m\*-separation, see [9, 24, 25, 28, 33]), which relates the conditional independencies of the model to its underlying

graphical structure, does not apply in general if the structural equations are *non-linear* and the graph contains *cycles* (see [12, 31] or example 2.17).

Luckily, one key property of mSCMs is that the variables of the mSCM always entail the conditional independences implied by  $\sigma$ -separation, a non-naive generalization of the d/m/m\*-separation (see [12]), which also works in the presence of cycles, non-linearities and latent confounders, and reduces to d-separation in the acyclic case.

To prove the  $\sigma$ -separation criterion, the authors of [12] have constructed an extensive theory for directed graphs with hyperedges. As a first contribution in this paper we give a simplified but equivalent definition of mSCMs plainly in terms of directed graphs and prove the  $\sigma$ -separation criterion directly under weaker assumptions.

As a second contribution we extend the definition of  $\sigma$ -separation to mixed graphs (including also bi- and undirected edges) by introducing additional structure. We will refer to this class of mixed graphs as  $\sigma$ -connection graphs ( $\sigma$ -CG), since they are inspired by the d-connection graphs introduced by [19]. We prove that  $\sigma$ -CGs and  $\sigma$ -separation are closed under marginalisation and conditioning, in analogy with the d-connection graphs (d-CG) from [19].

The work of [19] provides an elegant approach to causal discovery using a weighted SAT solver to find the causal graph that is most compatible with (weighted) conditional independences in the data, encoding the notion of d-separation into *answer set programming (ASP)*, a declarative programming language with an expressive syntax for implementing discrete or integer optimization problems. Our third contribution is to adapt the approach of [19] by replacing d-separation by  $\sigma$ -separation and d-CGs by  $\sigma$ -CGs. The results mentioned above will then ensure all the needed properties to make the adapted algorithm of [19] applicable to general mSCMs, i.e., to non-linear causal models with cycles and latent confounders, under the additional assumptions of *no selection bias* and of  $\sigma$ -faithfulness.

Finally, as a proof of concept, we will show the effectiveness of our proposed algorithm in recovering features of the causal graphs of mSCMs from simulated data.

## 2 THEORY

### 2.1 MODULAR STRUCTURAL CAUSAL MODELS

Structural causal or equation models (SCM/SEM) usually start with a set of variables  $(X_v)_{v \in V}$  attached to a

graph  $G$  that satisfy (or are even defined by) equations of the form:

$$X_v = g_{\{v\}}(X_{\text{Pa}^G(v)}, E_v),$$

with a function  $g_{\{v\}}$  and noise variable  $E_v$  attached to each node  $v \in V$ . Here  $\text{Pa}^G(v)$  denotes the set of (direct causal) parents of  $v$ . In *linear* models the functions  $g_{\{v\}}$  are linear functions, in *acyclic* models the nodes of  $V$  form an acyclic graph  $G$ , and under *causal sufficiency* the variables  $E_v$  are independent (i.e. “no latent confounders”). The functions  $g_{\{v\}}$  are usually interpreted as *local causal mechanisms* that produce the values of  $X_v$  from the values of  $X_{\text{Pa}^G(v)}$  and  $E_v$ . These local mechanisms  $g_{\{v\}}$  are—in the causal setting—assumed to be stable even when one intervenes upon some of the variables, i.e. one makes a causal *local compatibility* assumption. One important observation now is that one can also consider the *global mechanism*  $g$  that maps the values of the latent variables  $(E_v)_{v \in V}$  to the values of the observed variables  $(X_v)_{v \in V}$ . The assumption of acyclicity or invertible linearity will then guarantee the *global compatibility* of all these mechanisms  $g_{\{v\}}$  and  $g$ . However, if we now abstain from assuming acyclicity or linearity, the global compatibility does not follow from the local compatibility anymore (see figure 1). So in a general consistent causal setting this needs to be *guaranteed or assumed*.

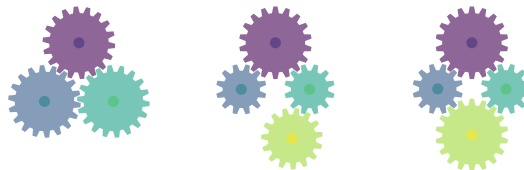


Figure 1: Gear analogy. Left: A cyclic mechanism that is locally compatible but not globally. Center: For acyclic mechanisms local compatibility implies global compatibility. Right: A cyclic mechanism that is locally and globally compatible. This shows that the assumption of global compatibility is needed when cycles are present.

The definition of modular structural causal models (mSCM) and the mentioned list of desirable properties basically follow directly from *causal* postulates:

**Postulate 2.1** (Causal postulates). *The observed world appears as the projection of an extended world such that:*

1. All latent and observed variables in this extended world are causally linked by a directed graph.
2. Every subsystem of this extended world can be expressed as the joint effect of its joint direct causes.
3. All these mechanisms are globally compatible.

Special subsystems of interest are the loops of a graph.

**Definition 2.2** (Loops). *Let  $G = (V, E)$  be a directed graph (with or without cycles).*

1. A loop of  $G$  is a set of nodes  $S \subseteq V$  such that for every two nodes  $v_1, v_2 \in S$  there are two directed paths  $v_1 \rightarrow \dots \rightarrow v_2$  and  $v_2 \rightarrow \dots \rightarrow v_1$  in  $G$  with all the intermediate nodes also in  $S$  (if any). The single element sets  $S = \{v\}$  are also considered as loops.
2. The strongly connected component of  $v$  in  $G$  is defined to be:

$$\text{Sc}^G(v) := \text{Anc}^G(v) \cap \text{Desc}^G(v),$$

*the set of nodes that are both ancestors and descendants of  $v$  (including  $v$  itself).*

3. Let  $\mathcal{L}(G) := \{S \subseteq G \mid S \text{ a loop of } G\}$  be the loop set of  $G$ .

**Remark 2.3.** *Note that the loop set  $\mathcal{L}(G)$  contains all single element loops  $\{v\} \in \mathcal{L}(G)$ ,  $v \in V$ , as the smallest loops and all strongly connected components  $\text{Sc}^G(v) \in \mathcal{L}(G)$ ,  $v \in V$ , as the largest loops, but also all non-trivial intermediate loops  $S$  with  $\{v\} \subsetneq S \subsetneq \text{Sc}^G(v)$  inside the strongly connected components (if existent). If  $G$  is acyclic then  $\mathcal{L}(G)$  only consists of the single element loops:  $\mathcal{L}(G) = \{\{v\} \mid v \in V\}$ .*

The definition of mSCM is made in such a way that it will automatically incorporate the causal postulates 2.1. In the following, all spaces are meant to be equipped with  $\sigma$ -algebras, forming standard measurable spaces, and all maps to be measurable.

**Definition 2.4** (Modular Structural Causal Model, [12]). *A modular structural causal model (mSCM) by definition consists of:*

1. a set of nodes  $V^+ = U \dot{\cup} V$ , where elements of  $V$  correspond to observed variables and elements of  $U$  to latent variables,
2. an observation/latent space  $\mathcal{X}_v$  for every  $v \in V^+$ ,  $\mathcal{X} := \prod_{v \in V^+} \mathcal{X}_v$ ,
3. a product probability measure  $\mathbb{P} := \mathbb{P}_U = \otimes_{u \in U} \mathbb{P}_u$  on the latent space  $\prod_{u \in U} \mathcal{X}_u$ ,<sup>1</sup>

<sup>1</sup>The assumption of independence of the noise variables here is not to be confused with causal sufficiency. The noise variables here might have two or more child nodes and thus can play the role of latent confounders. The independence assumption here also does not restrict the model class. If they were dependent we would just consider them as one variable and use a different graph that encoded this.

4. a directed graph structure  $G^+ = (V^+, E^+)$  with the properties:<sup>2</sup>

$$(a) V = \text{Ch}^{G^+}(U),$$

$$(b) \text{Pa}^{G^+}(U) = \emptyset,<sup>3</sup>$$

$$(c) \text{Ch}^{G^+}(u_1) \not\subseteq \text{Ch}^{G^+}(u_2) \text{ for every two distinct } u_1, u_2 \in U,<sup>3</sup>$$

where  $\text{Ch}^{G^+}$  and  $\text{Pa}^{G^+}$  stand for children and parents in  $G^+$ , resp.,

5. a system of structural equations  $g = (g_S)_{S \in \mathcal{L}(G^+): S \subseteq V}$ :

$$g_S : \prod_{v \in \text{Pa}^{G^+}(S) \setminus S} \mathcal{X}_v \rightarrow \prod_{v \in S} \mathcal{X}_v,<sup>24</sup>$$

that satisfy the following global compatibility conditions: For every nested pair of loops  $S' \subseteq S \subseteq V$  of  $G^+$  and every element  $x_{\text{Pa}^{G^+}(S) \cup S} \in \prod_{v \in \text{Pa}^{G^+}(S) \cup S} \mathcal{X}_v$  we have the implication:

$$\begin{aligned} g_S(x_{\text{Pa}^{G^+}(S) \setminus S}) &= x_S \\ \implies g_{S'}(x_{\text{Pa}^{G^+}(S') \setminus S'}) &= x_{S'}, \end{aligned}$$

where  $x_{\text{Pa}^{G^+}(S') \setminus S'}$  and  $x_{S'}$  denote the corresponding components of  $x_{\text{Pa}^{G^+}(S) \cup S}$ .

The mSCM can be summarized by the tuple  $M = (G^+, \mathcal{X}, \mathbb{P}, g)$ .

**Remark 2.5.** *Given the mechanisms attached to the nodes  $g_{\{v\}}$  the existence (and compatibility) of the other mechanisms  $g_S$  for non-trivial loops  $S$  can be guaranteed under certain conditions, e.g. trivially in the acyclic case, or if every cycle is contractive (see subsection 4.1), or more generally if the cycles are “uniquely solvable” (see [3, 12]).*

We are now going to define the actual random variables  $(X_u)_{u \in V^+}$  attached to any mSCM.

**Remark 2.6.** *Let  $M = (G^+, \mathcal{X}, \mathbb{P}, g)$  be a mSCM with  $G^+ = (U \dot{\cup} V, E^+)$ .*

<sup>2</sup>Even though we allow for selfloops in the directed graph  $G^+$  we note that the causal mechanisms  $g_S$  will depend only on  $\text{Pa}^{G^+}(S) \setminus S$ , removing the self-dependence on the functional level. Otherwise, the functions  $g_S$  would not hold up to a direct interventional interpretation and one would want to replace them with functions that do.

<sup>3</sup>This assumption is only necessary to give the mSCM a “reduced/summarized” form. In practice one could allow for more latent variables and more complex latent structure.

<sup>4</sup>Note that the index set runs over all “observable loops”  $S \subseteq V$ ,  $S \in \mathcal{L}(G^+)$ , which contains the usual single element sets  $S = \{v\}$ , which relate to the usual mechanisms  $g_{\{v\}}$ .

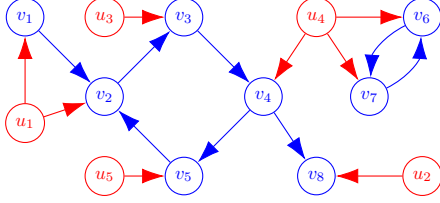


Figure 2: The graph  $G^+$  of a modular structural causal model (mSCM). The observed variables  $v_i \in V$  are in blue and the latent variables  $u_j \in U$  are in red. We have the four observed strongly connected components:  $\{v_1\}$ ,  $\{v_2, v_3, v_4, v_5\}$ ,  $\{v_6, v_7\}$ ,  $\{v_8\}$ .

1. The latent variables  $(X_u)_{u \in U}$  are given by the canonical projections  $E_u : \prod_{u' \in U} \mathcal{X}_{u'} \rightarrow \mathcal{X}_u$  and are jointly  $\mathbb{P}$ -independent (by 3). Sometimes we will write  $(E_u)_{u \in U}$  instead of  $(X_u)_{u \in U}$  to stress their interpretation as error/noise variables.
2. The observed variables  $(X_v)_{v \in V}$  are inductively defined by:

$$X_v := g_{S,v}((X_w)_{w \in \text{Pa}^{G^+}(S) \setminus S}),$$

where  $S := \text{Sc}^{G^+}(v) := \text{Anc}^{G^+}(v) \cap \text{Desc}^{G^+}(v)$  and where the second index  $v$  refers to the  $v$ -component of  $g_S$ . Note that the inductive definition is possible because when “aggregating” each of the biggest cycles  $\text{Sc}^{G^+}(v)$  into one node then only an acyclic graph is left, which can be totally ordered.

3. By the compatibility condition for  $g$  we then have that for every  $S \in \mathcal{L}(G^+)$  with  $S \subseteq V$  the following equality holds:

$$X_S = g_S(X_{\text{Pa}^{G^+}(S) \setminus S}),$$

where we put  $X_A := (X_v)_{v \in A}$  for subsets  $A$ .

As a consequence of the convenient definition 2.4 all the following desirable constructions (like marginalisations and interventions) are easily seen to be well-defined (for proofs see [12]). Note that already defining these constructions was a key challenge in the theory of causal models in the presence of cycles (see [3, 12], a.o.).

**Definition 2.7.** Let  $M = (G^+, \mathcal{X}, \mathbb{P}, g)$  be a mSCM with  $G^+ = (U \dot{\cup} V, E^+)$ .

1. By plugging the functions  $g_S$  into each other we can define the marginalised mSCM  $M'$  w.r.t. a subset  $W \subseteq V$ . For example, when marginalizing out  $W = \{w\}$  we can define (for the non-trivial case  $w \in \text{Pa}^{G^+}(S) \setminus S$ ):

$$g_{S',v}(x_{\text{Pa}^{G'}(S') \setminus S'}) := g_{S,v}(x_{\text{Pa}^{G^+}(S) \setminus (S \cup \{w\})}, g_{\{w\}}(x_{\text{Pa}^{G^+}(w) \setminus \{w\}})),$$

where  $G'$  is the marginalised graph of  $G^+$ ,  $S' \subseteq V'$  is any loop of  $G'$  and  $S$  the corresponding induced loop in  $G^+$ .

2. For a subset  $I \subseteq V$  and a value  $x_I \in \prod_{v \in I} \mathcal{X}_v$  we define the intervened graph  $G'$  by removing all the edges from parents of  $I$  to  $I$ . We put  $X_u^{\text{do}(x_I)} := X_u$  for  $u \in U$  and  $X_I^{\text{do}(x_I)} := x_I$  and inductively ( $S := \text{Sc}^{G'}(v)$ ):

$$X_v^{\text{do}(x_I)} := g_{S,v}(X_{\text{Pa}^{G'}(S) \setminus S}^{\text{do}(x_I)}).$$

By selecting all functions  $g_S$  where  $S$  is still a loop in the intervened graph  $G'$  we get the post-interventional mSCM  $M'$ . These constructions give us all interventional distributions, e.g. (cf. [25]):

$$\mathbb{P}(X_A | \text{do}(x_I), X_B) := \mathbb{P}(X_A^{\text{do}(x_I)} | X_B^{\text{do}(x_I)}).$$

Instead of fixing  $X_I^{\text{do}(x_I)}$  to a value  $x_I$  we could also specify a distribution  $\mathbb{P}'_I$  for it (“randomization”). In this way we define stochastic interventions  $\text{do}(\xi_I)$  with an independent random variable  $\xi_I$  taking values in  $\mathcal{X}_I$  and get a  $\mathbb{P}^{\text{do}(I)}$  similarly.

## 2.2 $\Sigma$ -SEPARATION IN MSCMS AND $\Sigma$ -CONNECTION GRAPHS

We now introduce  $\sigma$ -separation as a generalization of d-separation directly on the level of mixed graphs. To make the definition stable under marginalisation and conditioning we need to carry extra structure. The resulting graphs will be called  $\sigma$ -connection graphs ( $\sigma$ -CG), where the name is inspired by [19]. An example that shall clarify the difference between d- and  $\sigma$ -separation is given later in figure 2 and table 1.

**Definition 2.8** ( $\sigma$ -Connection Graphs ( $\sigma$ -CG)). A  $\sigma$ -connection graph ( $\sigma$ -CG) is a mixed graph  $G$  with a set of nodes  $V$  and directed ( $\rightarrow$ ), undirected ( $—$ ) and bi-directed ( $\leftrightarrow$ ) edges, together with an equivalence relation  $\sim_\sigma$  on  $V$  that has the property that every equivalence class  $\sigma(v)$ ,  $v \in V$ , is a loop in the underlying directed graph structure:  $\sigma(v) \in \mathcal{L}(G)$ . Undirected self-loops ( $v — v$ ) are allowed, (bi)-directed self-loops ( $v \rightarrow v$ ,  $v \leftrightarrow v$ ) are not.

In particular, every node is assigned to a unique fixed loop  $\sigma(v)$  in  $G$  with  $v \in \sigma(v)$  and two of such loops  $\sigma(v_1)$ ,  $\sigma(v_2)$  are either identical or disjoint. The reason for why we need such structure is illustrated in figure 5.

**Definition 2.9** ( $\sigma$ -Open Path in a  $\sigma$ -CG). Let  $G$  be a  $\sigma$ -CG with set of nodes  $V$  and  $Z \subseteq V$  a subset. Consider

a path  $\pi$  in  $G$  with  $n \geq 1$  nodes:

$$v_1 \rightleftarrows \dots \rightleftarrows v_n.^5$$

The path will be called  $Z$ - $\sigma$ -open if:

1. the endnodes  $v_1, v_n \notin Z$ , and
2. every triple of adjacent nodes in  $\pi$  that is of the form:

(a) collider:

$$v_{i-1} \rightleftarrows v_i \rightleftarrows v_{i+1},$$

satisfies  $v_i \in Z$ ,

(b) (non-collider) left chain:

$$v_{i-1} \leftarrow v_i \rightleftarrows v_{i+1},$$

satisfies  $v_i \notin Z$  or  $v_i \in Z \cap \sigma(v_{i-1})$ ,

(c) (non-collider) right chain:

$$v_{i-1} \rightleftarrows v_i \rightarrow v_{i+1},$$

satisfies  $v_i \notin Z$  or  $v_i \in Z \cap \sigma(v_{i+1})$ ,

(d) (non-collider) fork:

$$v_{i-1} \leftarrow v_i \rightarrow v_{i+1},$$

satisfies  $v_i \notin Z$  or  $v_i \in Z \cap \sigma(v_{i-1}) \cap \sigma(v_{i+1})$ ,

(e) (non-collider) with undirected edge:

$$v_{i-1} \text{ --- } v_i \rightleftarrows v_{i+1},$$

$$v_{i-1} \rightleftarrows v_i \text{ --- } v_{i+1},$$

satisfies  $v_i \notin Z$ .

The difference between  $\sigma$ - and  $d$ -separation lies in the additional conditions involving  $Z \cap \sigma(v_{i\pm 1})$ . The intuition behind them is that the dependence structure inside a loop  $\sigma(v_i)$  is so strong that non-colliders can only be blocked by conditioning if an edge is pointing out of the loop (see example 2.17 and table 1).

Similar to  $d$ -separation we can now define  $\sigma$ -separation in a  $\sigma$ -CG.

**Definition 2.10** ( $\sigma$ -Separation in a  $\sigma$ -CG). *Let  $G$  be a  $\sigma$ -CG with set of nodes  $V$ . Let  $X, Y, Z \subseteq V$  be subsets.*

1. We say that  $X$  and  $Y$  are  $\sigma$ -connected by  $Z$  or not  $\sigma$ -separated by  $Z$  if there exists a path  $\pi$  (with some  $n \geq 1$  nodes) in  $G$  with one endnode in  $X$  and one endnode in  $Y$  that is  $Z$ - $\sigma$ -open. In symbols this statement will be written as follows:

$$X \underset{G}{\overset{\sigma}{\not\perp}} Y | Z.$$

<sup>5</sup>The stacked edges are meant to be read as an ‘‘OR’’ at each place independently. We also allow for repeated nodes in the paths.

2. Otherwise, we will say that  $X$  and  $Y$  are  $\sigma$ -separated by  $Z$  and write:

$$X \underset{G}{\overset{\sigma}{\perp}} Y | Z.$$

**Remark 2.11.** 1. The finest/trivial  $\sigma$ -CG structure of a mixed graph  $G$  is given by  $\sigma(v) := \{v\}$  for all  $v \in V$ . In this way  $\sigma$ -separation in  $G$  coincides with the usual notion of  $d$ -separation in a  $d$ -connection graph ( $d$ -CG)  $G$  (see [19]). We will take this as the definition of  $d$ -separation and  $d$ -CG in the following.

2. The coarsest  $\sigma$ -CG structure of a mixed graph  $G$  is given by  $\sigma(v) := \text{Sc}^G(v) := \text{Anc}^G(v) \cap \text{Desc}^G(v)$  w.r.t. the underlying directed graph. Note that the definition of strongly connected component totally ignores the bi- and undirected edges of the  $\sigma$ -CG.
3. In any  $\sigma$ -CG we will always have that  $\sigma$ -separation implies  $d$ -separation, since every  $Z$ - $d$ -open path is also  $Z$ - $\sigma$ -open because  $\{v\} \subseteq \sigma(v)$ .
4. If a  $\sigma$ -CG  $G$  is acyclic (implying  $\text{Sc}^G(v) = \{v\}$ ) then  $\sigma$ -separation coincides with  $d$ -separation.

We now want to ‘‘hide’’ or marginalise out the latent nodes  $u \in U$  from the graph of any mSCM and represent their induced dependence structure with bidirected edges.

**Definition 2.12** (Induced  $\sigma$ -CG of a mSCM). *Let  $M = (G^+, \mathcal{X}, \mathbb{P}, g)$  be a mSCM with  $G^+ = (U \dot{\cup} V, E^+)$ . The induced  $\sigma$ -CG  $G$  of  $M$ , also referred to as the causal graph  $G$  of  $M$  is defined as follows:*

1. The nodes of  $G$  are all  $v \in V$ , i.e. all observed nodes of  $G^+$ .
2.  $G$  contains all the directed edges of  $G^+$  whose endnodes are both in  $V$ , i.e. observed.
3.  $G$  contains the bidirected edge  $v \leftrightarrow w$  with  $v, w \in V$  if and only if  $v \neq w$  and there exists a  $u \in U$  with  $v, w \in \text{Ch}^{G^+}(u)$ , i.e.  $v$  and  $w$  have a common latent confounder.
4.  $G$  contains no undirected edges.
5. We put  $\sigma(v) := \text{Sc}^G(v) = \text{Anc}^G(v) \cap \text{Desc}^G(v)$ .

**Remark 2.13.** Caution must be applied when going from  $G^+$  to  $G$ : It is possible that three observed nodes  $v_1, v_2, v_3$  have one joint latent common cause  $u_1$ , which can be read off  $G^+$ . This information will get lost when going from  $G^+$  to  $G$ , as we will represent this with three bidirected edges.  $G$  will nonetheless capture the conditional independence relations (see Theorem 2.14).

We now present the most important ingredient for our constraint-based causal discovery algorithm, namely a generalized directed global Markov property that relates the underlying causal graph ( $\sigma$ -CG)  $G$  of any mSCM  $M$  to the conditional independencies of the observed random variables  $(X_v)_{v \in V}$  via a  $\sigma$ -separation criterion.

**Theorem 2.14** ( $\sigma$ -Separation Criterion, see Corollary B.3). *The observed variables  $(X_v)_{v \in V}$  of any mSCM  $M$  satisfy the  $\sigma$ -separation criterion w.r.t. the induced  $\sigma$ -CG  $G$ . In other words, for all subsets  $W, Y, Z \subseteq V$  we have the implication:*

$$W \perp\!\!\!\perp_G^\sigma Y \mid Z \implies X_W \perp\!\!\!\perp_{\mathbb{P}} X_Y \mid X_Z.$$

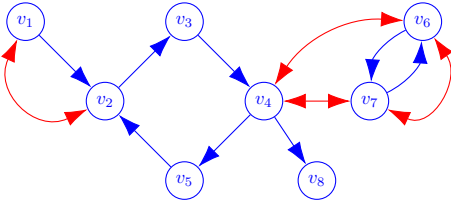


Figure 3: The induced  $\sigma$ -connection graph (causal graph) of the modular structural causal model (mSCM) of figure 2, with  $\sigma$ -equivalence classes  $\{\{v_1\}, \{v_2, v_3, v_4, v_5\}, \{v_6, v_7\}, \{v_8\}\}$ .

Table 1: d- and  $\sigma$ -separation in the  $\sigma$ -connection graph  $G$  from figure 3.  $\sigma$ -separation implies d-separation, but i.g. d-separation encodes more conditional independencies than  $\sigma$ -separation:

d-separation	$\sigma$ -separation
$\{v_2\} \perp\!\!\!\perp_G^d \{v_4\} \mid \{v_3, v_5\}$	$\{v_2\} \not\perp\!\!\!\perp_G^\sigma \{v_4\} \mid \{v_3, v_5\}$
$\{v_1\} \perp\!\!\!\perp_G^d \{v_6\}$	$\{v_1\} \perp\!\!\!\perp_G^\sigma \{v_6\}$
$\{v_1\} \perp\!\!\!\perp_G^d \{v_6\} \mid \{v_3, v_5\}$	$\{v_1\} \not\perp\!\!\!\perp_G^\sigma \{v_6\} \mid \{v_3, v_5\}$
$\{v_1\} \not\perp\!\!\!\perp_G^d \{v_8\}$	$\{v_1\} \not\perp\!\!\!\perp_G^\sigma \{v_8\}$
$\{v_1\} \perp\!\!\!\perp_G^d \{v_8\} \mid \{v_3, v_5\}$	$\{v_1\} \not\perp\!\!\!\perp_G^\sigma \{v_8\} \mid \{v_3, v_5\}$
$\{v_1\} \perp\!\!\!\perp_G^d \{v_8\} \mid \{v_4\}$	$\{v_1\} \perp\!\!\!\perp_G^\sigma \{v_8\} \mid \{v_4\}$

If we want to infer the causal graph ( $\sigma$ -CG)  $G$  of a mSCM from data with help of conditional independence tests in practice we usually need to assume also the reverse implication of the  $\sigma$ -separation criterion from 2.14 for the observational distribution  $\mathbb{P}(X_V)$  and the relevant interventional distributions  $\mathbb{P}(X_V \mid \text{do}(\xi_I))$  (see 2.7). This will be called  $\sigma$ -faithfulness.

**Definition 2.15** ( $\sigma$ -faithfulness). *We will say that the tuple  $(G, \mathbb{P})$  is  $\sigma$ -faithful if for every three subsets  $W, Y, Z \subseteq V$  we have the equivalence:*

$$W \perp\!\!\!\perp_G^\sigma Y \mid Z \iff X_W \perp\!\!\!\perp_{\mathbb{P}} X_Y \mid X_Z.$$

**Remark 2.16** (Strong  $\sigma$ -completeness, cf. [22]). *We do believe that the generic non-linear mSCM is  $\sigma$ -faithful to  $G$ , as the needed conditional dependence structure in all our simulated (sufficiently non-linear) cases was observed (cf. example 2.17). But proving such a strong  $\sigma$ -completeness result is difficult (and to our knowledge only done for multinomial and linear Gaussian DAG models, see [22]) and left for future research. Further note that the class of linear models, which even follow d-separation, would i.g. not be considered  $\sigma$ -faithful. Since linear models are of measure zero in the bigger class of general mSCMs this would not contradict our conjecture.*

**Example 2.17.** *Consider a directed four-cycle, e.g. the subgraph  $\{v_2, v_3, v_4, v_5\}$  from figure 2, where all other observed nodes are assumed to be absent. Consider only the non-linear causal mechanisms given by ( $i = 3, 4, 5$ ):*

$$\begin{aligned} g_{\{v_2\}}(X_5, E_1) &:= \tanh(0.9 \cdot X_5 + 0.5) + E_1, \\ g_{\{v_i\}}(X_{i-1}, E_i) &:= \tanh(0.9 \cdot X_{i-1} + 0.5) + E_i, \end{aligned}$$

where  $E_1, E_3, E_4, E_5$  are assumed to be independent. The equations  $X_i = g_{\{v_i\}}(X_{i-1}, E_i)$  and the one for  $X_2$  will imply the conditional dependence

$$X_2 \not\perp\!\!\!\perp_{\mathbb{P}} X_4 \mid (X_3, X_5),$$

which can be checked by computations and/or simulations. As one can read off table 1, d-separation fails to express the dependence, in contrast to  $\sigma$ -separation, which captures it correctly.

### 2.3 MARGINALISATION AND CONDITIONING IN $\Sigma$ -CONNECTION GRAPHS

Inspired by [19] we will define marginalisation and conditioning operations on  $\sigma$ -connection graphs ( $\sigma$ -CG) and prove the closedness of  $\sigma$ -separation (and thus its criterion) under these operations. These are key results to extend the algorithm of [19] to the setting of mSCMs.

**Definition 2.18** (Marginalisation of a  $\sigma$ -CG). *Let  $G$  be a  $\sigma$ -CG with set of nodes  $V$  and  $w \in V$ ,  $W := \{w\}$ . We define the marginalised  $\sigma$ -CG  $G^W$  with set of nodes  $V \setminus W$  via the rules for  $v_1, v_2 \in V \setminus W$ :*

$v_1 \overset{a}{\circ} \overset{b}{\circ} v_2 \in G^W$  with arrow heads/tails  $a$  and  $b$  if and only if there exists:

1.  $v_1 \overset{a}{\circ} \overset{b}{\circ} v_2$  in  $G$ , or
2.  $v_1 \overset{a}{\circ} w \overset{q}{\circ} \overset{b}{\circ} v_2$  in  $G$ , or
3.  $v_1 \overset{a}{\circ} \overset{q}{\circ} w \overset{b}{\circ} v_2$  in  $G$ , or
4.  $v_1 \overset{a}{\circ} \rightarrow w \leftarrow \overset{b}{\circ} v_2$  in  $G$ .

Note that directed paths in  $G$  have no colliders, so loops in  $G$  map to loops in  $G^W$  (if not empty). Thus we have the induced  $\sigma$ -CG structure  $\sim_\sigma$  on  $G^W$ .

**Definition 2.19** (Conditioning of a  $\sigma$ -CG). *Let  $G$  be a  $\sigma$ -CG with set of nodes  $V$  and  $c \in V$ ,  $C := \{c\}$ . We define the conditioned  $\sigma$ -CG  $G_C$  with set of nodes  $V \setminus C$  via the rules for  $v_1, v_2 \in V \setminus C$ :*

$v_1 a \overset{\sigma}{\leftarrow} v_2 \in G_C$  if and only if there exists:

1.  $v_1 a \overset{\sigma}{\leftarrow} v_2$  in  $G$ , or
2.  $v_1 a \overset{\sigma}{\rightarrow} c \overset{\sigma}{\leftarrow} v_2$  in  $G$ , or
3.  $v_1 a \overset{\sigma}{\leftarrow} c \overset{\sigma}{\leftarrow} v_2$  in  $G$ ,  $\sigma(v_1) = \sigma(c)$ , or
4.  $v_1 a \overset{\sigma}{\rightarrow} c \overset{\sigma}{\rightarrow} v_2$  in  $G$ ,  $\sigma(c) = \sigma(v_2)$ , or
5.  $v_1 a \overset{\sigma}{\leftarrow} c \overset{\sigma}{\rightarrow} v_2$  in  $G$ ,  $\sigma(v_1) = \sigma(c) = \sigma(v_2)$ .

Note that directed paths in  $\sigma(v)$  in  $G$  condition to directed paths, so loops in  $\sigma(v)$  in  $G$  map to loops in  $G_C$  (if not empty). Thus we have a well-defined induced  $\sigma$ -CG structure  $\sim_\sigma$  on  $G_C$ .

The proofs of the following theorem, stating the closedness of  $\sigma$ -separation under marginalisation and conditioning, can be found in the supplementary material (Theorem A.1 and Theorem A.2). See also figure 5.

**Theorem 2.20.** *Let  $G$  be a  $\sigma$ -CG with set of nodes  $V$  and  $X, Y, Z \subseteq V$  any subsets. For any nodes  $w, c \in V \setminus (X \cup Y \cup Z)$ ,  $W := \{w\}$ ,  $C := \{c\}$ , we then have the equivalences:*

$$\begin{aligned} X \underset{G^W}{\perp\!\!\!\perp} Y | Z &\iff X \underset{G}{\perp\!\!\!\perp} Y | Z, \\ X \underset{G_C}{\perp\!\!\!\perp} Y | Z &\iff X \underset{G}{\perp\!\!\!\perp} Y | Z \cup C. \end{aligned}$$

**Corollary 2.21.** *Let  $G$  be a  $\sigma$ -CG with set of nodes  $V$  and  $X, Y, Z \subseteq V$  pairwise disjoint subsets and  $W := V \setminus (X \cup Y \cup Z)$ . Then we have the equivalence:*

$$X \underset{G}{\perp\!\!\!\perp} Y | Z \iff X \underset{G_Z^W}{\perp\!\!\!\perp} Y,$$

where  $G_Z^W$  is any  $\sigma$ -CG with set of nodes  $X \cup Y$  obtained by marginalising out all the nodes from  $W$  and conditioning on all the nodes from  $Z$  in any order. This means that if  $X = \{x\}$  and  $Y = \{y\}$  then  $x$  and  $y$  are  $\sigma$ -separated by  $Z$  in  $G$  if and only if  $x$  and  $y$  are not connected by any edge in the  $\sigma$ -CG  $G_Z^W$ .

It is also tempting to introduce an intervention operator directly on the level of  $\sigma$ -CGs. However, since the interplay between conditioning and intervention is complicated (e.g. they do not commute i.g.) we do not investigate this further in this paper. The intervention operator on the level of mSCMs will be enough for our purposes as we assume no pre-interventional selection bias and then only encounter observational or post-interventional conditioning, which is covered by our framework.

### 3 ALGORITHM

In this section, we propose an algorithm for causal discovery that is based on the theory in the previous section. Given that theory, our proposed algorithm is a straightforward modification of the algorithm by [19]. The main idea is to formulate the causal discovery problem as an optimization problem that aims at finding the causal graph that best matches the data at hand. This is done by encoding the rules for conditioning, marginalisation, and intervention (see below) on a  $\sigma$ -CG into Answer Set Programming (ASP), an expressive declarative programming language based on stable model semantics that supports optimization [15, 20]. The optimization problem can then be solved by employing an off-the-shelf ASP solver.

#### 3.1 CAUSAL DISCOVERY WITH $\Sigma$ -CONNECTION GRAPHS

Let  $M = (G^+, \mathcal{X}, \mathbb{P}, g)$  be a mSCM with  $G^+ = (U \dot{\cup} V, E^+)$  and  $I \subseteq V$  a subset. Consider a (stochastic) perfect intervention  $\text{do}(\xi_I)$  that enforces  $X_I = \xi_I$  for an independent random variable  $\xi_I$  taking values in  $\mathcal{X}_I$ . Denote the (unique) induced distribution of the intervened mSCM  $M_{\text{do}(\xi_I)}$  by  $\mathbb{P}^{\text{do}(I)}$ , and the causal graph (i.e., induced  $\sigma$ -CG of the intervened mSCM on the observed variables) by  $G_{\text{do}(I)} = (G_{\text{do}(I)}^+)^U$ .

Under  $\mathbb{P}^{\text{do}(I)}$ , the observed variables  $(X_v)_{v \in V}$  satisfy the  $\sigma$ -separation criterion w.r.t.  $G_{\text{do}(I)}$  by Theorem 2.14. For the purpose of causal discovery, we will in addition assume  $\sigma$ -faithfulness (Definition 2.15), i.e., that each conditional independence between observed variables is due to a  $\sigma$ -separation in the causal graph. Taken together, and by applying Corollary 2.21, we get for all subsets  $W, Y, Z \subseteq V$  the equivalences:

$$\begin{aligned} X_W \underset{\mathbb{P}^{\text{do}(I)}}{\perp\!\!\!\perp} X_Y | X_Z &\iff W \underset{G_{\text{do}(I)}}{\perp\!\!\!\perp} Y | Z \\ &\iff W \underset{(G_{\text{do}(I)})_Z}{\perp\!\!\!\perp} Y. \end{aligned} \quad (1)$$

If  $W = \{w\}$  and  $Y = \{y\}$  consist of a single node each, the latter can be easily checked by testing whether  $w$  is non-adjacent to  $y$  in  $(G_{\text{do}(I)})_Z^{V \setminus W \cup Y \cup Z}$ .

#### 3.2 CAUSAL DISCOVERY AS AN OPTIMIZATION PROBLEM

Following [19], we formulate causal discovery as an optimization problem where a certain loss function is optimized over possible causal graphs. This loss function sums the weights of all the inputs that are violated assuming a certain underlying causal graph.

The input for the algorithm is a list  $S = ((w_j, y_j, Z_j, I_j, \lambda_j))_{j=1}^n$  of weighted conditional independence statements. Here, the weighted statement  $(w_j, y_j, Z_j, I_j, \lambda_j)$  with  $w_j, y_j \in V$ ,  $Z_j, I_j \subseteq V$ , and  $\lambda_j \in \mathbb{R} := \mathbb{R} \cup \{-\infty, +\infty\}$  encodes that  $X_{w_j} \perp\!\!\!\perp_{\mathbb{P}^{\text{do}(X_{I_j})}} X_{y_j} \mid X_{Z_j}$  holds with confidence  $\lambda_j$ , where a finite value of  $\lambda_j$  gives a “soft constraint” and a value of  $\lambda_j = \pm\infty$  imposes a “hard constraint”. Positive weights encode that we have empirical support *in favor* of the independence, whereas negative weights encode empirical support *against* the independence (in other words, in favor of *dependence*).

As in [19], we define a loss function that measures the amount of evidence *against* the hypothesis that the data was generated by an mSCM with causal graph  $G$ , by simply summing the absolute weights of the input statements that conflict with  $G$  under the  $\sigma$ -Markov and  $\sigma$ -faithfulness assumptions:

$$\begin{aligned} \mathcal{L}(G, S) \\ := \sum_{(w_j, y_j, Z_j, I_j, \lambda_j) \in S} \lambda_j (\mathbb{1}_{\lambda_j > 0} - \mathbb{1}_{w_j \perp\!\!\!\perp_{\sigma_{\text{do}(I_j)}} y_j \mid Z_j}) \end{aligned} \quad (2)$$

where  $\mathbb{1}$  is the indicator function. This loss function differs from the one used in [19] in that we use  $\sigma$ -separation instead of d-separation. Causal discovery can now be formulated as the optimization problem:

$$G^* = \arg \min_{G \in \mathbb{G}(V)} \mathcal{L}_R(G, S) \quad (3)$$

where  $\mathbb{G}(V)$  denotes the set of all possible causal graphs with variables  $V$ .

The optimization problem (3) may have multiple optimal solutions, because the underlying causal graph may not be identifiable from the inputs. Nonetheless, some of the features of the causal graph (e.g., the presence or absence of a certain directed edge) may still be identifiable. We employ the method proposed by [21] for scoring the confidence that a certain feature  $f$  is present by calculating the difference between the optimal losses under the additional hard constraints that the feature  $f$  is present vs. that the feature  $f$  is absent in  $G$ .

In our experiments, we will use the weights proposed in [21]:  $\lambda_j = \log p_j - \log \alpha$ , where  $p_j$  is the p-value of a statistical test with independence as null hypothesis, and  $\alpha$  is a significance level (e.g., 1%). This test is performed on the data measured in the context of the (stochastic) perfect intervention  $I_j$ . These weights have the desirable property that independences typically get a lower absolute weight than strong dependences. For the conditional independence test, we use a standard partial

correlation test after marginal rank-transformation of the data so as to obtain marginals with standard-normal distributions.

### 3.3 FORMULATING THE OPTIMIZATION PROBLEM IN ASP

In order to calculate the loss function (2), we make use of Corollary 2.21 to reduce the  $\sigma$ -separation test to a simple non-adjacency test in a conditioned and marginalised  $\sigma$ -CG, as in (1). We do this by encoding  $\sigma$ -CGs, Theorem 2.20 and the marginalisation and conditioning operations on  $\sigma$ -CGs (Definitions 2.18 and 2.19) in ASP. The details of the encoding are provided in the Supplementary Material.<sup>6</sup> The optimization problem in (3) can then be solved straightforwardly by running an off-the-shelf ASP solver with as input the encoding and the weighted independence statements.

A more precise statement of the following result is provided in the Supplementary Material. The proof is basically the same as the one given in [21].

**Theorem 3.1.** *The algorithm for scoring features  $f$  is sound for oracle inputs and asymptotically consistent under mild assumptions.*

## 4 EXPERIMENTS

### 4.1 CONSTRUCTING MSCMS AND SAMPLING FROM MSCMS

To construct a modular structural causal model (mSCM) in practice we need to specify the compatible system of functions  $(g_S)_{S \in \mathcal{L}(G)}$ . The following Theorem is helpful (and a direct consequence of Banach’s fixed point theorem).

**Theorem 4.1.** *Consider the functions  $g_{\{v\}}$  for the trivial loops  $\{v\} \in \mathcal{L}(G)$ ,  $v \in V$  and assume the following contractivity condition:*

*For every non-trivial loop  $S \in \mathcal{L}(G)$  and for every value  $x_{\text{Pa}^{G^+}(S) \setminus S}$  the multi-dimensional function:*

$$x'_S \mapsto \left( g_{\{v\}}(x'_{S \cap \text{Pa}^{G^+}(v) \setminus \{v\}}, x_{\text{Pa}^{G^+}(v) \setminus S}) \right)_{v \in S}$$

*is a contraction, i.e. Lipschitz continuous with Lipschitz constant  $L(x_{\text{Pa}^{G^+}(S) \setminus S}) < 1$  w.r.t. a suitable norm  $\|\cdot\|$ .*

*Then all the functions  $g_S$  for the non-trivial loops  $S \in \mathcal{L}(G)$  exist, are unique and  $g = (g_S)_{S \in \mathcal{L}(G)}$  forms a*

<sup>6</sup>The full source code for the algorithm and to reproduce our experiments is available under an open source license from <https://github.com/caus-am/sigmasep>.



globally compatible system.

More constructively, for every value  $x_{\text{Pa}G^+(S)\setminus S}$  and initialization  $x_S^{(0)}$  the iteration scheme (using vector notations):

$$x_S^{(t+1)} := (g_{\{v\}})_{v \in S}(x_S^{(t)}, x_{\text{Pa}G^+(S)\setminus S})$$

converges to a unique limit vector  $x_S$  (for  $t \rightarrow \infty$  and independent of  $x_S^{(0)}$ ).  $g_S$  is then given by putting:

$$g_S(x_{\text{Pa}G^+(S)\setminus S}) := x_S.$$

This provides us with a method for constructing very general non-linear mSCMs (e.g. neural networks, see Section C in Supplementary Material) and to sample from them: by sampling  $x_U$  from the external distribution and then apply the above iteration scheme until convergence for all loops, yielding the limit  $x_V$  as one data point.

## 4.2 RESULTS ON SYNTHETIC DATA

In our experiments we will—due to computational restrictions—only allow for  $d = 5$  observed nodes and  $k = 2$  additional latent confounders. We sample edges independently with a probability of  $p = 0.3$ . We model the non-linear function  $g_{\{v\}}$  as a neural network with tanh activation, bias terms that have a normal distribution with mean  $-0.5$  and standard deviation  $0.2$ , and weights sampled uniformly from the L1-unit ball to satisfy the contraction condition of Theorem 4.1 (see also Supplementary Material, Section C). We simulate 0–5 single-variable interventions with random (unique) targets. For each intervened model we sample from standard-normal noise terms and compute the observations. To also detect weak dependencies in cyclic models we allow for  $n = 10^4$  samples in each such model for each allowed intervention. We then run all possible conditional independence tests between every pair of single nodes and calculate their  $p$ -values. We used  $\alpha = 10^{-3}$  as the threshold between dependence and independence. For computational reasons we restrict to partial correlation tests of marginal Gaussian rank-transforms of the data. These tests are then fed into the ASP solver together with our encoding of the optimization problem (3). We query the ASP solver for the confidence for the absence or presence of each possible directed and bidirected edge. We simulate 300 models and aggregate results, using the confidence scores to compute ROC- and PR-curves for features. Figure 4 shows that, as expected, our algorithm recovers more directed edges of the underlying causal graph in the simulation setting as the number of single-variable interventions increases. More results (ROC- and PR-curves for directed edges and con-

founders for different numbers of single-variable interventions and for different encodings) are provided in the Supplementary Material.

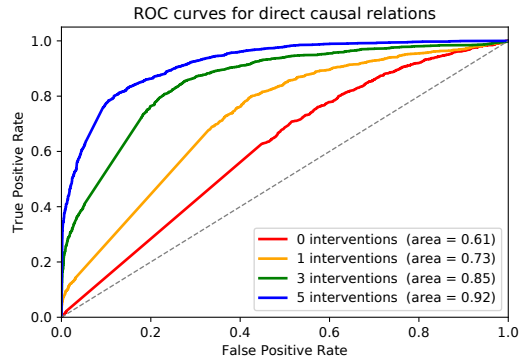


Figure 4: The ROC curves for identifying directed edges. See also figures 6 and 7 in the Supplementary Material.

## 5 CONCLUSION

We introduced  $\sigma$ -connection graphs ( $\sigma$ -CG) as a generalization of the d-connection graphs (d-CG) of [19] and extended the notion of  $\sigma$ -separation that was introduced in [12] to  $\sigma$ -CGs. We showed how  $\sigma$ -CGs behave under marginalisation and conditioning. This provides a graphical representation of how conditional independencies of modular structural causal models (mSCMs) behave under these operations. We provided a sufficient condition that allows constructing mSCMs and sampling from them. We extended the algorithm of [19] to deal with the more generally applicable notion of  $\sigma$ -separation instead of d-separation, thereby obtaining the first algorithm for causal discovery that can deal with cycles, nonlinearities, latent confounders and a combination of data sets corresponding to observational and different interventional settings. We illustrated the effectiveness of the algorithm on simulated data. In this work, we restricted attention to (stochastic) perfect (“surgical”) interventions, but a straightforward extension to deal with other types of interventions and to generalize the idea of randomized controlled trials can be obtained by applying the JCI framework [23]. In future work we wish to improve our algorithm to also handle selection bias, become more scalable and apply it to real world data sets.

### Acknowledgements

This work was supported by the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement 639466).

## References

- [1] F. Barthe, O. Guédon, S. Mendelson, and A. Naor. A probabilistic approach to the geometry of the  $l_p^p$ -ball. *Ann. Probab.*, 33(2):480–513, 2005.
- [2] V.I. Bogachev. *Measure Theory. Vol. I, II.* Springer, 2007.
- [3] S. Bongers, J. Peters, B. Schölkopf, and J. M. Mooij. Theoretical aspects of cyclic structural causal models. *arXiv.org preprint*, arXiv:1611.06221v2 [stat.ME], 2018.
- [4] T. Claassen and T. Heskes. Bayesian probabilities for constraint-based causal discovery. In *IJCAI-13*, pages 2992–2996, 2013.
- [5] T. Claassen, J.M. Mooij, and T. Heskes. Learning Sparse Causal Models is not NP-hard. In *UAI-13*, pages 172–181, 2013.
- [6] G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.
- [7] M. Drton, M. Eichler, and T. Richardson. Computing maximum likelihood estimates in recursive linear models with correlated errors. *Journal of Machine Learning Research*, 10:2329–2348, 2009.
- [8] R. Evans and T. Richardson. Maximum likelihood fitting of acyclic directed mixed graphs to binary data. In *UAI-10*, 2010.
- [9] R.J. Evans. Graphs for margins of Bayesian networks. *Scand. J. Stat.*, 43(3):625–648, 2016.
- [10] R.J. Evans. Margins of discrete Bayesian networks. *arXiv:1501.02103*, pages 1–41, 2017. Submitted to *Annals of Statistics*.
- [11] R.J. Evans and T.S. Richardson. Markovian acyclic directed mixed graphs for discrete data. *Ann. Statist.*, 42(4):1452–1482, 08 2014.
- [12] P. Forré and J. M. Mooij. Markov properties for graphical models with cycles and latent variables. *arXiv:1710.08775*, 2017.
- [13] M. Gebser, R. Kaminski, B. Kaufmann, and T. Schaub. *Clingo = ASP + control: Extended report.* Technical report, University of Potsdam, 2014. <http://www.cs.uni-potsdam.de/wv/pdfformat/gekakasc14a.pdf>.
- [14] D. Geiger and D. Heckerman. Learning Gaussian networks. In *UAI-94*, pages 235–243, 1994.
- [15] M. Gelfond. Answer sets. In *Handbook of Knowledge Representation*, pages 285–316. 2008.
- [16] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning.* MIT Press, 2016.
- [17] D. Heckerman, C. Meek, and G. Cooper. A Bayesian approach to causal discovery. In C. Glymour and G. F. Cooper, editors, *Computation, Causation, and Discovery*, pages 141–166. MIT Press, 1999.
- [18] A. Hyttinen, F. Eberhardt, and P.O. Hoyer. Causal discovery for linear cyclic models. In *Proceedings of the Fifth European Workshop on Probabilistic Graphical Models*, 2010.
- [19] A. Hyttinen, F. Eberhardt, and M. Järvisalo. Constraint-based causal discovery: Conflict resolution with answer set programming. In *UAI-14*, pages 340–349, 2014.
- [20] V. Lifschitz. What is Answer Set Programming? In *AAAI Conference on Artificial Intelligence*, pages 1594–1597, 2008.
- [21] S. Magliacane, T. Claassen, and J.M. Mooij. Ancestral causal inference. In *NIPS-16*, pages 4466–4474. 2016.
- [22] C. Meek. Strong completeness and faithfulness in Bayesian networks. In *UAI-95*, pages 411–418, 1995.
- [23] J. M. Mooij, S. Magliacane, and T. Claassen. Joint causal inference from multiple contexts. *arXiv.org preprint*, <https://arxiv.org/abs/1611.10351v3> [cs.LG], March 2018.
- [24] J. Pearl. Fusion, propagation and structuring in belief networks. Technical Report 3, UCLA Computer Science Department, 1986. Technical Report 850022 (R-42).
- [25] J. Pearl. *Causality: Models, reasoning, and inference.* Cambridge University Press, 2nd edition, 2009.
- [26] J. Peters, D. Janzing, and B. Schölkopf. *Elements of Causal Inference: Foundation and Learning Algorithms.* MIT press, 2017.
- [27] T. Richardson. A discovery algorithm for directed cyclic graphs. In *UAI-96*, pages 454–461. 1996.
- [28] T. Richardson. Markov properties for acyclic directed mixed graphs. *Scand. J. Stat.*, 30(1):145–157, 2003.
- [29] T. Richardson and P. Spirtes. Automated discovery of linear feedback models. In C. Glymour and G. F. Cooper, editors, *Computation, Causation, and Discovery*, pages 253–304. MIT Press, 1999.
- [30] D. Rothenhäusler, C. Heinze, J. Peters, and N. Meinshausen. BACKSHIFT: Learning causal cyclic graphs from unknown shift interventions. In *NIPS-15*, pages 1513–1521. 2015.
- [31] P. Spirtes. Directed cyclic graphical representations of feedback models. In *UAI-95*, pages 491–499, 1995.
- [32] P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search.* MIT press, 2000.
- [33] T.S. Verma and J. Pearl. Causal Networks: Semantics and Expressiveness. *UAI-90*, 4:69–76, 1990.
- [34] J. Zhang. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. *Artificial Intelligence*, 172(16-17):1873–1896, 2008.

---

# Marginal Weighted Maximum Log-likelihood for Efficient Learning of Perturb-and-Map models

---

**Tatiana Shpakova**

INRIA - ENS  
PSL Research University  
Paris, France

**Francis Bach**

INRIA - ENS  
PSL Research University  
Paris, France

**Anton Osokin**

National Research University  
Higher School of Economics  
Moscow, Russia

## Abstract

We consider the structured-output prediction problem through probabilistic approaches and generalize the “perturb-and-MAP” framework to more challenging weighted Hamming losses, which are crucial in applications. While in principle our approach is a straightforward marginalization, it requires solving many related MAP inference problems. We show that for log-supermodular pairwise models these operations can be performed efficiently using the machinery of dynamic graph cuts. We also propose to use *double* stochastic gradient descent, both on the data and on the perturbations, for efficient learning. Our framework can naturally take weak supervision (e.g., partial labels) into account. We conduct a set of experiments on medium-scale character recognition and image segmentation, showing the benefits of our algorithms.

## 1 INTRODUCTION

Structured-output prediction is an important and challenging problem in the field of machine learning. When outputs have a structure, often in terms of parts or elements (e.g., pixels, sentences or characters), methods that do take it into account typically outperform more naive methods that consider outputs as a set of independent elements. Structured-output methods based on optimization can be broadly separated in two main families: *max-margin methods*, such as structured support vector machines (SSVM) (Taskar et al., 2003; Tsochantaridis et al., 2005) and *probabilistic methods based on maximum likelihoods* such as conditional random fields (CRF) (Laferty et al., 2001).

Structured-output prediction faces many challenges:

(1) on top of large input dimensions, problems also have large outputs, leading to scalability issues, in particular when prediction or learning depends on combinatorial optimization problems (which are often polynomial-time, but still slow given they are run many times); (2) it is often necessary to use losses which go beyond the traditional 0-1 loss to shape the behavior of the learned models towards the final evaluation metric; (3) having fully labelled data is either rare or expensive and thus, methods should be able to deal with weak supervision.

Max-margin methods can be used with predefined losses, and have been made scalable by several recent contributions (see, e.g., Lacoste-Julien et al., 2013, and references therein), but do not deal naturally with weak supervision. However, a few works (Yu and Joachims, 2009; Kumar et al., 2010; Girshick et al., 2011) incorporate weak supervision into the max-margin approach via the CCCP (Yuille and Rangarajan, 2003) algorithm.

The flexibility of probabilistic modeling naturally allows (a) taking into consideration weak supervision and (b) characterizing the uncertainty of predictions, but it comes with strong computational challenges as well as a non-natural way of dealing with predefined losses beyond the 0-1 loss. The main goal of this paper is to provide new tools for structured-output inference with probabilistic models, thus making them more widely applicable, while still being efficient. There are two main techniques to allow for scalable learning in CRFs: stochastic optimization (Vishwanathan et al., 2006) and piecewise training (Sutton and McCallum, 2005, 2007; Kolesnikov et al., 2014); note that the techniques above can also be used for weak supervision (and we reuse some of them in this work).

Learning and inference in probabilistic structured-output models recently received a lot of attention from the research community (Bakir et al., 2007; Nowozin and Lempert, 2011; Smith, 2011). In this paper we consider only models for which maximum-a posteriori (MAP)

inference is feasible (a step often referred to as decoding in max-margin formulations, and which typically makes them tractable). A lot of efforts were spent to explore MAP-solvers algorithms for various problems, leveraging various structures, e.g., graphs of low tree-width (Bishop, 2006; Wainwright and Jordan, 2008; Sontag et al., 2008; Komodakis et al., 2011) and function submodularity (Boros and Hammer, 2002; Kolmogorov and Zabih, 2004; Bach, 2013; Osokin and Vetrov, 2015).

Naturally, the existence of even an exact and efficient MAP-solver does not mean that the partition function (a key tool for probabilistic inference as shown below) is tractable to compute. Indeed, the partition function computation is known to be  $\#P$ -hard (Jerrum and Sinclair, 1993) in general. For example, MAP-inference is efficient for log-supermodular probabilistic models, while computation of their partition function is not (Djolonga and Krause, 2014).

For such problems where MAP-inference is efficient, but partition function computation is not, “perturb-and-MAP” ideas such as proposed by Papandreou and Yuille (2011); Hazan and Jaakkola (2012) are a very suitable treatment. By adding random perturbations, and then performing MAP-inference, they can lead to estimates of the partition function. In Section 2, we review the existing approaches to the partition function approximation, parameter learning and inference.

An attempt to learn parameters via “perturb-and-MAP” ideas was made by Hazan et al. (2013), where the authors have developed a PAC-Bayesian-flavoured approach for the non-decomposable loss functions. While the presented algorithm has something in common with ours (gradient descent optimization of the objective upper bound), it differs in the sense of the objective function and the problem setup, which is more general but that requires a different (potentially with higher variance) estimates of the gradients. Such estimates are usual in reinforcement learning, e.g., the log-derivative trick from the REINFORCE algorithm (Williams, 1992).

The goal of this paper is to make the “perturb-and-MAP” technique applicable to practical problems, in terms of (a) scale, by increasing the problem size significantly, and (b) losses, by treating structured losses such as the Hamming loss or its weighted version, which are crucial to obtaining good performance in practice.

Overall, we make the following contributions:

- In Section 3, we generalize the “perturb-and-MAP” framework to more challenging weighted Hamming losses which are commonly used in applications. In principle, this is a straightforward marginalization but this requires solving many related MAP inference prob-

lems. We show that for graph cuts (our main inference algorithm for image segmentation), this can be done particularly efficiently. Besides that, we propose to use a *double* stochastic gradient descent, both on the data and on the perturbations.

- In Section 4, we show how weak supervision (e.g., partial labels) can be naturally dealt with. Our method in this case relies on approximating marginal probabilities that can be done almost at the cost of the partition function approximation.
- In Section 5, we conduct a set of experiments on medium-scale character recognition and image segmentation, showing the benefits of our new algorithms.

## 2 PERTURB-AND-MAP

In this section, we introduce the notation and review the relevant background. We study the following probabilistic model (a.k.a. a Gibbs distribution) over a discrete product space  $Y = Y_1 \times \dots \times Y_D$ ,

$$P(y) = \frac{1}{Z(f)} e^{f(y)}, \quad (1)$$

which is defined by a potential function  $f : Y \rightarrow \mathbb{R}$ . The constant  $Z(f) = \sum_{y \in Y} e^{f(y)}$  is called the partition function and normalizes  $P(y)$  to be a valid probability function, i.e., to sum to one.  $Z(f)$  is in general intractable to compute as the direct computation requires summing over exponentially (in  $D$ ) many elements.

Various partition function approximations methods have been used in parameter learning algorithms (Parise and Welling, 2005), e.g., mean-field (MF, Jordan et al., 1999), tree-reweighted belief propagation (TRW, Wainwright and Jordan, 2008) or loopy belief propagation (LBP, Weiss, 2001). We will work with the upper bound on the partition function proposed by Hazan and Jaakkola (2012) as it allows us to approximate the partition function via MAP-inference, calculate gradients efficiently, approximate marginal probabilities and guarantee tightness for some probabilistic models. We introduce this class of techniques below.

### 2.1 Gumbel perturbations

Recently, Hazan and Jaakkola (2012) provided a general-purpose upper bound on the *log-partition function*  $A(f) = \log Z(f)$ , based on the “perturb-and-MAP” idea (Papandreou and Yuille, 2011): maximize the potential function perturbed by Gumbel-distributed noise.<sup>1</sup>

<sup>1</sup>The Gumbel distribution on the real line has cumulative distribution function  $F(z) = \exp(-\exp(-(z+c)))$ , where  $c$  is the Euler constant.

**Proposition 1** (Hazan and Jaakkola (2012), Corollary 1). For any function  $f : Y \rightarrow \mathbb{R}$ , we have  $A(f) \leq A_G(f)$ , where

$$A_G(f) = \mathbb{E}_{z_1, \dots, z_D \sim \text{Gumbel}} \left[ \max_{y \in Y} \left( f(y) + \sum_{d=1}^D z_d(y_d) \right) \right]. \quad (2)$$

Gumbel denotes the Gumbel distribution and  $\{z_d(y_d)\}_{y_d \in Y_d}^{d=1, \dots, D}$  is a collection of independent Gumbel samples.

The bound is tight when  $f(y)$  is a separable function (i.e., a sum of functions of single variables), and the tightness of this bound was further studied by Shpakova and Bach (2016) for log-supermodular models (where  $f$  is supermodular). They have shown that the bound  $A_G$  is always lower (and thus provide a better bound) than the ‘‘L-field’’ bound proposed by Djolonga and Krause (2014, 2015), which is itself based on separable optimization on the base polytope of the associated supermodular function.

The partition function bound  $A_G$  can be approximated by replacing the expectation by an empirical average. That is, to approximate it we need to solve a large number (as many as the number of Gumbel samples used to approximate the expectation) of MAP-like problems (i.e., maximizing  $f$  plus a separable function) which are feasible by our assumption. Strictly speaking, the MAP-inference is NP-hard in general, but firstly, it is much easier than the partition function calculation, secondly, there are solvers for special cases, e.g., for log-supermodular models (which include functions  $f$  which are negatives of cuts (Kolmogorov and Zabih, 2004; Boykov and Kolmogorov, 2004)) and those solvers are often efficient enough in practice. In this paper, we will focus primarily on a subcase of supermodular potentials, namely negatives of graph cuts.

## 2.2 Parameter learning and Inference

In the standard supervised setting of structured prediction, we are given  $N$  pairs of observations  $\mathcal{D} = \{(x^n, y^n)\}_{n=1}^N$ , where  $x^n$  is a feature representation of the  $n$ -th object and  $y^n \in Y = Y_1 \times \dots \times Y_{D^n}$  is a structured vector of interest (e.g., a sequence of tags, a segmentation MAP or a document summarization representation). In the standard linear model, the potential function  $f(y|x)$  is represented as a linear combination:  $f(y|x) = w^T \Psi(x, y)$ , where  $w$  is a vector of weights and the structured feature map  $\Psi(x, y)$  contains the relevant information for the feature-label pair  $(x, y)$ . To learn the parameters using the predefined probabilistic model, one can use the (regularized) maximum likelihood approach:

$$\max_w \frac{1}{N} \sum_{n=1}^N \log P(y^n|x^n, w) - \frac{\lambda}{2} \|w\|^2, \quad (3)$$

where  $\lambda > 0$  is a regularization parameter and the likelihood  $P(y|x, w)$  is defined as  $\frac{\exp(f(y|x))}{Z(f, x)} = \exp(f(y|x) - A(f, x))$ , where  $A(f, x)$  is the log-partition function (that now depends on  $x$ , since we consider conditional models).

Hazan and Jaakkola (2012) proposed to learn parameters based on the Gumbel bound  $A_G(f, x)$  instead of the intractable log-partition function:

$$\begin{aligned} \log P(y|x) &= f(y|x) - A(f, x) \leq f(y|x) - A_G(f, x) \\ &= f(y|x) - \mathbb{E}_z \left[ \max_{y \in Y} \left\{ \sum_{d=1}^D z_d(y_d) + f(y) \right\} \right] \\ &\approx f(y|x) - \frac{1}{M} \sum_{m=1}^M \max_{y^{(m)} \in Y} \left\{ \sum_{d=1}^D z_d^{(m)}(y_d^{(m)}) + f(y^{(m)}) \right\}. \end{aligned}$$

Hazan and Jaakkola (2012) considered the fully-supervised setup where labels  $y^n$  were given for all data points  $x^n$ . Shpakova and Bach (2016) developed the approach, but also considered a setup with missing data (part of the labels  $y^n$  are unknown) for the small Weizmann Horse dataset (Borenstein et al., 2004). Leveraging the additional stochasticity present in the Gumbel samples, Shpakova and Bach (2016) extend the use of stochastic gradient descent, not on the data as usually done, but on the Gumbel randomization. It is equivalent to the choice of parameter  $M = 1$  for every gradient computation (but with a new Gumbel sample at every iteration). In our work, we use the stochastic gradient descent in a regime stochastic w.r.t. both the data and the Gumbel perturbations. This allows us to apply the method to large-scale datasets.

For linear models, we have  $f(y|x) = w^T \Psi(x, y)$  and  $\Psi(x, y)$  is usually given or takes zero effort to compute. We assume that the gradient calculation  $\nabla_w f(y|x) = \Psi(x, y)$  does not add complexity to the optimization algorithm. The gradient of  $\log P(y|x)$  is equal to  $\nabla_w f(y|x) - \nabla_w \max_{y \in Y} \left\{ \sum_{d=1}^D z_d(y_d) + f(y|x) \right\} = \nabla_w f(y|x) - \nabla_w f(y^*|x)$ , where  $y^*$  lies in  $\arg \max$  of the perturbed optimization problem. The gradient of  $\langle \log P(y|x) \rangle$  (the average over a subsample of data, typically a mini-batch) has the form  $\langle \nabla_w f(y|x) \rangle - \langle \nabla_w f(y^*|x) \rangle = \langle \Psi(x, y) \rangle_{emp.} - \langle \Psi(x, y^*) \rangle$ , where  $\langle \Psi(x, y) \rangle_{emp.}$  denotes the empirical average over the data. Algorithm 1 contains this double stochastic gradient descent (SGD) with stochasticity w.r.t. both sampled data and Gumbel samples. The choice of the stepsize  $\gamma_h = \frac{1}{\lambda h}$  is standard for strongly-convex problems (Shalev-Shwartz et al., 2011).

Note, that the classic log-likelihood formulation (3) is implicitly considering a ‘‘0-1 loss’’  $l_{0-1}(y, \hat{y}) = [y \neq \hat{y}]$  as it takes probability of the entire output object  $y^n$  conditioned on the observed feature representation  $x^n$ .

However, in many structured-output problems 0-1 loss

---

**Algorithm 1** Double SGD: stochasticity w.r.t. data and Gumbel samples

---

**Input:** dataset  $\mathcal{D} = \{(x^n, y^n)\}_{n=1}^N$ , number of iterations  $H$ , size of the mini-batch  $T$ , stepsize sequence  $\{\gamma_h\}_{h=1}^H$ , regularization parameter  $\lambda$

**Output:** model parameters  $w$

- 1: **Initialization:**  $w = 0$
  - 2: **for**  $h = 1$  **to**  $H$  **do**
  - 3:   Sample data mini-batch of small size  $T$  (that is,  $T$  pairs of observations)
  - 4:   Calculate sufficient statistics  $\langle \Psi(x, y) \rangle_{emp.}$  from the mini-batch
  - 5:   **for**  $t=1$  **to**  $T$  **do**
  - 6:     Sample  $z_d(y_d)$  as independent Gumbels for all  $y_d \in Y_d$  and for all  $d$
  - 7:     Find  $y^* \in \arg \max_{y \in Y} \left\{ \sum_{d=1}^D z_d(y_d) + f(y) \right\}$
  - 8:   **end for**
  - 9:   Make a gradient step:  
 $w_{h+1} \rightarrow w_h + \gamma_h \left( \langle \Psi(x, y) \rangle_{emp.} - \langle \Psi(x, y^*) \rangle - \lambda w_h \right)$
  - 10: **end for**
- 

evaluation is not an adequate performance measure. The Hamming or weighted Hamming losses that sum mistakes across the  $D$  elements of the outputs, are more in demand as they count misclassification per element.

### 2.3 Marginal probability estimation

Either at testing time (to provide an estimate of the uncertainty of the model) or at training time (in the case of weak supervision, see Section 4), we need to compute marginal probabilities for a single variable  $y_d$  out of the  $d$  ones, that is,

$$P(y_d|x) = \sum_{y_{-d}} P(y_{-d}, y_d|x),$$

where  $y_{-d}$  is a sub-vector of  $y$  obtained by elimination of the variable  $y_d$ . Following Hazan and Jaakkola (2012) and Shpakova and Bach (2016), this can be obtained by taking  $m$  Gumbel samples and the associated maximizers  $y^m \in Y = Y_1 \times \dots \times Y_D$ , and, for any particular  $d$ , counting the number of occurrences in each possible value in all the  $d$ -th components  $y_d^m$  of the maximizers  $y^m$ .

While this provides an estimate of the marginal probability, this is not an easy expression to optimize at it depends on several maximizers of potentially complex optimization problems. In the next section, we show how we can compute a different (and new) approximation which is easily differentiable and on which we can apply stochastic gradient descent.

## 3 MARGINAL LIKELIHOOD

In this section, we demonstrate the learning procedure for the element-decoupled losses. We consider the regularized empirical risk minimization problem in a general form:

$$\max_w \frac{1}{N} \sum_{n=1}^N \ell(w, x^n, y^n) - \frac{\lambda}{2} \|w\|^2, \quad (4)$$

where  $\ell(w, x, y)$  can take various forms from Table 1 and  $\lambda$  is the regularization parameter. The choice of the likelihood form is based on the problem setting such as presence of missing data and the considered test-time evaluation function.

### 3.1 Hamming loss

The Hamming loss is a loss function that counts misclassification per dimension:  $l_h(y, \hat{y}) = \frac{1}{D} \sum_{d=1}^D [y_d \neq \hat{y}_d]$ . For this type of loss instead of the classic log-likelihood objective it is more reasonable to consider the following decoupling representation from Table 1:

$$\ell(w, x, y) = \sum_{d=1}^D \log P(y_d|x, w), \quad (5)$$

where

$$\begin{aligned} P(y_d|x, w) &= \sum_{y_{-d}} \exp(f(w, y_{-d}|y_d, x) - A(f, x)) \\ &= \exp(B(f, y_d, x) - A(f, x)) \end{aligned}$$

is the marginal probability of the single element  $y_d$  given the entire input  $x$ , and  $B(f, y_d) = \log \sum_{y_{-d}} \exp(f(w, y_{-d}|y_d))$ , where  $y_{-d} \in Y_1 \times \dots \times Y_{d-1} \times Y_{d+1} \times \dots \times Y_D$ . Thus, the log-marginal probability may be obtained from the difference of two log-partition functions (which we will approximate below with Gumbel samples).

This idea of considering the marginal likelihood was proposed by Kakade et al. (2002). Our contribution is to consider the approximation by ‘‘perturb-and-MAP’’ techniques. We thus have a new objective function  $\ell(w, x, y)$ :  $\ell(w, x, y) = \sum_{d=1}^D [(B(f, x, y_d) - A(f, x))]$ , and now the following approximation could be applied:

$$\begin{aligned} A(f) &\approx A_G(f) = \mathbb{E}_z \left\{ \max_{y \in Y} \sum_{d=1}^D z_d(y_d) + f(y) \right\}, \\ B(f|y_d) &\approx B_G(f|y_d) \\ &= \mathbb{E}_z \left\{ \max_{y_{-d} \in Y_{-d}} \sum_{s=1: s \neq d}^D z_s(y_s) + f(y_{-d}|y_d) \right\}. \end{aligned}$$

It is worth noting, that the approximation is not anymore an upper bound of the marginal likelihood; moreover it is a difference of convex functions. Remarkably, the objective function exactly matches the log-likelihood in the

Table 1: Variants of the Objective Loss  $\ell(w, x, y)$  Function.  $\{\theta_d(y_d)\}_{d=1}^D$  are the weights of the weighted Hamming loss,  $\{q_d(y_d)\}_{d=1}^D$  are the marginal probabilities  $P(y_d|x)$ .

Loss	Labelled Data	Unlabelled Data
0-1	$\log P(w, y x)$	$\log \sum_{y \in Y} P(w, y, x)$
Hamming	$\sum_{d=1}^D \log P(w, y_d x_d)$	$\sum_{d=1}^D \sum_{y_d \in Y_d} q_d(y_d) \log P(w, y_d x_d)$
Weighted Hamming	$\sum_{d=1}^D \theta_d(y_d) \log P(w, y_d x_d)$	$\sum_{d=1}^D \sum_{y_d \in Y_d} q_d(y_d) \theta_d(y_d) \log P(w, y_d x_d)$

case of unary potentials (separable potential function) as the log-likelihood function becomes the sum of marginal likelihoods.

As noted, the objective  $\ell(w, x, y)$  is not convex anymore, but it is presented as the difference of two convex functions. We can still try to approximate with stochastic gradient descent (which then only converges to a stationary point, typically a local minimum). Algorithm 2 describes the implementation details.

---

**Algorithm 2** Double SGD for Marginal Likelihood

---

**Input:** dataset  $\mathcal{D} = \{(x^n, y^n)\}_{n=1}^N$ , number of iterations  $H$ , size of the mini-batch  $T$ , stepsize sequence  $\{\gamma_h\}_{h=1}^H$ , regularization parameter  $\lambda$

**Output:** model parameters  $w$

- 1: **Initialization:**  $w = 0$
  - 2: **for**  $h = 1$  **to**  $H$  **do**
  - 3:   Sample data mini-batch of small size  $T$  (that is,  $T$  pairs of observations)
  - 4:   **for**  $t=1$  **to**  $T$  **do**
  - 5:     Sample  $z_d(y_d)$  as independent Gumbels for all  $y_d \in Y_d$  and for all  $d$
  - 6:     Find  $y_A^* \in \arg \max_{y \in Y} \left\{ \sum_{d=1}^D z_d(y_d) + f(y) \right\}$
  - 7:     **for**  $d=1$  **to**  $D$  **do**
  - 8:       Find  $y_B^* \in \arg \max_{y-d \in Y-d} \left\{ \sum_{s=1: s \neq d}^D z_s(y_s) + f(y-d|y_d) \right\}$
  - 9:     **end for**
  - 10:   **end for**
  - 11:   Make a gradient step:  
 $w_{h+1} \rightarrow w_h + \gamma_h \left( \langle \Psi(x, y_B^*) \rangle - \langle \Psi(x, y_A^*) \rangle - \lambda w_h \right)$
  - 12: **end for**
- 

**Acceleration trick.** Interestingly we can use the same Gumbel perturbation realizations for approximating  $A_G(f)$  and  $B_G(f|y_d)$  through an empirical average. On the one hand, this restriction should not influence on the result as with a sufficient large averaging number  $M$ ,  $A_G(f)$  and  $B_G(f|y_d)$  converges to their expectations. This is the same for stochastic gradients: on every itera-

tion, we use a different Gumbel perturbation, but we share this one for the estimation of the gradients of  $A_G(f)$  and  $B_G(f|y_d)$ . This allows us to save some computations as shown below, while preserving convergence (the extra correlation added by using the same samples for the two gradients does not change the unbiasedness of our estimates).

Moreover, if  $y_A^*$  has the same label value  $y_d$  as the ground truth, then the MAP inference problem for  $y_A^*$  exactly matches the one for  $y_B^*$  (with the element  $y_d$  fixed from the ground truth). Then  $y_A^* = y_B^*$  and the corresponding difference of gradients gives zero impact into the gradient step. This fact allows us to reduce the number of MAP-inference problems. We should thus calculate  $y_B^*$  only for those indices  $d$  that leads to a mismatch between  $d$ -th label of  $y_A^*$  and the ground truth one. Remarkably, during the convergence to the optimal value, the reduction will occur more often and decrease the execution time with the number of iteration increase. Besides that, in the experiments with graph cuts in Section 5 we use dynamic graph cut algorithm for solving several optimization problems of similar structure (here  $D$  marginal probabilities calculation). We describe it in more details in Section 3.3.

### 3.2 Weighted Hamming loss

The weighted Hamming loss is used for performance evaluation in the models, where each dimension has its own level of importance, e.g., in an image segmentation with superpixels, proportional to the size of superpixels. It differs from the usual Hamming loss in this way:  $l_h(y, \hat{y}) = \frac{1}{D} \sum_{d=1}^D \theta_d(y_d) [y_d \neq \hat{y}_d]$ .

Thus we consider a dimension-weighted model as it can be adjusted for the problem of interest that gives the model more flexibility. The optimization problem of interest is transformed from the previous case by weighted multiplication:

$$\ell(w, x, y) = \sum_{d=1}^D \theta_d(y_d) [(B(f, x, y_d) - A(f, x))]. \quad (6)$$

To justify this objective function, we notice that in the case of unit weights, the weighted loss and objective

function (6) match the loss and the objective from the previous section. Furthermore,  $y_d$  with a large weight  $\theta_d(y_d)$  puts more importance towards making the right prediction for this  $y_d$ , and that is why we put more weight on the  $d$ -th marginal likelihood. This corresponds to the usual rebalancing used in binary classification (see, e.g., Bach et al., 2006, and references therein). Then, the algorithm for this case duplicated the one for the usual Hamming loss and the acceleration trick can be used as well.

### 3.3 Scalable algorithms for graph cuts

As a classical efficient MAP-solver for pairwise potentials problem we will use graph cut algorithms from Boykov and Kolmogorov (2004). The function  $f(y|x)$  should then be supermodular, i.e., with pairwise potentials, all pairwise weights of  $w$  should remain negative.

In both Sections 3.1 and 3.2 we can apply the dynamic graphcut algorithm proposed by Kohli and Torr (2007), which is a modification of the Boykov-Kolmogorov graphcut algorithms. It is dedicated to situations when a sequence of graphcut problems with slightly different unary potentials need to be solved. Then, instead of solving them separately, we can use the dynamic procedure and find the solutions for slightly different problems with less costs. This makes graphcut scalable for a special class of problems.

It can easily be seen that our sequence of problems  $y_B^* \in \arg \max_{y_{-d} \in Y_{-d}} \{ \sum_{s=1: s \neq d} z_s(y_s) + f(y_{-d}|y_d) \}$  for  $d = 1, \dots, D$  is a perfect application for the dynamic graph cut algorithm. At each iteration we solve  $T$  sets of graph cut problems, each of set contains  $1 + a_t$  problems solvable by the same dynamic cut, where  $a_t$  is the number of not matched pixels between  $y_A^*$  and ground truth  $y^n$ . Finally, using acceleration trick and dynamic cuts we reduce the gradient descent iteration complexity from  $\sum_{t=1}^T (1 + D_t)$  graphcut problems to  $T$  dynamic graph cut problems. We make the approach scalable and can apply it for large datasets.

## 4 PARAMETER LEARNING IN THE SEMISUPERVISED SETUP

In this section we assume the presence of objects with unknown labels in the train dataset. We can separate the given data in two parts: fully annotated data  $\mathcal{D}_1 = \{(x^n, y^n)\}_{n=1}^N$  as in the supervised case and unlabeled data  $\mathcal{D}_2 = \{x^l\}_{l=1}^L$ . Then, the optimal model parameter  $w$  is a solution of the following optimization problem:

$$\max_w L_1(w) + \kappa L_2(w) - \frac{\lambda}{2} \|w\|^2, \quad (7)$$

where  $L_1(w) = \sum_{n=1}^N \ell_1(w, x^n, y^n)$ ,  $L_2(w) =$

$\sum_{l=1}^L \ell_2(w, x^l)$  and the parameter  $\kappa$  governs the importance of the unlabeled data.  $\ell_1(w, x^n, y^n)$  can have a form from the left column of the Table 1, and  $\ell_2(w, x^l)$  from the right one.

**Marginal calculations.** It is worth reminding from Section 2.3, that we can approximate marginal probabilities  $q(y)$  of holding  $y_d = k$  along with the partition function approximation almost for free. This can be obtained by taking  $m$  Gumbel samples and the associated maximizers  $y^m \in Y = Y_1 \times \dots \times Y_D$ , and, for any particular  $d$ , counting the number of occurrences in each possible value in all the  $d$ -th components  $y_d^m$  of the maximizers  $y^m$ . The approximation accuracy depends on number of samples  $M$ . To calculate this we already need to have a trained weight vector  $w$  which we can obtain from the fully annotated dataset  $\mathcal{D} = \{(x^n, y^n)\}_{n=1}^N$ . We will calculate  $q(y)$  for the unlabelled data  $\mathcal{D}_2 = \{x^l\}_{l=1}^L$ . Those marginal probabilities contain much more information than MAP inference for the new data as can be seen on the example in Figure 1. We believe that proper use of the marginal probabilities will help to gain better result than using labels from the MAP inference (which we observe in experiments).

It is worth noting that for the inference and learning phases we use a different number of Gumbel samples. During the learning phase, we incorporate the double stochastic procedure and use 1 sample per 1 iteration and 1 label. For the marginal calculation (inference) we should use large number of samples (e.g. 100 samples) to get accurate approximation.

---

**Algorithm 3** Sketch for the semisupervised algorithm.

---

**Input:** fully annotated dataset  $\mathcal{D}_1 = \{(x^n, y^n)\}_{n=1}^N$ , number of iterations  $H$ , size of the mini-batch  $T$ , stepsize sequence  $\{\gamma_h\}_{h=1}^H$ , regularization param.  $\lambda$

**Output:** model parameters  $w_1$

1: **Initialization:**  $w_1 = 0$

2: **Find  $w_1$  via Algorithm 2**

**Input:** fully annotated dataset  $\mathcal{D}_1 = \{(x^n, y^n)\}_{n=1}^N$ , unlabeled dataset  $\mathcal{D}_2 = \{x^l\}_{l=1}^L$ , number of iterations  $H$ , size of the mini-batch  $T$ , stepsize sequence  $\{\gamma_h\}_{h=1}^H$ , regularization parameter  $\lambda$

**Output:** model parameters  $w_{1,2}$

3: **Initialization:**  $w_{1,2} = w_1$

4: **Calculate:**  $q(y)$  for unlabeled data via  $w_1$

5: **Find  $w_{1,2}$  via mixture of Algorithms 2 and 4**

---

We provide the sketch of the proposed optimization algorithm in Algorithm 3. The optimization of  $L_1$  is fully supervised and this can be done with tools of the previous section. The optimization of  $L_2$  requires the specification of  $\ell_2(w, x)$ , which we take as



$\ell_2(w, x) = \sum_{d=1}^D \sum_{y_d \in \{0, \dots, K\}} q_d(y_d) \log P(w, y_d | x_d) =$   
 $\sum_{d=1}^D \sum_{y_d \in \{0, \dots, K\}} q_d(y_d) B(f|y_d) - DA(f)$ , that is, the average of the fully supervised cost function with labels generated from the model  $q$ . The term  $L_2$  corresponds to the common way of treating unlabeled data through the marginal likelihood. The sub-algorithm for the optimization of  $\ell_2$  is presented as Algorithm 4.

---

**Algorithm 4** Double SGD for Unsupervised Learning

---

**Input:** unlabeled dataset  $\mathcal{D}_2 = \{x^l\}_{l=1}^L$ , parameter estimate  $w_1$ , number of iterations  $H$ , size of the mini-batch  $T$ , stepsize sequence  $\{\gamma_h\}_{h=1}^H$ , regularization parameter  $\lambda$

**Output:** model parameters  $w_{1,2}$

- 1: **Initialization:**  $w_{1,2} = w_1$
  - 2: **for**  $h = 1$  **to**  $H$  **do**
  - 3:   Sample data mini-batch of small size  $T$  (that is,  $T$  pairs of observations)
  - 4:   **for**  $t=1$  **to**  $T$  **do**
  - 5:     Sample  $z_d(y_d)$  as independent Gumbels for all  $y_d \in Y_d$  and for all  $d$
  - 6:     Find  $y_A^* \in \arg \max_{y \in Y} \left\{ \sum_{d=1}^D z_d(y_d) + f(y) \right\}$
  - 7:     **for**  $d=1$  **to**  $D$  **and**  $k=0$  **to**  $K$  **do**
  - 8:       Find  $y_{B,d,k}^* \in \arg \max_{y_{-d} \in Y_{-d}} \left\{ \sum_{s=1: s \neq d}^D z_s(y_s) + f(y_{-d} | y_d = k) \right\}$
  - 9:     **end for**
  - 10:   **end for**
  - 11:   Make a gradient step:  

$$w_{h+1} \rightarrow w_h + \gamma_h \left( \left\langle \sum_{k=0}^K q_d(k) \Psi(x, y_{B,d,k}^*) \right\rangle - \Psi(x, y_A^*) \right) - \lambda w_h$$
  - 12: **end for**
- 

**Acceleration trick.** Suppose, that  $y_d$  can take values in the range  $\{0, \dots, K\}$ . Again we use the same Gumbel perturbation for estimating  $A_G(f)$  and  $B_{dkG}(f|y_d = k)$  for all  $k \in \{0, \dots, K\}$ . The consequence of using the same perturbations is that if the  $d$ -th label  $y_d$  of  $y_A^*$  takes value  $k$ , then the corresponding  $d$ -th gradient will cancel out with one of the  $y_{B,k}^*$ . Thus, we will calculate only  $K$  (instead of  $K + 1$  labels) structured labels  $y_{B,l}^* (l \neq k)$  and reduce the number of optimization problems to be solved. Dynamic graph cuts are applied here as well.

Finally in Table 1 we see the relationships between the proposed objective functions. Firstly, the known labels  $y^n$  in the supervised case are equivalent to the binary marginal probabilities  $q(y^n) \in \{0, 1\}^{D^n}$ . Secondly, the unit weights  $\theta_d(y_d) = 1$  in the weighted Hamming loss are equivalent to the basic Hamming loss.

**Partial labels.** Another case that we would like to mention is annotation with partial labels, e.g., in an image segmentation application, the bounding boxes of the images are given. Then denote  $y^{given}$  as the set of given labels. In this setup the marginal probabilities become conditional ones  $q(y_d | y^{given})$  and to approximate this we need to solve several conditional MAP-inference problems. The objective function  $\ell_2(w) = \sum_d \sum_{y_d \in \{0, \dots, K\}} q_d(y_d | y^{given}) \log P(w, y_d | x_d, y^{given})$  remains feasible to optimize.

## 5 EXPERIMENTS

The experimental evaluation consists of two parts: Section 5.1 is dedicated to the chain model problem, where we compare the different algorithms for supervised learning; Section 5.2 is focused on evaluating our approach for the pairwise model on a weakly-supervised problem.

### 5.1 OCR dataset

The given OCR dataset from Taskar et al. (2003) consists of handwritten words which are separated in letters in a chain manner. The OCR dataset contains 10 folds of  $\sim 6000$  words overall. The average length of the word is  $\sim 9$  characters. Two traditional setups of these datasets are considered: 1) “small” dataset when one fold is considered as a training data and the rest is for test, 2) “large” dataset when 9 folds of 10 compose the train data and the rest is the test data. We perform cross-validation over both setups and present results in Table 2.

As the MAP oracle we use the dynamic programming algorithm of Viterbi (1967). The chain structure also allows us to calculate the partition function and marginal probabilities exactly. Thus, the CRF approach can be applied. We compare its performance with the structured SVM from Osokin et al. (2016), perturb-and-MAP (Hazan and Jaakkola, 2012) and the one we propose for marginal perturb-and-MAP (as Hamming loss is used for evaluation).

The goal of this experiment is to demonstrate that the CRF approach with exact marginals shows a slightly worse performance as the proposed one with approximated marginals but correct Hamming loss.

Table 2: OCR Dataset. Performance Comparison.

method	small dataset	large dataset
CRF	19.5 $\pm$ 0.4	13.1 $\pm$ 0.8
S-SVM+BCFW	19.5 $\pm$ 0.4	12.0 $\pm$ 1.1
perturb&MAP	19.1 $\pm$ 0.3	12.5 $\pm$ 1.1
marg. perturb&MAP	19.1 $\pm$ 0.3	12.8 $\pm$ 1.2

For the OCR dataset, we performed 10-fold cross-validation and the numbers of Table 2 correspond to the averaged loss function (Hamming loss) values over the 10 folds. As we can see from the result in Table 2, the approximate probabilistic approaches slightly outperforms the CRF on both datasets. The Gumbel approximation (with or without marginal likelihoods) does lead to a better estimation for the Hamming loss. Note that S-SVM performs better in the case of a larger dataset, which might be explained by stronger effects of model misspecification that hurts probabilistic models more than S-SVM (Pletscher et al., 2011).

## 5.2 HorseSeg dataset

The problem of interest is foreground/background superpixel segmentation. We consider a training set of images  $\{x^n\}_{n=1\dots N}$  that contain different numbers of superpixels. A hard segmentation of the image is expressed by an array  $y^n \in \{0, 1\}^{D^n}$ , where  $D^n$  is the number of superpixels for the  $n$ -th image.

The HorseSeg dataset was created by Kolesnikov et al. (2014) and contains horse images. The “small” dataset has images with manually annotated labels and contains 147 images. The second “medium” dataset is partially annotated (only bounding boxes are given) and contains 5974 images. The remaining “large” one has 19317 images with no annotations at all. A fully annotated hold out dataset was used for the test stage. It consists of 241 images.

The graphical model is a pairwise model with loops. We consider log-supermodular distribution and thus, the max oracle is available as the graph cut algorithm by Boykov and Kolmogorov (2004). Note that CRFs with exact inference cannot be used here.

Following Kolesnikov et al. (2014), for the performance evaluation the weighted Hamming loss is used, where the weight is governed by the superpixel size and foreground/background ratio in the particular image.

That is,  $l_h(y, \hat{y}) = \frac{1}{D} \sum_{d=1}^D \theta_d(y_d)[y_d \neq \hat{y}_d]$ , where

$$\theta_d(y_d) = \begin{cases} \frac{V_d}{2V_{foreground}}, & \text{if } y_d = 1. \\ \frac{V_d}{2V_{background}}, & \text{if } y_d = 0. \end{cases}$$

$V_d$  is the size of superpixel  $d$ ,  $V_{background}$  and  $V_{foreground}$  are the sizes of the background and the foreground respectively. In this way smaller object sizes have more penalized mistakes.

Since we incorporate  $\theta(y)$  into the learning process and for its evaluation we need to know the background and foreground sizes of the image, this formulation is only applicable for the supervised case, where  $y_d$  is given for

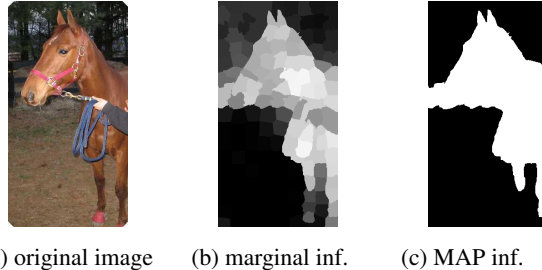


Figure 1: Example of the marginal and MAP inference for an image from the HorseSeg database Kolesnikov et al. (2014).

Table 3: HorseSeg Dataset. Performance Comparison.

method	“small”	“medium”	“large”
S-SVM+BCFW	12.3	10.9	10.9
perturb&MAP	20.9	21.0	20.9
w.m. perturb&MAP	11.6	10.9	10.9

all superpixels. However, in this dataset we have plenty of images with partial or zero annotation. For these set of images  $\mathcal{D}_2 = \{x^l\}_{l=1}^L$ , we handle approximate marginal probabilities  $q_d^l$  associated to the unknown labels. Using them we can approximate the foreground and background volumes:  $V_{foreground}^l \approx \sum_{d=1}^{D^l} q_d^l$  and  $V_{background}^l \approx \sum_{d=1}^{D^l} (1 - q_d^l)$ .

We provide an example of the marginal and MAP inference in Figure 1. The difference of the information compression between these two approaches is visually comparable. We believe that the smoother and accurate marginal approach should have a positive impact on the result, as the uncertainty about the prediction is well propagated.

As an example of the max-margin approaches we take S-SVM+BCFW from the paper of Osokin et al. (2016) which is well adapted to large-scale problems. For S-SVM+BCFW and perturb-and-MAP methods we use MAP-inference for labelling unlabelled data using  $w_1$  (see Section 4).

For the HorseSeg dataset (Table 3), the numbers correspond to the averaged loss function (weighted Hamming loss) values over the hold out test dataset. The results of the experiment in Table 3 demonstrate that the approaches taking into account the weights of the loss  $\theta_d(\cdot)$  (S-SVM+BCFW and w.m. perturb&MAP) give a much better accuracy than the regular perturb&MAP. S-SVM+BCFW uses loss-augmented inference and thereby augments the weighted loss structure into the learning phase. Weighted marginal perturb-and-MAP plugs the weights of the weighted Hamming loss inside the objective log-likelihood function. Basic perturb-and-MAP does

Table 4: Reduced HorseSeg Dataset. Performance Comparison.

method	10% of “small”	“medium” with bbox	“medium” w/o bbox
S-SVM+BCFW	17.3	14.0	16.1
perturb&MAP	23.2	23.4	23.0
w.m. p.&MAP	18.1	13.7	14.4

not use the weights  $\theta_a(\cdot)$  and loses a lot of accuracy. This shows us that the predefined loss for performance evaluation has a significant influence on the result and should be taken into account.

**Small dataset size influence.** We now investigate the effect of the reduced “small” dataset. We preserve the setup from the previous section and the only thing that we change is  $N$ , the size of the “small” fully-annotated dataset  $\mathcal{D}_1 = \{(x^n, y^n)\}_{n=1}^N$ . The new “small” dataset is 10% the size of the initial one, i.e., only 14 images. By taking a small labelled dataset, we test the limit of supervised learning when few labels are present.

For the HorseSeg dataset (Table 4), the numbers correspond to the averaged loss function (weighted Hamming loss) values over the hold out test dataset. The results of this experiments are presented in Table 4. In this setup, the probabilistic approach “weighted marginal perturb-and-MAP” gains more than max-margin “S-SVM+BCFW”. This could happen because of very limited fully supervised data. The learned parameter  $w_1$  gives a noisy model and this noisy model produces a lot of noisy labels for the unlabeled data, while weighted perturb-and-MAP is more cautious as it uses probabilities that contain more information (see Figure 1).

### 5.2.1 Acceleration trick impact

We now compare the execution time of the algorithm with and without our acceleration techniques (namely Dynamic Cuts [DC] and Gumbel Reduction[ GR]) to get an idea on how helpful they are. Table 5 shows the execution time for calculating all  $y_B^*$  (Algorithm 2) for different numbers of iterations on the HorseSeg small dataset. We conclude that the impact of DC does not depend on the total number of iterations always leading to acceleration around 1.3. For GR, acceleration goes from 3.5 for 100 iterations to 7.6 for one million iterations. Overall, we get acceleration of factor around 10 for one million iterations.

## 5.3 Experiments analysis

The experiments results mainly show that not taking into account the right loss in the learning procedure is detrimental to probabilistic technique such as CRFs, while

method \ it	100	$10^3$	$10^4$	$10^5$	$10^6$
basic	0.9	9.2	89.5	900	8993
DC	0.7	6.9	69.0	696	7171
GR	0.3	2.1	15.5	133.4	1186
DC+GR	0.2	1.5	10.9	83.5	727

Table 5: Execution time comparison in seconds. HorseSeg small dataset.

taking it into account (our novelty) improves results. Also, Tables 2 and 3 show that the proposed methods achieves (and sometimes surpasses) the level performance of the max-margin approach (with loss-augmented inference).

Further, we observed that the size of the training set influences the SSVM and perturb-and-MAP approaches differently. For smaller datasets, the max-margin approaches tend to lose information due to usage of the hard estimates for the unlabelled data (e.g. in Table 4: 16.1 against 14.4 for “medium” dataset without bounding boxes labeling).

Table 4 reports an experiment about using weakly-labeled data at the training stage (the results on the partially annotated “medium” dataset). This experiment studied the impact on the final prediction quality of the training set of “medium” size on top of the reduced “small” fully-labelled set. The results of Table 4 mean that the usage of our approach adopted to the correct test measure outperforms the default perturb-and-MAP by a large margin. Our approach also significantly outperformed the comparable baseline of SSVM due to reduced size of the “small” fully-labelled set.

## 6 CONCLUSION

In this paper, we have proposed an approximate learning technique for problems with non-trivial losses. We were able to make marginal weighted log-likelihood for perturb-and-MAP tractable. Moreover, we used it for semi-supervised and weakly-supervised learning. Finally, we have successfully demonstrated good performance of the marginal-based and weighted-marginal-based approaches on the middle-scale experiments. As a future direction, we can go beyond the graph cuts and image segmentation application and consider other combinatorial problems with feasible MAP-inference, e.g., matching.

### Acknowledgements

We acknowledge support the European Union’s H2020 Framework Programme (H2020-MSCA-ITN-2014) under grant agreement n°642685 MacSeNet. This research was also partially supported by Samsung Research, Samsung Electronics, and the European Research Council (grant SEQUOIA 724063).

## References

- F. Bach. Learning with submodular functions: a convex optimization perspective. *Foundations and Trends in Machine Learning*, 6(2-3):145–373, 2013.
- F. Bach, D. Heckerman, and E. Horvitz. Considering cost asymmetry in learning classifiers. *Journal of Machine Learning Research (JMLR)*, 7:1713–1741, 2006.
- G. Bakir, T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar, and S. V. N. Vishwanathan. *Predicting Structured Data*. The MIT Press, 2007.
- C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.
- E. Borenstein, E. Sharon, and S. Ullman. Combining Top-down and Bottom-up Segmentation. In *Proc. ECCV*, 2004.
- E. Boros and P. L. Hammer. Pseudo-boolean optimization. *Discrete Applied Mathematics*, 123(1):155–225, 2002.
- Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 26(9):1124–1137, 2004.
- J. Djolonga and A. Krause. From MAP to Marginals: Variational Inference in Bayesian Submodular Models. In *Adv. NIPS*, 2014.
- J. Djolonga and A. Krause. Scalable Variational Inference in Log-supermodular Models. In *Proc. ICML*, 2015.
- R. B. Girshick, P. F. Felzenszwalb, and D. A. Mcallester. Object detection with grammar models. In *Adv. NIPS*, 2011.
- T. Hazan and T. Jaakkola. On the partition function and random maximum a-posteriori perturbations. In *Proc. ICML*, 2012.
- T. Hazan, S. Maji, J. Keshet, and T. Jaakkola. Learning efficient random maximum a-posteriori predictors with non-decomposable loss functions. In *Adv. NIPS*, 2013.
- M. Jerrum and A. Sinclair. Polynomial-time approximation algorithms for the Ising model. *SIAM Journal on Computing*, 22(5):1087–1116, 1993.
- M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.
- S. Kakade, Y. W. Teh, and S. T. Roweis. An alternate objective function for markovian fields. In *Proc. ICML*, 2002.
- P. Kohli and P. H. Torr. Dynamic graph cuts for efficient inference in markov random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 29(12):2079–2088, 2007.
- A. Kolesnikov, M. Guillaumin, V. Ferrari, and C. H. Lampert. Closed-form training of conditional random fields for large scale image segmentation. In *Proc. ECCV*, 2014.
- V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 26(2):147–159, 2004.
- N. Komodakis, N. Paragios, and G. Tziritas. MRF energy minimization and beyond via dual decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 33(3):531–552, 2011.
- M. P. Kumar, B. Packer, and D. Koller. Self-paced learning for latent variable models. In *Adv. NIPS*, 2010.
- S. Lacoste-Julien, M. Jaggi, M. Schmidt, and P. Pletscher. Block-coordinate Frank-Wolfe optimization for structural SVMs. In *Proc. ICML*, 2013.
- J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. ICML*, 2001.
- S. Nowozin and C. H. Lampert. Structured learning and prediction in computer vision. *Foundations and Trends in Computer Graphics and Vision*, 6(3–4):185–365, 2011.
- A. Osokin and D. P. Vetrov. Submodular relaxation for inference in Markov random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 37(7):1347–1359, 2015.
- A. Osokin, J.-B. Alayrac, I. Lukasewitz, P. K. Dokania, and S. Lacoste-Julien. Minding the gaps for block Frank-Wolfe optimization of structured SVMs. In *Proc. ICML*, 2016.
- G. Papandreou and A. Yuille. Perturb-and-MAP random fields: Using discrete optimization to learn and sample from energy models. In *Proc. ICCV*, 2011.
- S. Parise and M. Welling. Learning in Markov random fields: an empirical study. In *Joint Statistical Meeting (JSM)*, 2005.
- P. Pletscher, S. Nowozin, P. Kohli, and C. Rother. Putting MAP back on the map. In *33rd Annual Symposium of the German Association for Pattern Recognition*, 2011.
- S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter. Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical Programming*, 127(1):3–30, 2011.
- T. Shpakova and F. Bach. Parameter learning for log-supermodular distributions. In *Adv. NIPS*, 2016.
- N. A. Smith. Linguistic structure prediction. *Synthesis Lectures on Human Language Technologies*, 4(2):1–274, 2011.

- D. Sontag, T. Meltzer, A. Globerson, Y. Weiss, and T. Jaakkola. Tightening LP relaxations for MAP using message-passing. In *Proc. UAI*, 2008.
- C. Sutton and A. McCallum. Piecewise training of undirected models. In *Proc. UAI*, 2005.
- C. Sutton and A. McCallum. Piecewise pseudolikelihood for efficient training of conditional random fields. In *Proc. ICML*, 2007.
- B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Adv. NIPS*, 2003.
- I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research (JMLR)*, 6:1453–1484, 2005.
- S.V.N. Vishwanathan, N. N. Schraudolph, M. W. Schmidt, and K. P. Murphy. Accelerated training of conditional random fields with stochastic gradient methods. In *Proc. ICML*, 2006.
- A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, IT-13(2):260–269, 1967.
- M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.
- Y. Weiss. Comparing the mean field method and belief propagation for approximate inference in MRFs. *Advanced Mean Field Methods: Theory and Practice*, pages 229–240, 2001.
- R. J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.
- C.-N. J. Yu and T. Joachims. Learning structural svms with latent variables. In *Proc. ICML*, 2009.
- A. L. Yuille and A. Rangarajan. The concave-convex procedure. In *Adv. NIPS*, 2003.

---

# Variational Inference for Gaussian Processes with Panel Count Data

---

Hongyi Ding<sup>13</sup>, Young Lee<sup>2</sup>, Issei Sato<sup>13</sup>, Masashi Sugiyama<sup>31</sup>

<sup>1</sup>The University of Tokyo, Japan

<sup>2</sup>National Univeristy of Singapore, Singapore

<sup>3</sup>The RIKEN Center for AIP, Tokyo, Japan

## Abstract

We present the first framework for Gaussian-process-modulated Poisson processes when the temporal data appear in the form of panel counts. Panel count data frequently arise when experimental subjects are observed only at discrete time points and only the numbers of occurrences of the events between subsequent observation times are available. The exact occurrence timestamps of the events are unknown. The method of conducting the efficient variational inference is presented, based on the assumption of a Gaussian-process-modulated intensity function. We derive a tractable lower bound to alleviate the problems of the intractable evidence lower bound inherent in the variational inference framework. Our algorithm outperforms classical methods on both synthetic and three real panel count sets.

## 1 INTRODUCTION

**Background and issues.** Temporal data frequently arise as outcomes of an underlying *temporal point process* (Kingman, 1993) in continuous time. Temporal data can generally be classified into two types. One is from experiments that monitor subjects in a continuous fashion; and thereby the exact timestamps of all occurrences of the events are fully observable. These data are usually referred to as *recurrent event data* (Cook and Lawless, 2007). On the other hand, we have the so-called *panel count data* (Sun and Zhao, 2016), which is the focus of our paper. Under this framework, subjects are examined or observed only at discrete time-points and thus give only the numbers of occurrences of the events between subsequent observation times.

**Characteristics of panel count data.** A common characteristic of the panel count data is that we only have the numbers of occurrences between subsequent observation times. In particular, the exact occurrence times of the events are unknown. Hence, panel counts are non-negative integers and they represent the number of occurrences of events within a fixed period. Classical examples often arise in the clinical trials (Thall and Lachin, 1988) where patients are required to go back to the hospital after a certain treatment and only the numbers of symptoms between subsequent visits are recorded, such as the number of vomits or new tumors. Figure 1 gives an example of panel count data.

**Objective of this study.** The purpose of this paper is to present the variational Bayesian inference on Gaussian-process-modulated Poisson processes (GP3) that permits panel data observations.

There have been extensive studies on GP3 models and various inference algorithms are introduced for *recurrent event data* when timestamps of the events are fully observable, e.g., Monte Carlo sampling (Diggle et al., 2013; Adams et al., 2009), Laplace approximation (Flaxman et al., 2015) and variational inference (Lloyd et al., 2015). Among these approaches, the variational inference method (Lloyd et al., 2015) provides a computationally efficient estimate of the intensity function and does not require a careful discretization of the underlying space.

To the best of our knowledge, however, there has not been any study carried out on the variational inference of the GP3 model when the data come in the form of panel counts. Our ultimate goal is to infer the underlying intensity function in the panel count data.

**Related statistical works.** Based on the maximum likelihood criterion, several non-parametric estimators have been proposed to infer the underlying intensity function (Sun and Zhao, 2016), e.g., a

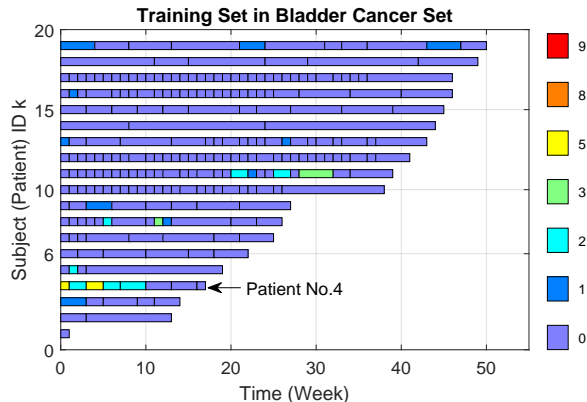


Figure 1: **Bladder Cancer Data Set.** This figure illustrates the panel count data from the patients. For the  $k$ th subject (or the  $k$ th patient), his/her observation window  $\mathcal{X}^{(k)}$  is divided into disjoint intervals. The  $i$ th interval is denoted as  $\mathcal{X}_i^{(k)}$ . For example, patient No. 4 ( $k = 4$ ) has an observation window which is divided into 8 disjoint intervals, i.e.,  $\bigcup_{i=1}^8 \mathcal{X}_i^{(4)} = \mathcal{X}^{(4)}$  and  $X_i \cap X_j = \emptyset$  for  $i \neq j$ . Patients may drop out from the study at any time and therefore their observation windows are different. An interval is shown by a rectangle. We use different colors to indicate the different numbers of new bladder tumors observed in this interval. Note that we only have access to *the number of events* in each interval.

non-parametric maximum pseudo-likelihood estimator (NPMPLE) (Wellner and Zhang, 2000), a non-parametric maximum pseudo-likelihood estimator with gamma frailty (NPMPPLGF) (Zhang and Jamshidian, 2003) and the local Expectation-Maximization (LocalEM) estimator (Fan et al., 2011). Unlike NPMPLE and NPMPPLGF, which only estimate the cumulative intensity function at a set of points, LocalEM provides a smooth estimate of the underlying intensity function due to the use of an exponential quadratic kernel (Fan et al., 2011).

Besides the computational cost in selecting the bandwidth of the exponential quadratic kernel, the estimators obtained by the LocalEM algorithm and other similar algorithms are point-estimates in the sense that the estimated intensity function is a point in the functional space. These point-estimates fail to capture the uncertainty in the data set. We show an example of the estimated intensity function by LocalEM in Figure 2. The uncertainty of the intensity function helps us understand the difficulty of the prediction at a given time.

**Contributions.** The contributions of our work are twofold. 1) In the first place it undertakes to construct a variational inference procedure for the **Gaussian-Process**

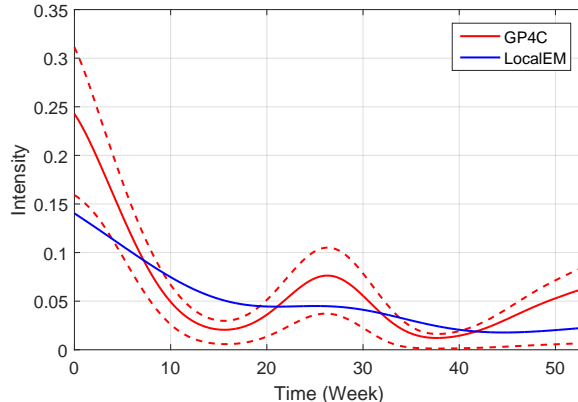


Figure 2: **Bladder Cancer Data Set.** Inferred intensity function by the LocalEM and GP4C methods. For GP4C, a 75% credible interval is given by dotted lines. Our estimator GP4C provides the additional uncertainty in the estimated intensity function compared with LocalEM. See Section 5 for details.

modulated **Poisson Process** model for **Panel Count** data (GP4C). 2) To carry out a variational inference in this setting, we derive a simple and tractable lower bound of the intractable evidence lower bound and demonstrate through empirical evidence that with this lower bound, GP4C outperforms a non-Bayesian method.

## 2 BACKGROUND

Throughout this paper, we denote the set of panel count data from  $K \in \mathbb{N}^+$  independent subjects as  $\mathcal{D}$ . Each subject will generate a sequence of events in the continuous space  $\mathcal{X}$ . We only consider the temporal point processes where the continuous space  $\mathcal{X}$  is a subset of  $\mathbb{R}$ . In the *recurrent event data*, the timestamps of the events are fully observable. We denote the timestamps from the  $k$ th subject as  $\{x_j^{(k)} \in \mathcal{X}\}$ .

In the *panel count data*, the  $k$ th subject is assessed in  $N_k$  disjoint intervals  $\{\mathcal{X}_i^{(k)}\}_{i=1}^{N_k}$ , where  $\bigcup_i \mathcal{X}_i^{(k)} = \mathcal{X}^{(k)} \subset \mathcal{X}$ . We have access to each interval  $\mathcal{X}_i^{(k)}$  and the number of events observed in this interval  $m_i^{(k)} = |\{x_j^{(k)} \in \mathcal{X}_i^{(k)}\}|$ . Let  $\mathbf{d}_k = \{(\mathcal{X}_i^{(k)}, m_i^{(k)})\}_{i=1}^{N_k}$  and  $\mathcal{D} = \{\mathbf{d}_k\}$ . Figure 1 illustrates an example of the panel count data.

### 2.1 LIKELIHOOD OF PANEL COUNT DATA

In the *recurrent event data*, one approach to modeling the events  $\{x_j^{(k)} \in \mathcal{X}\}$  from each subject is to use the inhomogeneous Poisson processes (IPP) (Kingman, 1993) and assume that there is a fixed underlying intensity function  $\lambda(x) : \mathcal{X} \rightarrow \mathbb{R}^+$ . Given the intensity function  $\lambda(x)$ ,

the likelihood for the observed events is

$$p(\{x_j^{(k)}\}|\lambda(x)) = \exp\left(-\int_{\mathcal{X}} \lambda(x)dx\right) \prod_j \lambda(x_j^{(k)}).$$

To derive the likelihood of the *panel count data*  $\mathcal{D}$ , we use two important features of an IPP (Kingman, 1993). The first is that given the intensity function  $\lambda(x)$ , the probability that we observe  $m_i^{(k)}$  events in the interval  $\mathcal{X}_i^{(k)}$  is given as follows:

$$p(m_i^{(k)}|\lambda(x); \mathcal{X}_i^{(k)}) = \frac{r_{ik}^{m_i^{(k)}}}{m_i^{(k)}!} \exp(-r_{ik}), \quad (1)$$

where  $r_{ik} \triangleq \int_{\mathcal{X}_i^{(k)}} \lambda(x)dx$  is the rate parameter of the Poisson distribution. Hereafter, we omit the dependency on  $\mathcal{X}_i^{(k)}$  for simplicity. However, the likelihood depends on the intervals and even for the same sequence, after censored with different intervals, the likelihood of the sequence will vary. See Appendix E.1 for a brief discussion.

The second feature is that on two disjoint intervals  $\mathcal{X}_i^{(k)}$  and  $\mathcal{X}_j^{(k)}$  ( $\mathcal{X}_i^{(k)} \cap \mathcal{X}_j^{(k)} = \emptyset$ ), the numbers of events on these intervals are independent random variables.

$$p(m_j^{(k)}, m_i^{(k)}|\lambda(x)) = p(m_j^{(k)}|\lambda(x))p(m_i^{(k)}|\lambda(x)). \quad (2)$$

Based on these two features, the likelihood of the panel count data  $\mathcal{D}$  can be derived. We assume that all subjects share the same intensity function  $\lambda(x)$ . Since  $K$  subjects are independent of each other and for the  $k$ th subject, the  $N_k$  intervals  $\{\mathcal{X}_i^{(k)}\}_{i=1}^{N_k}$  are disjoint, we obtain the following likelihood:

$$p(\mathcal{D}|\lambda(x)) = \prod_{k=1}^K p(\mathbf{d}_k|\lambda(x)) = \prod_{k=1}^K \prod_{i=1}^{N_k} p(m_i^{(k)}|\lambda(x)). \quad (3)$$

Several maximum likelihood estimators have been proposed on the basis of this likelihood or its variants, e.g., NPMPLE (Wellner and Zhang, 2000; Wellner et al., 2007), NPMPLEGF (Zhang and Jamshidian, 2003) and the LocalEM estimator (Fan et al., 2011). An estimate from LocalEM on the data set in Figure 1 is given in Figure 2. As we discussed, these estimators fail to model the uncertainty in the intensity function.

## 2.2 GP3 MODEL

In order to model the uncertainty of the intensity function  $\lambda(x)$  via a kernel, the traditional approach is to use the Cox process (Kingman, 1993). A Cox process is defined via a stochastic intensity function  $\lambda(x)$ . The stochastic

process to generate the intensity function is usually chosen to be a Gaussian process (GP) (Adams et al., 2009) and the model using a GP is called a GP3 model.

For the *recurrent event data*, GP3 models have been studied extensively (Adams et al., 2009; Gunter et al., 2014; Lloyd et al., 2015). The following model is an example of GP3 models (Lloyd et al., 2015),

$$\lambda(x) = f^2(x), \quad f \sim \mathcal{GP}(g(x), \kappa(x, x')), \quad (4)$$

where  $\mathcal{GP}(g(x), \kappa(x, x'))$  denotes the Gaussian process with mean function  $g(x)$  and covariance function  $\kappa(x, x')$ . The function  $f(x)$  drawn from a GP prior is squared to ensure the non-negativity of the intensity function. The GP3 model in Equation (4) admits a complete variational inference framework. Moreover, this intensity model can be enhanced with an independent variable for each subject or a mixture structure (Lloyd et al., 2016) to flexibly model the heterogeneity of the intensity functions across several subjects.

## 3 OUR MODEL GP4C : GP3 MODEL FOR PANEL COUNT DATA

In order to retain the scalability and efficiency of the variational inference approach (Lloyd et al., 2015) and add the uncertainty on the intensity function when we only observe the panel count data, we use the GP3 model defined in Equation (4) as the underlying intensity model.

The joint distribution  $p(\mathcal{D}, f)$  can be obtained by combining the likelihood model in Equation (3) and the intensity model in Equation (4).

$$p(\mathcal{D}, f) = \left[ \prod_{k=1}^K p(\mathbf{d}_k|\lambda(x)) \right] p(f; g, \kappa). \quad (5)$$

We call this model the **GP**-modulated **Poisson Process** model for **Panel Count** data (GP4C).

## 4 INFERENCE

In this section, we will discuss the problems when applying variational inference techniques on the GP4C model.

### 4.1 VARIATIONAL INFERENCE

We use sparse GPs to reduce the computational complexity with the set of pseudo inputs  $\{x_r\}_{r=1}^R$  on  $\mathcal{X}$  (Titsias, 2009). Let  $\mathbf{f}_R \triangleq [f(x_1), \dots, f(x_R)]^\top$ . The joint model with additional pseudo inputs is  $p(\mathcal{D}, f, \mathbf{f}_R) = p(\mathcal{D}|f)p(f|\mathbf{f}_R)p(\mathbf{f}_R)$  and the variational distribution is defined as follows:

$$q(f, \mathbf{f}_R) = p(f|\mathbf{f}_R)q(\mathbf{f}_R), \quad (6)$$



where  $q(\mathbf{f}_R) = \mathcal{N}(\boldsymbol{\mu}, \Sigma)$  and  $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$  denotes the normal distribution with mean  $\boldsymbol{\mu}$  and covariance matrix  $\Sigma$ . The evidence lower bound (ELBO)  $\mathcal{L}$  can be obtained by using Jensen's inequality.

$$\begin{aligned} \ln p(\mathcal{D}) &\geq \iint q(f, \mathbf{f}_R) \ln \frac{p(\mathcal{D}, f, \mathbf{f}_R)}{q(f, \mathbf{f}_R)} df d\mathbf{f}_R \\ &= \sum_{k=1}^K \sum_{i=1}^{N_k} \left( m_i^{(k)} \mathbb{E}_q \left[ \ln \int_{\mathcal{X}_i^{(k)}} f^2(x) dx \right] - \ln(m_i^{(k)}!) \right) \\ &\quad - \sum_{k=1}^K \mathbb{E}_q \left[ \int_{\mathcal{X}^{(k)}} f^2(x) dx \right] + \mathbb{E}_q \left[ \ln \frac{p(\mathbf{f}_R)}{q(\mathbf{f}_R)} \right] \triangleq \mathcal{L}. \end{aligned} \quad (7)$$

In ELBO, when assuming that the covariance function  $\kappa(x, x')$  is the automatic relevance determination (ARD) function  $\kappa(x, x') = \gamma \exp\left(-\frac{(x-x')^2}{2a^2}\right)$ ,  $x, x' \in \mathcal{X}$ , the second term in the ELBO can be analytically calculated (Lloyd et al., 2015) as follows:

$$\begin{aligned} \mathbb{E}_q \left[ \int_{\mathcal{X}^{(k)}} f^2(x) dx \right] &= \gamma |\mathcal{X}^{(k)}| - \text{tr}(K_{RR}^{-1} \Phi) \\ &\quad + \text{tr}(K_{RR}^{-1} \Phi K_{RR}^{-1} (\boldsymbol{\mu} \boldsymbol{\mu}^\top + \Sigma)), \end{aligned} \quad (8)$$

where  $\Phi$  is an  $R \times R$  matrix related to the pseudo inputs with its  $(i, j)$ -th entry equal to  $\int_{\mathcal{X}^{(k)}} \kappa(x_i, x) \kappa(x, x_j) dx$  and  $K_{RR}$  is the covariance matrix computed at the pseudo inputs. However, the ELBO  $\mathcal{L}$  is still intractable, since we can not analytically compute the expected integral  $\mathbb{E}_q \left[ \ln \int_{\mathcal{X}_i^{(k)}} f^2(x) dx \right]$  in the first term.

## 4.2 A TRACTABLE LOWER BOUND

We tackle the intractable expectation by deriving a tractable lower bound. First we introduce a relevant lemma on the expectation of the logarithm of the square of a normal-distributed random variable.

**Lemma 1.** *Let  $y \sim \mathcal{N}(\mu, \sigma^2)$  and  $\varphi = (\mu/\sigma)^2$ . Then*

$$\mathbb{E}_y[\ln y^2] = \ln(2\sigma^2) + \sum_{j=0}^{\infty} \frac{(\varphi/2)^j \exp(-\varphi/2)}{j!} \psi(j+1/2), \quad (9)$$

where  $\psi(\cdot)$  is the digamma function.

The proof of Lemma 1 can be found in Appendix A. Let

$$g_m(y) = \sum_{j=0}^{\infty} \frac{y^j \exp(-y)}{j!} \psi(j+m). \quad (10)$$

Then  $\mathbb{E}_y[\ln y^2] = \ln(2\sigma^2) + g_{0.5}(\varphi/2)$ . The function  $g_m(y)$ , where  $y$  is a positive real number and  $m$  is a positive integer, has been studied in the analysis of mobile

and wireless communication systems (Moser, 2007). For  $m = 1/2$ ,  $g_{0.5}(\varphi/2)$  can be computed using a confluent hyper-geometric function  $G(\cdot)$  (Lloyd et al., 2015), which is stored in a pre-computed look-up table.

$$g_{0.5}(\varphi/2) = -G(-\varphi/2) - 2 \ln 2 - C, \quad (11)$$

where  $C$  is Euler's constant and  $C \approx 0.5772$ . However, to the best of our knowledge, it is still not clear how to calculate the integral of the function  $G(-\varphi/2)$  when using a GP. To derive a tractable lower bound of the intractable expectation, we introduce the following lemma to give a lower bound of the function  $g_m(y)$  and the proof can be found in Appendix B.

**Lemma 2.** *Let  $y \sim \mathcal{N}(\mu, \sigma^2)$  and  $C$  be Euler's constant.*

$$\mathbb{E}_y[\ln y^2] \geq \ln(\mu^2 + b\sigma^2) - C - \ln 2, \quad \forall b \in [0, 1]. \quad (12)$$

Based on Lemma 2, we propose the following lower bound for the intractable expectation in the ELBO.

**Theorem 1.** *Let  $f$  be a GP as defined in Equation (4). For  $b \in [0, 1]$ , the following bound holds:*

$$\begin{aligned} \mathbb{E}_q \left[ \ln \int_{\mathcal{X}_i^{(k)}} f^2(x) dx \right] &\geq -C - \ln 2 \\ &\quad + \ln \left( \int_{\mathcal{X}_i^{(k)}} \left( \mathbb{E}_q^2 f(x) + b \text{Var}_q f(x) \right) dx \right), \end{aligned} \quad (13)$$

where the distribution  $q$  is given in Equation (6).

*Proof.* We first use Jensen's inequality on the logarithm function and then interchange the order of integration and expectation.

$$\begin{aligned} \mathbb{E}_q \left[ \ln \int_{\mathcal{X}_i^{(k)}} f^2(x) dx \right] &= \mathbb{E}_q \left[ \ln \int_{\mathcal{X}_i^{(k)}} \tilde{p}(x) \frac{f^2(x)}{\tilde{p}(x)} dx \right] \\ &\geq \int_{\mathcal{X}_i^{(k)}} \tilde{p}(x) \mathbb{E}_q \left[ \ln \frac{f^2(x)}{\tilde{p}(x)} \right] dx, \end{aligned} \quad (14)$$

where  $\tilde{p}(x)$  is a probability distribution on  $\mathcal{X}_i^{(k)}$ . Furthermore, maximizing this lower bound with respect to  $\tilde{p}(x)$  yields the optimal distribution:

$$\tilde{p}_{\text{opt}}(x) \propto \exp \left( \mathbb{E}_q \ln f^2(x) \right). \quad (15)$$

We remark that this result is analogous to that of the discrete version presented in Paisley (2010). Substituting Equation (15) into the right-hand side of Equation (14)

yields

$$\begin{aligned}
\mathbb{E}_q \left[ \ln \int_{\mathcal{X}_i^{(k)}} f^2(x) dx \right] &\geq \ln \left( \int_{\mathcal{X}_i^{(k)}} e^{\mathbb{E}_q \ln f^2(x)} dx \right) \\
&\stackrel{(13)}{\geq} \ln \left( \int_{\mathcal{X}_i^{(k)}} e^{\ln(\mathbb{E}_q^2 f(x) + b \text{Var}_q f(x)) - C - \ln 2} dx \right) \\
&= \ln \left( \int_{\mathcal{X}_i^{(k)}} \left( \mathbb{E}_q^2 f(x) + b \text{Var}_q f(x) \right) dx \right) - C - \ln 2,
\end{aligned}$$

where we have invoked Lemma 2 in the penultimate line whilst defining  $y := f(x)$ .  $\square$

It should be emphasized that we are making no further assumptions on the dimensionality of  $x$  in the proof of Theorem 1. Hence we may augment the dimensionality of  $x$  in Theorem 1 such that it can also be applied to problems in spatial point processes. In summary, the ELBO in Equation (7) inherits an analytical bound. We present the following:

**Theorem 2.** *A tractable lower bound of the ELBO  $\mathcal{L}$  in the GP4C model is given as follows:*

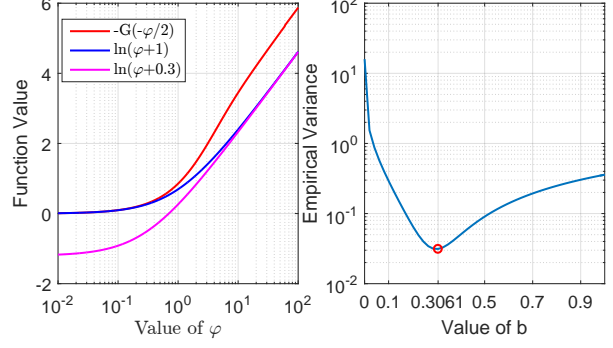
$$\begin{aligned}
\mathcal{L} &\geq \tilde{\mathcal{L}} \triangleq - \sum_{k=1}^K \mathbb{E}_q \left[ \int_{\mathcal{X}^{(k)}} f^2(x) dx \right] + \mathbb{E}_q \left[ \ln \frac{p(\mathbf{f}_R)}{q(\mathbf{f}_R)} \right] \\
&+ \sum_{k=1}^K \sum_{i=1}^{N_k} m_i^{(k)} \ln \left( \int_{\mathcal{X}_i^{(k)}} \left( \mathbb{E}_q^2 f(x) + b \text{Var}_q f(x) \right) dx \right) \\
&- \sum_{k=1}^K \sum_{i=1}^{N_k} \left( m_i^{(k)} (C + \ln 2) + \ln(m_i^{(k)}!) \right). \quad (16)
\end{aligned}$$

The details of the proof are deferred to Appendix C. The derivations of  $\mathbb{E}_q^2 f(x)$  and  $\text{Var}_q f(x)$  follow similar lines to the derivation of Equation (8). The third part of  $\tilde{\mathcal{L}}$  is a constant and thus can be omitted when maximizing the lower bound. Let  $\Psi = \{\boldsymbol{\mu}, \Sigma\}$  and  $\Phi = \{\gamma, a\}$  be the variational parameters and hyper-parameters in the covariance function of a GP, respectively. We use the variational Expectation-Maximization (vEM) algorithm (Dempster et al., 1977) to update the parameters  $\Psi$  and  $\Phi$  iteratively on the modified ELBO  $\tilde{\mathcal{L}}$ .

### 4.3 THE VALUE OF PARAMETER $b$

A natural question is, how do we select the parameter  $b$  in Theorem 1? Recall that two inequalities were used in the proof. It is cumbersome to evaluate Inequality (14) since it is an integral over  $\mathcal{X}_i^{(k)}$ . We first examine different choices of  $b$  in Lemma 2.

In Paisley et al. (2012), a more correlated lower bound of the ELBO serves as a better control variate in reducing the variance of a stochastic gradient. Inspired by this



**Figure 3: Influences of  $b$  in Lemma 2.** (Left) The true value of  $-G(-\varphi/2)$  by a look-up table and two simple lower bounds. The bound  $\ln(\phi + b)$  with  $b = 0.3$  correlates with the curve of the true value better. (Right). The variance  $\text{Var}[h(\varphi; b)]$  when varying the choices of  $b$  and the best  $b$  is shown with a red circle.

study, we introduce a heuristic method and conduct the following experiment to evaluate the correlation for different choices of  $b$ . In Lemma 2, the difference between the lower bound and the true value is

$$\begin{aligned}
&\ln(\mu^2 + b\sigma^2) - C - \ln 2 - \mathbb{E}_y[\ln y^2] \\
&= \ln(\varphi + b) + G(-\varphi/2) \triangleq h(\varphi; b). \quad (17)
\end{aligned}$$

For each choice of  $b$ , we vary  $\varphi = (\mu/\sigma)^2$  on a vector of 5000 logarithmically spaced points between  $10^{-6}$  and  $10^6$  and evaluate the correlation between the lower bound and the true value by the variance  $\text{Var}[h(\varphi; b)]$ . We calculate  $\text{Var}[h(\varphi; b)]$  on a vector of 50 evenly spaced choices of  $b$  between 0 and 1 and the result is shown in Figure 3. We see that the optimal choice of  $b$  is 0.3061 if  $\varphi$  ranges from  $10^{-6}$  to  $10^6$ . In the actual situation, this optimal value of  $b$  depends on the range of  $\varphi$  in the data and the influence of Inequality (14), we evaluate several choices of  $b$  on synthetic data sets in Section 5.

### 4.4 COMPUTATIONAL COMPLEXITY

Let each interval in temporal point processes be  $\mathcal{X}_i^{(k)} = [x_{ai}^{(k)}, x_{bi}^{(k)}]$  with two end points  $x_{ai}^{(k)}$  and  $x_{bi}^{(k)}$ . Two intervals are different if at least one end point is different. We denote the number of different intervals in the data set as  $N$  and the number of pseudo inputs as  $M$ . For each interval, the computation complexity of GP4C is  $\mathcal{O}(M^3)$  which is determined by the matrix-matrix calculation when evaluating  $\text{Var}_q f(x)$  in Equation (16). The computational complexity during one iteration of the vEM algorithm is  $\mathcal{O}(NM^3)$  since in our implementation, we calculate the integral of all  $N$  different intervals.

We analyze the computational complexity of the Lo-

calEM (Fan et al., 2011) algorithm for comparison. In LocalEM,  $\{x_{ai}^{(k)}\}$  and  $\{x_{bi}^{(k)}\}$  are first merged into a single ordered set  $X$  where duplicated values are removed. We denote the size of the merged set  $X$  as  $\bar{N}$  and generally  $\bar{N} \leq N$ . Then the Gaussian quadratic rule with  $\bar{M}$  points is used to calculate the integral of the intensity function between subsequent values in the set  $X$  and the computational complexity during one iteration is  $\mathcal{O}(\bar{N}^2 \bar{M}^2)$ . If the size of the merged set  $\bar{N}$  is significantly smaller than  $N$ , LocalEM may be computationally more efficient than GP4C. However, if  $\bar{N} \approx N$ , LocalEM may suffer from the term  $\bar{N}^2$  in the computational complexity. We provide additional experiments on the influence of the number  $\bar{N}$  in Appendix E.3.

## 5 EXPERIMENTS

We evaluate our proposed GP4C model and compare it with the benchmark methods on both synthetic and real-world data sets. The algorithms are programmed in Matlab R2015b and run on an Intel Xeon E5-2667 CPU with a memory of 64GB. Our code is available at [github.com/Dinghy/GP4C](https://github.com/Dinghy/GP4C).

### 5.1 EXPERIMENT SETTINGS

For each data set  $\mathcal{D}$ , we randomly partition the subjects into training and testing sets, which we denote as  $\mathcal{D}_{\text{train}}$  and  $\mathcal{D}_{\text{test}}$ , respectively. We repeat each setting for  $S = 40$  times. In the  $s$ th trial, the training and testing sets are denoted as  $\mathcal{D}_{\text{train}}^{(s)}$  and  $\mathcal{D}_{\text{test}}^{(s)}$ .

**Benchmark.** Two benchmark algorithms are used.

- a) **GP3** (Lloyd et al., 2015). This benchmark reflects the best performance that can be obtained if we obtain the recurrent event data set where we have the exact timestamps.
- b) **LocalEM** (Fan et al., 2011). Both LocalEM and GP4C are nonparametric estimators based on the maximum likelihood criterion. To fairly compare the computation time, we implemented the LocalEM algorithm in MATLAB based on the R code provided in Fan et al. (2011). This method produces a smooth estimate of the intensity function due to the use of an exponential quadratic kernel. We use a 5-fold cross-validation on the training set to select the bandwidth of the exponential quadratic kernel.

**Evaluation Metric.** We evaluate the performance of the algorithms in terms of three metrics.

- a) Mean of the integrated squared error (MISE). In synthetic data sets, we have the ground truth of the

intensity function  $\lambda_{\text{true}}$  and the integrated squared error can be calculated using our estimated intensity function  $\lambda_{\text{est}}^{(s)}$  during the  $s$ th trial. To measure the bias of each estimator, we calculate the mean of the integrated squared error as follows:

$$\text{MISE}(s) \triangleq \int_{\mathcal{X}} (\lambda_{\text{est}}^{(s)}(x) - \lambda_{\text{true}}(x))^2 dx. \quad (18)$$

For GP4C, to measure its bias, we omit the variance of the estimator and use the expectation of the intensity function  $\mathbb{E}_{q^{(s)}}[f^2(x)]$  as  $\lambda_{\text{est}}^{(s)}(x)$ .

- b) Test log likelihood  $\mathcal{L}_{\text{test}}$ . During the  $s$ th trial, the logarithm of the test likelihood can be written as follows:

$$\mathcal{L}_{\text{test}}(s) \triangleq \ln \int p(\mathcal{D}_{\text{test}}^{(s)} | f) p(f | \mathcal{D}_{\text{train}}^{(s)}) df. \quad (19)$$

For LocalEM, since this estimator provides a point-estimate and we directly use the estimated function  $f^{(s)}$  to calculate  $\mathcal{L}_{\text{test}}(s)$ . For GP4C and GP3, we need to sample the function  $f^{(s)}$  from the variational distribution and the detailed calculation can be found in Appendix D.

- c) Computation time  $T$ . We record the training time measured in seconds for each setting. For GP3 and GP4C, we record the computation time of the training process. For LocalEM, it includes the time of 5-fold cross-validation on the training set to select the bandwidth of the exponential quadratic kernel and the time of a training process over the whole training set.

**Optimization Settings.** For GP3 and GP4C, following Lian et al. (2015), we use the re-parametrization trick  $\Sigma = LL^T$  by Cholesky decomposition and add positivity constraints to the diagonal elements in  $L$ . Due to this constraint on  $L$ , we use the limited-memory projected quasi-Newton algorithm (Schmidt et al., 2009) to optimize the variational parameters  $\Psi = \{\mu, \Sigma\}$ . We add a jitter term  $\epsilon I$  where  $\epsilon = 10^{-6}$  to the covariance matrix  $K_{RR}$  to avoid numerical instability (Titsias, 2009).

### 5.2 SYNTHETIC DATA SETS

We test three synthetic data sets which we denote as the Synthetic A, B and C data sets, respectively.

On the Synthetic A data set, the intensity function is a square wave function  $h_1(x)$  as follows. See Figure 4 for an illustration of  $h_1(x)$ .

$$h_1(x) = \begin{cases} 7 & \text{if } \text{mod}\left(\left\lfloor \frac{x}{10} \right\rfloor, 2\right) = 0, \\ 2 & \text{otherwise.} \end{cases}$$

Table 1: **Synthetic data sets.** Mean and standard deviation of statistics about different choices of  $b$  over 40 runs. GP3 uses the *recurrent event data* while LocalEM and GP4C use the *panel count data*. For GP4C,  $b = 0.3$  and  $b = 0$  perform better than  $b = 1$  in terms of MISE and  $\mathcal{L}_{\text{test}}$ .

Method	MISE	$\mathcal{L}_{\text{test}}$	$T$ [s]
(Synthetic A)			
GP3	29.5±1.0	-1366.5±17.4	16±4
GP4C(1)	41.8±6.2	-3236.9±542.3	25±5
GP4C(0)	40.8±3.3	-1378.1±16.9	19±4
GP4C(0.3)	40.2±3.2	-1377.8±17.5	20±3
LocalEM	44.6±3.1	-1383.5±17.0	33±2
(Synthetic B)			
GP3	0.5±0.2	-783.1±20.7	8±1
GP4C(1)	1.9±2.1	-1005.8±81.5	55±44
GP4C(0)	2.7±0.8	-794.5±20.1	17±3
GP4C(0.3)	2.4±0.7	-794.2±20.2	17±4
LocalEM	3.5±0.7	-800.3±19.6	33±2
(Synthetic C)			
GP3	1.2±0.4	-864.1±14.9	8±3
GP4C(1)	2.3±1.5	-1194.6±100.5	52±53
GP4C(0)	2.1±0.6	-871.2±15.9	17±2
GP4C(0.3)	2.0±0.7	-872.0±15.7	18±3
LocalEM	5.2±1.1	-882.7±16.5	34±2

On the Synthetic B and C data set, the underlying intensity functions are drawn according to Equation (4). We first draw a function from a GP on a vector of 3001 evenly-spaced points in  $\mathcal{X} = [0, T]$ , where  $T = 60$ . We approximate the value of the function at an arbitrary position with linear interpolation. The function is then squared to guarantee the positiveness of the intensity function. See Figure 5 for an illustration.

During the  $s$ th trial, we first generate a *recurrent event data set* with 100 subjects on the same observation window  $\mathcal{X}^{(k)} = \mathcal{X}$ . Then we generate the corresponding *panel count data set*  $\mathcal{D}^{(s)}$  by censoring each subject with 10 intervals. We generate the censored intervals by a draw from a Dirichlet distribution  $\mathbf{w}^{(k)} \sim \text{Dir}(\boldsymbol{\theta})$  and  $\boldsymbol{\theta}$  is a 10-dimensional vector with all elements equal to 1. The  $i$ th interval of the  $k$ th subject can be computed as  $\mathcal{X}_i^{(k)} = [\sum_{j=1}^{i-1} w_j^{(k)} T, \sum_{j=1}^i w_j^{(k)} T]$ . We randomly partition  $\mathcal{D}^{(s)}$  into two parts, where 50 subjects are used for training and 50 for testing.

**Different choices of the hyper-parameter  $b$ .** On all three synthetic data sets, we test three different choices of  $b$  in  $\{0, 0.3, 1\}$ . We choose the number of pseudo inputs to be 30. We calculate the MISE and  $\mathcal{L}_{\text{test}}$  and the results

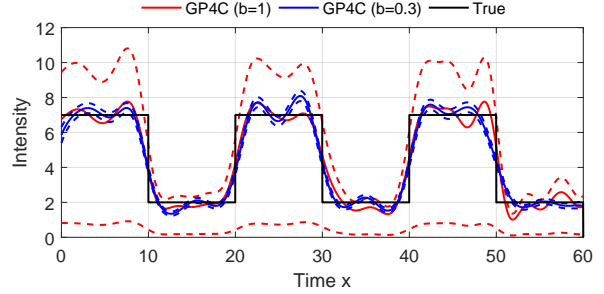


Figure 4: **Synthetic A Data Set.** The estimated intensity functions from GP4C ( $b = 1$ ) and GP4C ( $b = 0.3$ ) are shown with 75% credible intervals. True intensity function  $h_1(x)$  is given for comparison. We see that GP4C ( $b = 1$ ) over-estimates the variance of the intensity function.

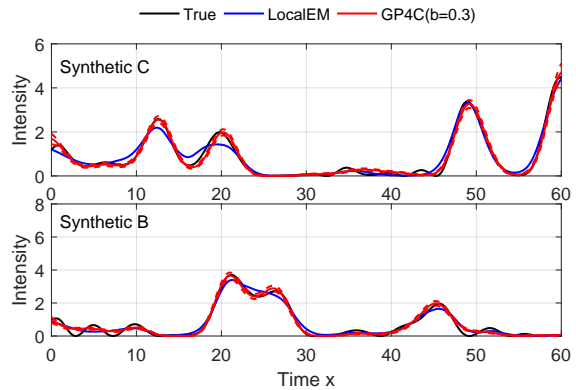


Figure 5: **Synthetic B & C Data Sets.** An illustration of the underlying intensity functions and inferred intensity functions by the LocalEM and GP4C methods. The underlying intensity function is drawn from a Gaussian process. For GP4C, a 75% credible interval is given by dotted lines.

are provided in Table 1. We see that  $b = 0, 0.3$  generally outperform  $b = 1$  on these simple synthetic data sets. However, the difference between  $b = 0$  and  $b = 0.3$  is not significant. The reason is that Inequality (14) and the range of  $\varphi$  on  $\mathcal{X}$  are also relevant to the actual performance of different  $b$ , as we discussed in Section 4.3.

To investigate the reason behind the bad performance of  $\mathcal{L}_{\text{test}}$  when  $b = 1$ , we plot the best result in terms of MISE during 40 trials in Figure 4. We see that GP4C ( $b = 1$ ) over-estimates the variance of the intensity function and the over-estimated variance leads to the poor performance in  $\mathcal{L}_{\text{test}}$ . We fix  $b = 0.3$  during the remaining experiments for simplicity.

**Number of the pseudo inputs.** We vary the number of pseudo inputs in GP3 and GP4C since this number de-

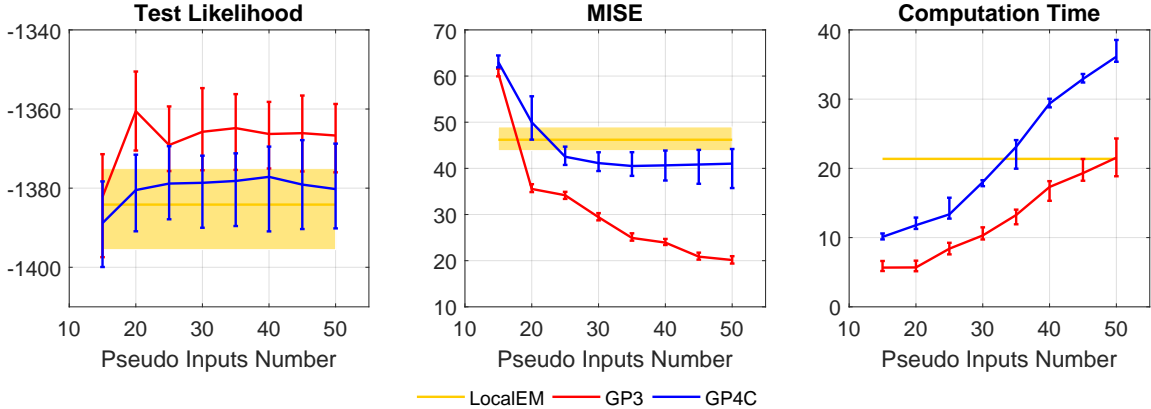


Figure 6: **Synthetic Data Set.** Comparison of performance of GP3, GP4C and LocalEM in terms of  $\mathcal{L}_{\text{test}}$ , MISE and  $T$  when varying the number of pseudo inputs for sparse GPs. For the test likelihood, MISE and the computation time, the median, the 0.25 and 0.75 quantiles of the statistics in 40 experiments are shown with error bars or shaded area. For GP3 and GP4C, MISE and  $\mathcal{L}_{\text{test}}$  stay relatively stable with the increase of the number of pseudo inputs.

terminates the accuracy of approximation in a sparse GP. We expect that for GP-based methods the test likelihood will be relatively stable when increasing the number of pseudo inputs according to previous studies on sparse GPs (Titsias, 2009).

The result for the Synthetic A data set is given in Figures 6. In Figure 6, we see that for GP3 and GP4C, MISE and  $\mathcal{L}_{\text{test}}$  stay relatively stable with the increase of the number of pseudo inputs. The computation time of GP3 and GP4C will grow with the increase of the number of pseudo inputs.

In both Table 1 and Figure 6, we see that GP4C outperforms LocalEM on these three datasets. However, we also notice that there is still a gap between GP3 and GP4C in terms of  $\mathcal{L}_{\text{test}}$  and MISE in Table 1. Two reasons may account for this fact. The first one is that the data are provided in the form of panel counts rather than exact timestamps. The second reason is that we use a lower bound of the true ELBO to perform the variational inference, which may lead to a bias. This bias can be alleviated with the stochastic variational inference (Paisley et al., 2012), where our lower bound can serve as a control variate. We leave this as a future study.

An additional experiment in which we increase the number of training subjects to evaluate the gain in performance on the Synthetic A data set is given in Appendix E.2.

### 5.3 REAL WORLD DATA SETS

Sun and Zhao (2016) provided three panel count data sets. Some statistics can be found in Table 2. A brief description about the these data sets can be found in Ap-

Table 2: Statistics about the three data sets, where  $K$ ,  $\mathcal{X}$ ,  $\bar{N}$  and  $N$  denote the number of subjects in each data set, the underlying continuous space, the number of different end points and the number of different intervals  $\mathcal{X}_i^{(k)}$ , respectively.

Data Set	$\mathcal{X}$	$K$	$\bar{N}$	$N$
Na-A	[0, 55]	65	45	109
Na-B	[0, 55]	48	38	84
Bl-A	[0, 53]	38	52	176
Bl-B	[0, 53]	47	52	201
Sk-A & Sk-B	[0, 61.57]	143	751	816
Sk-C & Sk-D	[0, 62.63]	147	808	887

pendix F.

We use 18 pseudo inputs for all real world experiments. In each trial, we randomly split each data set into two parts, which are  $\mathcal{D}_{\text{train}}^{(s)}$  (50%) and  $\mathcal{D}_{\text{test}}^{(s)}$  (50%). On these three data sets, since the original data are in the form of panel counts, GP3 is not tested. We compare GP4C with LocalEM in terms of  $\mathcal{L}_{\text{test}}$  and the computation time  $T$ .

The results are given in Table 3. The standard deviation of the likelihood is large since the likelihood depends on the censored intervals of the subjects, which vary greatly in different train/test split. We conduct an experiment to reduce the standard deviation in Appendix H. In Table 3, LocalEM performs better on the Nausea and Bladder data sets in terms of the computation time  $T$ . GP4C outperforms LocalEM in terms of test likelihood  $\mathcal{L}_{\text{test}}$  in all data sets.

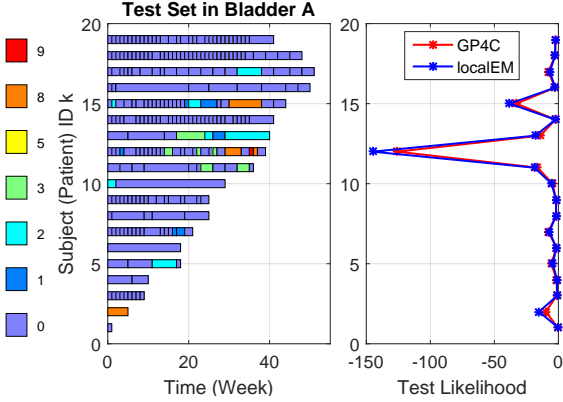


Figure 7: **Bladder A Data Set.** An illustration of the panel count data in the test set (Left) and the test likelihood from GP4C and LocalEM of each subject (Right). GP4C mainly outperforms LocalEM on two subjects whose numbers of newly-occurred cancers are large (No. 12 and 15).

To see the difference between GP4C and LocalEM, we show the result of inferred intensities by two algorithms during one trial on the Bladder A data set in Figure 2. We see that GP4C provides the additional uncertainty which helps improve  $\mathcal{L}_{\text{test}}$  compared with LocalEM. Since the Bladder A set is small, we plot the panel count data in the training set in Figure 1. The test set and the test likelihood of all its subjects are given in Figure 7. From the test likelihood of each subject, we see that GP4C outperforms LocalEM on two subjects whose counts of newly-occurred tumors are large (No. 12 and No. 15). The count 8 never occurs in the training set and a point-estimate will fail to model this uncertainty while a GP-modulated method will take the uncertainty into consideration.

Another observation about this data set is that there is a heterogeneity across all subjects. The traditional approach to modeling heterogeneity is to add an additional variable on the intensity function for each subject (Cook and Lawless, 2007). We briefly discuss how to add this change to GP4C in Appendix G.

## 6 CONCLUSION

We presented the first framework for GP-modulated Poisson processes when data appear in the form of panel counts. We derived a tractable lower bound for the intractable evidence lower bound when modeling the panel count data using the GP-modulated intensity function. Our model, GP4C, outperforms a non-Bayesian method using the maximum likelihood criterion in terms of test likelihood and achieves comparable results in terms of

Table 3: Mean and standard deviation of the test likelihood ( $\mathcal{L}_{\text{test}}$ ) and the computation time  $T$  measured in seconds on the three panel count data sets over 40 runs. LocalEM performs better on the Nausea and Bladder data sets in terms of computation time. In all data sets, GP4C performs better on the test likelihood and outperforms LocalEM on computation time in the Skin data sets.

Data Set	METHOD	$\mathcal{L}_{\text{test}}$	$T[s]$
Na-A	LocalEM	$-492.1 \pm 306.1$	$1 \pm 0$
	GP4C	$-484.9 \pm 201.8$	$10 \pm 10$
Na-B	LocalEM	$-473.2 \pm 212.2$	$1 \pm 0$
	GP4C	$-411.0 \pm 184.3$	$10 \pm 7$
Bl-A	LocalEM	$-201.8 \pm 46.9$	$1 \pm 0$
	GP4C	$-182.2 \pm 47.3$	$25 \pm 9$
Bl-B	LocalEM	$-313.1 \pm 54.2$	$1 \pm 0$
	GP4C	$-310.4 \pm 54.9$	$26 \pm 21$
Sk-A	LocalEM	$-259.1 \pm 27.3$	$39 \pm 3$
	GP4C	$-258.7 \pm 26.7$	$33 \pm 6$
Sk-B	LocalEM	$-198.1 \pm 47.1$	$39 \pm 3$
	GP4C	$-191.2 \pm 42.5$	$24 \pm 4$
Sk-C	LocalEM	$-358.0 \pm 35.8$	$47 \pm 4$
	GP4C	$-355.7 \pm 36.0$	$21 \pm 12$
Sk-D	LocalEM	$-200.9 \pm 31.9$	$46 \pm 3$
	GP4C	$-198.9 \pm 30.6$	$27 \pm 4$

computational time.

In the future, we plan to implement the stochastic variational inference algorithm to evaluate the bias in the tractable lower bound. We are also considering to find an applicable two-dimensional data set where we can extend our algorithm to spatial point processes.

## Acknowledgements

We thank the anonymous reviewers for their helpful suggestions. MS was supported by KAKENHI 17H00757.

## References

- Adams, R. P., Murray, I., and MacKay, D. J. (2009). Tractable nonparametric Bayesian inference in Poisson processes with Gaussian process intensities. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 9–16. ACM.
- Cook, R. J. and Lawless, J. (2007). *The Statistical Analysis of Recurrent Events*. Springer Science & Business Media.
- Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM

- algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38.
- Diggle, P. J., Moraga, P., Rowlingson, B., and Taylor, B. M. (2013). Spatial and spatio-temporal log-Gaussian Cox processes: extending the geostatistical paradigm. *Statistical Science*, pages 542–563.
- Fan, C.-P. S., Stafford, J., and Brown, P. E. (2011). Local-EM and the EMS algorithm. *Journal of Computational and Graphical Statistics*, 20(3):750–766.
- Flaxman, S., Wilson, A., Neill, D., Nickisch, H., and Smola, A. (2015). Fast Kronecker inference in Gaussian processes with non-Gaussian likelihoods. In *International Conference on Machine Learning*, pages 607–616.
- Gunter, T., Lloyd, C., Osborne, M. A., and Roberts, S. J. (2014). Efficient Bayesian nonparametric modeling of structured point processes. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pages 310–319. AUAI Press.
- Kingman, J. F. C. (1993). *Poisson Processes*. Wiley Online Library.
- Lian, W., Henaio, R., Rao, V., Lucas, J., and Carin, L. (2015). A multitask point process predictive model. In *International Conference on Machine Learning*, pages 2030–2038.
- Lloyd, C., Gunter, T., Osborne, M., and Roberts, S. (2015). Variational inference for Gaussian process modulated Poisson processes. In *International Conference on Machine Learning*, pages 1814–1822.
- Lloyd, C., Gunter, T., Osborne, M., Roberts, S., and Nickson, T. (2016). Latent point process allocation. In *Artificial Intelligence and Statistics*, pages 389–397.
- Moser, S. M. (2007). Some expectations of a non-central chi-square distribution with an even number of degrees of freedom. In *TENCON 2007-2007 IEEE Region 10 Conference*, pages 1–4. IEEE.
- Paisley, J. (2010). Two useful bounds for variational inference. Technical report, Technical report, Department of Computer Science, Princeton University, Princeton, NJ.
- Paisley, J., Blei, D. M., and Jordan, M. I. (2012). Variational Bayesian inference with stochastic search. In *Proceedings of the 29th International Conference on Machine Learning*, pages 1363–1370. Omnipress.
- Schmidt, M., Berg, E., Friedlander, M., and Murphy, K. (2009). Optimizing costly functions with simple constraints: A limited-memory projected quasi-Newton algorithm. In *Artificial Intelligence and Statistics*, pages 456–463.
- Sun, J. and Zhao, X. (2016). *Statistical Analysis of Panel Count Data*. Springer.
- Thall, P. F. and Lachin, J. M. (1988). Analysis of recurrent events: Nonparametric methods for random-interval count data. *Journal of the American Statistical Association*, 83(402):339–347.
- Titsias, M. K. (2009). Variational model selection for sparse Gaussian process regression. *Report, University of Manchester, UK*.
- Wellner, J. A. and Zhang, Y. (2000). Two estimators of the mean of a counting process with panel count data. *Annals of Statistics*, pages 779–814.
- Wellner, J. A., Zhang, Y., et al. (2007). Two likelihood-based semiparametric estimation methods for panel count data with covariates. *The Annals of Statistics*, 35(5):2106–2142.
- Zhang, Y. and Jamshidian, M. (2003). The gamma-frailty Poisson model for the nonparametric estimation of panel count data. *Biometrics*, 59(4):1099–1106.

---

# A Unified Probabilistic Model for Learning Latent Factors and Their Connectivities from High-Dimensional Data

---

Ricardo Pio Monti<sup>1</sup> and Aapo Hyvärinen<sup>1,2</sup>

<sup>1</sup>Gatsby Computational Neuroscience Unit, University College London, UK

<sup>2</sup>Department of Computer Science and HIIT, University of Helsinki, Finland

## Abstract

Connectivity estimation is challenging in the context of high-dimensional data. A useful preprocessing step is to group variables into clusters, however, it is not always clear how to do so from the perspective of connectivity estimation. Another practical challenge is that we may have data from multiple related classes (e.g., multiple subjects or conditions) and wish to incorporate constraints on the similarities across classes. We propose a probabilistic model which simultaneously performs both a grouping of variables (i.e., detecting community structure) and estimation of connectivities between the groups which correspond to latent variables. The model is essentially a factor analysis model where the factors are allowed to have arbitrary correlations, while the factor loading matrix is constrained to express a community structure. The model can be applied on multiple classes so that the connectivities can be different between the classes, while the community structure is the same for all classes. We propose an efficient estimation algorithm based on score matching, and prove the identifiability of the model. Finally, we present an extension to directed (causal) connectivities over latent variables. Simulations and experiments on fMRI data validate the practical utility of the method.

## 1 INTRODUCTION

Estimating the connectivity structure between observed variables is a fundamental problem in statistics and machine learning. Probabilistic methods are often based on estimation of the covariance matrix or its inverse. A number of estimators have been proposed for both (Dempster, 1972; Ledoit & Wolf, 2003). On a more general

level, such undirected connectivity estimation is a special case of modelling the covariance matrix, which is one of the goals of classical dimensionality reduction methods such as factor analysis and principal components analysis (PCA). In contrast, directed connectivity estimation studies the causal dependence structure across variables (Pearl, 2009). In this work we present methods to perform both directed and undirected connectivity estimation of latent variables in the context of high-dimensional data.

An important problem in practice is that many connectivity estimation methods assume we observe, and know how to choose, the variables between which the connectivity is to be estimated. However, in practice we often have very high-dimensional data, and it may not be useful or feasible to estimate the connectivities between all of them. It is important to somehow reduce the number of variables so that the connectivity estimation is feasible, and furthermore, such reduction can greatly facilitate interpretation of the results. It is often useful to perform the dimension reduction so that it can be interpreted as clustering, as in non-negative PCA (Sigg & Buhmann, 2008). A relevant challenge is how such reduction in the number of variables should be combined with connectivity estimation. In the past, approaches based on stochastic block models (Airoldi et al., 2008; Marlin & Murphy, 2009) or clustered factor analysis (Buesing et al., 2014) have been employed. However, such methods do not explicitly model the connectivity over latent variables and cannot easily be extended to accommodate multiple classes of related datasets, both of which are of interest in this work. Alternative methods recover correlation structure over latent variables but do not focus on dimensionality reduction (Sasaki et al., 2017). Conversely, in the context of directed connectivity, the causal clustering of observed variables has been studied by Silva et al. (2006), Shimizu et al. (2009) and Kummerfeld & Ramsey (2016).

A further challenge is estimating multiple related connectivity matrices, assuming the data is divided into a number of classes, such as subjects in a biomedical setting.



While the estimation of multiple related Gaussian graphical models, parameterized by the inverse covariance, has been extensively studied (Varoquaux et al., 2010; Danaher et al., 2014; Monti et al., 2017), we want to combine such multiple connectivity estimation with the variable reduction scheme described above as well as extend such methods to the domain of directed connectivities.

A practical application that motivates our theoretical developments is functional MRI (fMRI) data, where estimation of “functional connectivity” is widely practiced. Such analysis is a cornerstone of modern neuroscientific research, having provided fundamental insights into the structure and architecture of the human connectome. However, the existing methods are often not very rigorous and would benefit from a proper probabilistic formulation. Traditionally, functional connectivity networks have been modeled as covariance graphs, where the nodes in the network correspond to spatially remote brain regions and edges encode the marginal dependence structure (often simply the covariance). In fMRI, we very clearly see the importance of the theoretical points raised above, in the form of the following challenges:

- **Inter-subject consistency:** Data is often collected across a cohort of subjects. A hallmark of brain networks is their inter-subject consistency; observed patterns in connectivity have been shown to demonstrate reproducible properties across subjects (Damoiseaux et al., 2006). This suggests significant benefits can be obtained by sharing information across subjects in a judicious manner. In fact, some of the most recent developments are based on collecting hundreds or even thousands of subjects’ data in a single data base (Di Martino et al., 2014).
- **Modularity:** Current methods do not actively incorporate domain knowledge relating to brain networks, a prime example of which is their modular structure which suggests that variables can be aggregated into non-overlapping modules or sub-networks (Sporns & Betzel, 2016). We note this property is not unique to brain networks, but also present in many real-world networks (Newman, 2006).

While motivated by fMRI, we note that these two properties are relevant to wide range of applications such as cyber-security, gene expression data and econometrics.

In this work, we propose a probabilistic latent variable model which is able to directly address the aforementioned issues. The proposed model consists of a low dimensional set of latent variables in a factor analytic model. The associated factor loading matrix is shared across classes and constrained to be non-negative and

orthonormal, thereby encoding module/community membership along its columns. Thus, the factors are interpreted as the activities in modules or communities.

Importantly, and in contrast to almost all related models, these latent variables or factors have full (i.e., non-diagonal) covariance structure which we term *latent connectivities*, giving the connectivity structure of the non-overlapping modules. The connectivity structure can be different between classes; however, the model can equally well be applied on data from a single class. Thus, we model both the grouping of variables, and the connectivity between the groups in a single probabilistic model, which can be seen as a variant of factor analysis. We argue that such a formulation leads to important benefits from the viewpoint of interpretation and identifiability whilst remaining plausible from an application perspective.

In contrast to classical Gaussian factor analysis, we are able to prove the uniqueness of the solution: the factors and loadings are identifiable like in (non-Gaussian) independent component analysis (Comon, 1994), largely based on non-negativity inherent in the module structure (Paatero & Tapper, 1994; Seung & Lee, 1999; Donoho & Stodden, 2004). We further propose an efficient parameter estimation algorithm based on score matching. Finally, we demonstrate that the proposed model can be extended to modelling directed connectivities between the latent variables. In this context, the factor loading matrix can be seen as a “pure” measurement model of a Bayesian network (Silva et al., 2006) of causal relationships between high-dimensional observations and their latent variables.

The remainder of this manuscript is organized as follows; in Section 2 we present the proposed model in the context of undirected latent variables. Section 3 provides an identifiability analysis for the proposed method. An efficient estimation algorithm based on score matching is presented in Section 4. The proposed method is extended to recover causal structure over latent variables in Section 5. Experimental results are presented in Section 6.

## 2 LATENT CONNECTIVITIES MODEL

We propose a latent variable model to accurately find modules (communities, clusters) and model their connectivities, possibly across multiple related classes (conditions, subjects). We assume we have access to multivariate data over  $N$  distinct classes, but all our results allow for the simple case  $N = 1$  as well. For a given class  $i$ , we write  $X^{(i)} \in \mathbb{R}^p$  to denote the  $p$ -dimensional observed random vector. The  $i$ th class is associated with a  $k$ -dimensional latent vector,  $Z^{(i)}$ , which is related to observations,  $X^{(i)}$ , via a loading matrix  $W \in \mathbb{R}^{p \times k}$ . We note that the loading matrix is shared across all classes and will serve to encode

module memberships across classes.

We start by a model of undirected connectivities in this section. Here, we assume that the data for each class follows a stationary multivariate Gaussian distribution with zero mean and covariance  $\Sigma^{(i)} \in \mathbb{R}^{p \times p}$ . Both observations and latent variables are taken to follow multivariate Gaussian distributions, such that:

$$Z^{(i)} \sim \mathcal{N}\left(0, G^{(i)}\right) \quad (1)$$

$$X^{(i)}|Z^{(i)} = z^{(i)} \sim \mathcal{N}\left(Wz^{(i)}, v^{(i)}I\right). \quad (2)$$

If  $G^{(i)}$  were diagonal, equations (1) and (2) would correspond to the traditional factor analysis or probabilistic PCA models (Tipping & Bishop, 1999). Our model is able to capture low-rank covariance structure via the loading matrix,  $W$ , as follows:

$$\Sigma^{(i)} = WG^{(i)}W^T + v^{(i)}I. \quad (3)$$

From equation (3) it follows that the loading matrix  $W$  serves to encode reproducible covariance structure which is present across all classes. In this work, we extend the traditional factor analysis model as follows:

- The loading matrix,  $W$ , is constrained to be non-negative and orthonormal. This leads to a loading matrix with at most one non-zero entry per row. We may interpret the columns of  $W$  as encoding membership to  $k$  non-overlapping modules or sub-networks.
- We introduce latent variables with a non-diagonal covariance structure, which we term *latent connectivities*. While the columns of the loading matrix encode module membership, the non-trivial covariance structure over latent variables may be interpreted as modeling marginal dependencies (i.e., connectivity) across distinct modules or sub-networks. Such an interpretation is very natural in many applied settings.

We note that the introduction of marginally dependent latent variables is not possible in the context of traditional factor analysis, since the effects of factor connectivity and factor loadings cannot be distinguished. In fact, an ordinary factor analysis model is non-identifiable even with uncorrelated factors. However, in combination with the aforementioned constraints on the loading matrix, it is possible to identify the latent connectivities in our model (see next section). We thus argue that our model is able to capture the modular nature of many real-world datasets and, due to its identifiability, yields easily interpretable results. Figure 1 provides an overview of the proposed model in the context of estimating brain connectivity networks.

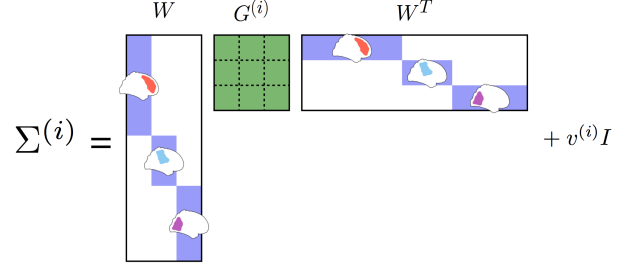


Figure 1: Visualization describing the various components of the proposed covariance model. The factor loading matrix,  $W$ , is shared across all subjects and serves to denote membership into non-overlapping brain modules. The *latent connectivity* across modules, parameterized by  $G^{(i)}$ , is allowed to vary across subjects.

### 3 IDENTIFIABILITY ANALYSIS

We note that without the introduction of constraints on  $W$ , the covariance model proposed in equation (3) is not unique. For example, it would be possible to reparameterize  $W$  such that  $G^{(i)}$  is diagonal matrix by using an eigen-value decomposition of  $G^{(i)}$ . However, the following properties demonstrate that non-negativity and orthonormality constraints are sufficient to ensure the solution is identifiable.

**Property 1.** *Assume non-negative, orthonormal  $W$ . Then at most one entry per row of  $W$  can be non-zero.*

*Proof.* Directly from the constraints on  $W$ , we can express the  $(i, j)$  entry of  $W^T W$  as:

$$(W^T W)_{ij} = \sum_{r=1}^k (W^T)_{ir} W_{rj} = \sum_{r=1}^k W_{ri} W_{rj} = \delta_{ij}$$

which, combined with non-negativity, implies that the  $i$  and  $j$  columns of  $W$  can have no overlapping support for  $i \neq j$ .  $\square$

**Property 2.** *Assume non-negative, orthonormal  $W$ . Then any matrix  $V \in \mathbb{R}^{k \times k}$  for which  $\tilde{W} = WV$  is non-negative and orthonormal must be the identity matrix or some permutation of the identity.*

*Proof.* By Property 1 we have that both  $W$  and  $\tilde{W}$  have at most one non-zero entry per row. Since  $\tilde{W}^T \tilde{W} = I$  we have that  $V^T V = I$ . Define  $c_i$  and  $\tilde{c}_i$  to be the index of the non-zero entry along the  $i$ th row for  $W$  and  $\tilde{W}$  respectively. By construction:

$$\tilde{W}_{i\tilde{c}_i} = (WV)_{i\tilde{c}_i} = \sum_{r=1}^k W_{ir} V_{r\tilde{c}_i} = W_{ic_i} V_{c_i\tilde{c}_i} > 0$$

where the final equality follows from the fact that  $W_{ic_i}$  is the only non-zero entry along the  $i$ th row of  $W$ . Since  $W_{ic_i} > 0$ , this implies that  $V_{c_i \tilde{c}_i} > 0$ . Furthermore, for  $j \neq \tilde{c}_i$  we have

$$\tilde{W}_{ij} = (WV)_{ij} = \sum_{r=1}^k W_{ir} V_{rj} = W_{ic_i} V_{c_i j} = 0$$

Since  $W_{ic_i} > 0$ , we must have that  $V_{c_i j} = 0$  whenever  $j \neq \tilde{c}_i$ . When combined with the fact that  $V^T V = I$ , it follows that  $V$  must either be the identity matrix of a permutation it.  $\square$

Property 2 indicates that the matrix  $W$  is uniquely defined in our model, and there is nothing like an undetermined factor rotation in conventional Gaussian factor analysis. By similar logic, Property 2 also implies the uniqueness of  $G^{(i)}$ .

## 4 ESTIMATION BY SCORE MATCHING

The parameters associated with the proposed model consist of the loading matrix,  $W$ , the latent variable covariances,  $\{G^{(i)}\}$ , and the observation noise,  $\{v^{(i)}\}$ . One potential strategy is to estimate latent variables in an expectation-maximization framework. However, due to relative simplicity of the proposed covariance model we propose to directly marginalize out latent variables.

Parameters may also be estimated via maximum likelihood estimation, however, this results in an iterative algorithm where the computational cost of each parameter update is  $\mathcal{O}(p^3)$  (a derivation of which is provided in the Supplementary materials). Instead, we propose to estimate parameters by score matching (Hyvärinen, 2005), leading to an algorithm with a computational cost of  $\mathcal{O}(p^2 k)$  per iteration. This is a significant reduction as we will typically expect  $k \ll p$ . While score matching is typically used in the context of unnormalized statistical models, it may often result in optimization-related benefits for normalized models as well (Hyvärinen, 2007; Lin et al., 2016).

In the context of multivariate Gaussian data, the score matching objective function is defined as (Hyvärinen, 2005):

$$J = \sum_{i=1}^N -\text{tr}(\Omega^{(i)}) + \frac{1}{2} \text{tr}(\Omega^{(i)} \Omega^{(i)} K^{(i)}), \quad (4)$$

where  $K^{(i)}$  is the sample covariance matrix for class  $i$  and  $\Omega^{(i)}$  is the inverse covariance, which may be computed by the Sherman-Woodbury identity as:

$$\Omega^{(i)} = v^{(i)-1} \left( I - W G^{(i)} (G^{(i)} + v^{(i)} I)^{-1} W^T \right).$$

Directly optimizing the score matching objective (equation (4)) under non-negativity and orthonormality constraints on the loading matrix is challenging. One potential strategy is to employ projected gradient descent as suggested by Hirayama et al. (2016). However, projecting onto the non-negative Stiefel manifold is undesirable as it requires  $W$  to have at most one non-zero entry per row at each step of the optimization algorithm (see Property 1 above). Such an approach is therefore highly dependent to the random initialization of the loading matrix.

In this work we seek to minimize equation (4) in a constrained optimization framework. This allows for the orthonormality constraints to be enforced via an augmented Lagrangian penalty (Bertsekas, 2014), while the non-negativity is enforced at each iteration by projecting onto the non-negative orthant.

The objective function associated with the augmented Lagrangian is defined as:

$$\tilde{J} = J + \frac{\rho}{2} \|W^T W - I_k\|_2^2 + \text{tr}(\Lambda^T (W^T W - I_k)),$$

where  $\Lambda \in \mathbb{R}^{k \times k}$  are Lagrange multipliers enforcing the orthonormality constraints,  $\rho$  is a positive scalar parameter parameterizing the augmented penalty term and  $J$  is the original score matching objective. We may then proceed to iteratively optimize each of the parameters using gradient descent. In particular, the gradient of the score matching objective with respect to the loading matrix can be computed as:

$$\frac{\partial J}{\partial W} = - \sum_{i=1}^N K^{(i)} W A^{(i)} \left( I - \frac{1}{2} A^{(i)} \right),$$

where we define  $A^{(i)} = G^{(i)} (G^{(i)} + v^{(i)} I)^{-1}$ . Similarly, in the case of the latent connectivities,  $G^{(i)}$ , and observation noise,  $v^{(i)}$ , we have

$$\frac{\partial J}{\partial G^{(i)}} = v^{(i)-2} \left[ v^{(i)} I - W^T K^{(i)} W \left( I - A^{(i)} \right) \right] \times \left[ (G^{(i)} + v^{(i)} I)^{-1} \left( I - A^{(i)} \right) \right].$$

$$\begin{aligned} \frac{\partial J}{\partial v^{(i)}} &= \sum_{i=1}^N -v^{(i)-3} \text{tr} \left( K^{(i)} - v^{(i)} I \right) \\ &\quad + v^{(i)-3} \text{tr} \left( W^T K^{(i)} W H_1^{(i)} \right) + H_2^{(i)} \end{aligned}$$

where  $H_1^{(i)} \in \mathbb{R}^{k \times k}$  and  $H_2^{(i)} \in \mathbb{R}$  are defined, together with the relevant derivations, in the Supplementary material.

Estimation proceeds by iteratively updating each of the parameters in a gradient descent framework. In the context of the loading matrix the step-size,  $\eta$ , is selected via the

Armijo rule and we project onto the non-negative orthant at each iteration, resulting in an update of the form:

$$W \leftarrow \mathcal{P}_+ \left( W - \eta \left( \frac{\partial J}{\partial W} + \rho(WW^T W - W) + W\Lambda \right) \right)$$

where  $\mathcal{P}_+(x) = \max(0, x)$  is the projection onto the non-negative orthant. Moreover, in the case of latent variable connectivities we have:

$$\frac{\partial J}{\partial G^{(i)}} = 0 \iff G^{(i)}(G^{(i)} + v^{(i)}I)^{-1} = I - v^{(i)}(W^T K^{(i)} W)$$

which after some manipulation yields a closed form update for the latent connectivity structure as:

$$G^{(i)} \leftarrow W^T K^{(i)} W - v^{(i)} I \quad (5)$$

By writing  $W^T K^{(i)} W = (X^{(i)} W)^T (X^{(i)} W)$  we note that this update has an intuitive interpretation as the covariance across estimated modules. The Lagrange multipliers are updated as (Bertsekas, 2014):

$$\Lambda \leftarrow \Lambda + \rho(W^T W - I)$$

It is important to note that the proposed method only enforces orthonormality on the loading matrix in the limit of convergence. However, the updates provided above are premised on the assumption that  $W$  is orthonormal. As such, the aforementioned updates only correspond to approximations in the case where  $W$  is non-orthonormal.

**Hyper-parameter Tuning** In practice, the proposed model requires the selection of a single hyper-parameter,  $k$ , which determines the dimensionality of latent variables. We propose to tune  $k$  by minimizing the negative log-likelihood over held-out data.

## 5 EXTENSION TO DIRECTED CONNECTIVITY

In this section we describe a natural extension of the aforementioned model to estimate causal structure (directed connectivity) across latent variables. As in the previous section, we restrict ourselves to linear latent variables models where each observed variable is conditionally dependent on a single latent variable. In the context of Bayesian networks such models are known as *Pure 1-Factor* models (Silva et al., 2006; Kummerfeld & Ramsey, 2016). We note that such an assumption directly corresponds to each row of the loading matrix,  $W$ , containing at most one non-zero entry. This is precisely what is enforced by the non-negativity and orthonormality assumptions introduced in this work. As such, restricting the loading matrix in this manner corresponds to a natural and frequently employed assumption when attempting to recover causal structure (Silva et al., 2006).

We follow Shimizu et al. (2006) and study a non-Gaussian variant of Bayesian networks over latent variables. Formally, we assume variables  $Z_j^{(i)}, j \in \{1 \dots k\}$  can be arranged in a causal ordering such no later variables causes a variable ahead of it in the order. We denote such an ordering by  $k(j)$  and assume that each variable,  $Z_j^{(i)}$ , is a linear function of earlier variables together with a non-Gaussian disturbance,  $e_j^{(i)}$ , such that:

$$Z_j^{(i)} = \sum_{k(r) < k(j)} b_{jr}^{(i)} Z_r^{(i)} + e_j^{(i)}. \quad (6)$$

Due to the linear nature of dependencies, we can write equation (6) as follows:

$$Z^{(i)} = B^{(i)} Z^{(i)} + e^{(i)} \quad (7)$$

$$= \left( I - B^{(i)} \right)^{-1} e^{(i)}. \quad (8)$$

The matrix  $B^{(i)}$  encodes a Directed Acyclic Graph (DAG), which corresponds to the *structural model* over latent variables. We note that equation (7) corresponds to a Linear, non-Gaussian, acyclic model (LiNGAM; Shimizu et al., 2006). As in the previous section, observed data are subsequently related as follows:

$$X^{(i)} | Z^{(i)} = z^{(i)} \sim \mathcal{N}(W z^{(i)}, v^{(i)} I) \quad (9)$$

where the loading matrix encodes the *measurement model*.

To date, a wide range of algorithms have been proposed to estimate measurement models. Prominent examples include the `BuildPureClusters` and `FindOneFactorCluster` algorithms. However, such methods cannot easily be extended to the context of multiple related datasets where the underlying structural models are heterogeneous. Moreover, such methods do not scale well to high-dimensional data. In contrast, our method proposed below can easily accommodate such data and is therefore a good candidate to accurately recover the measurement model. Once the measurement model has been estimated, we may proceed to infer the causal structure over latent variables using established methods such as LiNGAM.

We now outline our two-stage procedure to estimate *Pure 1-Factor* latent variable models. First, the score matching algorithm detailed in Section 4 is employed to estimate the measurement model (i.e., the loading matrix  $W$ ). Given a measurement model, there are a variety of algorithms to estimate the structural model. In this work we follow Shimizu et al. (2009) and propose to recover causal dependencies over latent variables by applying LiNGAM to projected observations,  $\hat{W}^T X^{(i)}$ . This step is performed independently for each of the  $N$  classes.

We note that the likelihood proposed in Section 2 is misspecified in the context of non-Gaussian Bayesian networks considered here (as latent variables follow non-Gaussian distribution). However, in the first stage we are only interested in the estimation of the loading matrix,  $W$ , using covariance information which is unaffected by non-Gaussianity over latent variables. In fact, as we allow for arbitrary (i.e., non-diagonal) latent connectivities, the proposed model is able to accommodate covariances induced by the causal structure whilst estimating the loading matrix. Alternative approaches, such as the `BuildPureClusters` algorithm, are also based exclusively on studying covariance structure, albeit while introducing additional higher-order algebraic constraints.

## 6 EXPERIMENTAL RESULTS

We systematically assess the performance of the proposed model using simulated data under the Gaussian factor analysis model of Section 2 as well as the directed latent Pure Bayesian network model described in Section 5. Finally, we present an application to resting-state fMRI data from the ABIDE consortium (Di Martino et al., 2014).

### 6.1 PERFORMANCE METRICS

We assess the performance of the proposed method in the context of both a single class and multiple related classes. Throughout these simulations, we quantify the performance of various methods based on three distinct tasks:

1. Recovery of the loading matrix,  $W$ . This corresponds to accurately recovering the mixing matrix, and by implication, the module memberships for each variable.
2. Recovery of the latent connectivity structure. In the context of undirected latent connectivities, this corresponds to accurately recovering the covariance structure,  $G^{(i)}$ , across estimated modules. Conversely, in the context of directed latent Bayesian network models it corresponds to accurately recovering the causal dependence structure over latent variables, encoded in  $B^{(i)}$ .
3. In the context of undirected connectivities we also measure the negative log-likelihood over unseen data. This provides an objective quantification of how well the proposed method is able to model the data in comparison to alternative methods.

### 6.2 GAUSSIAN FACTOR ANALYSIS MODEL

Data was generated according to the model described in Section 2. The covariance structure for latent variables,  $G^{(i)}$ , was randomly generated for each class by sampling the lower triangular entries from a standard Gaussian distribution and multiplying by its transpose. We note that generating  $G^{(i)}$  in this fashion will randomly introduce both positive and negative correlations across modules. A random loading matrix,  $W$ , was generated by sampling uniform random variables and projecting onto the non-negative Stiefel manifold (this involved retaining the largest entry per row and setting all other entries to zero). Observations for each class were subsequently generated according to equations (1) and (2). The dimensionality of observations and latent variables was set to  $p = 50$  and  $k = 5$  respectively. Data was generated in this manner for  $N$  subjects. We consider two cases:  $N = 1$  and  $N = 10$ , which correspond to the single class and multiple class scenarios. Each experiment was repeated 500 times.

The proposed method was benchmarked against several widely used alternatives. In the context of recovering the loading matrix,  $W$ , and covariance structure of latent variables,  $G^{(i)}$ , we compare to non-negative PCA (using the method proposed by Sigg & Buhmann (2008)) and traditional factor analysis (where Varimax rotation was employed, since it should be able to recover module structure in the factor loadings). We note that while these methods do not explicitly model the covariance structure across latent variables, this can be estimated by first projecting observations using the estimated loading matrix and subsequently studying the covariance structure. Indeed, this is related to the update performed by the proposed method in equation (5). When measuring the negative log-likelihood over unseen data we add additional comparisons against the sample covariance matrix and the estimate proposed by Ledoit & Wolf (2003) and the graphical Lasso.

Simulation results for a single ( $N = 1$ ) class are shown along the top panel of Figure 2. The top left panel plots the squared error when estimating the loading matrix. In the presence of small sample sizes, the proposed method is comparable to non-negative PCA. However, as the sample size increases the proposed method consistently outperforms alternative methods as it is able to model the connectivity of latent variables. Additional results relating the clustering implied by estimated loading matrices are provided in the supplementary materials. Results for the estimation of latent connectivities,  $G^{(i)}$ , are shown in the top middle panel. We note that the proposed method comfortably outperforms competing methods. Finally, the right panel shows mean negative log-likelihood over unseen data; the proposed method consistently outperforms

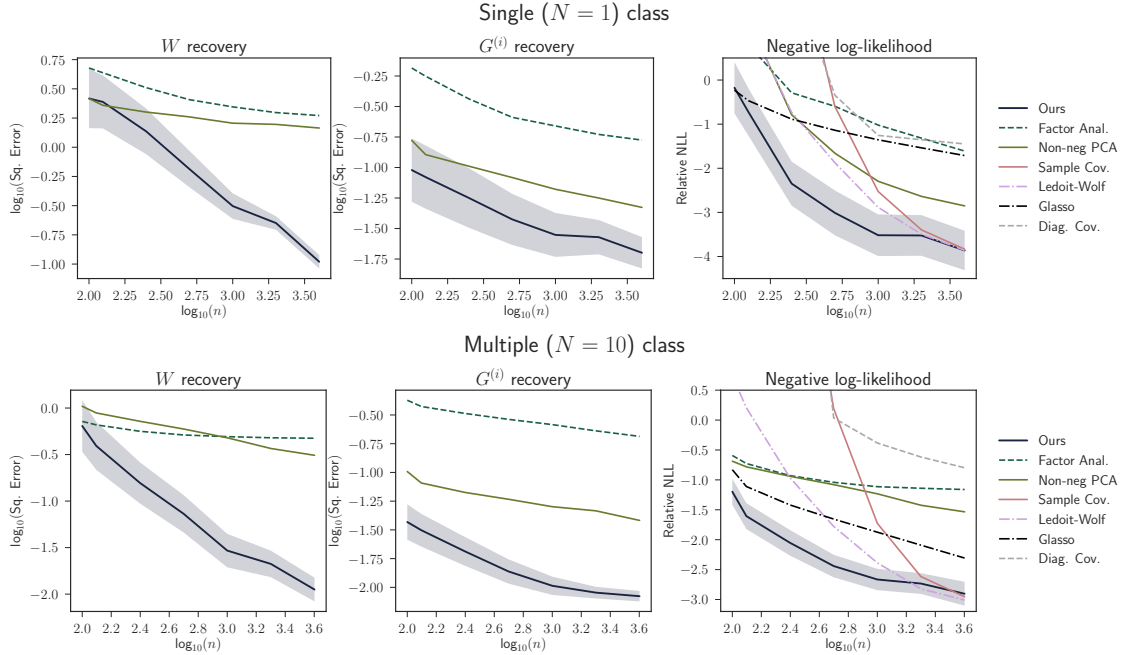


Figure 2: Simulated data results for Gaussian latent variable models with single ( $N = 1$ ) and multiple ( $N = 10$ ) classes are shown along the top and bottom panels respectively. Left and middle panels plot the mean squared error for the estimated loading and latent variable covariance matrices as a function of sample size,  $n$ . Right panels shows the mean negative log-likelihood for unseen data as a function of sample size,  $n$ . Shaded regions correspond to 95% error bars.

alternative methods for small and moderate sample sizes and remains competitive as sample size increases.

The bottom panel of Figure 2 plots results for the general case of multiple subjects. While the loading matrix is shared across all classes, the latent covariance structure is heterogeneous. While the proposed method is well-suited to accommodate such data, methods such as PCA and factor analysis are not directly applicable. As such, non-negative PCA and factor analysis models were applied using a naive aggregation of the data which concatenated observations across all classes. As before, by accurately modeling the marginal dependencies across latent variables, the proposed method is able to obtain far more accurate estimates of both the loading matrix as well as the latent connectivity structure. The bottom right panel of Figure 2 plots the mean negative log-likelihoods over unseen data and provides empirical evidence that the proposed model provides an accurate estimate of covariance structure.

In addition to quantifying the recovery of the loading matrix and latent connectivities, we also quantify the computation cost associated with each algorithm. The top panel of Figure 4 shows the mean running time as a function of the number of observed variables,  $p$ . The results validate our prior claims that the score matching

algorithm yields significant computational improvements compared to the maximum likelihood algorithm described in the Supplementary material. For high-dimensional data, the proposed score matching algorithm also improves on the running time compared to both factor analysis and non-negative PCA.

### 6.3 LATENT PURE BAYESIAN NETWORKS

In this section we perform experiments where the data is generated as described in Section 5. This involved generating latent variables following a non-Gaussian variant of Bayesian networks where the disturbances,  $e^{(i)}$ , were simulated according to a Logistic distribution. The weights for the structural model, encoded in  $B^{(i)}$ , were randomly generated together with a distinct random causal ordering for each class. The loading matrix,  $W$ , was generated as in Section 6.2.

In the context of recovering the measurement model (i.e., the loading matrix  $W$ ) we benchmark the proposed method with factor analysis and non-negative PCA as well as the `FindOneFactorClusters` (FOFC) algorithm<sup>1</sup> proposed by Kummerfeld & Ramsey (2016). Formally, the FOFC algorithm only returns non-overlapping

<sup>1</sup>The Tetrad project implementation was employed.

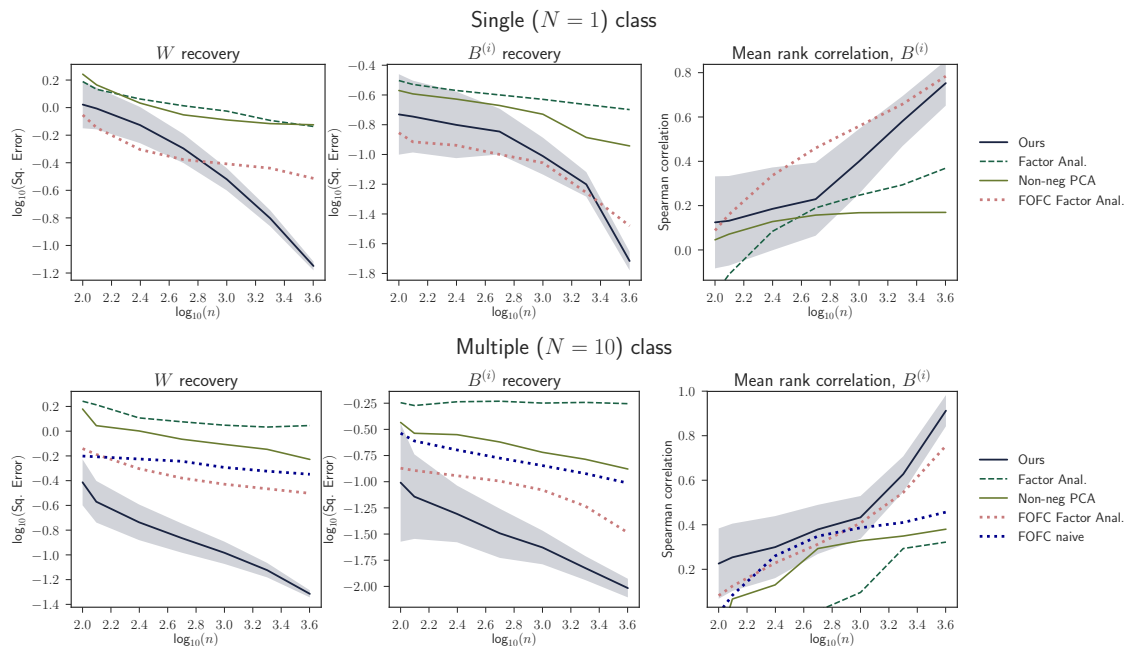


Figure 3: Simulated data results for latent Bayesian networks with single ( $N = 1$ ) and multiple ( $N = 10$ ) classes are shown along the top and bottom panels respectively. Left and middle panels plot the mean squared error for the estimated loading matrix and structural dependency matrices as a function of sample size,  $n$ . The right panels show the mean correlation between the estimated causal ordering of latent variables and the true causal order. For all algorithms the causal structure over latent variables was estimated by LiNGAM. Shaded regions correspond to 95% error bars.

clusters of observed variables which share the same latent parent. In order to estimate the associated loading matrix we subsequently employ factor analysis whilst preserving the 1-Factor structure as suggested by Shimizu et al. (2009). Given an accurate estimate of  $W$ , we may directly project observations,  $\hat{Z}^{(i)} = \hat{W}^T X^{(i)}$  and apply traditional causal discovery algorithms by treating  $\hat{Z}^{(i)}$  as observed variables (Silva et al., 2006; Shimizu et al., 2009). Throughout these experiments, the causal structure of latent variables was inferred using LiNGAM.

Figure 3 shows results for a single ( $N = 1$ ) and multiple ( $N = 10$ ) class cases along the top and bottom row respectively. The left panels show the squared error when estimating the loading matrix. In the context of causal models this corresponds to accurately recovering the measurement model. When data is only available for a single class (top left panel) the performance of the proposed method is similar to that of the FOFC algorithm. However, when data across multiple classes is available, the proposed method is able to exploit this information and improve upon the FOFC algorithm as shown in the bottom left panel. We also plot the performance of running the FOFC when naively aggregating data across multiple subjects. Such a naive aggregation leads to worse performance as each class has its own latent causal structure.

We observe a similar pattern when studying the recovery of the structural equations, as shown in the middle and right panels. The proposed method out-performs both factor analysis and PCA and is comparable to FOFC in the context of a single ( $N = 1$ ) class. However, in the context of multiple classes the proposed method is able to out-perform alternative methods.

Finally, the bottom panel of Figure 4 plots the mean running time as the number of observed variables,  $p$ , increases. In terms of running time, the proposed method is significantly faster than the FOFC algorithm.

#### 6.4 APPLICATION TO FMRI DATA

In this section we apply the proposed method to resting-state fMRI data taken from the ABIDE consortium (Di Martino et al., 2014). Data was collected from the University of Maryland site corresponding to 53 healthy controls as well as 53 age matched Autism Spectrum Disorder (ASD) subjects. Data from each subject was treated as a distinct class, resulting in  $N = 106$  classes. Data were preprocessed via the CPAC pipeline from the ABIDE repository<sup>2</sup>. Time courses were then extracted from 116

<sup>2</sup><http://preprocessed-connectomes-project.org/abide/>

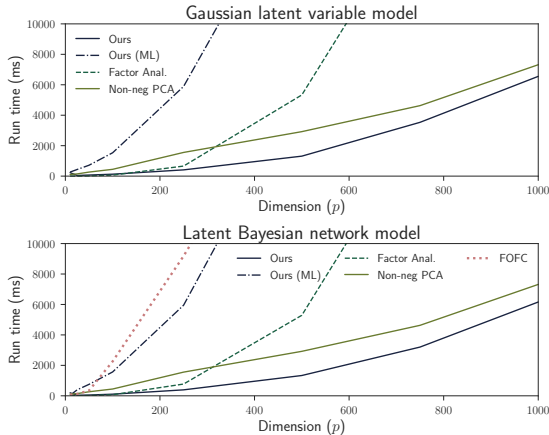


Figure 4: Mean running times (in milliseconds) taken to estimate the factor loading matrix,  $W$ , when data is generated according to the Gaussian latent variable model (top) and latent Bayesian network model (bottom). Mean run times based on 10 experiments run on a Macbook Pro (3.5 GHz Intel Core i7, 16 GB RAM).

regions defined by the Automated Anatomical Labeling (AAL) atlas, yielding 296 observations over 116 nodes for each subject. The data was analyzed under the assumption of undirected latent connectivity structure as there is a large literature discussing differences in covariance structure between healthy controls and ASD subjects (Fox & Greicius, 2010).

The proposed method requires the specification of a single parameter,  $k$ , which dictates the dimensionality of latent variables. As discussed in Section 4, this parameter was selected by minimizing the negative log-likelihood over held-out data, resulting in an choice of  $k = 5$  modules. Figure 5 shows the  $k = 5$  estimated modules obtained by applying the proposed method. The spatial consistency and inter-hemispheric symmetry of module assignments reflects the anatomical and functional architecture of the brain. Moreover, edges in Figure 5 highlight significant differences in covariance structure of latent variables between healthy controls and ASD subjects. Permutation tests were performed on each edge of the latent connectiv-

Table 1: Mean log-likelihood scores on unseen data for  $N = 106$  subjects (standard deviations are provided in brackets).

Method	Log-likelihood
Ours	<b>-163.76 (8.86)</b>
Non-neg. PCA	-190.47 (9.26)
Factor Anal.	-193.89 (8.05)
Glasso	-198.58 (9.05)
Lediot-Wolf	-247.91 (10.88)
Sample Cov.	-329.75 (14.98)

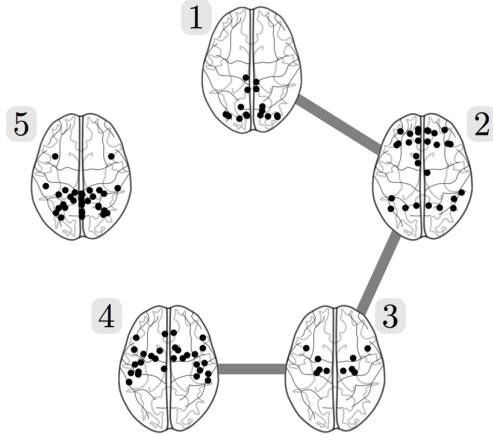


Figure 5: Visualization of estimation brain modules. Each black node represents a distinct brain region. Note that estimated modules are both spatially consistent and symmetric across hemispheres. Edges indicate significant increase in inter-module marginal dependence for ASD subjects compared to healthy controls (edge-wise Bonferroni corrected permutation tests,  $p < 0.01$ ).

ity structure and Bonferroni corrected for multiple testing. Results indicate that ASD subjects demonstrate increased connectivity for which there is growing evidence (Keown et al., 2013). In particular, we note increased connectivity between the frontoparietal regions (module 2) and both the occipital regions (module 1) and the hippocampus, amygdala and temporal lobes (module 3). Finally, Table 1 reports the mean log-likelihood scores on unseen data for all  $N = 106$  subjects, demonstrating the proposed model accurately captures covariance structure of fMRI data.

## 7 CONCLUSION

We have proposed a probabilistic model which simultaneously performs grouping of variables as well as estimation of the *latent connectivities* between groups. The proposed method can be seen as an extension of traditional factor analysis with the important difference that latent variables are allowed to have a full (i.e., non-diagonal) covariance structure while the loading matrix is restricted to encode module membership. The proposed model can directly accommodate datasets across multiple related classes under the assumption that variables across classes share the same modularity or community structure. While the proposed method is introduced in the context of Gaussian latent variable models, we also demonstrate that it may be extended to latent Bayesian network models. We present experiments on synthetic and fMRI data which demonstrate the capabilities of our approach, in particular showing it successfully scales to high-dimensional data.



## References

- Airoldi, Edoardo et al. Mixed Membership Stochastic Blockmodels. *J. Mach. Learn. Res.*, 9(2008):1981–2014, 2008. ISSN 1532-4435.
- Bertsekas, Dimitri P. *Constrained optimization and Lagrange multiplier methods*. Academic Press, 2014.
- Buesing, Lars et al. Clustered factor analysis of multineuronal spike data. *NIPS*, 27:3500–3508, 2014.
- Comon, Pierre. Independent component analysis - a new concept? *Signal Processing. Signal Processing*, 36: 287–314, 1994.
- Damoiseaux, J S et al. Consistent resting-state networks across healthy subjects. *Proc. Natl. Acad. Sci.*, 103(37): 13848–13853, 2006. ISSN 0027-8424.
- Danaher, Patrick et al. The joint graphical lasso for inverse covariance estimation across multiple classes. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 76(2):373–397, 2014.
- Dempster, A P. Covariance Selection. *Biometrics*, 28(1): 157, 1972.
- Di Martino, Adriana et al. The autism brain imaging data exchange: Towards a large-scale evaluation of the intrinsic brain architecture in autism. *Mol. Psychiatry*, 19(6):659–667, 2014. ISSN 14765578.
- Donoho, David and Stodden, Victoria. When does non-negative matrix factorization give a correct decomposition into parts? *NIPS*, pp. 1141–1148, 2004.
- Fox, Michael D. and Greicius, Michael. Clinical applications of resting state functional connectivity. *Front. Syst. Neurosci.*, 4, 2010.
- Hirayama, Jun Ichiro et al. Characterizing variability of modular brain connectivity with constrained principal component analysis. *PLoS One*, 11(12), 2016.
- Hyvärinen, Aapo. Estimation of non-normalized statistical models by score matching. *J. Mach. Learn. Res.*, 6: 695–708, 2005.
- Hyvärinen, Aapo. Some extensions of score matching. *Comput. Stat. Data Anal.*, 51(5):2499–2512, 2007.
- Keown, Christopher Lee et al. Local functional overconnectivity in posterior brain regions is associated with symptom severity in autism spectrum disorders. *Cell Rep.*, 5(3):567–572, 2013.
- Kummerfeld, Erich and Ramsey, Joseph. Causal Clustering for 1-Factor Measurement Models. *KDD*, pp. 1655–1664, 2016.
- Ledoit, Olivier and Wolf, Michael. Improved estimation of the covariance matrix of stock returns with an application to portfolio selection. *J. Empir. Financ.*, 10(5): 603–621, 2003.
- Lin, Lina et al. Estimation of high-dimensional graphical models using regularized score matching. *Electron. J. Stat.*, 10:806–854, 2016.
- Marlin, Benjamin and Murphy, Kevin. Sparse Gaussian Graphical Models with Unknown Block Structure. *ICML*, pp. 705–712, 2009.
- Monti, Ricardo Pio et al. Learning population and subject-specific brain connectivity networks via Mixed Neighborhood Selection. *Ann. Appl. Stat.*, 11(4):2142–2164, 2017.
- Newman, M E J. Modularity and community structure in networks. *Proc. Natl. Acad. Sci.*, 103(23):8577–8582, 2006.
- Paatero, Pentti and Tapper, Unto. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.
- Pearl, Judea. *Causality*. Cambridge University Press, 2009.
- Sasaki, Hiroaki et al. Simultaneous estimation of non-gaussian components and their correlation structure. *Neural Comput.*, 29:2887–2924, 2017.
- Seung, H. Sebastian and Lee, Daniel D. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.
- Shimizu, Shohei et al. A Linear Non-Gaussian Acyclic Model for Causal Discovery. *J. Mach. Learn. Res.*, 7: 2003–2030, 2006.
- Shimizu, Shohei et al. Estimation of linear non-Gaussian acyclic models for latent factors. *Neurocomputing*, 72 (7-9):2024–2027, 2009.
- Sigg, Christian D and Buhmann, Joachim M. Expectation-maximization for sparse and non-negative PCA. *ICML*, pp. 960–967, 2008.
- Silva, Ricardo et al. Learning the structure of linear latent variable models. *J. Mach. Learn. Res.*, 7:191–246, 2006.
- Sporns, Olaf and Betzel, Richard F. Modular Brain Networks. *Annu. Rev. Psychol.*, 67(1):613–640, 2016.
- Tipping, M. E. and Bishop, C. M. Probabilistic Principle Component Analysis. *J. R. Stat. Soc. Ser. B (Statistical Methodol.)*, 11(2):150–210, 1999.
- Varoquaux, Gaël et al. Brain covariance selection: better individual functional connectivity models using population prior. *NIPS*, 2010.

---

# Improved Stochastic Trace Estimation using Mutually Unbiased Bases

---

Jack Fitzsimons<sup>1</sup>

Michael Osborne<sup>1</sup>

Stephen Roberts<sup>1</sup>

Joseph Fitzsimons<sup>2,3</sup>

<sup>1</sup> Information Engineering, University of Oxford, UK

<sup>2</sup> Singapore University of Technology and Design, Singapore

<sup>3</sup> Centre for Quantum Technologies, National University of Singapore, Singapore

## Abstract

The paper begins by introducing the definition and construction of mutually unbiased bases, which are a widely used concept in quantum information processing but have received little to no attention in the machine learning and statistics literature. We demonstrate their usefulness by using them to create a new sampling technique which offers an improvement on the previously well established bounds of stochastic trace estimation. This approach offers a new state of the art single shot sampling variance while requiring  $\mathcal{O}(\log(n))$  random bits for  $\mathbf{x} \in \mathbb{R}^n$  which significantly improves on traditional methods such as fixed basis methods, Hutchinson’s and Gaussian estimators in terms of the number of random bits required and worst case sample variance.

## 1 INTRODUCTION

Function space representations and transformations are at the heart of many machine learning techniques. For example, the relationship between computational space and Fourier space arises throughout machine learning literature, from classic shift invariant filters studied in image processing through to modern techniques for kernel approximation such as random Fourier features [1]. The power of random Fourier features, for example, is introduced by the unbiased relationship of the computational and Fourier bases, that is to say that a Dirac-delta distribution in one basis is represented with uniformly distributed mass in the other.

In this work we take advantage of not only pairs of mutually unbiased bases, but entire sets of them. We believe that the application of mutually unbiased bases has potential to improve kernel matrix approximation

and feature learning. To exemplify their applicability, we demonstrate their ability to create a novel sampling method which improves upon the well established error bounds of stochastic trace estimation.

The problem of stochastic trace estimation is relevant to a range of problems from physics and applied mathematics such as electronic structure calculations [2], seismic waveform inversion [3], discretized parameter estimation problems with PDEs as constraints [4] and approximating the log determinant of symmetric positive semi-definite matrices [5]. Machine learning, in particular, is a research domain which has many uses for stochastic trace estimation. They have been used efficiently by Generalised Cross Validation (GCV) in discretized iterative methods for fitting Laplacian smoothing splines to very large datasets [6], computing the number of triangles in a graph [7, 8], string pattern matching [9, 10] and training Gaussian Processes using score functions [11]. Motivated by accelerating Gaussian graphical models, Markov random fields, variational methods and Bregman divergences, work based on stochastic trace estimation has also been developed to improve the computational efficiency of log determinant calculations [12].

Stochastic trace estimation endeavours to choose  $n$ -dimensional vectors  $\mathbf{x}$  such that the expectation of  $\mathbf{x}^T A \mathbf{x}$  is equal to the trace of the implicit symmetrical positive semi definite matrix  $A \in \mathbb{R}^{n \times n}$ . It can be seen that many sampling policies satisfy this condition. Due to this, several metrics are used in order to choose a sampling policy such as the single shot sampling variance, the number of samples to achieve a  $(\epsilon, \delta)$ -approximation and the number of random bits required to create  $\mathbf{x}$  [13]. This last metric is motivated in part by the relatively long timescales for hardware random number generation, and concerns about parallelising pseudo-random number generators.

In this work we propose a new stochastic trace estimator based on mutually unbiased bases (MUBs) [14], and

quantify the single shot sampling variance of the proposed MUBs sampling method and its corresponding required number of random bits. We will refer to methods which sample from a fixed set of basis functions as being fixed basis sampling methods. For example, we can randomly sample the diagonal values of the matrix  $A$  by sampling  $\mathbf{x}$  from the set of columns which form the identity matrix. This is referred to as the unit vector estimator in the literature [13]. Other similar methods sample from the columns Discrete Fourier Transform (DFT), the Discrete Hartley Transform (DHT), the Discrete Cosine Transform (DCT) or a Hadamard matrix. We prove that sampling from the set of mutually unbiased bases significantly reduces this single shot sample variance, in particular in the worst case bound.

The paper is laid out as follows: Section 2 gives a brief introduction to mutually unbiased basis and their construction, Section 3 describes our novel approach of using mutually unbiased bases for trace estimation and Section 3.2 gives a rigorous analysis of the new estimator. Section 4 compares the proposed MUBs estimator to established approaches both in terms of the analytic expectation of sample variance and as applied to synthetic and real data. The tasks of counting the number of triangles in a graph and of estimating the log determinant of kernel matrices are considered as an example application.

## 2 MUTUALLY UNBIASED BASES

Linear algebra has found application in a diverse range of fields, with each field drawing from a common set of tools. However, occasionally, techniques developed in one field do not become well known outside of that community, despite the potential for wider use. In this work, we will make extensive use of mutually unbiased bases, sets of bases that arise from physical considerations in the context of quantum mechanics [14] and which have been extensively exploited within the quantum information community [15]. In quantum mechanics, physical states are represented as vectors in a complex vector space, and the simplest form of measurement projects the state onto one of the vectors from some fixed orthonormal basis for the space, with the probability for a particular outcome given by the square of the length of the projection onto the corresponding basis vector<sup>1</sup>. In such a setting, it is natural to ask about the existence of pairs or sets of measurements where the outcome of one measurement reveals nothing about the outcome of another measurement, and effectively erases any information about the outcome had the alternate measure-

<sup>1</sup>For a more comprehensive introduction to the mathematics of quantum mechanics in finite-dimensional systems, we refer the reader to [16]

ment instead been performed. As each measurement corresponds to a particular basis, such a requirement implies that the absolute value of the overlap between pairs of vectors drawn from bases corresponding to different measurements be constant. This leads directly to the concept of mutually unbiased bases (MUBs).

A set of orthonormal bases  $\{B_1, \dots, B_n\}$  are said to be mutually unbiased if for all choices of  $i$  and  $j$ , such that  $i \neq j$ , and for every  $\mathbf{u} \in B_i$  and every  $\mathbf{v} \in B_j$ ,  $|\mathbf{u}^\dagger \mathbf{v}| = \frac{1}{\sqrt{n}}$ , where  $n$  is the dimension of the space. While for real vector spaces the number of mutually unbiased bases has a complicated relationship with the dimensionality [17], for complex vector spaces the number of mutually unbiased bases is known to be exactly  $n + 1$  when  $n$  is either a prime or an integer power of a prime [18]. Furthermore, a number of constructions are known for finding such bases [18]. When  $n$  is neither prime nor a power of a prime, the number of mutually unbiased bases remains open, even for the case of  $n = 6$  [19], but is known to be at least  $p_1^{d_1} + 1$ , where  $n = \prod_i p_i^{d_i}$  and  $p_i$  are prime numbers such that  $p_i < p_{i+1}$  for all  $i$ .

One practical method for constructing MUBs is to use the unitary operators method with finite fields [20], which is effective when the dimensionality of the space is either prime or a prime power. For conciseness, we will outline the procedure for only the prime dimensionality case but note that any integer dimensional space is at most bounded by two times its closest prime power dimension which adds a constant cost to the memory and runtime performance. First, let us construct the matrix  $X$  as the identity matrix with the columns shifted one to the left creating the form,

$$X = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ 1 & 0 & 0 & \dots & 0 \end{bmatrix}$$

and letting  $Z$  be a diagonal matrix with elements set to the roots of unity,  $Z_{k,k} = \exp\left(\frac{2k\pi i}{n}\right)$ . Given these two matrices a set of mutually unbiased bases are found as the eigenvectors of the matrices,

$$X, Z, XZ, XZ^2, \dots, XZ^{n-1}.$$

At first glance it may appear that the computational cost of constructing vectors from these bases is  $\mathcal{O}(d^3)$  due to the cost decomposing these matrices in  $\mathbb{C}^{n \times n}$ , however, under more scrutiny we can see that  $X$  is a circulant permutation matrix and as such its eigenvectors are equal to  $U_{k,j} = \frac{1}{\sqrt{n}} \exp\left(\frac{jk2\pi i}{n}\right)$  irrespective of the dimensionality,

where  $j$  indexes the elements of the eigenvector and  $v$  indexes which eigenvector is under consideration.

As the elements the diagonal matrix  $Z$  are the roots of unity in ascending order, it can be seen that  $\exp\left(\frac{2\pi i}{n}\right)^k Z^k X = Q^k X = X Z^k$ , where  $Q$  is some matrix of the same form as  $Z$  but with a shift of phase of the non-zero elements. As such, by writing the eigenbasis of  $X = U^{-1}\Sigma U$  we can derive the eigenbasis of  $XZ^k$  for arbitrary value  $k$  with eigen decomposition  $XZ^k = \hat{U}^{-1}\hat{\Sigma}\hat{U}$ ,

$$\begin{aligned} XZ^k &= U^\dagger \Sigma U Z^k \\ &= Q^k U^{-1} \Sigma U \end{aligned}$$

Next, we pull  $Q^{\frac{k}{2}}$  and  $Z^{\frac{k}{2}}$  through the eigenvectors by observing that we can transform the eigenvectors as  $\Sigma = Z^{\frac{1}{2}} \hat{\Sigma} Q^{\frac{k}{2}}$ ,

$$\begin{aligned} XZ^k &= Q^{\frac{k}{2}} U^{-1} \Sigma U Z^{\frac{k}{2}} \\ &= Q^{\frac{k}{2}} U^{-1} Z^{\frac{1}{2}} \hat{\Sigma} Q^{\frac{k}{2}} U Z^{\frac{k}{2}} \\ &= \hat{U}^{-1} \hat{\Sigma} \hat{U} \end{aligned}$$

where  $\hat{U} = Q^{-\frac{k}{2}} Z^{\frac{k}{2}} U$  and hence the  $\hat{U}_{i,j} = \frac{1}{\sqrt{n}} \exp\left(\frac{2\pi i}{n} \left(jv + \frac{(j+1)(j+2)}{2} k\right)\right)$  using the same indexing as before.

As a result, we can simply use the following procedure to sample the vector  $\mathbf{x}$  in linear computational time and memory:

- 1 Choose  $k$  and  $v$ , representing the basis and the vector to select respectively, uniformly at random.
- 2 If  $k = 0$ , then we select the vector  $v$  from the computational basis, that is to say the columns of the identity matrix.
- 3 Else, let  $x_j = \frac{1}{\sqrt{n}} \exp\left(\frac{2\pi i}{n} \left(jv + \frac{(j+1)(j+2)}{2} k\right)\right)$

### 3 TRACE ESTIMATORS

In order to estimate the trace of a  $n \times n$  positive semi-definite matrix  $A$  from a single call to an oracle for  $\mathbf{x}^\dagger A \mathbf{x}$ , we consider four strategies:

- **Fixed basis estimator:** For a fixed orthonormal basis  $B$ , choose  $\mathbf{x}$  uniformly at random from the elements of  $B$ . The trace is then estimated to be  $n \mathbf{x}^\dagger A \mathbf{x}$ .

- **Mutually unbiased bases (MUBs) estimator:** For a fixed choice of a set of  $b$  mutually unbiased bases  $\mathbb{B} = \{B_1, \dots, B_b\}$ , choose  $B$  uniformly at random from  $\mathbb{B}$  and then choose  $\mathbf{x}$  uniformly at random from the elements of  $B$ . Here  $b$  is taken to be the maximum number of mutually unbiased bases for a complex vector space of dimension  $n$ . As in the fixed basis strategy, the trace is then estimated to be  $n \mathbf{x}^\dagger A \mathbf{x}$ .
- **Hutchinson's estimator:** Randomly choose the elements of  $\mathbf{x}$  independently and identically distributed from a Rademacher distribution ( $Pr(x_i = \pm 1) = \frac{1}{2}$ ). The trace is then estimated to be  $n \mathbf{x}^\dagger A \mathbf{x}$ .
- **Gaussian estimator:** Randomly choose the elements of  $\mathbf{x}$  independently and identically distributed from a zero mean unit variance Gaussian distribution. The trace is then estimated to be  $n \mathbf{x}^\dagger A \mathbf{x}$ .

The first strategy is a generic formulation of approaches which sample vectors from a fixed orthogonal basis, the most efficient sampling method in terms of the number of random bits required in the literature [13], while the second strategy is novel and represents our main contribution. Both strategies have similar randomness requirements: In the first strategy at least  $\lceil \log_2(n) \rceil$  random bits are necessary to ensure the possibility of choosing every element of  $B$ . In the second strategy, an identical number of random bits is necessary to choose  $\mathbf{x}$  for a fixed  $B$ , and  $\lceil \log_2(b) \rceil$  random bits are necessary to choose  $B$ . Note that an upper bound on the number of mutually unbiased bases is one greater than dimensionality of the space, and this bound is saturated for spaces where the dimensionality is prime or an integer power of a prime, i.e.  $b \leq n + 1$ . Thus the number of random bits necessary to implement these strategies differs by a factor of approximately two. The third and fourth strategies significantly outperform the fixed basis estimator in terms of single-shot variance, at the cost of a dramatic increase in the amount of randomness required, and have been extensively studied in the literature [13, 21, 22]. For conciseness we will not repeat the analysis of these methods in this paper but will compare the fixed basis estimator and MUBs estimator to them in Table 4.1.

#### 3.1 ANALYSIS OF FIXED BASIS ESTIMATOR

We first analyse the worst case variance of the fixed base estimator. In this analysis and the analysis for the MUBs estimator which follows, we make no assumption on  $A$  and consider the worst case variance.

We begin from the definition of the variance of the estimator for a single query. Let  $X$  be a random variable

such that  $X = \mathbf{x}^\dagger A \mathbf{x}$ , where  $\mathbf{x}$  is chosen according to the fixed basis strategy. Then

$$\text{Var}(X) = E(X^2) - E(X)^2, \quad (1)$$

where  $E(\cdot)$  denotes the expectation value of the argument. We compute this term by term. First

$$E(X) = \frac{1}{n} \sum_{\mathbf{x} \in B} \mathbf{x}^\dagger A \mathbf{x} = \frac{\text{Tr}(A)}{n},$$

where  $n = \dim A$ , and hence the second term in Eq. 1 is equal to  $\frac{\text{Tr}(A)^2}{n^2}$ . Turning to the first term,

$$\begin{aligned} E(X^2) &= \frac{1}{n} \sum_{\mathbf{x} \in B} (\mathbf{x}^\dagger A \mathbf{x})^2 \\ &= \frac{1}{n} \sum_{i=1}^n M_{ii}^2, \end{aligned}$$

where  $M = UAU^\dagger$  for some fixed unitary matrix  $U$  such that  $U^\dagger \mathbf{x}$  is a vector in the standard basis for all  $x \in B$ , and  $M_{ii}$  is the  $i$ th entry on the main diagonal of  $M$ . The variance for the fixed basis estimator is then given by  $V_{\text{fixed}} = n \sum_{i=1}^n M_{ii}^2 - \text{Tr}(A)^2$ . The worst case occurs when the value of  $\sum_{i=1}^n M_{ii}^2$  is maximized for fixed trace of  $A$  (and hence  $M$ ), and so the worst case single shot variance for the fixed basis estimator is  $V_{\text{fixed}}^{\text{worst}} = (n-1)\text{Tr}(A)^2$ .

### 3.2 ANALYSIS OF MUBS ESTIMATOR

We now turn to analysis of the MUBs estimator. We assume that  $n$  is either prime or a prime raised to some integer power. In this case, it has been established that  $b = n + 1$  [18]. The variance is defined as in Eq. 1, except that  $X$  is defined in terms of  $\mathbf{x}$  chosen according to the MUBs strategy. Again, we analyse the individual terms making up the variance. We begin with

$$E(X) = \frac{1}{nb} \sum_{B \in \mathbb{B}} \sum_{\mathbf{x} \in B} \mathbf{x}^\dagger A \mathbf{x} = \frac{\text{Tr}(A)}{n},$$

and hence the second term in the variance is the same as for the fixed basis estimator. Analysing the first term is, however, more difficult. We begin with the observation that  $E(X^2)$  can be expressed in terms of the trace of the Kronecker product of two matrices, as follows

$$\begin{aligned} E(X^2) &= \frac{1}{nb} \sum_{B \in \mathbb{B}} \sum_{\mathbf{x} \in B} (\mathbf{x}^\dagger A \mathbf{x})^2 \\ &= \frac{1}{nb} \sum_{B \in \mathbb{B}} \sum_{\mathbf{x} \in B} \text{Tr}((\mathbf{x} \mathbf{x}^\dagger A)^{\otimes 2}). \end{aligned}$$

Moving the summations inside the equation we obtain

$$\begin{aligned} E(X^2) &= \frac{1}{nb} \text{Tr} \left( \sum_{B \in \mathbb{B}} \sum_{\mathbf{x} \in B} (\mathbf{x} \mathbf{x}^\dagger)^{\otimes 2} A^{\otimes 2} \right) \\ &= \frac{2}{nb} \text{Tr}(PA^{\otimes 2}), \end{aligned} \quad (2)$$

where  $P = \frac{1}{2} \sum_{B \in \mathbb{B}} \sum_{\mathbf{x} \in B} (\mathbf{x} \mathbf{x}^\dagger)^{\otimes 2}$ .

While this form of  $P$  may appear intimidating, we now prove that  $P$  is in fact a projector with each eigenvalue being either 0 or 1. We prove this indirectly, first by showing that  $P$  has rank at most  $n(n+1)/2$ , and then using the relationship between the traces of  $P$  and  $P^2$  to conclude that the non-zero  $n(n+1)/2$  eigenvalues are equal to unity. Any vector of the form  $\mathbf{w} = \mathbf{u} \otimes \mathbf{v} - \mathbf{u} \otimes \mathbf{v}$  for  $\mathbf{u}, \mathbf{v} \in B_1$  trivially satisfies  $P\mathbf{w} = \mathbf{0}$ . Since such vectors form a basis for a subspace of dimension  $n(n-1)/2$ , we conclude that  $\text{rank}(P) \leq n^2 - n(n-1)/2 = n(n+1)/2$ . Turning now to the issue of trace, we have

$$\begin{aligned} \text{Tr}(P) &= \text{Tr} \left( \frac{1}{2} \sum_{B \in \mathbb{B}} \sum_{\mathbf{x} \in B} (\mathbf{x} \mathbf{x}^\dagger)^{\otimes 2} \right) \\ &= \frac{1}{2} \sum_{B \in \mathbb{B}} \sum_{\mathbf{x} \in B} (\mathbf{x}^\dagger \mathbf{x})^2 \\ &= \frac{nb}{2}. \end{aligned}$$

We can similarly compute the trace of  $P^2$  to obtain

$$\begin{aligned} \text{Tr}(P^2) &= \text{Tr} \left( \frac{1}{4} \sum_{B, B' \in \mathbb{B}} \sum_{\mathbf{x} \in B} \sum_{\mathbf{y} \in B'} (\mathbf{x} \mathbf{x}^\dagger)^{\otimes 2} (\mathbf{y} \mathbf{y}^\dagger)^{\otimes 2} \right) \\ &= \frac{1}{4} \sum_{B, B' \in \mathbb{B}} \sum_{\mathbf{x} \in B} \sum_{\mathbf{y} \in B'} |\mathbf{x}^\dagger \mathbf{y}|^4 \\ &= \frac{nb}{4} + \frac{n^2 b(b-1)}{4n^2} \\ &= \frac{b(n+b-1)}{4}. \end{aligned}$$

Notice that this implies that  $\text{Tr}(P) = \text{Tr}(P^2)$  for dimensions which are prime or integer powers of a prime, since in such cases  $b = n + 1$ . This implies that the eigenvalues on the non-zero subspace minimize the sum of their squares for a fixed sum, and since  $P$  is positive semi-definite, we can conclude that each non-zero eigenvalue must be equal to unity.

Returning to the calculation of variance, we then have

$$\begin{aligned} E(X^2) &\leq \frac{2}{nb} \text{Tr}(A^{\otimes 2}) \\ &= \frac{2}{nb} \text{Tr}(A)^2, \end{aligned}$$

and hence

$$\text{Var}(X) \leq \left( \frac{2}{nb} - \frac{1}{n^2} \right) \text{Tr}(M)^2 \leq \frac{\text{Tr}(A)^2}{n^2}. \quad (3)$$

This implies that the variance on the estimate of  $\text{Tr}(A)$  is bounded from above by  $\text{Tr}(A)^2$ . It is, in fact, possible to compute the variance exactly from Eq. 2 by observing

that  $M$  is the projector onto the symmetric subspace when  $n$  is an integer power of a prime. That is to say, for any vector  $\mathbf{u}$  and any vector  $\mathbf{v}$  orthogonal to  $\mathbf{u}$ , the vectors  $\mathbf{u} \otimes \mathbf{v} + \mathbf{v} \otimes \mathbf{u}$ ,  $\mathbf{u} \otimes \mathbf{u}$  and  $\mathbf{v} \otimes \mathbf{v}$  are in the  $+1$  eigenspace of  $M$ , whereas the vector  $\mathbf{u} \otimes \mathbf{v} - \mathbf{v} \otimes \mathbf{u}$  is in the null space of  $M$ . Thus we can compute the exact variance of the MUBs estimator, using the spectral decomposition  $A = \sum_i \lambda_i \mathbf{u}_i \mathbf{u}_i^\dagger$  as

$$\begin{aligned} V_{\text{MUBs}} &= \frac{2n}{n+1} \text{Tr}(PA^{\otimes 2}) - \text{Tr}(A)^2 \\ &= \frac{2n}{n+1} \sum_{i=1}^n \sum_{j=1}^n \lambda_i \lambda_j \\ &\quad \text{Tr}(P(\mathbf{u}_i \otimes \mathbf{u}_j)(\mathbf{u}_i \otimes \mathbf{u}_j)^\dagger) - \text{Tr}(A)^2 \\ &= \frac{2n}{n+1} \sum_{i=1}^n \left( \lambda_i^2 + \frac{1}{2} \sum_{j \neq i} \lambda_i \lambda_j \right) - \text{Tr}(A)^2 \\ &= \frac{n}{n+1} \text{Tr}(A^2) - \frac{1}{n+1} \text{Tr}(A)^2. \end{aligned}$$

Since for all positive semi-definite matrices  $A$  the value of  $\text{Tr}(A)^2$  is bounded from below by  $\text{Tr}(A^2)$ , the single shot variance on the MUBs estimator is bounded by  $V_{\text{MUBs}}^{\text{worst}} = \frac{n-1}{n+1} \text{Tr}(A^2)$  in the worst case, a significant improvement on the bound stemming from Eq. 3. The worst case single shot variance of the MUBs estimator is then at least a factor of  $n+1$  better than that of any fixed basis estimator. Furthermore, the variance for the widely used Hutchinson estimator [21, 13], is given by  $V_H = 2(\text{Tr}(A^2) - \sum_{i=1}^n A_{ii}^2)$ . In the worst case,  $\sum_{i=1}^n A_{ii}^2 = \frac{1}{n} \text{Tr}(A^2)$ , and hence the worst case single shot variance for Hutchinson estimator is  $V_H^{\text{worst}} = \frac{2(n-1)}{n} \text{Tr}(A^2)$ . Thus, the MUBs estimator has better worst case performance than the Hutchinson estimator by a factor  $\frac{2(n+1)}{n}$  which approaches 2 from above for large  $n$ .

## 4 RESULTS

### 4.1 THEORETICAL RESULTS

Table 1 compares the single shot variance, worst case single shot variance and randomness requirements of the trace estimators. As can be seen from the comparison the MUBs estimator has strictly smaller variance than either the Hutchinson or Gaussian methods, while requiring significantly less randomness to implement. Given the drastic reduction in randomness requirements, and the improved worst case performance, the MUBs estimator provides an attractive alternative to previous methods for estimating the trace of implicit matrices.

## 4.2 NUMERICAL RESULTS

### 4.2.1 Example Matrices

Before we demonstrate the use of the MUBs estimator on example applications we draw the readers attention to a situation where the traditional methods perform poorly. This occurs when the values of the matrix  $A$  are close to the ones matrix with a small proportion of the diagonal values much greater. The most extreme example being when this small proportion is only one element of the matrix. Due to the relationship between each of the unbiased bases this ‘spikiness’ only appears in one of the  $n+1$  bases and hence the MUBs estimator appears very robust to the condition.

It is worth noting the reason we observe an order of magnitude improvement in this setting over the competing methods. The spikes matrix described can be written as the sum of two rank one matrices. Each of these matrices will perform very poorly for the unitary estimator in that basis but gets exactly the correct result in the  $n$  other mutually unbiased bases. Naturally as  $n$  becomes large and the number of samples utilised is relatively small, then we sample the exact result with high probability.

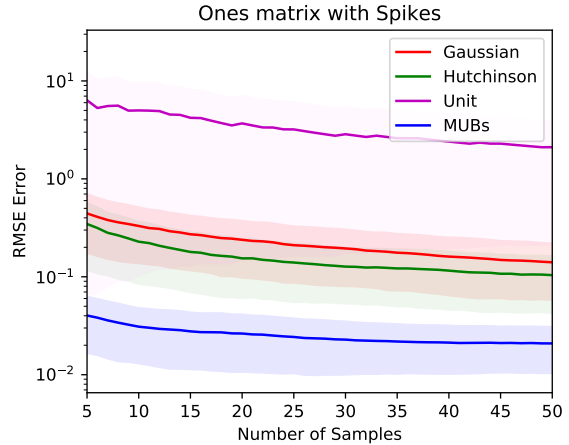


Figure 1: Convergence of the methods when estimating the trace of a  $1000 \times 1000$  ones matrix with 1 diagonal element replaced with 1001. This ‘spike’ has little effect of the convergence of the MUBs estimator and hence the method vastly out performs the others. The experiment was run 500 times and the mean and standard deviation have been plotted for each method.

We can generalise this result to low rank matrices more broadly. Any given rank- $m$  matrix can be written as the sum of  $m$  rank-1 matrices. Figure 2, demonstrates the convergence of of the stochastic trace estimators to rank-10  $1000 \times 1000$  matrices. These were created by sam-

Estimator	$V$	$V^{\text{worst}}$	$R$
Fixed basis	$n \sum_{i=1}^n M_{ii}^2 - \text{Tr}(A)^2$	$(n-1)\text{Tr}(A)^2$	$\log_2(n)$
MUBs	$\frac{n}{n+1} \text{Tr}(A^2) - \frac{1}{n+1} \text{Tr}(A)^2$	$\frac{n-1}{n+1} \text{Tr}(A^2)$	$\log_2(n) + \log_2(n+1)$
Hutchinson [21]	$2(\text{Tr}(A^2) - \sum_{i=1}^n A_{ii}^2)$	$\frac{2(n-1)}{n} \text{Tr}(A^2)$	$n$
Gaussian [22]	$2\text{Tr}(A^2)$	$2\text{Tr}(A^2)$	$\infty$ for exact; $\mathcal{O}(n)$ for fixed precision

Table 1: Comparison of single shot variance  $V$ , worst case single shot variance  $V^{\text{worst}}$  and number of random bits  $R$  required for commonly used trace estimators and the MUBs estimator.

pling 10 eigenvalues from a standard  $\chi^2$  distribution and sampling the first 10 eigenvectors of a Gaussian random matrix.

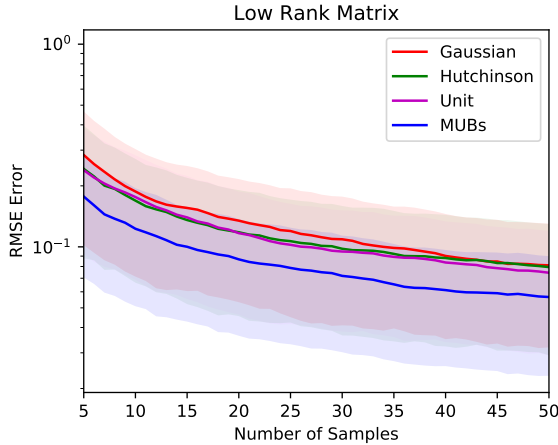


Figure 2: Convergence of the methods when estimating the trace of a  $1000 \times 1000$  rank-10 matrix. The eigenvalues were sampled from a standard  $\chi^2$ -distribution. As the rank of the matrix is only 1% of the dimensionality of the space we once again see substantially improved convergence rates. The experiment was run 30 times and the mean and standard deviation have been plotted for each method.

#### 4.2.2 Counting Triangles in Graphs

As an example application we will consider counting the number of triangles in a graph. This is an important problem in a number of application domains such as identifying the number of ‘friend of a friend’ connections in a social network which is important for friendship suggestions [23, 24], identifying spam like behaviour [25] and even identifying thematic structures in the internet [26]. An efficient method to do this is the Trace Triangle

algorithm [9]. The algorithm is based on a relationship between the adjacency matrix,  $A$ , and the number of triangles for an undirected graph,  $\Delta_g$ ,

$$\Delta_g = \frac{\text{Tr}(A^3)}{6}.$$

The trace of the adjacency matrix cubed can be sampled in  $\mathcal{O}(n^2)$  per sample as opposed to being explicitly computed in  $\mathcal{O}(n^3)$ . We compared Gaussian, Hutchinson’s, Unit and MUBs estimators performance at predicting the number of triangles for the graphs presented in Table 2 and the results of the experiment are presented in Figure 3. An efficient Python implementation for generating the MUBs sample vectors in  $\mathcal{O}(n)$ , is available at [www.github.com/OxfordMLRG/traceEst](http://www.github.com/OxfordMLRG/traceEst). The MUBs estimator outperforms each of the classical methods in all of the experiments, as would be implied by the theory.

Dataset	Vertices	Edges	Triangles
Arxiv-HEP-th	27,240	341,923	1,478,735
CA-AstroPh	18,772	198,050	1,351,441
CA-GrQc	5,242	14,484	48,260
wiki-vote	7,115	100,689	608,389

Table 2: Datasets used for the comparison of stochastic trace estimation methods in the counting of triangles in graphs. All datasets can be found at [snap.stanford.edu/data](http://snap.stanford.edu/data)

#### 4.2.3 Log Determinant of Covariance Matrix

Next, let us consider a common linear algebraic calculation required in the training of Gaussian processes, determinantal point processes and Gauss Markov random field modelling to name just a few applications, namely the log determinant of a kernel matrix.

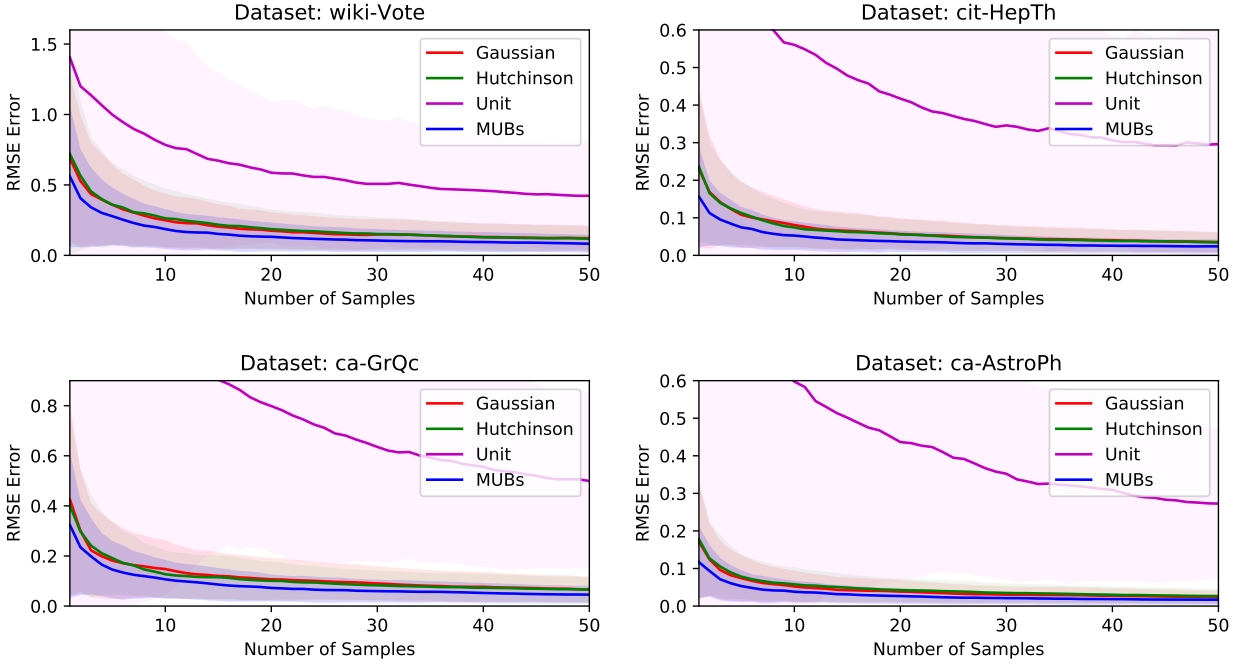


Figure 3: A comparison of the performance of the stochastic trace estimation methods on the four datasets. The experiments were performed 500 times each. The solid line indicated the empirical mean absolute relative error and the surrounding transparent region indicates one empirical standard deviation of the 500 trials.

The use of stochastic trace estimation to approximate log determinant calculations of kernel matrices has been well studied [12, 27, 28] and a range of methods are feasible. Most notably, polynomial approaches such as truncated Taylor approximations and Chebyshev approximations [12, 29] have been applied, with the latter achieving consistently better results. The general concept relies on the fact that the trace of a matrix is simply the sum of its eigenvalues and the log determinant is the sum of the log of its eigenvalues. Stochastic trace estimation aids us in approximating the sum of the eigenvalues squared, cubed and so on which we can use in a polynomial approximation of the log function,

$$\log(x) \approx \sum_{j=0}^m c_j x^j \quad \rightarrow \quad \log(|K|) \approx \sum_{j=0}^m c_j \text{Tr}(K^j)$$

where the constants  $c_j$  refer to the coefficients of the polynomial approximation. In practice, the trace of  $K^0$  is simply the dimensionality of the matrix,  $K^1$  is the trace of the explicit matrix and  $K^2$  can be found as  $\sum_{i,j} K_{i,j}^2$  due to the relationship between the matrix elements and the Frobenius norm. As such, the approximation error incurred is only due to the trace of the matrix raised to

three and above.

In order to demonstrate the effect of improved stochastic trace estimation on log determinant estimations, we sampled 1000 points from a 5-dimensional hypercube uniformly at random. These points in turn formed a covariance matrix using an isotropic Gaussian kernel function. This aimed to emulate a realistic dataset which may be used by practitioners.

We used a order-6 Chebyshev polynomial approximation and recorded estimation errors of the relative root mean squared error (RMSE) for each power of the covariance matrix. These can be seen in Figure 4. Also plotted is the estimation error of the log determinant itself, as it compounds both the polynomial approximation error and the error due to the stochastic trace estimation. A fixed budget of 25 probing vectors was allowed for each of the approaches. As can be seen in the figure, the error incurred due to the stochastic trace estimation is non-negligible and for the higher order estimates the MUBs approach was achieving improved results in terms of both its expectation and standard error.



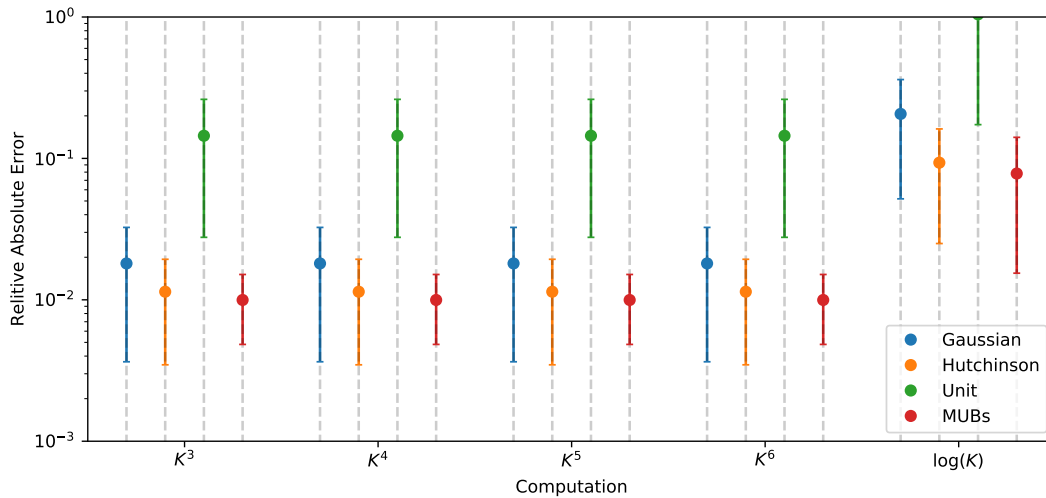


Figure 4: The performance of estimating the trace of  $K^3$ ,  $K^4$ ,  $K^5$ ,  $K^6$  and their combined result in the Chebyshev polynomial approximation of  $\log(|K|)$ . The experiment we ran 20 times and their expectation and standard error have been shown above.

## 5 CONCLUSION

We have introduced a new MUBs sampler for stochastic trace estimation which combines the efficiency of fixed basis methods with performance which outperforms the state of the art methods. We offer both empirical and theoretical comparisons to the previously established state of the art techniques and clearly demonstrate the benefit of using mutually unbiased bases for stochastic linear algebraic procedures to accelerate machine learning algorithms.

## Acknowledgements

JFF acknowledges support from the Singapore Ministry of Education and the US Air Force Office of Scientific Research under AOARD grant FA2386-15-1-4082. This material is based on research funded in part by the Singapore National Research Foundation under NRF Award NRF-NRFF2013-01.

## References

[1] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008.

[2] Zhaojun Bai, Mark Fahey, Gene H Golub, M Menon, and E Richter. Computing partial

eigenvalue sums in electronic structure calculations. Technical report, Citeseer, 1998.

[3] Tristan van Leeuwen, Aleksandr Y Aravkin, and Felix J Herrmann. Seismic waveform inversion by stochastic optimization. *International Journal of Geophysics*, 2011, 2011.

[4] Eldad Haber, Matthias Chung, and Felix Herrmann. An effective method for parameter estimation with pde constraints with multiple right-hand sides. *SIAM Journal on Optimization*, 22(3):739–757, 2012.

[5] Christos Boutsidis, Petros Drineas, Prabhanjan Kambadur, and Anastasios Zouzias. A randomized algorithm for approximating the log determinant of a symmetric positive definite matrix. *arXiv preprint arXiv:1503.00374*, 2015.

[6] Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 19(2):433–450, 1990.

[7] Mikhail J Atallah, Frédéric Chyzak, and Philippe Dumas. A randomized algorithm for approximate string matching. *Algorithmica*, 29(3):468–486, 2001.

[8] Mikhail J Atallah, Elena Grigorescu, and Yi Wu. A lower-variance randomized algorithm for approxi-

- mate string matching. *Information Processing Letters*, 113(18):690–692, 2013.
- [9] Haim Avron. Counting triangles in large graphs using randomized matrix trace estimation. In *Workshop on Large-scale Data Mining: Theory and Applications*, volume 10, pages 10–9, 2010.
- [10] Charalampos E Tsourakakis. Fast counting of triangles in large real networks without counting: Algorithms and laws. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 608–617. IEEE, 2008.
- [11] Michael L Stein, Jie Chen, Mihai Anitescu, et al. Stochastic approximation of score functions for gaussian processes. *The Annals of Applied Statistics*, 7(2):1162–1191, 2013.
- [12] Insu Han, Dmitry Malioutov, and Jinwoo Shin. Large-scale log-determinant computation through stochastic chebyshev expansions. In *International Conference on Machine Learning*, pages 908–917, 2015.
- [13] Haim Avron and Sivan Toledo. Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix. *Journal of the ACM (JACM)*, 58(2):8, 2011.
- [14] Julian Schwinger. Unitary operator bases. *Proceedings of the National Academy of Sciences*, 46(4):570–579, 1960.
- [15] Thomas Durt, Berthold-Georg Englert, Ingemar Bengtsson, and Karol Życzkowski. On mutually unbiased bases. *International journal of quantum information*, 8(04):535–640, 2010.
- [16] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.
- [17] P Oscar Boykin, Meera Sitharam, Mohamad Tarifi, and Pawel Wocjan. Real mutually unbiased bases. *arXiv preprint quant-ph/0502024*, 2005.
- [18] Andreas Klappenecker and Martin Rötteler. Constructions of mutually unbiased bases. In *Finite fields and applications*, pages 137–144. Springer, 2004.
- [19] Paul Butterley and William Hall. Numerical evidence for the maximum number of mutually unbiased bases in dimension six. *Physics Letters A*, 369(1):5–8, 2007.
- [20] Somshubhro Bandyopadhyay, P Oscar Boykin, Vwani Roychowdhury, and Farrokh Vatan. A new proof for the existence of mutually unbiased bases. *Algorithmica*, 34(4):512–528, 2002.
- [21] MF Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989.
- [22] RN Silver and H Röder. Calculation of densities of states and spectral functions by chebyshev recursion and maximum entropy. *Physical Review E*, 56(4):4822, 1997.
- [23] Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27(1):415–444, 2001.
- [24] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.
- [25] Luca Becchetti, Paolo Boldi, Carlos Castillo, and Aristides Gionis. Efficient semi-streaming algorithms for local triangle counting in massive graphs. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 16–24. ACM, 2008.
- [26] Jean-Pierre Eckmann and Elisha Moses. Curvature of co-links uncovers hidden thematic layers in the world wide web. *Proceedings of the national academy of sciences*, 99(9):5825–5829, 2002.
- [27] Jack Fitzsimons, Kurt Cutajar, Michael Osborne, Stephen Roberts, and Maurizio Filippone. Bayesian inference of log determinants. *arXiv preprint arXiv:1704.01445*, 2017.
- [28] Jack Fitzsimons, Diego Granziol, Kurt Cutajar, Michael Osborne, Maurizio Filippone, and Stephen Roberts. Entropic trace estimates for log determinants. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 323–338. Springer, 2017.
- [29] R Kelley Pace and James P LeSage. Chebyshev approximation of log-determinants of spatial weight matrices. *Computational Statistics & Data Analysis*, 45(2):179–196, 2004.

# Unsupervised Multi-view Nonlinear Graph Embedding

Jiaming Huang<sup>1,2</sup>, Zhao Li<sup>1\*</sup>, Vincent W. Zheng<sup>3</sup>, Wen Wen<sup>2</sup>, Yifan Yang<sup>1</sup>, Yuanmi Chen<sup>1</sup>

<sup>1</sup>Alibaba Group, Hangzhou, China

<sup>2</sup>School of Computers, Guangdong University of Technology, Guangzhou, China

<sup>3</sup>Advanced Digital Sciences Center, Singapore

## Abstract

In this paper, we study the unsupervised multi-view graph embedding (UMGE) problem, which aims to learn graph embedding from multiple perspectives in an unsupervised manner. However, the vast majority of multi-view learning work focuses on non-graph data, and surprisingly there are limited work on UMGE. By systematically analyzing different existing methods for UMGE, we discover that cross-view and nonlinearity play a vital role in efficiently improving graph embedding quality. Motivated by this concept, we develop an unsupervised Multi-viEW nonlinear Graph Embedding (MERGE) approach to model relational multi-view consistency. Experimental results on five benchmark datasets demonstrate that MERGE significantly outperforms the state-of-the-art baselines in terms of accuracy in node classification tasks without sacrificing the computational efficiency.

## 1 INTRODUCTION

Most of the recent work on graph embedding [25, 9, 5] focuses on the network information alone. In practice, a graph may consist of additional node features; *e.g.*, in a paper citation network, each paper node includes text content. Thus some pioneer work has started to consider the node features in graph embedding; *e.g.*, Planetoid [33] uses a semi-supervised framework to learn a network embedding for each node from its network features and a content embedding from its content features. Generally, network structure and node features are considered as two different “views” for a node in the graph. Motivated by *multi-view learning* [31, 11], we allow the

Zhao Li and Vincent W. Zheng are co-first authors.

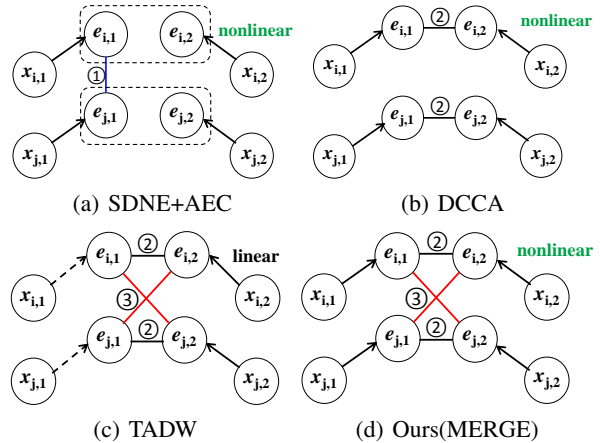


Figure 1: A systematic comparison of different methods.  $x_{i,j}$  denotes the  $i^{th}$  instance (or node)’s  $j^{th}$ -view features, and  $e_{i,j}$  is its embedding. In (c) ~ (d), we let view 1’s features be network, and view 2’s be content. ① denotes the cross-instance in single view. ② denotes the intra-instance–cross-view. ③ denotes the cross-instance–cross-view.

two views to reinforce each other, so as to obtain a better graph embedding.

In this paper, we try to solve the *Unsupervised Multi-view Graph Embedding* (UMGE) problem. Given a graph with node features, we aim to learn a network embedding and a content embedding simultaneously for each node in an unsupervised manner. Most of the multi-view learning work focuses on non-graph data, such as image and caption features in [18], acoustic and articulatory features in speech [2], and so on. Combining network and node features has been popular for graph classification [17] and graph clustering [36]. However, it has been under-explored in the literature for UMGE [2, 32, 37]. Moreover, despite the success of these prior methods, their development appears ad hoc and underlying connections are not investigated yet.

To approach the problem of UMGE, we start with systematically analyzing the underlying connections among different prior methods in Fig. 1 (two-view example). For simple discussion, let us denote  $\mathbf{x}_{i,k}$  as the features corresponding to the  $i^{\text{th}}$  instance in  $k^{\text{th}}$  view, and  $\mathbf{e}_{i,k}$  as its embedding. A naive way to extend the single view graph embedding methods into multi-view scenarios is concatenating the graph embedding (learned in network structure only) and the node features directly. For example, as shown in Fig. 1(a), SDNE [27] learns the graph embedding taking no account of the node features. By using a deep AutoEncoder [4] to model the node features and concatenating these two kinds of embeddings as node representation, SDNE can be extended to multi-view scenarios. However, simply concatenating the two views has no guarantee to reinforce each other. Typically, multi-view learning focuses on enforcing intra-instance–cross-view consistency on irrelevant data. In Fig. 1(b), DCCA [2] first embeds each instance’s multi-view features  $\mathbf{x}_{i,k}$ ’s into low-dimensional representations  $\mathbf{e}_{i,k}$ ’s, then enforces the maximal correlation between  $\mathbf{e}_{i,1}$  and  $\mathbf{e}_{i,2}$  for each  $i$ , which does not take the relational information into consideration. Some recent work exploits a different approach to model the relational information by enforcing cross-instance–cross-view consistency. In Fig. 1(c), TADW [32] recovers an adjacency matrix with the multiplication between a network embedding matrix and a text embedding matrix. Thus, two nodes with similar neighbors tend to have similar multi-view embedding.

Despite the recent advance of UMGE in network representation, there are still rooms for improvement, as current work has some limitations. First, the feature embedding in TADW is linear. However, in practice, data non-linearity is common for both network features [27] and content features [30]. Second, the network structure is under-explored. For example, DCCA does not preserve any relational proximity. However, preserving network structure is useful for graph embedding [25]. TADW preserves the second-order proximity by using matrix decomposition, which is computationally expensive. It takes a complexity of  $O(\text{nnz}(M) + |V|)$  to decompose the adjacency matrix  $M$  for a graph  $G = (V, E)$ , where  $V$  and  $E$  are the sets of nodes and edges respectively, and  $\text{nnz}(M)$  is the number of non-zero entries in  $M$ . In other words, its complexity is at least  $O(|E| + |V|)$ , since  $\text{nnz}(M) \geq |E|$ . Therefore, our goal is to efficiently solve UMGE with nonlinearity. We consider nonlinear embedding for both views in order to deal with the non-linear data nature. Besides, we also preserve the second-order proximity by enforcing cross-instance–cross-view consistency, yet with less computational cost compared against TADW.

In this paper, we propose a simple, yet effective un-

supervised Multi-view nonlinear Graph Embedding (MERGE) model. Our insights are two-fold. First, inspired by SDNE [27], MERGE encodes the nonlinearity of the network/content by taking the network/content features as input, and then applying a deep AutoEncoder [4] to learn a nonlinear network/content embedding for each node. Second, MERGE preserves the second-order proximity by extending DeepWalk [20] to the multi-view setting. On the one hand, DeepWalk preserves the second-order proximity by using one node’s network embedding to interpret its “neighbor” node’s context embedding. MERGE is easy to extend DeepWalk for using one node’s network embedding to interpret its “neighbor” node’s content embedding, so as to enforce cross-instance–cross-view consistency. Besides, MERGE also fully leverages this formulation to enforce intra-instance–cross-view consistency by using one node’s network embedding to interpret its own content embedding. On the other hand, DeepWalk employs a graph sampling approach to achieve comparable performance, which takes an  $O(|V| \log |V|)$  complexity with a hierarchical softmax objective function. MERGE further reduces the complexity to  $O(|V|)$  by using *negative sampling* [19]. We back up this argument later with a detailed complexity analysis of the MERGE algorithm in Sec. 4.

Our contributions are summarize as follows. It is the first time that different existing methods for UMGE are systematically analyzed, where cross-view and non-linearity are discovered to be critical in improving graph embedding quality, and we develop a simple, yet effective MERGE model to efficiently incorporate nonlinearity in graph embedding. MERGE is evaluated on five benchmark datasets, and it exceedingly outperforms the state-of-the-art baselines by at least relatively 1.5% ~ 17.9% (macro-F1) and 0.4% ~ 14.9% (micro-F1) over all the datasets. Moreover, the computational complexity of MERGE scales linearly with  $|V|$ , which is more favorably applied to real-world scenarios.

## 2 RELATED WORD

Graph embedding is an important task for graph analytics. Generally, it aims to learn a vector representation for each node, such that two nodes being “close” on the graph have similar vectors. Earlier work on graph embedding, such as MDS [7], LLE [23], IsoMap [26] and Laplacian eigenmap [3], typically treats to solve the leading eigenvectors of graph affinity matrices as node embedding. Recent methods explore deep learning for graph embedding. For example, LINE [25] models the node closeness by using both first-order and second-order proximity, therefore two nodes either connect di-

rectly or share the common neighbors result in similar embedding. Node2Vec [9] considers how to sample paths for DeepWalk, with awareness of node homophily and structural role. Besides, there are a number of works study the heterogeneous graph embedding [24, 6, 16]. Most existing graph embedding work focuses on single-view setting; *i.e.*, only network structure is considered. Only little work considers a multi-view setting, such as COLDA [12], TADW [32] and GCN [13]. Yet, COLDA is supervised; GCN is semi-supervised; only TADW is unsupervised.

Multi-view learning is a machine learning paradigm, which handles the data with multiple views of features in its instances [28]. Some detailed surveys about multi-view learning are available in [31, 15]. A large portion of multi-view learning literatures focus on supervised or semi-supervised settings, such as recommendation [8] and classification [10, 35]. Some other work considers an unsupervised setting, but not particularly for representation learning. For example, in [14], multi-view learning is used to improve spectral clustering; whereas in [34], the focus is to tackle the corrupted view with intra-view and inter-view noises. An overall review of recent multi-view embedding models is in [29]. Among them, a notable method is DCCA [2]. It extends CCA [1] by learning embedding for each view with a deep neural network and then enforcing the maximal correlation across two views' embedding. However, some recent multi-view embedding work, including COLDA [12], and TADW [32], has paid less attention to graph data.

### 3 UNSUPERVISED MULTI-VIEW GRAPH EMBEDDING

The UMGE is formalized as follows. Given a graph  $G = (V, E)$  as the input, each node  $v_i \in V$  has a network feature vector  $\mathbf{x}_{i,1} \in \mathbb{R}^{m_1}$  and a content feature vector  $\mathbf{x}_{i,2} \in \mathbb{R}^{m_2}$ . UMGE then generates two embedding vectors in a  $d$ -dimensional common space for each node as the output. For instance, the network features  $\mathbf{x}_{i,1}$  is extracted from  $G$ 's adjacency matrix, and the content features  $\mathbf{x}_{i,2}$  is extracted from each node's text information or attributes. They are treated as two different views to represent node  $v_i$ 's by generating the network embedding  $\mathbf{e}_{i,1} \in \mathbb{R}^d$  and the content embedding  $\mathbf{e}_{i,2} \in \mathbb{R}^d$  accordingly.

To tackle the problem, we propose MERGE, as shown in Fig. 1(d), to efficiently model the relational multi-view consistency with nonlinearity. MERGE enforces the cross-instance–cross-view and intra-instance–cross-view consistencies by DeepWalk style formulations with an inexpensive realization, which models the relational

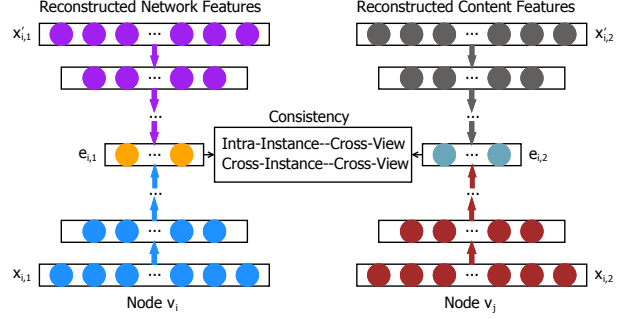


Figure 2: Illustration of the MERGE model.

information with second-order proximity and the multiple views correspondingly. Additionally, MERGE applies a deep neural network on the multi-view features and couples the nonlinear embedding outputs with the rich relational multi-view consistencies to model the feature nonlinearity. Fig. 2 summarizes the three major components of the proposed MERGE model:

- A direct formulation of intra-instance–cross-view and cross-instance–cross-view consistencies between node  $v_i$ 's network embedding  $\mathbf{e}_{i,1}$  and its “neighbor” node  $v_j$ 's content embedding  $\mathbf{e}_{j,2}$  (by setting  $i$  as a special “neighbor” node of itself, this component handles the intra-instance–cross-view as well);
- A nonlinear formulation of learning a network embedding  $\mathbf{e}_{i,1}$  from node  $v_i$ 's network features  $\mathbf{x}_{i,1}$ ;
- A nonlinear formulation of learning a content embedding  $\mathbf{e}_{j,2}$  from node  $v_j$ 's content features  $\mathbf{x}_{j,2}$ .

By integrating all these components together, MERGE learns the embedding  $\mathbf{e}_{i,1}$  and  $\mathbf{e}_{j,2}$  from  $G$ . In the following sections, we introduce the methodology to model these consistencies.

**Cross-instance–cross-view consistency for second-order proximity.** Formally, we define this consistency between one node  $v_i$ 's view-1 embedding  $\mathbf{e}_{i,1}$  and another node  $v_j$ 's view-2 embedding  $\mathbf{e}_{j,2}$  as a probability  $p(\mathbf{e}_{j,2}|\mathbf{e}_{i,1})$ . To enforce this cross-instance–cross-view consistency between  $\mathbf{e}_{i,1}$  and  $\mathbf{e}_{j,2}$ , MERGE maximizes  $p(\mathbf{e}_{j,2}|\mathbf{e}_{i,1})$ . Inspired by DeepWalk [20], which preserves second-order proximity and uses graph sampling for efficient realization, MERGE defines  $p(\mathbf{e}_{j,2}|\mathbf{e}_{i,1})$  with a softmax function over  $\mathbf{e}_{i,1}$  and  $\mathbf{e}_{j,2}$ :

$$p(\mathbf{e}_{j,2}|\mathbf{e}_{i,1}) = \frac{\exp(\mathbf{e}_{j,2}^\top \mathbf{e}_{i,1})}{\sum_{v_k \in V} \exp(\mathbf{e}_{k,2}^\top \mathbf{e}_{i,1})}. \quad (1)$$

Following the similar intuition as DeepWalk, MERGE uses node  $v_i$ 's network embedding  $\mathbf{e}_{i,1}$  to interpret its "neighbor"  $v_j$ 's content embedding  $\mathbf{e}_{j,2}$ . Different from DeepWalk, Eq. 1 does not contain any "context embedding" as additional parameters. Instead, MERGE uses one view's embedding to interpret the other view's embedding across the nodes. If two nodes have similar neighbors, the interpretations to the other view of these neighbors tend to be similar. Therefore MERGE still favors the second-order proximity with Eq. 1. Generally, it is possible to symmetrically define  $p(\mathbf{e}_{i,1}|\mathbf{e}_{j,2}) \propto \exp(\mathbf{e}_{i,1}^\top \mathbf{e}_{j,2})$ . Since  $\exp(\mathbf{e}_{i,1}^\top \mathbf{e}_{j,2}) = \exp(\mathbf{e}_{j,2}^\top \mathbf{e}_{i,1})$ , introducing  $p(\mathbf{e}_{i,1}|\mathbf{e}_{j,2})$  tends to achieve a similar effect as Eq. 1. For simplicity, only  $p(\mathbf{e}_{j,2}|\mathbf{e}_{i,1})$  is used to represent cross-instance–cross-view consistency in this paper.

Additionally, MERGE uses graph sampling to enforce  $p(\mathbf{e}_{j,2}|\mathbf{e}_{i,1})$  on a size-controllable set of  $(v_i, v_j)$ 's to reduce computation cost. On the one hand,  $v_i$  and  $v_j$  are supposed to be "close" in  $G$ ; otherwise it is not desirable to require their network embedding and content embedding to be consistent. We exploit a broad definition of "closeness" for  $v_i$  and  $v_j$  as [20, 9]; *i.e.*, a node  $v_j$  is the neighbor of  $v_i$  if  $v_j$  appears in a context window of  $v_i$  on a randomly sampled path from  $G$ . On the other hand, MERGE controls the number of "neighbors" of  $v_i$  by an adjustable amount of sampled paths from  $G$ . Usually, the number of sampled paths is set as a constant to avoid computing all the edges in  $G$ , which leads the complexity of TADW to be  $O(|E|)$ .

**Intra-instance–cross-view consistency for multiple views.** Formally, MERGE defines this consistency between node  $v_i$ 's view-1 embedding  $\mathbf{e}_{i,1}$  and its view-2 embedding  $\mathbf{e}_{i,2}$  as a probability  $p(\mathbf{e}_{i,2}|\mathbf{e}_{i,1})$ . To enforce this intra-instance–cross-view consistency between  $\mathbf{e}_{i,1}$  and  $\mathbf{e}_{i,2}$ , MERGE maximizes the likelihood  $\prod_i p(\mathbf{e}_{i,2}|\mathbf{e}_{i,1})$ . Considering the similar intuition as Eq. 1 that utilizes a given node's network embedding to interpret its content embedding, MERGE defines

$$p(\mathbf{e}_{i,2}|\mathbf{e}_{i,1}) = \frac{\exp(\mathbf{e}_{i,2}^\top \mathbf{e}_{i,1})}{\sum_{v_k \in V} \exp(\mathbf{e}_{k,2}^\top \mathbf{e}_{i,1})}. \quad (2)$$

In this regard, we simplify the objective function with a unified formulation about the cross-view consistency.

**Nonlinearity for feature embedding.** To handle the nonlinearity of each view's features, we propose to learn each  $v_i$ 's embedding  $\mathbf{e}_{i,k}$  as a nonlinear function of its corresponding feature vector  $\mathbf{x}_{i,k}$ . Specifically, we introduce AutoEncoder [4], a deep neural network, to enable this nonlinear embedding. For simplicity, we use a three-layer architecture as an example to explain how AutoEncoder works. For any input vector  $\mathbf{z} \in \mathbb{R}^d$ , we denote

$\tanh(\mathbf{z})$  to return a vector value of the hyperbolic tanh function over each dimension of  $\mathbf{z}$ . For any view  $k$ , we model each  $\mathbf{e}_{i,k}$  as

$$\mathbf{e}_{i,k} = \tanh(W_1^{(k)} \mathbf{x}_{i,k} + \mathbf{b}_1^{(k)}), \quad (3)$$

where  $W_1^{(k)} \in \mathbb{R}^{d \times |V|}$ ,  $\mathbf{x}_{i,k} \in \mathbb{R}^{m_k}$  and  $\mathbf{b}_1^{(k)} \in \mathbb{R}^d$  are parameters. Here  $\mathbf{x}_{i,k}$  is assumed to be reconstructible from  $\mathbf{e}_{i,k}$  by a nonlinear function:

$$\mathbf{x}'_{i,k} = \tanh((W_2^{(k)})^\top \mathbf{e}_{i,k} + \mathbf{b}_2^{(k)}), \quad (4)$$

where  $W_2^{(k)} \in \mathbb{R}^{d \times m_k}$  and  $\mathbf{b}_2^{(k)} \in \mathbb{R}^{m_k}$  are parameters. Finally, MERGE minimizes the  $\ell_2$ -distance between  $\mathbf{x}_{i,k}$  and  $\mathbf{x}'_{i,k}$  over all possible  $(i, k)$  pairs.

## 4 LEARNING WITH MERGE

In this section, we derive the overall objective function to compute the graph embedding. First of all, we consider the two cross-view consistencies in Eq. 1 and Eq. 2. Ideally, to enforce the two cross-view consistencies, we can directly maximize both  $p(\mathbf{e}_{j,2}|\mathbf{e}_{i,1})$  and  $p(\mathbf{e}_{i,2}|\mathbf{e}_{i,1})$  for all the  $i$ 's and their neighboring  $j$ 's. However, since we need to sum up all the nodes in  $V$ , it is time consuming to compute the normalization terms in Eq. 1 and Eq. 2. DeepWalk exploits a hierarchical approximation to the softmax function in Eq. 1 and Eq. 2, which reduces the overall complexity to  $O(|V| \log |V|)$ . MERGE applies negative sampling [19] to replace the softmax function to further reduce the complexity. Specifically, instead of maximizing  $\log p(\mathbf{e}_{j,2}|\mathbf{e}_{i,1}) + \log p(\mathbf{e}_{i,2}|\mathbf{e}_{i,1})$ , MERGE introduces a surrogate function:

$$\begin{aligned} \Delta_{i,j} &= \log \sigma(\mathbf{e}_{j,2}^\top \mathbf{e}_{i,1}) \\ &\quad + \sum_{t=1}^{\ell_1} \mathbb{E}_{v_h \sim P_n(v_i)} [\log \sigma(-\mathbf{e}_{h,2}^\top \mathbf{e}_{i,1})]. \end{aligned} \quad (5)$$

Here  $\sigma(x) = 1/(1 + \exp(-x))$  is the sigmoid function;  $v_h \sim P_n(v_i)$  indicates sampling a node  $v_h$  as a *negative context* of node  $v_i$  from a distribution  $P_n(v_i)$ . We follow [19] to define  $P_n(v_i) \propto \text{deg}(v_i)^{3/4}$ , where  $\text{deg}(v_i)$  is  $v_i$ 's degree. Besides,  $\mathbb{E}_{v_h \sim P_n(v_i)}[\cdot]$  is the expectation over  $P_n(v_i)$ . Denote the positive context set of node  $v_i$  as  $C_i$ , where  $C_i \subset V$ . Then, we define the objective function by enforcing the two cross-view consistencies as

$$O_1 = -\frac{1}{n} \sum_{v_i \in V} \sum_{v_j \in C_i \cup v_i} \Delta_{i,j}, \quad (6)$$

where  $n = \sum_{i: v_i \in V} |C_i \cup v_i|$  is the number of  $\Delta_{i,j}$ 's. By minimizing  $O_1$ , we enforce the two cross-view consistencies with second-order proximity. This  $O_1$  formulation Eq. 6 also leads to reduce the complexity to  $O(|V|)$ , as we will see later.

Next, we consider the nonlinear embedding for two

---

**Algorithm 1** Learning Algorithm for MERGE

---

**Require:** graph  $G = (V, E)$  with node features  $\mathbf{x}_{i,k}$ 's, context window size  $\ell_0$ , # of negative samples  $\ell_1$ , sample path length  $\tau$ , # of paths per node  $\ell_3$ .

**Ensure:** embedding  $\mathbf{e}_{i,k}$ 's, parameters  $\Theta$ .

- 1: Initialize the parameters in  $\mathcal{M}$ ;
  - 2: **while** not converged **do**
  - 3: Paths  $P \leftarrow \text{SamplePath}(G, \ell_0, \ell_1, \tau, \ell_3)$ ;
  - 4: Batches  $\{(v_i, v_j, \lambda)\}, B \leftarrow \text{ConstructBatch}(P)$ ;
  - 5: **for** each batch  $\{(v_i, v_j, \lambda)\}$  **do**
  - 6:  $\nabla_{\mathcal{M}} \leftarrow \frac{\partial \mathcal{L}}{\partial \mathcal{M}}$  by AdaDelta;
  - 7:  $\mathcal{M} \leftarrow \text{GradDescent}(\nabla_{\mathcal{M}}, \mathcal{M})$ ;
  - 8: **end for**
  - 9: **end while**
- 

views. Specifically, we learn the nonlinear network embedding and nonlinear content embedding by minimizing:

$$\begin{cases} O_2 = \sum_{v_i \in V} \|\mathbf{x}'_{i,1} - \mathbf{x}_{i,1}\|^2 \\ O_3 = \sum_{v_i \in V} \|\mathbf{x}'_{i,2} - \mathbf{x}_{i,2}\|^2. \end{cases} \quad (7)$$

**Objective function.** MERGE combines  $O_1$ ,  $O_2$  and  $O_3$  together to learn  $\mathbf{e}_{i,k}$ 's. Denote our model  $\mathcal{M}$  as  $\{\mathbf{e}_{i,k}, \Theta | i = 1, \dots, |V|, k = 1, 2\}$ , which parameters are listed in  $\Theta = \{W_1^{(k)}, W_2^{(k)}, \mathbf{b}_1^{(k)}, \mathbf{b}_2^{(k)}\}$ . The overall objective function to minimize is

$$\mathcal{L}(\mathcal{M}) = O_1 + \alpha O_2 + \beta O_3 + \gamma \Omega(\mathcal{M}), \quad (8)$$

where  $\alpha, \beta, \gamma$  are a set of positive trade-off parameters.  $\Omega(\mathcal{M})$  is a regularization term over  $\mathcal{M}$ ; *e.g.*, it sums up the  $\ell_2$ -norm of each parameter in  $\mathcal{M}$ .

**Learning algorithm.** To optimize  $\mathcal{L}$  in Eq. 8, we use batch stochastic gradient descent algorithm. The detailed derivations of the parameter gradients are complex, especially when the layers of deep AutoEncoder increases. Due to space limit, we skip the gradient details in this paper. Empirically, these gradients can be automatically computed in the state-of-the-art deep learning toolkits. Since all the  $O_1$ ,  $O_2$  and  $O_3$  are larger than zero,  $\mathcal{L}$  is bounded, the minimization will converge in the end. Empirically, the MERGE algorithm converges quickly, as shown later in Sec. 5.

We summarize learning algorithm for MERGE in Alg. 1. In line 1, we initialize all the parameters  $\mathbf{e}_{i,k}$ 's and  $\Theta$ . In line 3, we sample a set of paths from  $G$ , in order to identify the context neighbors for each node  $v_i$ . In particular, starting from each node  $v \in V$ , we sample  $\ell_3$  paths by random walk. Each path's length is  $\tau$ . As we use nega-

tive sampling for  $O_1$ , each node  $v_i$  has both positive context and negative context. Thus, in line 4, we construct  $B$  batches, each of which has a set of tuples  $\{(v_i, v_j, \lambda)\}$  from the sampled paths  $P$ . Here,  $v_j$  is  $v_i$ 's positive context when  $\lambda = 1$ , negative when  $\lambda = -1$ . Empirically, we find the batch size range from 100 to 1000 working well. In lines 5 ~ 8, we compute the batch gradient for each parameter in  $\mathcal{M}$  and do gradient descent.

**Algorithm complexity.** We analyze the complexity of Alg. 1. Parameter initialization in line 1 takes  $O(|V|)$ . Path sampling in line 3 takes  $O(|V|\tau\ell_3)$ . In line 4, the following steps are implemented:

1. extracting all the  $v_i$  and its context neighbor pairs, which takes  $O(|V|\tau\ell_3)$ ;
2. sampling  $\ell_4$  negative context nodes for  $v_i$ , which takes  $O(\ell_4)$ ;
3. sampling  $B$  batches, each of size  $\ell_4$ , which takes  $O(B\ell_4)$ .

In line 6, we compute the gradient of  $\mathcal{L}$  w.r.t. each batch. For  $O_1$ , it takes  $O(1)$  for a single node. For  $O_2$ , as each network feature vector has the length of  $O(|V|)$ , it takes  $O(|V|)$  to compute the gradient. For  $O_3$ , it takes  $O(1)$  for a single node. For  $\Omega$ , it takes  $O(|V|)$  since the network embedding has  $O(|V|)$  dimensions. In total, line 6 takes  $O(|V|\ell_4)$ . Line 7 takes  $O(|V|)$  to update the parameters. The complexity of Alg. 1 is  $O(|V|) = O(|V| + |V|\tau\ell_3 + \ell_4 + B\ell_4 + B(|V|\ell_4 + |V|))$ , which is much smaller than the TADW's  $O(|E| + |V|)$ .

## 5 EXPERIMENTS

In this section, we demonstrate the effectiveness of MERGE by comparing it against the current state-of-the-art baselines. MERGE shows a superior advantage over all other models in extensive experiments. In addition, we present a comprehensive analysis on MERGE in terms of its parameter sensitivity, algorithm convergence and computational complexity.

### 5.1 Experimental Setup

To quantitatively evaluate the quality of our node embedding, we follow [20, 5] to use node classification as the evaluation task. We feed the embedding output from each method into a logistic regression model for node classification. We use *micro-F1* (*i.e.*, the overall F1 w.r.t. all the classes) and *macro-F1* (*i.e.*, the average F1 w.r.t. each class) as the evaluation metrics. For each dataset, we randomly sample 50% of nodes as the test set  $D_{test}$ ,

Table 1: Benchmark datasets.

Dataset	#(node)	#(edge)	#(class)
<b>Cora</b>	2708	5429	7
<b>PubMed</b>	19717	44338	3
<b>CiteSeer</b>	3264	4591	6
<b>Wikipedia</b>	2405	17981	19
<b>MicroblogPCU</b>	781	3315	2

30% as the training set  $D_{trn}$  and the remaining as the development set  $D_{dev}$ . We first train the model on  $D_{trn}$  and tune the hyperparameters with  $D_{dev}$ . Then we fix the hyperparameters, re-train our model on  $D_{trn} \cup D_{dev}$  and test it on  $D_{tst}$ . We repeat this procedure for 10 times and report the average results. All experiments are implemented on a Linux server with 64GB memory, 24 Intel Xeon CPUs (2.40GHz) and 1 Intel Tesla K80 GPU.

**Datasets.** Five different benchmark datasets are used for evaluation:

1. Three paper citation networks<sup>1</sup> as used in [33], including *Cora*, *PubMed* and *CiteSeer*. In these networks, nodes are papers, edges are citations and classes are paper categories.
2. One web document network<sup>2</sup>, *i.e.*, *Wikipedia*, as used in [32]. In this network, nodes are Wikipedia articles, edges are hyperlinks and classes are web page categories.
3. One microblog user network, *i.e.*, *MicroblogPCU*<sup>3</sup>. In this network, nodes are users, edges are follow-ship and classes indicate spam users.

The network statistics of these datasets are summarized in Table 1. Next we introduce the node features of each dataset. For *Cora* and *Citeseer*, each node contains a pre-processed bag-of-words text feature vector. For *PubMed* and *Wikipedia*, each node corresponds a preprocessed *tf-idf* text feature vector. For *MicroblogPCU*, each node contains a raw document. As the vocabulary size is large and the text is noisy, we follow TADW [32] to apply PCA for dimensionality reduction on the node-by-text feature matrix of each dataset. Then we keep  $m_2$  dimensions corresponding to the largest singular values, which retain 90% of the *energy* [21]. Thus each node has an  $m_2$ -dimensional content feature vector. Similarly, we also apply PCA to reduce each dataset’s node adjacency matrix to  $m_1$ -dimensional network feature vector.

**Baselines.** To give a clear illustration on the contribution of cross-view consistency and model nonlinearity

<sup>1</sup><http://linqs.umiacs.umd.edu/projects/projects/lbc>

<sup>2</sup><http://linqs.cs.umd.edu/projects/projects/lbc/index.html>

<sup>3</sup><https://archive.ics.uci.edu/ml/datasets/microblogPCU>

respectively, we classify our baselines into several categories:

1. Models which only use content information, such as *AEC*: AutoEncoder of Content applies AutoEncoder [4] to the content features.
2. Models which only explore relational information, such as *DeepWalk* [20], *Node2Vec* [9], *SDNE* [27] and *struc2vec* [22].
3. Models which naively concatenate a content feature vector and a network embedding vector. These approaches do not model the cross-view consistency, which include: *DeepWalk+AEC*, *Node2Vec+AEC*, *SDNE+AEC*, *struc2vec+AEC* and *AEG+AEC*, where AutoEncoder of Graph (AEG) is a method that applies AutoEncoder to the network features to learn a network embedding for each node.
4. *TADW* [32]: it learns both network embedding and content embedding by matrix decomposition, which only models the linearity.
5. *DCCA* [2]: it learns a network embedding and a content embedding for each node by applying a deep neural network to the corresponding features and enforcing the embedding to be maximally correlated. It does not consider relational information.

**Parameters.** MERGE has a similar set of hyperparameters as DeepWalk, which includes the context window size  $\ell_0$ , the number of negative samples  $\ell_1$ , the sample path length  $\tau$ , the number of paths per node  $\ell_3$  and the embedding dimension  $d$ . Typically, we set  $\ell_0 = 4$ ,  $\ell_1 = 9$ ,  $\tau = 10$ ,  $\ell_3 = 10$  and  $d = 128$  for all of the datasets. An exception is in *PubMed*, which is larger than the other datasets: we follow the suggestion in [19] to set a larger dimension  $d = 256$  and a smaller number of negative samples  $\ell_1 = 2$ . In addition, we have trade-off hyperparameters  $\{\alpha, \beta, \gamma\}$ . We use grid search over  $\{1e-6, 1e-5, 1e-4, 1e-3, 1e-2\}$  to tune each of them (results to be shown later). Typically, we set  $\alpha = 1e-4$ ,  $\beta = 1e-3$ ,  $\gamma = 1e-5$  for all the datasets.

For each baseline’s hyperparameters, we take their suggested values as in references and fine tune them with the development set on each dataset. For DeepWalk and Node2Vec, we set  $\ell_0 = 10$ ,  $\tau = 10$ ,  $\ell_3 = 80$ ,  $d = 128$ . For Node2Vec, we optimize  $p$  and  $q$  with a grid search over  $p, q \in 0.25, 0.50, 1, 2, 4$  as suggested by [9] on each dataset’s development set. For TADW, we use the optimal parameter values reported by [32] for *Cora*, *CiteSeer* and *Wikipedia*; besides, we set its hyperparameter  $k =$



200 for *PubMed* and  $k = 100$  for *MicroblogPCU*. For DCCA, we set its hyperparameters  $\alpha = 1e-4$  and  $\beta = 1e-4$ . For SDNE, we optimize the hyperparameters of  $\alpha \in \{1, 10, 100, 300, 500, 700\}$ ,  $\beta \in \{1, 10, 30, 50, 100\}$ , and  $\gamma \in \{1e-4, 1e-3, 1e-2, 1e-1, 1, 10\}$  on the validation set. For struc2vec, we fine tune the hyperparameters of  $\ell_0 \in (0, 10)$ ,  $\tau \in \{10, 30, 50, 70, 90\}$  and  $\ell_3 \in \{1, 5, 10, 15\}$  on the validation set. Besides, we set the maximum layer used in struc2vec as six and enable all of the optimization options in the paper. Note that it is time consuming to get the comparable results when the optimization options are disabled.

## 5.2 Model Performances

We report the results in Table 2. Note that all the improvements discussed below are calculated in relative values. For each dataset, we average the improvements over all the training ratios due to space limit. We draw conclusions as follows.

- Jointly considering both content and network information for graph embedding is critical, as indicated by the uncompetitive performance of AEC, DeepWalk, Node2Vec, SDNE, and struc2vec. In contrast, the naive method of concatenating content-based feature with network embedding vector already improves the overall performance. DeepWalk + AEC improves by 1% ~ 28% over DeepWalk alone, Node2Vec + AEC improves by 1% ~ 24%, SDNE + AEC improves by 1% ~ 60% and struc2vec improves by 5% ~ 320% in terms of macro-F1. Similarly, AEG+AEC improves AEC by 1% ~ 24%. However, directly concatenating two embeddings does not exploit the cross-view relations among data, and thus cannot guarantee the performance improvements. For example, AEC outperforms both DeepWalk+AEC and Node2Vec+AEC in PubMed and CiteSeer.
- Utilization of either cross-instance consistency (DeepWalk+AEC, node2vec+AEC, AEG+AEC, SDNE+AEC, struc2vec+AEC) or cross-view consistency (DCCA) alone is not sufficient to model the relations in data. Particularly, by enforcing cross-instance-cross-view consistency, MERGE outperforms cross-instance-only baselines by at least 2% ~ 17% (macro-F1) and 2% ~ 9% (micro-F1) on average, and it improves DCCA by averagely 4% ~ 20% (macro-F1) and 4% ~ 17% (micro-F1). This again proves that maximizing the utility of both views for graph embedding, we need to carefully leverage the interdependency between the two views rather than simply concatenating them.

- Modeling data nonlinearity in graph embedding gives rise to an enhancement of performances. It is shown that MERGE is better than TADW by 6.87% ~ 20% (macro-F1) and 4% ~ 16% (micro-F1) on average.
- By comprehensively modeling cross-instance-cross-view consistency with nonlinearity, MERGE consistently and significantly outperforms all the baselines on all the five datasets with different training ratios. Specifically, in most datasets training with only 30% of labeled nodes, MERGE outperforms the baselines which are trained with 100% of the training data(micro-F1) except Wikipedia and MicroblogPCU. On average, MERGE improves the best baselines by 2.1% ~ 10.3% (macro-F1) and 2.3% ~ 9.3% (micro-F1) across all the datasets. The advantage of MERGE is more favorable especially with less training data. For example, in Cora, MERGE outperforms the baselines at least by 17.8% when using 10% training data and 6.2% when using 100% training data in terms of macro-F1.

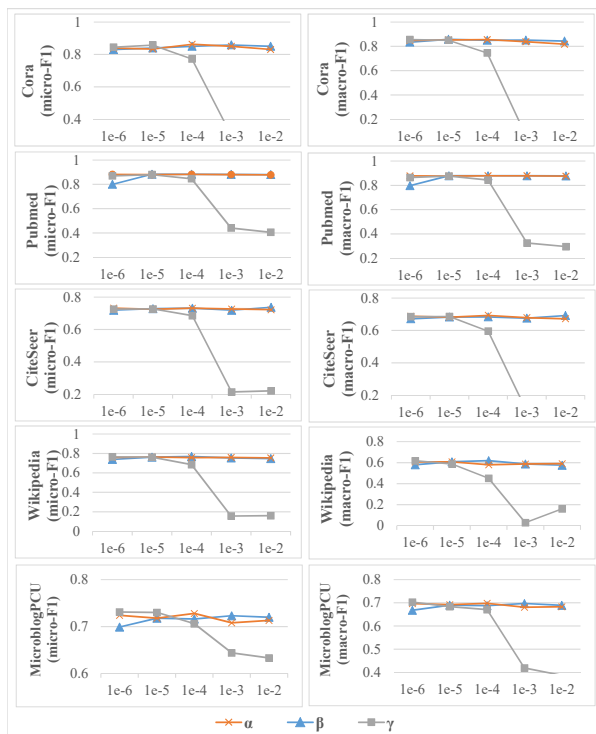


Figure 3: Results on parameter sensitivity.

## 5.3 Model Analysis

We validate the parameter sensitivity for MERGE by tuning our model parameters  $\alpha$ ,  $\beta$  and  $\gamma$ . As shown in

Table 2: Comparison with the baselines under different amounts of training labels. The bigger F1 value is better.

		macro-F1 (%)										micro-F1 (%)									
		% Labels	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	10%	20%	30%	40%	50%	60%	70%	80%	90%
Cora	AEC	27.8	43.7	52.1	54.8	59.4	62.1	64.0	65.2	66.0	67.0	44.5	54.9	60.1	62.8	65.6	67.2	68.9	69.5	70.3	70.9
	DeepWalk	61.8	68.1	71.2	72.8	73.9	74.8	75.8	76.2	76.7	77.1	64.2	69.7	72.5	74.3	75.0	76.1	76.9	77.3	77.6	78.1
	Node2Vec	63.1	70.2	72.8	74.5	76.0	76.5	77.4	77.9	78.1	78.3	65.7	72.0	74.4	76.1	77.5	77.9	78.9	79.3	79.5	79.7
	SDNE	60.1	67.7	70.3	71.4	72.4	73.9	74.6	75.1	75.7	76.2	62.9	69.4	71.6	72.7	73.7	74.9	75.6	76.1	76.6	77.0
	struc2vec	21.0	23.2	24.6	26.1	27.4	28.2	28.5	29.6	29.9	30.6	26.2	27.7	29.2	30.9	32.3	33.3	34.0	35.2	35.7	36.7
	DeepWalk+AEC	62.5	68.8	72.4	74.2	75.9	76.8	78.4	78.9	79.8	80.3	64.8	70.6	73.7	75.7	77.2	78.2	79.6	80.2	80.9	81.5
	Node2Vec+AEC	63.6	70.7	73.7	75.5	77.2	78.0	79.0	79.6	80.2	80.9	66.3	72.6	75.3	77.2	78.7	79.5	80.5	81.2	81.6	82.3
	AEG+AEC	37.9	57.6	66.3	69.0	72.8	75.2	77.0	77.9	79.0	79.7	51.0	64.0	69.8	72.8	75.5	77.4	79.1	79.7	80.6	81.1
	SDNE+AEC	66.3	71.7	73.7	75.2	76.0	77.1	77.9	78.2	79.0	79.7	69.8	73.8	75.5	76.9	77.7	78.6	79.5	79.9	80.6	81.1
	struc2vec+AEC	45.5	51.5	53.7	55.2	57.1	58.4	59.7	60.8	61.7	62.6	51.2	56.0	57.9	59.0	60.7	61.9	63.2	64.1	65.1	65.8
	TADW	59.3	68.3	70.2	70.7	71.6	72.0	72.4	72.4	73.1	73.1	65.5	71.1	72.8	73.4	74.1	74.3	74.8	74.9	75.3	75.5
	DCCA	58.8	66.4	68.6	69.7	71.0	71.6	72.8	72.9	73.3	73.5	64.0	69.4	71.4	72.5	73.5	73.9	75.0	75.3	75.5	75.7
	MERGE	78.1	82.1	83.3	83.9	84.1	84.8	85.3	85.4	85.8	85.9	80.2	83.4	84.6	85.0	85.3	85.9	86.5	86.6	86.9	86.9
	PubMed	AEC	82.3	83.6	84.3	84.7	85.0	85.0	85.2	85.3	85.5	85.5	82.2	83.6	84.2	84.6	84.9	85.0	85.2	85.3	85.4
DeepWalk		64.7	65.5	66.3	67.0	67.0	67.2	67.2	67.3	67.4	67.5	67.8	69.5	70.4	71.2	71.3	71.6	71.6	71.7	71.9	72.0
Node2Vec		66.7	67.7	68.3	68.7	68.9	69.1	69.1	69.4	69.5	69.6	69.5	71.2	72.0	72.5	72.9	73.0	73.1	73.4	73.5	73.7
SDNE		59.4	61.2	61.9	62.3	62.4	62.5	62.5	62.5	62.6	62.7	62.2	64.2	64.9	65.2	65.4	65.5	65.5	65.5	65.6	65.7
struc2vec		39.5	41.4	42.2	42.6	43.1	43.4	43.4	43.7	43.8	43.7	42.2	44.9	46.0	46.7	47.3	47.7	47.8	48.2	48.4	48.4
DeepWalk+AEC		78.7	83.4	84.9	85.7	86.3	86.5	86.7	86.9	87.1	87.2	79.2	83.7	85.2	86.0	86.6	86.7	86.9	87.2	87.3	87.4
Node2Vec+AEC		79.0	83.7	85.1	85.9	86.4	86.6	86.9	87.0	87.1	87.2	79.6	83.9	85.4	86.1	86.6	86.8	87.0	87.2	87.3	87.4
AEG+AEC		83.2	84.7	85.4	85.9	86.3	86.3	86.5	86.7	86.8	86.9	83.2	84.8	85.5	86.0	86.4	86.4	86.7	86.8	87.0	87.1
SDNE+AEC		78.0	80.7	83.1	84.2	85.0	85.4	85.7	85.9	86.0	86.2	78.1	80.8	83.2	84.2	84.9	85.3	85.7	85.9	86.0	86.2
struc2vec+AEC		75.9	76.8	79.4	80.9	82.2	83.2	83.8	84.2	84.6	84.7	75.6	76.8	79.3	80.9	82.2	83.1	83.7	84.1	84.5	84.6
TADW		67.8	76.2	79.2	80.3	81.2	81.7	82.1	82.4	82.7	82.9	71.5	76.9	79.5	80.4	81.3	81.7	82.1	82.4	82.7	82.8
DCCA		82.9	83.9	84.2	84.5	84.6	84.7	84.8	84.8	84.7	84.8	83.1	84.1	84.4	84.7	84.8	84.8	84.9	85.0	84.9	85.0
MERGE		86.2	87.2	87.6	87.8	88.1	88.1	88.3	88.3	88.4	88.5	86.6	87.5	87.9	88.1	88.4	88.4	88.6	88.6	88.7	88.8
CiteSeer		AEC	45.4	55.2	57.7	58.8	59.5	60.2	60.1	61.0	61.3	61.6	55.3	64.3	66.8	67.7	68.3	69.0	69.1	69.4	69.7
	DeepWalk	39.7	43.3	45.9	47.3	48.4	49.5	49.6	50.0	50.2	50.7	43.4	46.6	49.6	51.1	52.5	53.7	54.0	54.5	54.9	55.5
	Node2Vec	40.4	45.5	48.3	50.5	52.0	52.6	52.9	53.3	54.2	53.8	44.1	49.1	52.3	54.5	56.3	56.9	57.3	57.9	58.6	58.5
	SDNE	33.6	37.2	38.9	39.7	40.3	41.0	42.0	42.3	42.6	42.8	37.8	41.6	43.1	44.4	45.2	45.8	46.7	47.0	47.2	47.6
	struc2vec	19.6	20.5	21.4	22.4	23.0	23.5	23.4	24.1	24.5	24.8	21.4	22.3	23.3	24.6	25.3	25.9	26.1	27.0	27.4	28.0
	DeepWalk+AEC	40.9	45.8	49.7	53.0	55.3	57.6	59.1	60.4	61.4	62.2	44.6	49.4	53.8	57.2	60.0	62.3	64.1	65.2	66.4	67.3
	Node2Vec+AEC	41.3	47.5	51.9	55.5	57.6	59.4	60.4	61.9	63.2	63.7	45.1	51.3	56.1	59.6	62.3	64.0	65.2	66.6	67.8	68.4
	AEG+AEC	49.0	57.8	60.1	61.1	62.2	62.8	62.6	63.6	64.0	64.4	58.6	66.7	69.1	70.0	70.7	71.1	71.4	71.6	71.9	72.2
	SDNE+AEC	53.7	56.9	59.1	60.6	61.2	62.0	63.4	63.8	64.3	64.6	57.4	61.1	63.5	65.0	66.1	66.7	68.0	68.5	69.1	69.5
	struc2vec+AEC	48.8	51.7	51.6	53.0	54.0	54.8	56.6	57.6	58.2	59.4	53.1	55.6	55.7	57.3	58.4	59.2	60.9	62.1	62.7	64.0
	TADW	58.8	62.0	62.8	63.4	64.1	64.8	64.7	65.5	65.6	65.7	65.5	67.5	68.4	68.7	69.5	69.9	70.1	70.5	70.7	70.8
	DCCA	50.3	55.2	57.3	58.2	58.9	59.8	60.2	60.6	61.1	61.2	55.3	60.2	62.2	63.2	64.0	64.6	65.3	65.5	65.9	66.2
	MERGE	62.3	65.7	66.4	68.0	68.7	69.6	69.3	70.1	70.5	70.6	69.5	71.9	72.8	73.7	74.2	74.8	75.1	75.1	75.4	75.5
	Wikipedia	AEC	32.5	40.3	46.5	49.6	50.8	53.6	55.6	56.4	58.4	59.4	51.7	62.1	67.1	69.8	70.5	72.5	73.2	74.1	74.9
DeepWalk		36.4	40.9	44.3	46.1	47.2	49.2	50.1	50.9	52.1	52.3	49.3	54.6	57.1	59.5	60.7	61.7	62.6	63.4	64.2	64.4
Node2Vec		37.3	42.4	44.6	46.8	48.2	49.4	51.6	51.9	52.9	53.4	50.6	54.8	58.0	59.5	61.0	61.9	63.1	63.3	64.0	64.5
SDNE		31.8	37.1	39.3	42.0	43.6	45.4	46.1	46.9	48.2	48.7	41.4	48.1	51.1	53.4	55.4	56.7	57.3	58.4	59.2	59.8
struc2vec		7.8	8.6	9.5	10.2	10.3	10.6	11.1	11.1	11.5	11.5	12.7	13.5	14.4	15.1	15.0	15.3	16.0	16.3	16.3	16.5
DeepWalk+AEC		37.1	42.3	46.5	49.0	51.1	53.1	55.7	56.5	58.9	59.2	50.0	56.1	60.1	62.7	65.2	66.4	67.8	69.8	71.3	72.1
Node2Vec+AEC		37.9	43.7	46.5	49.6	51.7	53.8	56.3	57.9	59.6	60.2	51.4	56.3	60.4	62.9	65.3	66.6	68.5	70.0	71.3	72.1
AEG+AEC		37.9	45.3	50.5	54.3	55.2	57.9	59.3	60.0	61.7	62.4	56.1	65.3	69.8	72.5	73.5	75.1	75.9	76.8	77.7	78.2
SDNE+AEC		33.4	41.9	46.6	49.6	51.6	54.0	55.1	57.3	59.3	59.4	46.4	56.0	61.0	64.4	66.8	68.7	69.8	71.3	72.4	73.2
struc2vec+AEC		11.2	18.9	27.0	32.4	36.4	40.0	42.6	45.0	46.9	48.4	19.0	29.8	40.3	47.7	52.4	56.6	59.7	62.1	64.3	65.8
TADW		46.7	53.2	56.0	57.0	60.8	60.3	62.3	63.4	62.7	64.2	59.5	64.3	67.4	67.8	69.7	70.0	70.0	71.4	71.2	71.9
DCCA		43.5	48.6	50.4	53.1	54.2	54.7	56.4	57.0	57.0	57.4	59.4	64.4	66.7	68.4	69.8	70.3	71.0	71.6	71.8	72.3
MERGE		50.4	57.9	59.4	62.1	63.7	64.4	66.9	67.7	68.2	69.2	68.2	73.0	75.0	76.6	77.4	77.9	78.9	79.4	79.7	80.1
microblogPCU		AEC	39.3	44.4	48.4	52.0	52.4	52.8	58.2	59.6	59.2	60.2	63.2	64.4	66.0	67.4	67.0	67.8	69.4	70.1	69.8
	DeepWalk	60.3	60.8	58.5	59.1	63.1	66.6	64.9	66.1	64.1	65.6	63.9	64.9	63.3	63.0	66.0	70.1	68.8	69.9	68.1	69.7
	Node2Vec	58.5	59.4	56.9	57.9	60.8	62.9	62.9	65.0	63.6	63.4	62.3	63.8	62.6	62.3	64.4	67.4	68.0	70.0	68.4	68.3
	SDNE	53.5	53.9	53.2	58.6	59.1	58.2	57.1	56.9	58.3	61.8	55.6	58.8	57.7	61.3	64.9	63.6	61.3	63.4	63.0	65.6
	struc2vec	47.4	55.6	57.1	58.3	56.5	58.1	56.7	58.1	57.8	60.0	50.3	57.7	58.7	60.6	59.1	60.8	59.2	60.6	60.2	62.6
	DeepWalk+AEC	60.4	61.0	58.3	59.6	64.0	67.5	65.1	67.1	65.9	67.3	64.2	65.3	64.0	63.6	66.7	70.8	68.6	70.8	69.7	70.9
	Node2Vec+AEC	57.4	58.3	57.4	58.2	61.7	63.7	64.1	65.5	65.1	64.7	61.9	63.6	63.4	62.8	65.1	68.1	68.9	70.3	69.8	69.4
	AEG+AEC	43.9	47.8	51.0	57.8	57.0	62.4	64.4	67.0	65.6	67.4	63.8	65.1	66.7	68.7	67.9	71.6	71.8	74.3	72.9	73.8
	SDNE+AEC	54.0	58.3	61.1	64.6	64.8	66.9	67.7	68.4	69.1	69.3	56.4	60.0	6							

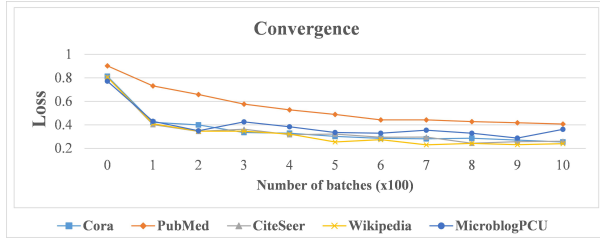


Figure 4: Results on convergence.

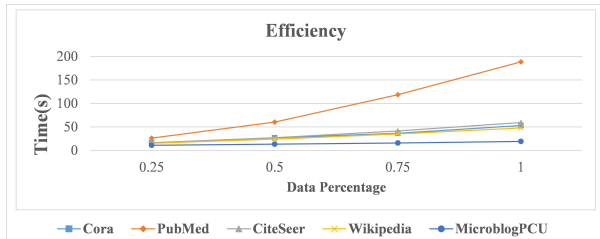


Figure 5: Results on running time.

Fig. 3, MERGE performs optimally when  $\alpha$  and  $\beta \in \{1e-5, 1e-4, 1e-3, 1e-2\}$  and  $\gamma \in \{1e-6, 1e-5\}$ . Detailed parameter settings are described in Sec. 5.1. Specifically, our model is relatively robust to the values of  $\alpha$  and  $\beta$ . While taking large value for  $\gamma$  causes over-regularization of the model, which leads to poor performance. The suggested  $\gamma$  value is less than  $1e-4$ . We further validate the convergence and time complexity of MERGE. As shown in Fig. 4, the loss of the model normalized by the number of nodes converges quickly within a few iterations. In terms of asymptotic computation complexity, we test the algorithm complexity with different amounts of nodes (*i.e.*, different  $|V|$  values). As shown in Fig. 5, it is observed that the running time of MERGE scales linearly with  $|V|$ .

## 6 CONCLUSION

In this paper, we systematically investigate the problem of UMGE, where multiple perspectives such as content and network features are exploited to learn graph embedding in an unsupervised fashion. We discover that cross-view and nonlinearity play a critical role in improving graph embedding quality. Therefore, we develop a simple, yet effective approach, Multi-viEW non-linear Graph Embedding (MERGE), to model relational multi-view consistency with nonlinearity. MERGE incorporates both network embedding and content embedding with a nonlinear AutoEncoder. Then, it enforces both intra-instance-cross-view consistency and cross-instance-cross-view consistency among nodes' network embedding with its neighbors' content embedding. The

experimental results demonstrate that MERGE consistently outperforms the state-of-the-art baselines by at least relatively 2.12%  $\sim$  10.91% (macro-F1) and 2.29%  $\sim$  10.12% (micro-F1) on average over five public datasets, yet with  $O(|V|)$  complexity. It is worthy to note that, in most cases, MERGE outperforms the baselines in favor of using relatively one third of training data yet with superior performance.

## References

- [1] ANDERSON, T. W. *An Introduction to Multivariate Statistical Analysis*, second ed. Wiley, New York, NY, 1984.
- [2] ANDREW, G., ARORA, R., BILMES, J. A., AND LIVESCU, K. Deep canonical correlation analysis. In *ICML (2013)*, pp. 1247–1255.
- [3] BELKIN, M., AND NIYOGI, P. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS (2001)*, pp. 585–591.
- [4] BENGIO, Y. Learning deep architectures for AI. *Foundations and Trends in Machine Learning* 2, 1 (2009), 1–127.
- [5] CAO, S., LU, W., AND XU, Q. Grarep: Learning graph representations with global structural information. In *CIKM (2015)*, pp. 891–900.
- [6] CHANG, S., HAN, W., TANG, J., QI, G., AGGARWAL, C. C., AND HUANG, T. S. Heterogeneous network embedding via deep architectures. In *KDD (2015)*, pp. 119–128.
- [7] COX, T. F., AND COX, M. *Multidimensional Scaling, Second Edition*, 2 ed. Chapman and Hall/CRC, 2000.
- [8] ELKAHKY, A. M., SONG, Y., AND HE, X. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *WWW (2015)*, pp. 278–288.
- [9] GROVER, A., AND LESKOVEC, J. node2vec: Scalable feature learning for networks. In *KDD (2016)*.
- [10] HE, J., DU, C., ZHUANG, F., YIN, X., HE, Q., AND LONG, G. Online bayesian max-margin subspace multi-view learning. In *IJCAI (2016)*, pp. 1555–1561.
- [11] HOU, C., NIE, F., TAO, H., AND YI, D. Multi-view unsupervised feature selection with adaptive similarity and view weight. *IEEE Transactions on Knowledge and Data Engineering* 29, 9 (2017), 1998–2011.

- [12] HUANG, Z., SHAN, S., ZHANG, H., LAO, S., AND CHEN, X. Cross-view graph embedding. In *ACCV* (2013), pp. 770–781.
- [13] KIPF, T. N., AND WELLING, M. Semi-supervised classification with graph convolutional networks.
- [14] LI, Y., NIE, F., HUANG, H., AND HUANG, J. Large-scale multi-view spectral clustering via bipartite graph. In *AAAI* (2015), pp. 2750–2756.
- [15] LI, Y., YANG, M., AND ZHANG, Z. Multi-view representation learning: A survey from shallow methods to deep methods. *CoRR abs/1610.01206* (2016).
- [16] LIU, Z., ZHENG, V. W., ZHAO, Z., AND LI, Z. Interactive paths embedding for semantic proximity search on heterogeneous graphs. In *KDD* (2018).
- [17] MACSKASSY, S. A., AND PROVOST, F. J. Classification in networked data: A toolkit and a univariate case study. *Journal of Machine Learning Research* 8 (2007), 935–983.
- [18] MANSIMOV, E., PARISOTTO, E., BA, L. J., AND SALAKHUTDINOV, R. Generating images from captions with attention. In *ICLR* (2016).
- [19] MIKOLOV, T., SUTSKEVER, I., CHEN, K., CORRADO, G. S., AND DEAN, J. Distributed representations of words and phrases and their compositionality. In *NIPS* (2013), pp. 3111–3119.
- [20] PEROZZI, B., AL-RFOU, R., AND SKIENA, S. Deepwalk: Online learning of social representations. In *KDD* (2014), pp. 701–710.
- [21] RAJARAMAN, A., AND ULLMAN, J. D. *Mining of Massive Datasets*. Cambridge University Press, New York, NY, USA, 2011.
- [22] RIBEIRO, L. F., SAVERESE, P. H., AND FIGUEIREDO, D. R. struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2017), ACM, pp. 385–394.
- [23] ROWEIS, S. T., AND SAUL, L. K. Nonlinear dimensionality reduction by locally linear embedding. *Science* 290, 5500 (2000), 2323–2326.
- [24] TANG, J., QU, M., AND MEI, Q. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *KDD* (2015), pp. 1165–1174.
- [25] TANG, J., QU, M., WANG, M., ZHANG, M., YAN, J., AND MEI, Q. Line: Large-scale information network embedding. In *WWW* (2015), pp. 1067–1077.
- [26] TENENBAUM, J. B., DE SILVA, V., AND LANGFORD, J. C. A global geometric framework for nonlinear dimensionality reduction. *Science* 290, 5500 (2000), 2319–2323.
- [27] WANG, D., CUI, P., AND ZHU, W. Structural deep network embedding. In *KDD* (2016), pp. 1225–1234.
- [28] WANG, S., DING, Z., AND FU, Y. Coupled marginalized auto-encoders for cross-domain multi-view learning. In *IJCAI* (2016), pp. 2125–2131.
- [29] WANG, W., ARORA, R., LIVESCU, K., AND BILMES, J. A. On deep multi-view representation learning. In *ICML* (2015), pp. 1083–1092.
- [30] XIE, R., LIU, Z., JIA, J., LUAN, H., AND SUN, M. Representation learning of knowledge graphs with entity descriptions. In *AAAI* (2016), pp. 2659–2665.
- [31] XU, C., TAO, D., AND XU, C. A survey on multi-view learning. *CoRR abs/1304.5634* (2013).
- [32] YANG, C., LIU, Z., ZHAO, D., SUN, M., AND CHANG, E. Y. Network representation learning with rich text information. In *IJCAI* (2015), pp. 2111–2117.
- [33] YANG, Z., COHEN, W. W., AND SALAKHUTDINOV, R. Revisiting semi-supervised learning with graph embeddings. In *ICML* (2016), pp. 40–48.
- [34] ZHANG, L., WANG, S., ZHANG, X., WANG, Y., LI, B., SHEN, D., AND JI, S. Collaborative multi-view denoising. In *KDD* (2016), pp. 2045–2054.
- [35] ZHOU, D., AND BURGESS, C. J. C. Spectral clustering and transductive learning with multiple views. In *ICML* (2007), pp. 1159–1166.
- [36] ZHOU, Y., CHENG, H., AND YU, J. X. Graph clustering based on structural/attribute similarities. *PVLDB* 2, 1 (2009), 718–729.
- [37] ZHUGE, W., NIE, F., HOU, C., AND YI, D. Unsupervised single and multiple views feature extraction with structured graph. *IEEE Transactions on Knowledge and Data Engineering* 29, 10 (2017), 2347–2359.

---

# Graph-based Clustering under Differential Privacy

---

Rafael Pinot<sup>\*1,2</sup>, Anne Morvan<sup>\*†1,2</sup>, Florian Yger<sup>1</sup>, Cédric Gouy-Pailler<sup>2</sup>, and Jamal Atif<sup>1</sup>

<sup>1</sup>Université Paris-Dauphine, PSL Research University, CNRS, 75016 Paris, France

<sup>2</sup>CEA, LIST, 91191 Gif-sur-Yvette, France

## Abstract

In this paper, we present the first differentially private clustering method for arbitrary-shaped node clusters in a graph. This algorithm takes as input only an approximate Minimum Spanning Tree (MST)  $\mathcal{T}$  released under weight differential privacy constraints from the graph. Then, the underlying nonconvex clustering partition is successfully recovered from cutting optimal cuts on  $\mathcal{T}$ . As opposed to existing methods, our algorithm is theoretically well-motivated. Experiments support our theoretical findings.

## 1 INTRODUCTION

Weighted graph data is known to be a useful representation data type in many fields, such as bioinformatics or analysis of social, computer and information networks. More generally, a graph can always be built based on the data dissimilarity where points of the dataset are the vertices and weighted edges express “distances” between those objects. For both cases, graph clustering is one of the key tools for understanding the underlying structure in the graph (Schaeffer, 2007). These clusters can be seen as groups of nodes close in terms of some specific similarity.

Nevertheless, it is critical that the data representation used in machine learning applications protects the private characteristics contained into it. Let us consider an application where one wants to identify groups of similar web pages in the sense of traffic volume *i.e.* web pages with similar audience. In that case, the nodes stand for the websites. The link between two vertices represents

the fact that some people consult them both. In such a framework, the web browsing history of an individual is used to set the edge weights. We consider that this history can be a very sensitive information for the user, since he/she could have visited sensible content web pages. Treating such datasets as non-private could lead to leaking information such as his/her political, sexual, or religious preferences. As a standard for data privacy preservation, differential privacy (Dwork et al., 2006b) has been designed: an algorithm is differentially private if, given two close databases, it produces statistically indistinguishable outputs. Since then, its definition has been extended to weighted graphs. Though, machine learning applications ensuring data privacy remain rare, in particular for clustering which encounters severe theoretical and practical limitations. Indeed, some clustering methods lack of theoretical support and most of them restrict the data distribution to convex-shaped clusters (Nissim et al., 2007; Blum et al., 2008; McSherry, 2009; Dwork, 2011) or unstructured data (Ho and Ruan, 2013; Chen et al., 2015). Hence, the aim of this paper is to offer a theoretically motivated private graph clustering. Moreover, to the best of our knowledge, this is the first weight differentially-private clustering algorithm able to detect clusters with an arbitrary shape for weighted graph data.

Our method belongs to the family of Minimum Spanning Tree (MST)-based approaches. An MST represents a useful summary of the graph, and appears to be a natural object to describe it at a lower cost. For clustering purposes, it has the appealing property to help retrieving non-convex shapes (Zahn, 1971; Asano et al., 1988; Grygorash et al., 2006; Morvan et al., 2017). Moreover, they appear to be well-suited for incorporating privacy constraints as will be formally proved in this work.

**Contributions:** Our contributions are threefold: 1) we provide the first theoretical justifications of MST-based clustering algorithms. 2) We endow DBMSTCLU algorithm (Morvan et al., 2017), an MST-based clustering algorithm from the literature, with theoretical guarantees.

---

\*Rafael Pinot and Anne Morvan contributed equally.

†Partly supported by the *Direction Générale de l'Armement* (French Ministry of Defense).

3) We introduce a differentially-private version of DBM-STCLU and give several results on its privacy/utility tradeoff.

## 2 PRELIMINARIES

### 2.1 NOTATIONS

Let  $\mathcal{G} = (V, E, w)$  be a simple undirected weighted graph with a vertex set  $V$ , an edge set  $E$ , and a weight function  $w := E \rightarrow \mathbb{R}$ . One will respectively call the edge set and the node set of a graph  $\mathcal{G}$  using the applications  $E(\mathcal{G})$  and  $V(\mathcal{G})$ . Given a node set  $S \subset V$ , one denotes by  $\mathcal{G}_{|S}$  the subgraph induced by  $S$ . We call  $G = (V, E)$  the topology of the graph, and  $\mathcal{W}_E$  denotes the set of all possible weight functions mapping  $E$  to weights in  $\mathbb{R}$ . For the remaining of this work, cursive letter are used to represent weighted graphs and straight letters refer to topological arguments. Since graphs are simple, the path  $\mathcal{P}_{u-v}$  between two vertices  $u$  and  $v$  is characterized either as the ordered sequence of vertices  $\{u, \dots, v\}$  or corresponding binding edges depending on the context. We also denote  $V_{\mathcal{P}_{u-v}}$  the unordered set of such vertices. Besides, edges  $e_{ij}$  denote an edge between nodes  $i$  and  $j$ . Finally, for all positive integer  $K$ ,  $[K] := \{1, \dots, K\}$ .

### 2.2 DIFFERENTIAL PRIVACY IN GRAPHS

As opposed to node-differential privacy (Kasiviswanathan et al., 2013) and edge-differential privacy (Hay et al., 2009), both based on the graph topology, the privacy framework considered here is weight-differential privacy where the graph topology  $G = (V, E)$  is assumed to be public and the private information to protect is the weight function  $w := E \rightarrow \mathbb{R}$ . Under this model introduced by Sealfon (2016), two graphs are said to be neighbors if they have the same topology, and *close* weight functions. This framework allows one to release an almost minimum spanning tree with weight-approximation error of  $O(|V| \log |E|)$  for fixed privacy parameters. Differential privacy is ensured in that case by using the Laplace mechanism on every edges weight to release a spanning tree based on a perturbed version of the weight function. The privacy of the spanning tree construction is thus provided by post-processing (cf. Th. 2.5). However, under a similar privacy setting, Pinot (2018) recently manages to produce the topology of a tree under differential privacy without relying on the post-processing of a more general mechanism such as the ‘‘Laplace mechanism’’. Their algorithm, called PAMST, privately releases the topology of an almost minimum spanning tree thanks to an iterative use of the ‘‘Exponential mechanism’’ instead.

For fixed privacy parameters, the weight approximation error is  $O\left(\frac{|V|^2}{|E|} \log |V|\right)$ , which outperforms the former method from Sealfon (2016) on arbitrary weighted graphs under weak assumptions on the graph sparseness. Thus, we keep here privacy setting from Pinot (2018).

**Definition 2.1** (Pinot (2018)). *For any edge set  $E$ , two weight functions  $w, w' \in \mathcal{W}_E$  are neighboring, denoted  $w \sim w'$ , if  $\|w - w'\|_\infty := \max_{e \in E} |w(e) - w'(e)| \leq \mu$ .*

$\mu$  represents the sensitivity of the weight function and should be chosen according to the application and the range of this function. The neighborhood between such graphs is clarified in the following definition.

**Definition 2.2.** *Let  $\mathcal{G} = (V, E, w)$  and  $\mathcal{G}' = (V', E', w')$ , two weighted graphs,  $\mathcal{G}$  and  $\mathcal{G}'$  are said to be neighbors if  $V = V'$ ,  $E = E'$  and  $w \sim w'$ .*

The so-called weight-differential privacy for graph algorithms is now formally defined.

**Definition 2.3** (Sealfon (2016)). *For any graph topology  $G = (V, E)$ , let  $\mathcal{A}$  be a randomized algorithm that takes as input a weight function  $w \in \mathcal{W}_E$ .  $\mathcal{A}$  is called  $(\epsilon, \delta)$ -differentially private on  $G = (V, E)$  if for all pairs of neighboring weight functions  $w, w' \in \mathcal{W}_E$ , and for all set of possible outputs  $S$ , one has*

$$\mathbb{P}[\mathcal{A}(w) \in S] \leq e^\epsilon \mathbb{P}[\mathcal{A}(w') \in S] + \delta.$$

*If  $\mathcal{A}$  is  $(\epsilon, \delta)$ -differentially private on every graph topology in a class  $\mathcal{C}$ , it is said to be  $(\epsilon, \delta)$ -differentially private on  $\mathcal{C}$ .*

One of the first, and most used differentially private mechanisms is the Laplace mechanism. It is based on the process of releasing a numerical query perturbed by a noise drawn from a centered Laplace distribution scaled to the sensitivity of the query. We present here its graph-based reformulation.

**Definition 2.4** (reformulation Dwork et al. (2006b)). *Given some graph topology  $G = (V, E)$ , for any  $f_G : \mathcal{W}_E \rightarrow \mathbb{R}^k$ , the sensitivity of the function is defined as  $\Delta f_G = \max_{w \sim w' \in \mathcal{W}_E} \|f_G(w) - f_G(w')\|_1$ .*

**Definition 2.5** (reformulation Dwork et al. (2006b)). *Given some graph topology  $G = (V, E)$ , any function  $f_G : \mathcal{W}_E \rightarrow \mathbb{R}^k$ , any  $\epsilon > 0$ , and  $w \in \mathcal{W}_E$ , the graph-based Laplace mechanism is  $\mathcal{M}_L(w, f_G, \epsilon) = f_G(w) + (Y_1, \dots, Y_k)$  where  $Y_i$  are i.i.d. random variables drawn from  $\text{Lap}(\Delta f_G / \epsilon)$ , and  $\text{Lap}(b)$  denotes the Laplace distribution with scale  $b$  (i.e probability density  $\frac{1}{2b} \exp\left(-\frac{|x|}{b}\right)$ ).*

**Theorem 2.1** (Dwork et al. (2006b)). *The Laplace mechanism is  $\epsilon$ -differentially private.*

We define hereafter the graph-based Exponential mechanism. In the sequel we refer to it simply as Exponential mechanism. The Exponential mechanism represents a way of privately answering arbitrary range queries. Given some range of possible responses to the query  $\mathcal{R}$ , it is defined according to a utility function  $u_G := \mathcal{W}_E \times \mathcal{R} \rightarrow \mathbb{R}$ , which aims at providing some total preorder on the range  $\mathcal{R}$  according to the total order in  $\mathbb{R}$ . The sensitivity of this function is denoted  $\Delta u_G := \max_{r \in \mathcal{R}} \max_{w \sim w' \in \mathcal{W}_E} |u_G(w, r) - u_G(w', r)|$ .

**Definition 2.6.** *Given some graph topology  $G = (V, E)$ , some output range  $\mathcal{R} \subset E$ , some privacy parameter  $\epsilon > 0$ , some utility function  $u_G := \mathcal{W}_E \times \mathcal{R} \rightarrow \mathbb{R}$ , and some  $w \in \mathcal{W}_E$  the graph-based Exponential mechanism  $\mathcal{M}_{Exp}(G, w, u_G, \mathcal{R}, \epsilon)$  selects and outputs an element  $r \in \mathcal{R}$  with probability proportional to  $\exp\left(\frac{\epsilon u_G(w, r)}{2\Delta u_G}\right)$ .*

The Exponential mechanism defines a distribution on a potentially complex and large range  $\mathcal{R}$ . As the following theorem states, sampling from such a distribution preserves  $\epsilon$ -differential privacy.

**Theorem 2.2** (reformulation McSherry and Talwar (2007)). *For any non-empty range  $\mathcal{R}$ , given some graph topology  $G = (V, E)$ , the graph-based Exponential mechanism preserves  $\epsilon$ -differential privacy, i.e if  $w \sim w' \in \mathcal{W}_E$ ,*

$$\begin{aligned} \mathbb{P}[\mathcal{M}_{Exp}(G, w, u_G, \mathcal{R}, \epsilon) = r] \\ \leq e^\epsilon \mathbb{P}[\mathcal{M}_{Exp}(G, w', u_G, \mathcal{R}, \epsilon) = r]. \end{aligned}$$

Further, Th 2.3 highlights the trade-off between privacy and accuracy for the Exponential mechanism when  $0 < |\mathcal{R}| < +\infty$ . Th 2.4 presents the ability of differential privacy to comply with composition while Th 2.5 introduces its post-processing property.

**Theorem 2.3** (reformulation Dwork and Roth (2013)). *Given some graph topology  $G = (V, E)$ , some  $w \in \mathcal{W}_E$ , some output range  $\mathcal{R}$ , some privacy parameter  $\epsilon > 0$ , some utility function  $u_G := \mathcal{W}_E \times \mathcal{R} \rightarrow \mathbb{R}$ , and denoting  $OPT_{u_G}(w) = \max_{r \in \mathcal{R}} u_G(w, r)$ , one has  $\forall t \in \mathbb{R}$ ,*

$$\begin{aligned} u_G(G, w, \mathcal{M}_{Exp}(w, u_G, \mathcal{R}, \epsilon)) \\ \leq OPT_{u_G}(w) - \frac{2\Delta u_G}{\epsilon} (t + \ln |\mathcal{R}|) \end{aligned}$$

with probability at most  $\exp(-t)$ .

**Theorem 2.4** (Dwork et al. (2006a)). *For any  $\epsilon > 0$ ,  $\delta \geq 0$  the adaptive composition of  $k$   $(\epsilon, \delta)$ -differentially private mechanisms is  $(k\epsilon, k\delta)$ -differentially private.*

**Theorem 2.5** (Post-Processing Dwork and Roth (2013)). *Let  $\mathcal{A} : \mathcal{W}_E \rightarrow B$  be a randomized algorithm that is  $(\epsilon, \delta)$ -differentially private, and  $h : B \rightarrow B'$  a deterministic mapping. Then  $h \circ \mathcal{A}$  is  $(\epsilon, \delta)$ -differentially private.*

## 2.3 DIFFERENTIALLY-PRIVATE CLUSTERING

Differentially private clustering for unstructured datasets has been first discussed in Nissim et al. (2007). This work introduced the first method for differentially private clustering based on the k-means algorithm. Since then most of the work in the field focused on adaptation of this method (Blum et al., 2008; McSherry, 2009; Dwork, 2011). The main drawback of this work is that it is not able to deal with arbitrary shaped clusters. This issue has been recently investigated in Ho and Ruan (2013) and Chen et al. (2015). They proposed two new methods to find arbitrary shaped clusters in unstructured datasets respectively based on density clustering and wavelet decomposition. Even though both of them allow one to produce non-convex clusters, they only deal with unstructured datasets and thus are not applicable to node clustering in a graph. Our work focuses on node clustering in a graph under weight-differential privacy. Graph clustering has already been investigated in a topology-based privacy framework (Mülle et al., 2015; Nguyen et al., 2016), however, these works do not consider weight-differential privacy. Our work is, to the best of our knowledge, the first attempt to define node clustering in a graph under weight differential privacy.

## 3 DIFFERENTIALLY-PRIVATE TREE-BASED CLUSTERING

We aim at producing a private clustering method while providing bounds on the accuracy loss. Our method is an adaptation of an existing clustering algorithm DBMST-CLU. However, to provide theoretical guarantees under differential privacy, one needs to rely on the same kind of guarantees in the non-private setting. Morvan et al. (2017) did not bring them in their initial work. Hence, our second contribution is to demonstrate the accuracy of this method, first in the non-private context.

In the following we present 1) the theoretical framework motivating MST-based clustering methods, 2) accuracy guarantees of DBMSTCLU in the non-private setting, 3) PTCLUST our private clustering algorithm, 4) its accuracy under differential privacy constraints.

### 3.1 THEORETICAL FRAMEWORK FOR MST-BASED CLUSTERING METHODS

MST-based clustering methods, however efficient, lack proper motivation. This Section closes this gap by providing a theoretical framework for MST-based clustering. In the sequel, notations from Section 2.1 are kept. The minimum path distance between two nodes in the graph is defined which enables to explicit our notion of

Cluster.

**Definition 3.1** (Minimum path distance). *Let be  $\mathcal{G} = (V, E, w)$  and  $u, v \in V$ . The minimum path distance between  $u$  and  $v$  is*

$$d(u, v) = \min_{\mathcal{P}_{u-v}} \sum_{e \in \mathcal{P}_{u-v}} w(e)$$

with  $\mathcal{P}_{u-v}$  a path (edge version) from  $u$  to  $v$  in  $\mathcal{G}$ .

**Definition 3.2** (Cluster). *Let be  $\mathcal{G} = (V, E, w)$ ,  $0 < w(e) \leq 1 \forall e \in E$  a graph,  $(V, d)$  a metric space based on the minimum path distance  $d$  defined on  $\mathcal{G}$  and  $D \subset V$  a node set.  $C \subset D$  is a cluster iff.  $|C| > 2$  and  $\forall C_1, C_2$  s.t.  $C = C_1 \cup C_2$  and  $C_1 \cap C_2 = \emptyset$ , one has:*

$$\operatorname{argmin}_{z \in D \setminus C_1} \left\{ \min_{v \in C_1} d(z, v) \right\} \subset C_2$$

Assuming that a cluster is built of at least 3 points makes sense since singletons or groups of 2 nodes can be legitimately considered as noise. For simplicity of the proofs, the following theorems hold in the case where noise is neglected. However, they are still valid in the setting where noise is considered as singletons (with each singleton representing a generalized notion of cluster).

**Theorem 3.1.** *Let be  $\mathcal{G} = (V, E, w)$  a graph and  $\mathcal{T}$  a minimum spanning tree of  $\mathcal{G}$ . Let also be  $C$  a cluster in the sense of Def. 3.2 and two vertices  $v_1, v_2 \in C$ . Then,  $V_{\mathcal{P}_{v_1-v_2}} \subset C$  with  $\mathcal{P}_{v_1-v_2}$  a path from  $v_1$  to  $v_2$  in  $\mathcal{G}$ , and  $V_{\mathcal{P}_{v_1-v_2}}$  the set of vertices contained in  $\mathcal{P}_{v_1-v_2}$ .*

*Proof.* Let be  $v_1, v_2 \in C$ . If  $v_1$  and  $v_2$  are neighbors, the result is trivial. Otherwise, as  $\mathcal{T}$  is a tree, there exist a unique path within  $\mathcal{T}$  between  $v_1$  and  $v_2$  denoted by  $\mathcal{P}_{v_1-v_2} = \{v_1, \dots, v_2\}$ . Let now prove by *reductio ad absurdum* that  $V_{\mathcal{P}_{v_1-v_2}} \subset C$ . Suppose there is  $h \in V_{\mathcal{P}_{v_1-v_2}}$  s.t.  $h \notin C$ . We will see that it leads to a contradiction. We set  $C_1$  to be the largest connected component (regarding the number of vertices) of  $\mathcal{T}$  s.t.  $v_1 \in C_1$ , and every nodes from  $C_1$  are in  $C$ . Because of  $h$ 's definition,  $v_2 \notin C_1$ . Let be  $C_2 = C \setminus C_1$ .  $C_2 \neq \emptyset$  since  $v_2 \in C_2$ . Let be  $z^* \in \operatorname{argmin}_{z \in V \setminus C_1} \left\{ \min_{v \in C_1} d(z, v) \right\}$  and  $e^* = (z^*, v^*)$  an edge that reaches this minimum. Let us show that  $z^* \notin C$ . If  $z^* \in C$ , then two possibilities hold:

1. There is an edge  $e_{z^*} \in \mathcal{T}$ , s.t.  $e_{z^*} = (z^*, z')$  with  $z' \in C_1$ . This is impossible, otherwise by definition of a connected component,  $z^* \in C_1$ . Contradiction.
2. For all  $e_{z^*} = (z^*, z')$  s.t.  $z' \in C_1$ , one has  $e_{z^*} \notin \mathcal{T}$ . In particular  $e^* \notin \mathcal{T}$ . Since  $h$  is the neighbor of  $C_1$  in  $\mathcal{G}$  there is also  $e_h \in \mathcal{T}$ , s.t.  $e_h = (h, h')$  with  $h' \in C_1$ . Once again two possibilities hold:

$$(a) w(e_{z^*}) = \min_{z \in V \setminus C_1} \left\{ \min_{v \in C_1} d(z, v) \right\} < w(e_h).$$

Then, if we replace  $e_h$  by  $e_{z^*}$  in  $\mathcal{T}$ , its total weight decreases. So  $\mathcal{T}$  is not a minimum spanning tree. Contradiction.

$$(b) w(e_{z^*}) = w(e_h), \text{ therefore } h \in \operatorname{argmin}_{z \in V \setminus C_1} \left\{ \min_{v \in C_1} d(z, v) \right\}. \text{ Since } h \notin C, \text{ one gets that } \operatorname{argmin}_{z \in V \setminus C_1} \left\{ \min_{v \in C_1} d(z, v) \right\} \not\subset C_2.$$

Thus,  $C$  is not a cluster. Contradiction.

We proved that  $z^* \notin C$ . In particular,  $z^* \notin C_2$ . Then,  $\operatorname{argmin}_{z \in V \setminus C_1} \left\{ \min_{v \in C_1} d(z, v) \right\} \not\subset C_2$ . Thus,  $C$  is not a cluster. Contradiction. Finally  $h \in C$  and  $V_{\mathcal{P}_{v_1-v_2}} \subset C$ .  $\square$

This theorem states that, given a graph  $\mathcal{G}$ , an MST  $\mathcal{T}$ , and any two nodes of  $C$ , every node in the path between them is in  $C$ . This means that a cluster can be characterized by a subtree of  $\mathcal{T}$ . It justifies the use of all MST-based methods for data clustering or node clustering in a graph. All the clustering algorithms based on successively cutting edges in an MST to obtain a subtree forest are meaningful in the sense of Th.3.1. In particular, this theorem holds for the use of DBMSTCLU (Morvan et al., 2017) presented in Section 3.2.1.

## 3.2 DETERMINISTIC MST-BASED CLUSTERING

This Section introduces DBMSTCLU (Morvan et al., 2017) that will be adapted to be differentially-private, and provide accuracy results on the recovery of the ground-truth clustering partition.

### 3.2.1 DBMSTCLU algorithm

Let us consider  $\mathcal{T}$  an MST of  $\mathcal{G}$ , as the unique input of the clustering algorithm DBMSTCLU. The clustering partition results then from successive cuts on  $\mathcal{T}$  so that a new cut in  $\mathcal{T}$  splits a connected component into two new ones. Each final connected component, a subtree of  $\mathcal{T}$ , represents a cluster. Initially,  $\mathcal{T}$  is one cluster containing all nodes. Then, at each iteration, an edge is cut if some criterion, called *Validity Index of a Clustering Partition* (DBCVI) is improved. This edge is greedily chosen to locally maximize the DBCVI at each step. When no improvement on DBCVI can be further made, the algorithm stops. The DBCVI is defined as the weighted average of all *cluster validity indices* which are based on two positive quantities, the *Dispersion* and the *Separation* of a cluster:

**Definition 3.3** (Cluster Dispersion). *The Dispersion of a cluster  $C_i$  (DISP) is defined as the maximum edge*



weight of  $C_i$ . If the cluster is a singleton (i.e. contains only one node), the associated Dispersion is set to 0. More formally:

$$\forall i \in [K], \text{DISP}(C_i) = \begin{cases} \max_{j, e_j \in C_i} w_j & \text{if } |E(C_i)| \neq 0 \\ 0 & \text{otherwise.} \end{cases}$$

**Definition 3.4** (Cluster Separation). *The Separation of a cluster  $C_i$  (SEP) is defined as the minimum distance between the nodes of  $C_i$  and the ones of all other clusters  $C_j, j \neq i, 1 \leq i, j \leq K, K \neq 1$  where  $K$  is the total number of clusters. In practice, it corresponds to the minimum weight among all already cut edges from  $\mathcal{T}$  comprising a node from  $C_i$ . If  $K = 1$ , the Separation is set to 1. More formally, with  $\text{incCuts}(C_i)$  denoting cut edges incident to  $C_i$ ,*

$$\forall i \in [K], \text{SEP}(C_i) = \begin{cases} \min_{j, e_j \in \text{incCuts}(C_i)} w_j & \text{if } K \neq 1 \\ 1 & \text{otherwise.} \end{cases}$$

**Definition 3.5** (Validity Index of a Cluster). *The Validity Index of a cluster  $C_i$  is defined as:*

$$V_C(C_i) = \frac{\text{SEP}(C_i) - \text{DISP}(C_i)}{\max(\text{SEP}(C_i), \text{DISP}(C_i))} \in [-1; 1]$$

**Definition 3.6** (Validity Index of a Clustering Partition). *The Density-Based Validity Index of a Clustering partition  $\Pi = \{C_i\}, 1 \leq i \leq K$ ,  $\text{DBCVI}(\Pi)$  is defined as the weighted average of the Validity Indices of all clusters in the partition where  $N$  is the number of vertices.*

$$\text{DBCVI}(\Pi) = \sum_{i=1}^K \frac{|C_i|}{N} V_C(C_i) \in [-1, 1]$$

DBMSTCLU is summarized in Algorithm 1: `evaluateCut(.)` computes the DBCVI when the cut in parameter is applied to  $\mathcal{T}$ . Initial DBCVI is set  $-1$ . Interested reader could refer to (Morvan et al., 2017) Section 4. for a complete insight on this notions.

### 3.2.2 DBMSTClu exact clustering recovery proof

In this section, we provide theoretical guarantees for the cluster recovery accuracy of DBMSTClu. Let us first begin by introducing some definitions.

**Definition 3.7** (Cut). *Let us consider a graph  $\mathcal{G} = (V, E, w)$  with  $K$  clusters,  $\mathcal{T}$  an MST of  $\mathcal{G}$ . Let denote  $(C_i^*)_{i \in [K]}$  the set of the clusters. Then,  $\text{Cut}_{\mathcal{G}}(\mathcal{T}) := \{e_{kl} \in \mathcal{T} \mid k \in C_i^*, l \in C_j^*, i, j \in [K]^2, i \neq j\}$ . In the sequel, for simplicity, we denote  $e^{(ij)} \in \text{Cut}_{\mathcal{G}}(\mathcal{T})$  the edge between cluster  $C_i^*$  and  $C_j^*$ .*

$\text{Cut}_{\mathcal{G}}(\mathcal{T})$  is basically the set of effective cuts to perform on  $\mathcal{T}$  in order to ensure the exact recovery of the clustering partition. More generally, trees on which  $\text{Cut}_{\mathcal{G}}(\cdot)$

---

#### Algorithm 1 DBMSTCLU( $\mathcal{T}$ )

---

```

1: Input:  $\mathcal{T}$ , the MST
2:  $\text{dbcvi} \leftarrow -1.0$ 
3:  $\text{clusters} \leftarrow \emptyset$ 
4:  $\text{cut\_list} \leftarrow \{E(\mathcal{T})\}$ 
5: while  $\text{dbcvi} < 1.0$  do
6:    $\text{cut\_tp} \leftarrow \emptyset$ 
7:    $\text{dbcvi\_tp} \leftarrow \text{dbcvi}$ 
8:   for each  $\text{cut}$  in  $\text{cut\_list}$  do
9:      $\text{newDbcvi} = \text{evaluateCut}(\mathcal{T}, \text{cut})$ 
10:    if  $\text{newDbcvi} \geq \text{dbcvi\_tp}$  then
11:       $\text{cut\_tp} \leftarrow \text{cut}$ 
12:       $\text{dbcvi\_tp} \leftarrow \text{newDbcvi}$ 
13:    if  $\text{cut\_tp} \neq \emptyset$  then
14:       $\text{clusters} = \text{cut}(\text{clusters}, \text{cut\_tp})$ 
15:       $\text{dbcvi} \leftarrow \text{dbcvi\_tp}$ 
16:       $\text{cut\_list} \leftarrow \text{cut\_list} \setminus \{\text{cut\_tp}\}$ 
17:    else
18:      break
19: return  $\text{clusters}$ 

```

---

enables to find the right partition are said to be a partitioning topology.

**Definition 3.8** (Partitioning topology). *Let us consider a graph  $\mathcal{G} = (V, E, w)$  with  $K$  clusters  $C_1^*, \dots, C_K^*$ . A spanning tree  $\mathcal{T}$  of  $\mathcal{G}$  is said to have a partitioning topology if  $\forall i, j \in [K], i \neq j, |\{e = (u, v) \in \text{Cut}_{\mathcal{G}}(\mathcal{T}) \mid u \in C_i^*, v \in C_j^*\}| = 1$ .*

Def. 3.7 and 3.8 introduce a topological condition on the tree as input of the algorithm. Nevertheless, conditions on weights are necessary too. Hence, we define homogeneous separability which expresses the fact that within a cluster the edge weights are spread in a controlled manner.

**Definition 3.9** (Homogeneous separability condition). *Let us consider a graph  $\mathcal{G} = (V, E, w)$ ,  $s \in E$  and  $\mathcal{T}$  a tree of  $\mathcal{G}$ .  $\mathcal{T}$  is said to be homogeneously separable by  $s$ , if*

$$\alpha_{\mathcal{T}} \max_{e \in E(\mathcal{T})} w(e) < w(s) \text{ with } \alpha_{\mathcal{T}} = \frac{\max_{e \in E(\mathcal{T})} w(e)}{\min_{e \in E(\mathcal{T})} w(e)} \geq 1.$$

One will write for simplicity that  $H_{\mathcal{T}}(s)$  is verified.

**Definition 3.10** (Weak homogeneity condition of a Cluster). *Let us consider a graph  $\mathcal{G} = (V, E, w)$  with  $K$  clusters  $C_1^*, \dots, C_K^*$ . A given cluster  $C_i^*, i \in [K]$ ,  $C_i^*$  is weakly homogeneous if: for all  $\mathcal{T}$  an MST of  $\mathcal{G}$ , and  $\forall j \in [K], j \neq i$ , s.t.  $e^{(ij)} \in \text{Cut}_{\mathcal{G}}(\mathcal{T})$ ,  $H_{\mathcal{T}_{C_i^*}}(e^{(ij)})$  is verified. For simplicity, one denote  $\alpha_i = \max_{\mathcal{T} \text{ MST of } \mathcal{G}} \alpha_{\mathcal{T}_{C_i^*}}$*

**Definition 3.11** (Strong homogeneity condition of a Cluster). *Let us consider a graph  $\mathcal{G} = (V, E, w)$  with*

$K$  clusters  $C_1^*, \dots, C_K^*$ . A given cluster  $C_i^*$ ,  $i \in [K]$ ,  $C_i^*$  is strongly homogeneous if: for all  $\mathcal{T}$  a spanning tree (ST) of  $\mathcal{G}$ , and  $\forall j \in [K]$ ,  $j \neq i$ , s.t.  $e^{(ij)} \in \text{Cut}_{\mathcal{G}}(\mathcal{T})$ ,  $H_{\mathcal{T}|_{C_i^*}}(e^{(ij)})$  is verified. For simplicity, one denote  $\bar{\alpha}_i = \max_{\mathcal{T} \text{ ST of } \mathcal{G}} \alpha_{\mathcal{T}|_{C_i^*}}$

Weak homogeneity is indeed really natural considering the non-private cases using an MST. Strong homogeneity is more demanding, but still a reachable condition. Section 4 presents an experiment where the graph respects strong homogeneity as well as being organised in arbitrary shaped clusters. We show that the weak homogeneity condition is implied by the strong homogeneity condition.

**Proposition 3.1.** *Let us consider a graph  $\mathcal{G} = (V, E, w)$  with  $K$  clusters  $C_1^*, \dots, C_K^*$ . If a given cluster  $C_i^*$ ,  $i \in [K]$  is strongly homogeneous, then, it is weakly homogeneous.*

*Proof.* If  $\mathcal{T}$  a spanning tree of  $\mathcal{G}$ , and  $\forall j \in [K]$ ,  $j \neq i$ , s.t.  $e^{(ij)} \in \text{Cut}_{\mathcal{G}}(\mathcal{T})$ ,  $H_{\mathcal{T}|_{C_i^*}}(e^{(ij)})$  is verified, then in particular, it is true for any MST.  $\square$

Strong homogeneity condition appears to be naturally more constraining on the edge weights than the weak one. The accuracy of DBMSTCLU is proved under the weak homogeneity condition, while the accuracy of its differentially-private version is only given under the strong homogeneity condition.

**Theorem 3.2.** *Let us consider a graph  $\mathcal{G} = (V, E, w)$  with  $K$  homogeneous clusters  $C_1^*, \dots, C_K^*$  and  $\mathcal{T}$  an MST of  $\mathcal{G}$ . Let now assume that at step  $k < K - 1$ , DBMSTClu built  $k + 1$  subtrees  $\mathcal{C}_1, \dots, \mathcal{C}_{k+1}$  by cutting  $e_1, e_2, \dots, e_k \in E$ .*

*Then,  $\text{Cut}_k := \text{Cut}_{\mathcal{G}}(\mathcal{T}) \setminus \{e_1, e_2, \dots, e_k\} \neq \emptyset \implies \text{DBCVI}_{k+1} \geq \text{DBCVI}_k$ , i.e. if there are still edges in  $\text{Cut}_k$ , the algorithm will continue to perform some cut.*

*Proof.* See supplementary material.  $\square$

**Theorem 3.3.** *Let us consider a graph  $\mathcal{G} = (V, E, w)$  with  $K$  homogeneous clusters  $C_1^*, \dots, C_K^*$  and  $\mathcal{T}$  an MST of  $\mathcal{G}$ .*

*Assume now that at step  $k < K - 1$ , DBMSTClu built  $k + 1$  subtrees  $\mathcal{C}_1, \dots, \mathcal{C}_{k+1}$  by cutting  $e_1, e_2, \dots, e_k \in E$ . We still denote  $\text{Cut}_k := \text{Cut}_{\mathcal{G}}(\mathcal{T}) \setminus \{e_1, e_2, \dots, e_k\}$ .*

*If  $\text{Cut}_k \neq \emptyset$  then  $\arg\max_{e \in \mathcal{T} \setminus \{e_1, e_2, \dots, e_k\}} \text{DBCVI}_{k+1}(e) \subset \text{Cut}_k$  i.e. the cut edge at step  $k + 1$  is in  $\text{Cut}_k$ .*

*Proof.* See supplementary material.  $\square$

**Theorem 3.4.** *Let us consider a graph  $\mathcal{G} = (V, E, w)$  with  $K$  weakly homogeneous clusters  $C_1^*, \dots, C_K^*$  and  $\mathcal{T}$  an MST of  $\mathcal{G}$ . Let now assume that at step  $K - 1$ , DBMSTClu built  $K$  subtrees  $\mathcal{C}_1, \dots, \mathcal{C}_K$  by cutting  $e_1, e_2, \dots, e_{K-1} \in E$ . We still denote  $\text{Cut}_{K-1} := \text{Cut}_{\mathcal{G}}(\mathcal{T}) \setminus \{e_1, e_2, \dots, e_{K-1}\}$ .*

*Then, for all  $e \in \mathcal{T} \setminus \{e_1, e_2, \dots, e_{K-1}\}$ ,  $\text{DBCVI}_K(e) < \text{DBCVI}_{K-1}$  i.e. the algorithm stops: no edge gets cut during step  $K$ .*

*Proof.* See supplementary material.  $\square$

**Corollary 3.1.** *Let us consider a graph  $\mathcal{G} = (V, E, w)$  with  $K$  weakly homogeneous clusters  $C_1^*, \dots, C_K^*$  and  $\mathcal{T}$  an MST of  $\mathcal{G}$ . DBMSTClu( $\mathcal{T}$ ) stops after  $K - 1$  iterations and the  $K$  subtrees produced match exactly the clusters i.e. under homogeneity condition, the algorithm finds automatically the underlying clustering partition.*

*Proof.* Th. 3.2 and 3.4 ensure that under homogeneity condition on all clusters, the algorithm performs the  $K - 1$  distinct cuts within  $\text{Cut}_{\mathcal{G}}(\mathcal{T})$  and stops afterwards. By definition of  $\text{Cut}_{\mathcal{G}}(\mathcal{T})$ , it means the DBMSTClu correctly builds the  $K$  clusters.  $\square$

### 3.3 PRIVATE MST-BASED CLUSTERING

This section presents our new node clustering algorithm PTCLUST for weight differential privacy. It relies on a mixed adaptation of PAMST algorithm (Pinot, 2018) for recovering a differentially-private MST of a graph and DBMSTCLU.

#### 3.3.1 PAMST algorithm

Given a simple-undirected-weighted graph  $\mathcal{G} = (V, E, w)$ , PAMST outputs an almost minimal weight spanning tree topology under differential privacy constraints. It relies on a Prim-like MST algorithm, and an iterative use of the graph-based Exponential mechanism. PAMST takes as an input a weighted graph, and a utility function. It outputs the topology of a spanning tree which weight is almost minimal. Algorithm 2 presents this new method, using the following utility function:

$$u_{\mathcal{G}} : \mathcal{W}_E \times \mathcal{R} \rightarrow \mathbb{R} \\ (w, r) \mapsto -|w(r) - \min_{r' \in \mathcal{R}} w(r')|.$$

PAMST starts by choosing an arbitrary node to construct iteratively the tree topology. At every iteration, it uses the Exponential mechanism to find the next edge to be added to the current tree topology while keeping the weights private. This algorithm is the state of the art to find a spanning tree topology under differential privacy. For

readability, let us introduce some additional notations. Let  $S$  be a set of nodes from  $G$ , and  $\mathcal{R}_S$  the set of edges that are incident to one and only one node in  $S$  (also denoted xor-incident). For any edge  $r$  in such a set, the incident node to  $r$  that is not in  $S$  is denoted  $r_{\rightarrow}$ . Finally, the restriction of the weight function to an edge set  $\mathcal{R}$  is denoted  $w|_{\mathcal{R}}$ .

---

**Algorithm 2** PAMST( $G, u_G, w, \epsilon$ )

---

- 1: **Input:**  $\mathcal{G} = (V, E, w)$  a weighted graph (separately the topology  $G$  and the weight function  $w$ ),  $\epsilon$  a degree of privacy and  $u_G$  utility function.
  - 2: **Pick**  $v \in V$  at random
  - 3:  $S_V \leftarrow \{v\}$
  - 4:  $S_E \leftarrow \emptyset$
  - 5: **while**  $S_V \neq V$  **do**
  - 6:    $r = \mathcal{M}_{Exp}(\mathcal{G}, w, u_G, \mathcal{R}_{S_V}, \frac{\epsilon}{|V|-1})$
  - 7:    $S_V \leftarrow S_V \cup \{r_{\rightarrow}\}$
  - 8:    $S_E \leftarrow S_E \cup \{r\}$
  - 9: **return**  $S_E$
- 

Theorem 3.5 states that using PAMST to get an almost minimal spanning tree topology preserves weight-differential privacy.

**Theorem 3.5.** *Let  $G = (V, E)$  be the topology of a simple-undirected graph, then  $\forall \epsilon > 0$ , PAMST( $G, u_G, \bullet, \epsilon$ ) is  $\epsilon$ -differentially private on  $G$ .*

### 3.3.2 Differentially-private clustering

The overall goal of this Section is to show that one can obtain a differentially-private clustering algorithm by combining PAMST and DBMSTCLU algorithms. However, PAMST does not output a weighted tree which is inappropriate for clustering purposes. To overcome this, one could rely on a sanitizing mechanism such as the Laplace mechanism. Moreover, since DBMSTCLU only takes weights from  $(0,1]$ , two normalizing parameters  $\tau$  and  $p$  are introduced, respectively to ensure lower and upper bounds to the weights that fit within DBMSTCLU needs. This sanitizing mechanism is called the Weight-Release mechanism. Coupled with PAMST, it will allow us to produce a weighted spanning tree with differential privacy, that will be exploited in our private graph clustering.

**Definition 3.12** (Weight-Release mechanism). *Let  $\mathcal{G} = (G, w)$  be a weighted graph,  $\epsilon > 0$  a privacy parameter,  $s$  a scaling parameter,  $\tau \geq 0$ , and  $p \geq 1$  two normalization parameters. The Weight-Release mechanism is defined as*

$$\mathcal{M}_{w,r}(G, w, s, \tau, p) = \left( G, w' = \frac{w + (Y_1, \dots, Y_{|E|}) + \tau}{p} \right)$$

where  $Y_i$  are i.i.d. random variables drawn from  $Lap(0, s)$ . With  $w + (Y_1, \dots, Y_{|E|})$  meaning that if one gives an arbitrary order to the edges  $E = (e_i)_{i \in [|E|]}$ , one has  $\forall i \in [|E|], w'(e_i) = w(e_i) + Y_i$ .

The following theorem presents the privacy guarantees of the Weight-Release mechanism.

**Theorem 3.6.** *Let  $G = (V, E)$  be the topology of a simple-undirected graph,  $\tau \geq 0$ ,  $p \geq 1$ , then  $\forall \epsilon > 0$ ,  $\mathcal{M}_{w,r}(G, \bullet, \frac{\mu}{\epsilon}, \tau, p)$  is  $\epsilon$ -differentially private on  $G$ .*

*Proof.* Given  $\tau \geq 0$ ,  $p \geq 1$ , and  $\epsilon > 0$ , the Weight release mechanism scaled to  $\frac{\mu}{\epsilon}$  can be break down into a Laplace mechanism and a post-processing consisting in adding  $\tau$  to every edge and dividing them by  $p$ . Using Theorems 2.1 and 2.5, one gets the expected result.  $\square$

So far we have presented DBMSTCLU and PAMST algorithms, and the Weight-Release mechanism. Let us now introduce how to compose those blocks to obtain a Private node clustering in a graph, called PTCLUST. The

---

**Algorithm 3** PTCLUST( $G, w, u_G, \epsilon, \tau, p$ )

---

- 1: **Input:**  $\mathcal{G} = (V, E, w)$  a weighted graph (separately the topology  $G$  and the weight function  $w$ ),  $\epsilon$  a degree of privacy and  $u_G$  utility function.
  - 2:  $T = \text{PAMST}(G, w, u_G, \epsilon/2)$
  - 3:  $\mathcal{T}' = \mathcal{M}_{w,r}(T, w|_{E(T)}, \frac{2\mu}{\epsilon}, \tau, p)$
  - 4: **return** DBMSTCLU( $\mathcal{T}'$ )
- 

algorithm 3 takes as an input a weighted graph (dissociated topology and weight function), a utility function, a privacy degree and two normalization parameters. It outputs a clustering partition. To do so, a spanning tree topology is produced using PAMST with time and space complexities respectively equal to  $O(|V|^2)$  and  $O(|E|)$ . Afterward a randomized and normalized version of the associated weight function is released using the Weight-release mechanism. Finally the obtained weighted tree is given as an input to DBMSTCLU that performs a clustering partition with  $O(|V|)$  time and space complexities. The following theorem ensures that our method preserves  $\epsilon$ -differential privacy.

**Theorem 3.7.** *Let  $G = (V, E)$  be the topology of a simple-undirected graph,  $\tau \geq 0$ , and  $p \geq 1$ , then  $\forall \epsilon > 0$ , PTCLUST( $G, \bullet, u_G, \epsilon, \tau, p$ ) is  $\epsilon$ -differentially private on  $G$ .*

*Proof.* Using Theorem 3.5 one has that  $T$  is produced with  $\epsilon/2$ -differential privacy, and using Theorem 3.6 one has that  $w'$  is obtained with  $\epsilon/2$ -differential privacy as

well. Therefore using Theorem 2.4,  $\mathcal{T}'$  is released with  $\epsilon$ -differential privacy. Using the post-processing property (Theorem 2.5) one gets the expected result.  $\square$

### 3.4 DIFFERENTIAL PRIVACY TRADE-OFF OF CLUSTERING

The results stated in this section present the security/accuracy trade-off of our new method in the differentially-private framework. PTCLUST relies on two differentially -private mechanisms, namely PAMST and the Weight-Release mechanism. Evaluating the accuracy of this method amounts to check whether using these methods for ensuring privacy does not deteriorate the final clustering partition. The accuracy is preserved if PAMST outputs the same topology as the MST-based clustering and if the Weight-Release mechanism preserves enough the weight function. According to Def. 3.8, if a tree has a partitioning topology, then it fits the tree-based clustering. The following theorem states that with high probability PAMST outputs a tree with a partitioning topology.

**Theorem 3.8.** *Let us consider a graph  $\mathcal{G} = (V, E, w)$  with  $K$  strongly homogeneous clusters  $C_1^*, \dots, C_K^*$  and  $T = \text{PAMST}(\mathcal{G}, u_{\mathcal{G}}, w, \epsilon)$ ,  $\epsilon > 0$ .  $T$  has a partitioning topology with probability at least*

$$1 - \sum_{i=1}^K (|C_i^*| - 1) \exp\left(-\frac{A}{2\Delta u_{\mathcal{G}}(|V| - 1)}\right)$$

$$\text{with } A = \epsilon \left( \begin{array}{cc} \bar{\alpha}_i \max_{e \in E(\mathcal{G}_{|C_i^*})}(w(e)) & - \min_{e \in E(\mathcal{G}_{|C_i^*})}(w(e)) \end{array} \right) + \ln |E|.$$

*Proof.* See supplementary material.  $\square$

The following theorem states that given a tree  $\mathcal{T}$  under the strong homogeneity condition, if the subtree associated to a cluster respects Def. 3.9, then it still holds after applying the Weight-Release mechanism to this tree.

**Theorem 3.9.** *Let us consider a graph  $\mathcal{G} = (V, E, w)$  with  $K$  strongly homogeneous clusters  $C_1^*, \dots, C_K^*$  and  $T = \text{PAMST}(\mathcal{G}, u_{\mathcal{G}}, w, \epsilon)$ ,  $\mathcal{T} = (T, w|_{\mathcal{T}})$  and  $\mathcal{T}' = \mathcal{M}_{w,r}(T, w|_{\mathcal{T}}, s, \tau, p)$  with  $s \ll p, \tau$ . Given some cluster  $C_i^*$ , and  $j \neq i$  s.t.  $e^{(ij)} \in \text{Cut}_{\mathcal{G}}(\mathcal{T})$ , if  $H_{\mathcal{T}|_{C_i^*}}(e^{(ij)})$  is verified, then  $H_{\mathcal{T}'|_{C_i^*}}(e^{(ij)})$  is verified with probability at least*

$$1 - \frac{\mathbb{V}(\varphi)}{\mathbb{V}(\varphi) + \mathbb{E}(\varphi)^2}$$

with the following notations :

$$\bullet \varphi = \left( \max_{j \in [|C_i^*| - 1]} Y_j \right)^2 - \min_{j \in [|C_i^*| - 1]} Z_j \times X^{\text{out}}$$

- $Y_j \underset{iid}{\sim} \text{Lap}\left(\frac{\max_{e \in E(\mathcal{T})} w(e) + \tau}{p}, \frac{s}{p}\right)$
- $Z_j \underset{iid}{\sim} \text{Lap}\left(\frac{\min_{e \in E(\mathcal{T})} w(e) + \tau}{p}, \frac{s}{p}\right)$
- $X^{\text{out}} \sim \text{Lap}\left(\frac{w(e^{(ij)}) + \tau}{p}, \frac{s}{p}\right),$

*Proof.* See supplementary material.  $\square$

Note that Theorem 3.9 is stated in a simplified version. A more complete version (specifying an analytic version of  $\mathbb{V}(\varphi)$  and  $\mathbb{E}(\varphi)$ ) is given in the supplementary material.

## 4 EXPERIMENTS

So far we have exhibited the trade-off between clustering accuracy and privacy and we experimentally illustrate it with some qualitative results. We have performed experiments on two classical synthetic graph datasets for clustering with nonconvex shapes: two concentric circles and two moons, both in their noisy versions. For the sake of readability and for visualization purposes, both graph datasets are embedded into a two dimensional Euclidean space. Each dataset contains 100 data nodes that are represented by a point of two coordinates. Both graphs have been built with respect to the strong homogeneity condition: edge weights within clusters are between  $w_{\min} = 0.1$  and  $w_{\max} = 0.3$  while edges between clusters have a weight strictly above  $w_{\max}^2/w_{\min} = 0.9$ . In practice, the complete graph has trimmed from its irrelevant edges (*i.e.* not respecting the strong homogeneity condition). Hence, those graphs are not necessarily Euclidean since close nodes in the visual representation may not be connected in the graph. Finally, weights are normalized between 0 and 1.

Figures 1 and 2 (best viewed in color) show for each dataset (a) the original homogeneous graph  $\mathcal{G}$  built by respecting the homogeneity condition, (b) the clustering partition<sup>1</sup> of DBMSTCLU with the used underlying MST, the clustering partitions for PTCLUST with  $\mu = 0.1$  obtained respectively with different privacy degrees<sup>2</sup> : (c)  $\epsilon = 0.5$ , (d)  $\epsilon = 0.7$  and (e)  $\epsilon = 1.0$ . The utility function  $u_{\mathcal{G}}$  corresponds to the graph weight. Each experiment is carried out independently and the tree topology obtained by PAMST will eventually be different. This explains why the edge between clusters may not be the same when the experiment is repeated with a

<sup>1</sup>For the sake of clarity, the edges in those Figures are represented based on the original weights and not on the privately released weights.

<sup>2</sup>Note that, although the range of  $\epsilon$  is in  $\mathbb{R}_+^*$ , it is usually chosen in practice in  $(0, 1]$  (Dwork and Roth, 2013, Chap 1&2).

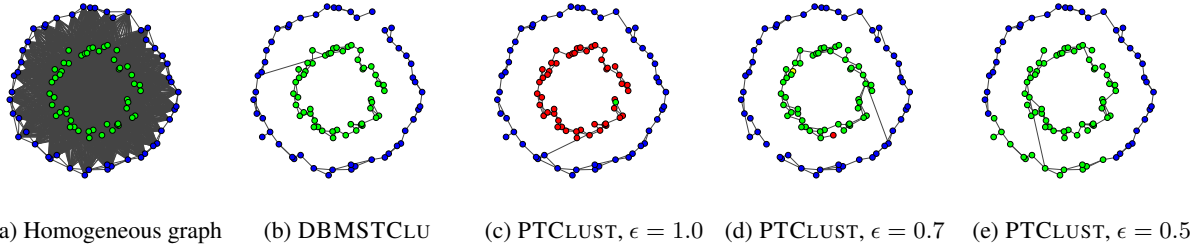


Figure 1: Circles experiments for  $n = 100$ . PTCLUST parameters:  $w_{min} = 0.1$ ,  $w_{max} = 0.3$ ,  $\mu = 0.1$ .

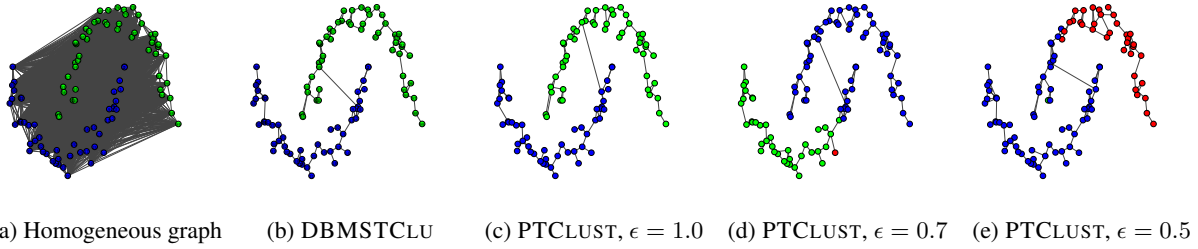


Figure 2: Moons experiments for  $n = 100$ . PTCLUST parameters:  $w_{min} = 0.1$ ,  $w_{max} = 0.3$ ,  $\mu = 0.1$ .

different level of privacy. However, this will marginally affect the overall quality of the clustering.

As expected, DBMSTCLU recovers automatically the right partition and the results are shown here for comparison with PTCLUST. For PTCLUST, the true MST is replaced with a private approximate MST obtained for suitable  $\tau$  and  $p$  ensuring final weights between 0 and 1.

When the privacy degree is moderate ( $\epsilon \in \{1.0, 0.7\}$ ), it appears that the clustering result is slightly affected. More precisely, in Figures 1c and 1d the two main clusters are recovered while one point is isolated as a singleton. This is due to the randomization involved in determining the edge weights for the topology returned by PAMST. In Figure 2c, the clustering is identical to the one from DBMSTCLU in Figure 2b. In Figure 1d, the clustering is very similar to the DBMSTCLU one, with the exception of an isolated singleton. However, as expected from our theoretical results, when  $\epsilon$  is decreasing (even below 0.5), the clustering quality deteriorates, as DBMSTCLU is sensitive to severe changes in the MST (cf. Figure 1e, 2e).

## 5 CONCLUSION

In this paper, we introduced PTCLUST, a novel graph clustering algorithm able to recover arbitrarily-shaped clusters while preserving differential privacy on the weights of the graph. It is based on the release of a private approximate minimum spanning tree of the graph of

the dataset, by performing suitable cuts to reveal the clusters. To the best of our knowledge, this is the first differentially private graph-based clustering algorithm adapted to nonconvex clusters. The theoretical analysis exhibited a trade-off between the degree of privacy and the accuracy of the clustering result. Differential privacy is investigated in the framework of strong homogeneity but this is quite restrictive. A smoother result would be very interesting but it is more challenging. This will be a focus of our future work. Our work suits to applications where privacy is a critical issue and it could pave the way to metagenomics and genes classification using individual gene maps while protecting patient privacy. Future work will also be devoted to deeply investigate these applications.

## References

- T. Asano, B. Bhattacharya, M. Keil, and F. Yao. Clustering algorithms based on minimum and maximum spanning trees. In *Proceedings of the Fourth Annual Symposium on Computational Geometry, SCG '88*, pages 252–257, New York, NY, USA, 1988. ACM.
- A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing, STOC '08*, pages 609–618, New York, NY, USA, 2008. ACM.
- L. Chen, T. Yu, and R. Chirkova. Wavecluster with differential privacy. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, CIKM '15*, pages 1011–1020, New York, NY, USA, 2015. ACM.
- C. Dwork. A firm foundation for private data analysis. *Commun. ACM*, 54(1):86–95, January 2011.
- C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3-4):211–407, 2013.
- C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In *Eurocrypt*, volume 4004, pages 486–503. Springer, 2006a.
- C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography*, pages 265–284. Springer Berlin Heidelberg, 2006b.
- O. Grygorash, Y. Zhou, and Z. Jorgensen. Minimum spanning tree based clustering algorithms. In *2006 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06)*, pages 73–81, Nov 2006.
- M. Hay, C. Li, G. Miklau, and D. Jensen. Accurate estimation of the degree distribution of private networks. In *2009 Ninth IEEE International Conference on Data Mining*, pages 169–178, Dec 2009.
- S.-S. Ho and S. Ruan. Preserving privacy for interesting location pattern mining from trajectory data. *Trans. Data Privacy*, 6(1):87–106, April 2013.
- S. P. Kasiviswanathan, K. Nissim, S. Raskhodnikova, and A. Smith. Analyzing graphs with node differential privacy. In *Proceedings of the 10th Theory of Cryptography Conference on Theory of Cryptography, TCC'13*, pages 457–476, Berlin, Heidelberg, 2013. Springer-Verlag.
- F. McSherry. Privacy integrated queries. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data (SIGMOD)*. Association for Computing Machinery, Inc., June 2009.
- F. McSherry and K. Talwar. Mechanism design via differential privacy. In *Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, Providence, RI, October 2007. IEEE.
- A. Morvan, K. Choromanski, C. Gouy-Pailler, and J. Atif. Graph sketching-based massive data clustering. *SIAM Data Mining 2018 (to appear)*, 2017.
- Y. Mülle, C. Clifton, and K. Böhm. Privacy-integrated graph clustering through differential privacy. In *EDBT/ICDT Workshops*, 2015.
- H. H. Nguyen, A. Imine, and M. Rusinowitch. Detecting communities under differential privacy. In *Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society, WPES '16*, pages 83–93, New York, NY, USA, 2016. ACM.
- K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing - STOC*. ACM Press, 2007.
- R. Pinot. Minimum spanning tree release under differential privacy constraints. *ArXiv e-prints*, January 2018.
- S. E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27 – 64, 2007.
- A. Sealfon. Shortest paths and distances with differential privacy. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems - PODS*. ACM Press, 2016.
- C. T. Zahn. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Trans. Comput.*, 20(1):68–86, January 1971.

---

# GaAN: Gated Attention Networks for Learning on Large and Spatiotemporal Graphs

---

Jiani Zhang<sup>1,\*</sup>, Xingjian Shi<sup>2,\*</sup>, Junyuan Xie<sup>3</sup>, Hao Ma<sup>4</sup>, Irwin King<sup>1</sup>, Dit-Yan Yeung<sup>2</sup>

<sup>1</sup>The Chinese University of Hong Kong, Hong Kong, China, {jnzhang, king}@cse.cuhk.edu.hk

<sup>2</sup>Hong Kong University of Science and Technology, Hong Kong, China, {xshiab, dyyeung}@cse.ust.hk

<sup>3</sup>Amazon Web Services, WA, USA, junyuanx@amazon.com

<sup>4</sup>Microsoft Research, WA, USA, haoma@microsoft.com

## Abstract

We propose a new network architecture, *Gated Attention Networks* (GaAN), for learning on graphs. Unlike the traditional multi-head attention mechanism, which equally consumes all attention heads, GaAN uses a convolutional sub-network to control each attention head’s importance. We demonstrate the effectiveness of GaAN on the inductive node classification problem on large graphs. Moreover, with GaAN as a building block, we construct the *Graph Gated Recurrent Unit* (GGRU) to address the traffic speed forecasting problem. Extensive experiments on three real-world datasets show that our GaAN framework achieves state-of-the-art results on both tasks.

## 1 INTRODUCTION

Many crucial machine learning tasks involve graph-structured datasets, such as classifying posts in a social network (Hamilton et al., 2017a), predicting interfaces between proteins (Fout et al., 2017) and forecasting the future traffic speed in a road network (Li et al., 2018). The main difficulty in solving these tasks is how to find the right way to express and exploit the graph’s underlying structural information. Traditionally, this is achieved by calculating various graph statistics like degree and centrality, using graph kernels, or extracting human engineered features (Hamilton et al., 2017b).

Recent research, however, has pivoted to solving these problems by *graph convolution* (Duvenaud et al., 2015; Atwood and Towsley, 2016; Kipf and Welling, 2017; Fout et al., 2017; Hamilton et al., 2017a; Veličković et al., 2018; Li et al., 2018), which generalizes the stan-

dard definition of convolution over a regular grid topology (Gehring et al., 2017; Krizhevsky et al., 2012) to ‘convolution’ over graph structures. The basic idea behind ‘graph convolution’ is to develop a localized parameter-sharing operator on a set of neighboring nodes to aggregate a local set of lower-level features. We refer to such an operator as a *graph aggregator* (Hamilton et al., 2017a) and the set of local nodes as the *receptive field* of the aggregator. Then, by stacking multiple graph aggregators, we build a deep neural network model which can be trained end-to-end to extract the local and global features across the graph. Note that we use the spatial definition instead of the spectral definition (Hammond et al., 2011; Bruna et al., 2014) of graph convolution because the full spectral treatment requires eigen-decomposition of the Laplacian matrix, which is computationally intractable on large graphs, while the localized versions (Defferrard et al., 2016; Kipf and Welling, 2017) can be interpreted as graph aggregators (Hamilton et al., 2017a).

Graph aggregators are the basic building blocks of graph convolutional neural networks. A model’s ability to capture the structural information of graphs is largely determined by the design of its aggregators. Most existing graph aggregators are based on either pooling over neighborhoods (Kipf and Welling, 2017; Hamilton et al., 2017a) or computing a weighted sum of the neighboring features (Monti et al., 2017). In essence, functions that are permutation invariant and can be dynamically resizing are eligible graph aggregators. One class of such functions is the neural attention network (Bahdanau et al., 2015), which uses a subnetwork to compute the correlation weight of the elements in a set. Among the family of attention models, the multi-head attention model has been shown to be effective for machine translation task (Lin et al., 2017; Vaswani et al., 2017). It has later been adopted as a graph aggregator to solve the node classification problem (Veličković et al., 2018). A single attention head sums up the elements that are similar

---

\* These two authors contributed equally.

to the query vector in one representation subspace. Using multiple attention heads allows exploring features in different representation subspaces, which provides more modeling power in nature. However, treating each attention head equally loses the opportunity to benefit from some attention heads which are inherently more important than others. For instance, assume there are ten different relationships in the graph and only two of them are valid for each node. If we use ten attention heads to model these relationships and treat them equally, each node will still aggregate features from eight non-existent subspaces. This will mislead the final prediction.

To this end, we propose the *Gated Attention Networks* (GaAN) for learning on graphs. GaAN uses a small convolutional subnetwork to compute a soft gate at each attention head to control its importance. Unlike the traditional multi-head attention that admits all attended contents, the gated attention can modulate the amount of attended content via the introduced gates. From this perspective, the gate-generation network acts as a high-level controller that determines how to aggregate the features extracted by the attention heads. Moreover, since only a simple and light-weighted subnetwork is introduced in constructing the gates, the computational overhead is negligible and the model is easy to train. We demonstrate the effectiveness of our new aggregator by applying it to the inductive node classification problem. To train our model and other graph aggregators on relatively large graphs, we also improve the sampling strategy introduced in (Hamilton et al., 2017a) to reduce the memory cost and increase the run-time efficiency. Furthermore, since our proposed aggregator is very general, we extend it to construct a *Graph Gated Recurrent Unit* (GGRU), which is directly applicable for spatiotemporal forecasting problem. Extensive experiments on two node classification datasets, PPI and the large Reddit dataset (Hamilton et al., 2017a), and one traffic speed forecasting dataset, METR-LA (Li et al., 2018), show that GaAN consistently outperforms the baseline models and achieves the state-of-the-art performance.

In summary, our main contributions include: (a) a new multi-head attention-based aggregator with additional gates on the attention heads; (b) a unified framework for transforming graph aggregators to graph recurrent neural networks; and (c) the state-of-the-art prediction performance on three real-world datasets. To the best of our knowledge, this is the first work to study the attention-based aggregators on large and spatiotemporal graphs.

## 2 NOTATIONS

We denote vectors with bold lowercase letters, matrices with bold uppercase letters, and sets with calligraphy let-

ters. We denote a single fully-connected layer with a non-linear activation  $\alpha(\cdot)$  as  $\text{FC}_\theta^\alpha(\mathbf{x}) = \alpha(\mathbf{W}\mathbf{x} + \mathbf{b})$ , where  $\theta = \{\mathbf{W}, \mathbf{b}\}$  are the parameters. Also,  $\theta$  with different subscripts mean different transformation parameters. For activation functions, we denote  $h(\cdot)$  to be the LeakyReLU activation (Xu et al., 2015a) with negative slope equals to 0.1 and  $\sigma(\cdot)$  to be the sigmoid activation.  $\text{FC}_\theta(\mathbf{x})$  means applying no activation function after the linear transform. We denote  $\oplus$  as the concatenation operation and  $\|_{k=1}^K \mathbf{x}_k$  as sequentially concatenating  $\mathbf{x}_1$  through  $\mathbf{x}_K$ . We denote the Hadamard product as ‘ $\circ$ ’ and the dot product between two vectors as  $\langle \cdot, \cdot \rangle$ .

## 3 RELATED WORK

In this section, we will review relevant research on learning on graphs. Our model is also related to many graph aggregators proposed by previous work. We will discuss these aggregators in Section 4.3.

**Neural attention mechanism** Neural attention mechanism is widely adopted in deep learning literature and many variants have been proposed (Xu et al., 2015b; Seo et al., 2017; Zhang et al., 2017; Vaswani et al., 2017; Cheng et al., 2017; Zhang et al., 2018). Among them, our model takes inspiration from the multi-head attention architecture proposed in (Vaswani et al., 2017). Given a query vector  $\mathbf{q}$  and a set of key-value pairs  $\{(\mathbf{k}_1, \mathbf{v}_1), \dots, (\mathbf{k}_n, \mathbf{v}_n)\}$ , a single attention head computes a weighted combination of the value vectors  $\sum_{i=1}^n w_i \mathbf{v}_i$ . The weights are generated by applying softmax to the inner product between the query and keys, i.e.,  $\mathbf{w} = \text{softmax}(\{\mathbf{q}^T \mathbf{k}_1, \dots, \mathbf{q}^T \mathbf{k}_n\})$ . In the multi-head case, the outputs of  $K$  different heads are concatenated to form an output vector with fixed dimensionality. The difference between the proposed GaAN and the multi-head attention mechanism is that we compute additional gates to control the importance of each head’s output.

**Graph convolutional networks on large graph** Applying graph convolution on large graphs is challenging because the memory complexity is proportional to the total number of nodes, which could be hundreds of thousands of nodes in large graphs (Hamilton et al., 2017a). To reduce memory usage and computational cost, (Hamilton et al., 2017a) proposed the GraphSAGE framework that uses a sampling algorithm to select a small subset of the nodes and edges. On each iteration, GraphSAGE first uniformly samples a mini-batch of nodes. Then, for each node, only a fixed number of neighborhoods are selected for aggregation. More recently, Chen et al. (Chen et al., 2018) proposed a new sampling method that randomly samples two sets of nodes according to a proposed distribution. However, this method is only applicable to one aggregator,



i.e., the *Graph Convolutional Network* (GCN) (Kipf and Welling, 2017). For the usage of the gate mechanism, the gate in Li et al. (2016) refers to the gate in Gated Recurrent Units, which are imposed on the activations of the neural network, while our gates are added to the attention heads to control each head’s relative importance.

**Graph convolution networks for spatiotemporal forecasting** Recently, researchers have applied graph convolution, which is commonly used for learning on static graphs, to spatiotemporal forecasting (Yuan et al., 2017). (Seo et al., 2016) proposed *Graph Convolutional Recurrent Neural Network* (GCRNN), which replaced the fully-connected layers in LSTM (Hochreiter and Schmidhuber, 1997) with the ChebNet operator (Defferrard et al., 2016), and applied it to a synthetic video prediction task. Li et al. (Li et al., 2018) proposed *Diffusion Convolutional Recurrent Neural Network* (DCRNN) to address the traffic forecasting problem, where the goal is to predict future traffic speeds in a sensor network given historical traffic speeds and the underlying road graph. DCRNN replaces the fully-connected layers in GRU (Chung et al., 2014) with the diffusion convolution operator (Atwood and Towsley, 2016). Furthermore, DCRNN takes the direction of graph edges into account. The difference between our GGRU with GCRNN and DCRNN is that we proposed a unified method for constructing a recurrent neural network based on an arbitrary graph aggregator rather than proposing a single model.

## 4 GATED ATTENTION NETWORKS

In this section, we first give a generic formulation of graph aggregators followed by the multi-head attention mechanism. Then, we introduce the proposed gated attention aggregator. Finally, we review the other kinds of graph aggregators proposed by previous work and explain their relationships with ours.

**Generic formulation of graph aggregators** Given a node  $i$  and its neighboring nodes  $\mathcal{N}_i$ , a graph aggregator is a function  $\gamma$  in the form of  $\mathbf{y}_i = \gamma_{\Theta}(\mathbf{x}_i, \{\mathbf{z}_{\mathcal{N}_i}\})$ , where  $\mathbf{x}_i$  and  $\mathbf{y}_i$  are the input and output vectors of the center node  $i$ .  $\mathbf{z}_{\mathcal{N}_i} = \{\mathbf{z}_j | j \in \mathcal{N}_i\}$  is a set of the reference vectors in the neighboring nodes and  $\Theta$  is the learnable parameters of the aggregator. In this paper, we do not consider aggregators that use edge features. However, it is straightforward to incorporate edges in our definition by defining  $\mathbf{z}_j$  to contain the edge feature vectors  $\mathbf{e}_{i,j}$ .

### 4.1 MULTI-HEAD ATTENTION AGGREGATOR

We linearly project the center node feature  $\mathbf{x}_i$  to get the query vector and project the neighboring node features to get the key and value vectors. We then apply the

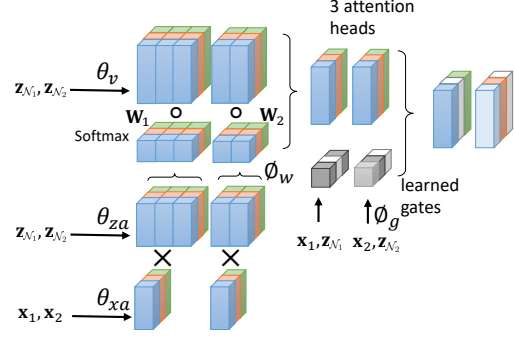


Figure 1: Illustration of a three-head gated attention aggregator with two center nodes in a mini-batch.  $|\mathcal{N}_1| = 3$  and  $|\mathcal{N}_2| = 2$  respectively. Different colors indicate different attention heads. Gates in darker color stands for larger values. (Best viewed in color)

multi-head attention mechanism (Vaswani et al., 2017) to obtain the final aggregation function. For the multi-head attention mechanism, different heads capture features from different representation subspaces. The detailed formulation of the multi-head attention aggregator is as follows:

$$\mathbf{y}_i = \text{FC}_{\theta_o}(\mathbf{x}_i \oplus \prod_{k=1}^K \sum_{j \in \mathcal{N}_i} w_{i,j}^{(k)} \text{FC}_{\theta_v}^h(\mathbf{z}_j)),$$

$$w_{i,j}^{(k)} = \frac{\exp(\phi_w^{(k)}(\mathbf{x}_i, \mathbf{z}_j))}{\sum_{l=1}^{|\mathcal{N}_i|} \exp(\phi_w^{(k)}(\mathbf{x}_i, \mathbf{z}_l))},$$

$$\phi_w^{(k)}(\mathbf{x}, \mathbf{z}) = \langle \text{FC}_{\theta_{xa}}^{(k)}(\mathbf{x}), \text{FC}_{\theta_{za}}^{(k)}(\mathbf{z}) \rangle.$$

Here,  $K$  is the number of attention heads.  $w_{i,j}^{(k)}$  is the  $k$ th attentional weights between the center node  $i$  and the neighboring node  $j$ , which is generated by applying a softmax to the dot product values.  $\theta_{xa}^{(k)}$ ,  $\theta_{za}^{(k)}$  and  $\theta_v^{(k)}$  are the parameters of the  $k$ th head for computing the query, key, and value vectors, which have dimensions of  $d_a$ ,  $d_a$  and  $d_v$  respectively. The  $K$  attention outputs are concatenated with the input vector and passed to a fully-connected layer parameterized by  $\theta_o$  to get the final output  $\mathbf{y}_i$ , which has dimension  $d_o$ . The difference between our aggregator and that in GAT (Veličković et al., 2018) is that we have adopted the key-value attention mechanism and the dot product attention while GAT does not compute additional value vectors and uses a fully-connected layer to compute  $\phi_w^{(k)}$ .

### 4.2 GATED ATTENTION AGGREGATOR

While the multi-head attention aggregator can explore multiple representation subspaces between the center node and its neighborhoods, not all of these subspaces are equally important; some subspaces may not even exist for specific nodes. Feeding the output of an attention

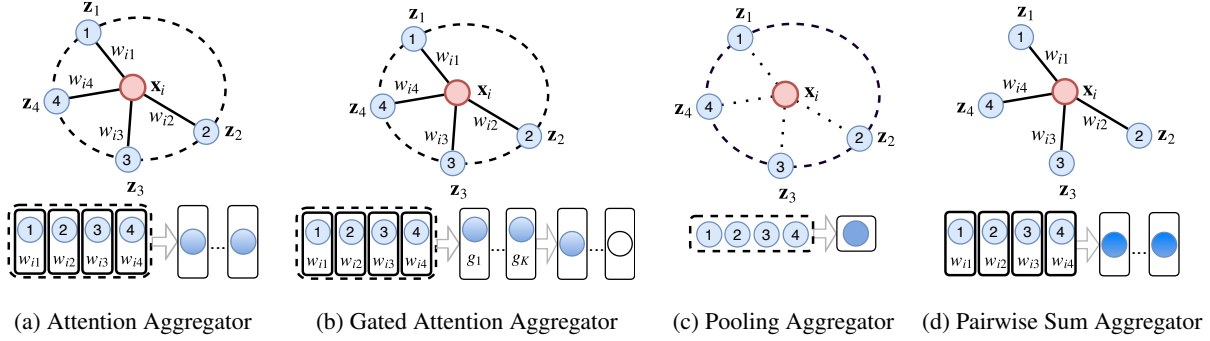


Figure 2: Comparison of different graph aggregators. The aggregators are drawn for only one aggregation step. The nodes in red are center nodes and the nodes in blue are neighboring nodes. The bold black lines between the center node and neighbor nodes indicate that a learned pairwise relationship is used for calculating the relative importance. The oval in dash line around the neighbors means the interaction among neighbors is utilized when determining the weights. (Best viewed in color)

head that captures a useless representation can mislead the mode’s final prediction.

Therefore, we compute an additional soft gate between 0 (low importance) and 1 (high importance) to assign different importance to each head. In combination with the multi-head attention aggregator, we get the formulation of the gated attention aggregator:

$$\begin{aligned}
 \mathbf{y}_i &= \text{FC}_{\theta_o}(\mathbf{x}_i \oplus \parallel_{k=1}^K (g_i^{(k)} \sum_{j \in \mathcal{N}_i} w_{i,j}^{(k)} \text{FC}_{\theta_v}^h(\mathbf{z}_j))), \quad (2) \\
 \mathbf{g}_i &= [g_i^{(1)}, \dots, g_i^{(K)}] = \psi_g(\mathbf{x}_i, \mathcal{Z}_{\mathcal{N}_i}),
 \end{aligned}$$

where  $g_i^{(k)}$  is a scalar, the gate value of the  $k$ th head at node  $i$ . To make sure adding gates will not introduce too many additional parameters, we use a convolutional network  $\psi_g$  that takes the center node and neighboring node features as the input to generate the gate values. All the other parameters have the same meanings as in Eqn. (1).

There are multiple possible designs of the  $\psi_g$  network. In this paper, we combine average pooling and max pooling to construct the network. The detailed formula is:

$$\mathbf{g}_i = \text{FC}_{\theta_g}^\sigma(\mathbf{x}_i \oplus \max_{j \in \mathcal{N}_i} \{\text{FC}_{\theta_m}(\mathbf{z}_j)\}) \oplus \frac{\sum_{j \in \mathcal{N}_i} \mathbf{z}_j}{|\mathcal{N}_i|}. \quad (3)$$

Here,  $\theta_m$  maps the neighbor features to a  $d_m$  dimensional vector before taking the element-wise max and  $\theta_g$  maps the concatenated features to the final  $K$  gates. By setting a small  $d_m$ , the subnetwork for computing the gates will have negligible computational overhead. A visual illustration of GaAN aggregator’s structure can be found in Figure 1. Also, we compare the general structures of the multi-head attention aggregator and the gated attention aggregator in Figure 2a and Figure 2b.

### 4.3 OTHER GRAPH AGGREGATORS

Most previous graph aggregators except attention-based aggregators can be summarized into two general categories: graph pooling aggregators and graph pairwise sum aggregators. In this section, we first describe these two types of aggregators and then explain their relationship with the attention-based aggregator. Finally, we give a list of the baseline aggregators used in the experiments.

**Graph pooling aggregators** The main characteristic of graph pooling aggregators is that they do not consider the correlation between neighboring nodes and the center node. Instead, neighboring nodes’ features are directly aggregated and the center node’s feature is simply concatenated or added to the aggregated vector and then passed through an output function  $\phi_o$ :

$$\mathbf{y}_i = \phi_o(\mathbf{x}_i \oplus \text{pool}_{j \in \mathcal{N}_i}(\phi_v(\mathbf{z}_j))). \quad (4)$$

Here, the projection function  $\phi_v$  and the output function  $\phi_o$  can be a single fully-connected layer and the  $\text{pool}(\cdot)$  operator can be average pooling, max pooling, or sum pooling. The majority of existing graph aggregators are special cases of the graph pooling aggregators. Some models only integrate the node features of neighborhoods (Duvenaud et al., 2015; Kipf and Welling, 2017; Hamilton et al., 2017a), while others integrated edge features as well (Atwood and Towsley, 2016; Fout et al., 2017; Schütt et al., 2017). In Figure 2c, we illustrate the architecture of the graph pooling aggregators.

**Graph pairwise sum aggregators** Like attention-based aggregators, graph pairwise sum aggregators also aggregate the neighborhood features by taking  $K$  weighted sums. The difference is that the weight between node  $i$  and its neighbor  $j$  is not related to the other neighbors in  $\mathcal{N}_i$ . The formula of graph pairwise sum aggregator is given as follows:

$$\mathbf{y}_i = \phi_o(\mathbf{x}_i \oplus \prod_{k=1}^K \sum_{j \in \mathcal{N}_i} w_{i,j}^{(k)} \phi_v^{(k)}(\mathbf{z}_j)), \quad (5)$$

$$w_{i,j}^{(k)} = \phi_w^{(k)}(\mathbf{x}_i, \mathbf{z}_j).$$

Here,  $w_{i,j}^{(k)}$  is only related to the pair  $\mathbf{x}_i$  and  $\mathbf{z}_j$ , while in attention-based models  $w_{i,j}^{(k)}$  is related to features of all neighbors  $\mathbf{z}_{\mathcal{N}_i}$ . Models like the adaptive forget gate strategy in *Graph LSTM* (Liang et al., 2016) and MoNet (Monti et al., 2017) employed pairwise sum aggregators with a single head or multiple heads. In Figure 2d, we illustrate the architecture of the graph pairwise sum aggregators.

**Baseline aggregators** To fairly evaluate the effectiveness of GaAN against previous work, we choose two representative aggregators in each category as baselines:

- **Avg. pooling:**  $\mathbf{y}_i = \text{FC}_{\theta_o}(\mathbf{x}_i \oplus \text{pool}_{j \in \mathcal{N}_i}^{\text{avg}}(\text{FC}_{\theta_v}^h(\mathbf{z}_j)))$ .
- **Max pooling:**  $\mathbf{y}_i = \text{FC}_{\theta_o}(\mathbf{x}_i \oplus \text{pool}_{j \in \mathcal{N}_i}^{\text{max}}(\text{FC}_{\theta_v}^h(\mathbf{z}_j)))$ .

- **Pairwise + sigmoid:**

$$\mathbf{y}_i = \text{FC}_{\theta_o}(\mathbf{x}_i \oplus \prod_{k=1}^K \sum_{j \in \mathcal{N}_i} w_{i,j}^{(k)} \text{FC}_{\theta_v}^h(\mathbf{z}_j)),$$

$$w_{i,j}^{(k)} = \frac{1}{|\mathcal{N}_i|} \sigma(\langle \text{FC}_{\theta_{x_a}}^{(k)}(\mathbf{x}_i), \text{FC}_{\theta_{z_a}}^{(k)}(\mathbf{z}_j) \rangle).$$

- **Pairwise + tanh:** Replace the sigmoid activation in **Pairwise + sigmoid** to tanh.

## 5 INDUCTIVE NODE CLASSIFICATION

### 5.1 MODEL

In the inductive node classification task, every node is assigned one or multiple labels. During training, the validation and testing nodes are not observable. And the goal is to predict the labels of the unseen testing nodes. Our approach follows that of (Hamilton et al., 2017a), where a mini-batch of nodes are sampled on each iteration during training and multiple layers of graph aggregators are stacked to compute the predictions.

With a stack of  $M$  layers of graph aggregators, we will first sample a mini-batch of nodes  $\mathcal{B}_0$  and then recursively expand  $\mathcal{B}_\ell$  to be  $\mathcal{B}_{\ell+1}$  by sampling the neighboring nodes of  $\mathcal{B}_\ell$ . After  $M$  sampling steps, we can get a hierarchy of node batches:  $\mathcal{B}_1, \dots, \mathcal{B}_M$ . The node representations, which are initialized to be the node features, will be aggregated in reverse order from  $\mathcal{B}_M$  to  $\mathcal{B}_0$ . The representations of the last layer, i.e., the final representations of the nodes in  $\mathcal{B}_0$ , are projected to get the output. We use the sigmoid activation for multi-label classification

Table 1: Effect of the merge operation. Both methods sample a maximum of 15 neighborhoods without replacement for three recursive steps on the Reddit dataset. We start from 512 seed nodes. The total number of nodes after the  $\ell$ th sampling step is denoted as  $|\mathcal{B}_\ell|$ . The sampling process is repeated for ten times and the mean is reported.

Strategy/Sample Step	$ \mathcal{B}_0 $	$ \mathcal{B}_1 $	$ \mathcal{B}_2 $	$ \mathcal{B}_3 $
Sample without merge	512	7.8K	124.4K	1.9M
Sample and merge	512	7.5K	70.7K	0.2M

and the softmax activation for multi-class classification. Also, we use the cross-entropy loss to train the model.

A naive sampling algorithm is always to sample all neighbors. However, it is not practical on large graphs because the memory complexity is  $O(|\mathcal{V}|)$  and the time complexity is  $O(|\mathcal{E}|)$ , where  $|\mathcal{V}|$  and  $|\mathcal{E}|$  are the total number of nodes and edges. Instead, similar to GraphSAGE (Hamilton et al., 2017a), we only sample a subset of the neighborhoods for each node. In our implementation, at the  $\ell$ th sampling step, we sample  $\min(|\mathcal{N}_i|, S_\ell)$  neighbors without replacement for the node  $i$ , where  $S_\ell$  is a hyperparameter that controls the maximum number of sampled neighbors at the  $\ell$ th step. Moreover, to improve over GraphSAGE and further reduce memory cost, we merge repeated nodes that are sampled from different seeds’ neighborhoods within each mini-batch. This greatly reduces the size of  $\mathcal{B}_\ell$ s as shown in Table 1.

Note that  $\min(|\mathcal{N}_i|, S_\ell)$  is not the same for all the nodes  $i$ . Thus, instead of padding the sampled neighborhood set to the same size for utilizing fast tensor operation, we implemented new GPU kernels that directly operate on inputs with variable lengths to accelerate computations.

### 5.2 EXPERIMENTAL SETUP

We performed a thorough comparison of GaAN with the state-of-the-art models, five aggregator-based models in our framework and a two-layer fully connected neural network on the PPI and Reddit datasets (Hamilton et al., 2017a). The five baseline aggregators include the multi-head attention aggregator, two pooling based aggregators, and two pairwise sum based aggregators mentioned in Section 4.3. We also conducted comprehensive ablation analysis.

The PPI dataset was collected from the molecular signatures database (Subramanian et al., 2005). Each node represents a protein and edges represent the interaction between proteins. Labels represent the cellular functions of each protein from gene ontology. Reddit is an online discussion forum where users can post and discuss contents on different topics. Each node represents a post and

Table 2: Datasets for inductive node classification. ‘multi’ stands for multilabel classification and ‘single’ otherwise.

Data	#Nodes	#Edges	#Fea	#Classes
PPI	56.9K	806.2K	50	121(multi)
Reddit	233.0K	114.6M	602	41(single)

two nodes are connected if they are commented by the same user. The labels indicate the community a post belongs to. Detailed statistics are listed in Table 2.

### 5.3 MODEL ARCHITECTURES AND IMPLEMENTATION DETAIL

The GaAN and other five aggregator-based networks are stacked with two graph aggregators. Each aggregator is followed by the LeakyReLU activation with negative slope equals to 0.1 and a dropout layer with dropout rate set to be 0.1. The output dimension  $d_o$  of all layers are fixed to be 128 except when we compare the relative performance with different output dimensions. To keep the number of parameters comparable for the multi-head models with a different number of heads, we fix the product of the dimension of the value vector and the number of heads, i.e.,  $d_v \times K$  to be the same when evaluating the effect of varying the number of heads. Also, the hyperparameters of the first and the second layer are assumed to be the same if no special explanation is given.

In the PPI experiments, both pooling aggregators have  $d_v = 512$ , where  $d_v$  means the dimensionality of the value vector projected by  $\theta_v$ . For the pairwise sum aggregators, the dimension of the keys  $d_a$  is set to be 24,  $d_v = 64$ , and  $K = 8$ . For both GaAN and the multi-head attention based aggregator,  $d_a$  is set to be 24 and the product  $d_v \times K$  is fixed to be 256. For GaAN, we set  $d_m$  to be 64 in the gate-generation network. Also, we use the entire neighborhoods in the mini-batch training algorithm. In the Reddit experiments, both pooling aggregators have  $d_v = 1024$ . For the pairwise sum aggregators,  $d_a = 32$ ,  $d_v = 256$  and  $K = 4$ . For the attention based aggregators,  $d_a$  is set to be 32 and  $d_v \times K$  is fixed to be 512. We set the gate-generation network in GaAN to have  $d_m = 64$ . Also, the number of heads is fixed to 1 in the first layer for both attention-based models. The maximum number of sampled neighbors in the first and second sampling steps are denoted as  $S_1$  and  $S_2$  and are respectively set to be 25 and 10 in Table 3. In the ablation analysis, we also show the performance when setting them to be (50, 20), (100, 40), and (200, 80).

To illustrate the effectiveness of incorporating graph structures, we also evaluate a two-layer fully-connected neural network with the hidden dimension of 1024 and

Table 3: Summary of different models’ test micro F1 scores in the inductive node classification task. In the first block, we include the best-reported results in the previous papers. In the second block, we report the results obtained by our models. For the PPI dataset, we do not use any sampling strategies. For the Reddit dataset, we use the maximum number sampling strategy with  $S_1=25$  and  $S_2=10$ .

Models / Datasets	PPI	Reddit
GraphSAGE (Hamilton et al., 2017a)	(61.2) <sup>1</sup>	95.4
GAT (Veličković et al., 2018)	97.3 ± 0.2	-
Fast GCN (Chen et al., 2018)	-	93.7
2-Layer FNN	54.07±0.06	73.58±0.09
Avg. pooling	96.85±0.19	95.78±0.07
Max pooling	98.39±0.05	95.62±0.03
Pairwise+sigmoid	98.39±0.05	95.86±0.08
Pairwise+tanh	98.32±0.18	95.80±0.03
Attention-only	98.46±0.09	96.19±0.07
GaAN	<b>98.71±0.02</b>	<b>96.36±0.03</b>

ReLU activation. It only takes node features as input and ignores graph structures.

We train all the aggregator-based models with Adam (Kingma and Ba, 2015) and early stopping on the validation set. Besides, we use the validation set to perform learning rate decay scheduler. For Reddit, before training we normalize all the features and project all the features to a hidden dimension of 256. The initial learning rate is 0.001 and gradually decreases to 0.0001 with the decay rate of 0.5 each time the validation F1 score does not decrease in a window of 4 epochs and early stopping occurs for 10 epochs. The gradient normalization value clips no larger than 1.0. For the PPI dataset, all the input features are projected to a 64-dimension hidden state before passing to the aggregators. The learning rate begins at 0.01 and decays to 0.001 with the decay rate of 0.5 if the validation F1 score does not increase for 15 epochs and stops training for 30 epochs.

The training batch size is fixed to be 512. Also, in all experiments, we use the validation set to select the optimal hyperparameters for training. The training, validation, and testing splits are the same as that in (Hamilton et al., 2017a). The micro-averaged F1 score is used to evaluate the prediction accuracy for both datasets. We repeat the training five times for Reddit and three times for PPI with different random seeds and report the average test F1 score along with the standard deviation.

<sup>1</sup>The performance reported in the paper is relatively low because the author has not trained their model into convergence. Also, it is not fair to compare it with the other scores because it uses the sampling strategy while the others have not.

Table 4: Comparison of the test F1 score on the Reddit and PPI datasets with different sampling neighborhood sizes and attention head number  $K$ .  $S_1$  and  $S_2$  are the maximum number of sampled neighborhoods in the 1st and 2nd sampling steps. ‘all’ means to sample all the neighborhoods.

Models	Reddit					PPI	
	#Param	$S_1, S_2$ 25,10	$S_1, S_2$ 50,20	$S_1, S_2$ 100,40	$S_1, S_2$ 200,80	#Param	$S_1, S_2$ all, all
2-Layer FNN	1.71M	73.58±0.09	73.58±0.09	73.58±0.09	73.58±0.09	1.23M	54.07±0.06
Avg. pooling	866K	95.78±0.07	96.11±0.07	96.28±0.05	96.35±0.02	274K	96.85±0.19
Max pooling	866K	95.62±0.03	96.06±0.09	96.18±0.11	96.33±0.04	274K	98.39±0.05
Pairwise+sigmoid	965K	95.86±0.08	96.19±0.04	96.33±0.05	96.38±0.08	349K	98.39±0.05
Pairwise+tanh	965K	95.80±0.03	96.11±0.05	96.26±0.03	96.36±0.04	349K	98.32±0.18
Attention-only-K1	562K	96.15±0.06	96.40±0.05	96.48±0.02	96.54±0.07	168K	96.31±0.08
Attention-only-K2	571K	96.19±0.07	96.40±0.04	96.52±0.02	96.57±0.02	178K	97.36±0.08
Attention-only-K4	587K	96.11±0.06	96.40±0.02	96.49±0.03	96.56±0.02	196K	98.09±0.07
Attention-only-K8	620K	96.10±0.03	96.38±0.01	96.50±0.04	96.53±0.02	233K	98.46±0.09
GaAN-K1	620K	96.29±0.05	96.50±0.08	96.67±0.04	96.73±0.05	201K	96.95±0.09
GaAN-K2	629K	96.33±0.02	96.59±0.02	96.71±0.05	96.82±0.05	211K	97.92±0.05
GaAN-K4	645K	<b>96.36±0.03</b>	<b>96.60±0.03</b>	96.73±0.04	<b>96.83±0.03</b>	230K	98.42±0.02
GaAN-K8	678K	96.31±0.13	<b>96.60±0.02</b>	<b>96.75±0.03</b>	96.79±0.08	267K	<b>98.71±0.02</b>

## 5.4 MAIN RESULTS

We compare our model with the previous state-of-the-art methods on inductive node classification. This includes GraphSAGE (Hamilton et al., 2017a), GAT (Veličković et al., 2018), and FastGCN (Chen et al., 2018). The GraphSAGE model used a 2-layer sample and aggregate model with a neighborhood size of  $S^{(1)} = 25$  and  $S^{(2)} = 10$  without dropout. The 3-layer GAT model consisted of 4, 4, and 6 heads in the first, second, and third layer respectively. Each attention head had 256 dimensions. GAT did not use neighborhood sampling, L2 regularization, or dropout. The FastGCN model is a fast version of the 3-layer, 128-dimension GCN with sampled neighborhood size being 400, 100, and 400 for each layer and no sampling is done during testing.

Table 3 summarizes all results of the state-of-the-art models and the models proposed in this paper. We denote the multi-head attention aggregator as ‘Attention-only’ in the tables and figures. We find that the proposed model, GaAN, achieves the best F1 score on both datasets and the other baseline aggregators can also show competitive results to the state-of-the-art. We note that aggregator-based models produce much higher F1 score than the fully-connected model, which shows the effectiveness of the graph aggregators. Our max pooling and avg. pooling baselines have higher scores on Reddit than that in the original GraphSAGE record. This mainly contributes to our usage of dropout and the LeakyReLU activation.

Regarding the training time, the average training time of the attention-only model for the first 100 epochs on PPI is 36.5s and that of GaAN is 37.0s when we run on the

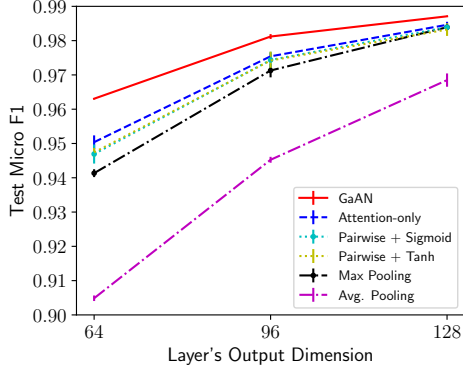
machine with a single TitanX GPU and Intel Xeon CPU 3.70 GHz. This shows that the computational overhead of adding the gates is negligible.

## 5.5 ABLATION ANALYSIS

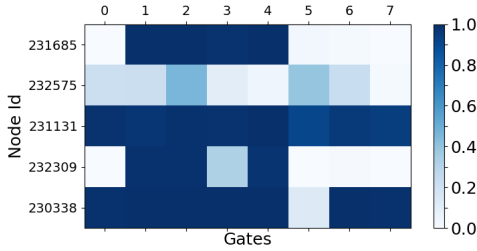
We ran some ablation experiments to analyze the performance of different graph aggregators when different hyperparameters were used. We also visualized the gates of the GaAN model.

**Effect of the number of attention heads and the sample size** We compare the performance of the aggregators when a different number of attention heads and sampling strategies are used. Results are shown in Table 4. We find that attention-based models consistently outperform pooling and pairwise sum based models with the fewer number of parameters, which demonstrates the effectiveness of the attention mechanism in this task. Moreover, GaAN consistently beats the multi-head attention model with the same number of attention heads  $K$ . This proves that adding additional gates to control the importance of the attention heads is beneficial to the final classification performance. From the last two row blocks of Table 4, we note that increasing the number of attention heads will not always produce better results on Reddit. In contrast, on PPI, the larger the  $K$ , the better the prediction results. Also, we can see steady improvement with larger sampling sizes, which is consistent with the observation in (Hamilton et al., 2017a).

**Effect of output dimensions in the PPI dataset** We changed the output dimension to be 64, 96, and 128 in the models for training in the PPI dataset. The test F1 score



(a) Performance of different models with a varying number of output dimensions on PPI.



(b) Visualization of 8 gate values of 5 example nodes on Reddit. Each row represents a learned gate vector for one node.

Figure 3: Ablation analysis on PPI and Reddit

is shown in Figure 3a. All multi-head models have  $K=8$ . We find that the performance becomes better for larger output dimensions and the proposed GaAN consistently outperforms the other models.

**Visualization of gate values** In Figure 3b, we visualized the gate values of five different nodes output by the GaAN-K8 model trained on the Reddit dataset. It illustrates the diversity of the learned gate combinations for different nodes. In most cases, the gates vary across attention heads, which shows that the gate-generation network can be learned to assign different importance to different heads.

## 6 TRAFFIC SPEED FORECASTING

### 6.1 GRAPH GRU

Following (Lin et al., 2017), we formulate traffic speed forecasting as a spatiotemporal sequence forecasting problem where the input and the target are sequences defined on a fixed spatiotemporal graph, e.g., the road network. To simplify notations, we denote  $\mathbf{Y} = \Gamma_{\Theta}(\mathbf{X}, \mathbf{Z}; \mathcal{G})$  as applying the  $\gamma$  aggregator for all nodes in  $\mathcal{G}$ , i.e.,  $\mathbf{y}_i = \gamma_{\Theta}(\mathbf{x}, \mathbf{z}_{\mathcal{N}_i})$ . Based on a given graph aggregator  $\Gamma$ , we can construct a GRU-like RNN structure

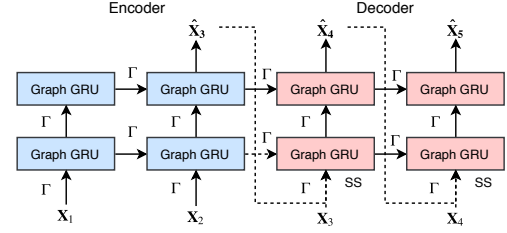


Figure 4: Illustration of the encoder-decoder structure used in the paper. We use two layers of Graph GRUs to predict a length-3 output sequence based on a length-2 input sequence. ‘SS’ denotes the scheduled sampling step.

Table 5: The Dataset used for traffic speed forecasting.

Data	#Nodes	#Edges	#Timestamps
METR-LA	207	1,515	34,272

using the following equations:

$$\begin{aligned}
 \mathbf{U}_t &= \sigma(\Gamma_{\Theta_{xu}}(\mathbf{X}_t, \mathbf{X}_t; \mathcal{G}) + \Gamma_{\Theta_{hu}}(\mathbf{X}_t \oplus \mathbf{H}_{t-1}, \mathbf{H}_{t-1}; \mathcal{G})), \\
 \mathbf{R}_t &= \sigma(\Gamma_{\Theta_{xr}}(\mathbf{X}_t, \mathbf{X}_t; \mathcal{G}) + \Gamma_{\Theta_{hr}}(\mathbf{X}_t \oplus \mathbf{H}_{t-1}, \mathbf{H}_{t-1}; \mathcal{G})), \\
 \mathbf{H}'_t &= h(\Gamma_{\Theta_{xh}}(\mathbf{X}_t, \mathbf{X}_t; \mathcal{G}) + \mathbf{R}_t \circ \Gamma_{\Theta_{hh}}(\mathbf{X}_t \oplus \mathbf{H}_{t-1}, \mathbf{H}_{t-1}; \mathcal{G})), \\
 \mathbf{H}_t &= (1 - \mathbf{U}_t) \circ \mathbf{H}'_t + \mathbf{U}_t \circ \mathbf{H}_{t-1}.
 \end{aligned} \tag{6}$$

Here,  $\mathbf{X}_t \in \mathbb{R}^{|\mathcal{V}| \times d_i}$  are the input features and  $\mathbf{H}_t \in \mathbb{R}^{|\mathcal{V}| \times d_o}$  are the hidden states of the nodes at the  $t$ th timestamp.  $|\mathcal{V}|$  is the total number of nodes,  $d_i$  is the dimension of the input, and  $d_o$  is the dimension of the state.  $\mathbf{U}_t$  and  $\mathbf{R}_t$  are the update gate and reset gate that controls how  $\mathbf{H}_t$  is calculated.  $\mathcal{G}$  is the graph that defines the connection structure between all the nodes.

We refer to this RNN structure as *Graph GRU* (GGRU). GGRU can be used as the basic building block for RNN encoder-decoder structure (Lin et al., 2017) to predict the future  $K$  steps of traffic speeds, i.e.,  $\hat{\mathbf{X}}_{J+1}, \hat{\mathbf{X}}_{J+2}, \dots, \hat{\mathbf{X}}_{J+K}$ , based on the previous  $J$  steps of observed traffic speeds, i.e.,  $\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_J$ . In the decoder, we use the scheduled sampling (Bengio et al., 2015) technique described in (Lin et al., 2017). Figure 4 illustrates our encoder-decoder structure. When attention-based aggregators are used, i.e., the multi-head attention aggregator or our GaAN aggregator, the connection structure in the recurrent step will also be learned based on the attention process. This can be viewed as an extension of *Trajectory GRU* (TrajGRU) (Shi et al., 2017) on irregular or graph-structured data.

### 6.2 EXPERIMENTAL SETUP

To evaluate the proposed GGRU model on traffic speed forecasting, we use the METR-LA dataset from (Li et al.,

Table 6: Performance comparison of different models for traffic speed forecasting on the METR-LA dataset. Models marked with ‘†’ treat sensor map as a directed graph while other models convert it into an undirected graph. Scores under “ $\tau$ min” are the scores at the  $\frac{\tau}{5}$ th predicted frame. The last three columns contain the average scores of the 15 min, 30 min, and 60 min forecasting horizons.

Models / T	15 min			30 min			60 min			Average		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
FC-LSTM (Li et al., 2018)	3.44	6.30	9.6%	3.77	7.23	10.9%	4.37	8.69	13.2%	3.86	7.41	11.2%
GCRNN (Li et al., 2018)	2.80	5.51	7.5%	3.24	6.74	9.0%	3.81	8.16	10.9%	3.28	6.80	9.13%
DCRNN† (Li et al., 2018)	2.77	5.38	7.3%	3.15	6.45	8.8%	<b>3.60</b>	<b>7.60</b>	<b>10.5%</b>	3.17	6.48	8.87%
Avg Pool	2.79	5.42	7.26%	3.20	6.52	8.84%	3.69	7.69	10.73%	3.22	6.54	8.94%
Max Pool	2.77	5.36	7.21%	3.18	6.45	8.78%	3.69	7.73	10.80%	3.21	6.51	8.93%
Pairwise + Sigmoid	2.76	5.36	7.14%	3.18	6.46	8.72%	3.70	7.73	10.77%	3.22	6.52	8.88%
Pairwise + Tanh	2.76	5.34	7.14%	3.18	6.46	8.73%	3.70	7.73	10.73%	3.21	6.51	8.87%
Attention-only	2.74	5.33	7.09%	3.16	6.45	8.69%	3.67	7.61	10.77%	3.19	6.49	8.85%
GaAN	<b>2.71</b>	<b>5.24</b>	<b>6.99%</b>	<b>3.12</b>	<b>6.36</b>	<b>8.56%</b>	3.64	7.65	10.62%	<b>3.16</b>	<b>6.41</b>	<b>8.72%</b>

2018). The nodes in the dataset represent sensors measuring traffic speed and edges denote proximity between sensor pairs measured by road network distance. The sensor speeds are recorded every five minutes. Complete dataset statistics are given in Table 5. We follow (Li et al., 2018)’s way to split the dataset. The first 70% of the sequences are used for training, the middle 10% are used for validation, and the final 20% are used for testing. We also use the same evaluation metrics as in (Li et al., 2018) for evaluation, including *Mean Absolute Error* (MAE), *Root Mean Squared Error* (RMSE), and *Mean Absolute Percentage Error* (MAPE). A sequence of length 12 is used as the input to predict the future traffic speed in one hour (12 steps).

### 6.3 MAIN RESULTS

We compare six variations of the proposed GGRU architecture with three baseline models, including fully-connected LSTM, GCRNN, and DCRNN (Li et al., 2018). We use the same set of six aggregators as in the inductive node classification experiment to construct the GGRU and we use two layers of GGRUs with the state dimension of 64 both in the encoder and the decoder. For attention based models, we set  $K = 4$ ,  $d_a = 16$ , and  $d_v = 16$ . For GaAN, we set  $d_m = 64$  and only use max pooling in the gate-generation network. For pooling based aggregators, we set  $d_v = 128$ . For pairwise sum aggregators, we set  $K = 4$ ,  $d_a = 32$ , and  $d_v = 16$ .

Since the road map is directed and our model does not deal with edge information, we first convert the road map into an undirected graph and use it as the  $\mathcal{G}$  in Eqn. (6). All models are trained by minimizing MAE loss with Adam optimizer. The initial learning rate is set to 0.001 and the batch-size is 64. We use the same scheduled sam-

pling strategy as in (Li et al., 2018). Table 1 shows the comparison of different approaches for 15 minutes, 30 minutes and 1 hour ahead forecasting on both datasets.

The scores for 15 minutes, 30 minutes, and 1 hour ahead forecasting as well as the average scores over three forecasting horizons are shown in Table 6. For the average score, we can see that the proposed GGRU models consistently give better results than GCRNN, which models the traffic network as an undirected graph. Moreover, the GaAN based GGRU model, which does not use edge information, achieves higher accuracy than DCRNN, which uses edge information in the road network.

## 7 CONCLUSION AND FUTURE WORK

We introduced the GaAN model and applied it to two challenging tasks: inductive node classification and traffic speed forecasting. GaAN beats previous state-of-the-art algorithms in both cases. In the future, we plan to extend GaAN by integrating edge features and processing massive graphs with millions or even billions of nodes. Moreover, our model is not restricted to graph learning. A particularly exciting direction for future work is to apply GaAN to natural language processing tasks like machine translation.

## 8 ACKNOWLEDGEMENT

The work described in this paper was partially supported by the Research Grants Council of the Hong Kong Special Administrative Region, China (No. CUHK 14208815 and No. HKUST 16207316 of the General Research Fund) and 2016 MOE of China (Project No. D.01.16.00101).

## References

- J. Atwood and D. Towsley. Diffusion-convolutional neural networks. In *NIPS*, pages 1993–2001, 2016.
- D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.
- S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *NIPS*, pages 1171–1179, 2015.
- J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun. Spectral networks and locally connected networks on graphs. In *ICLR*, 2014.
- J. Chen, T. Ma, and C. Xiao. FastGCN: Fast learning with graph convolutional networks via importance sampling. In *ICLR*, 2018.
- J. Cheng, S. Zhao, J. Zhang, I. King, X. Zhang, and H. Wang. Aspect-level sentiment classification with heat (hierarchical attention) network. In *CIKM*, pages 97–106, 2017.
- J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *NIPS, Workshop on Deep Learning and Representation Learning*, 2014.
- M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*, pages 3844–3852, 2016.
- D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams. Convolutional networks on graphs for learning molecular fingerprints. In *NIPS*, pages 2224–2232, 2015.
- A. Fout, J. Byrd, B. Shariat, and A. Ben-Hur. Protein interface prediction using graph convolutional networks. In *NIPS*, pages 6533–6542, 2017.
- J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin. Convolutional sequence to sequence learning. In *ICML*, pages 1243–1252, 2017.
- W. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *NIPS*, pages 1025–1035, 2017a.
- W. L. Hamilton, R. Ying, and J. Leskovec. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*, 2017b.
- D. K. Hammond, P. Vandergheynst, and R. Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2): 129–150, 2011.
- S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.
- T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.
- A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012.
- Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel. Gated graph sequence neural networks. In *ICLR*, 2016.
- Y. Li, R. Yu, C. Shahabi, and Y. Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *ICLR*, 2018.
- X. Liang, X. Shen, J. Feng, L. Lin, and S. Yan. Semantic object parsing with graph lstm. In *ECCV*, pages 125–143, 2016.
- Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio. A structured self-attentive sentence embedding. In *ICLR*, 2017.
- F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *CVPR*, pages 5115–5124, 2017.
- K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, and A. Tkatchenko. Quantum-chemical insights from deep tensor neural networks. *Nature communications*, 8:13890, 2017.
- M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi. Bidirectional attention flow for machine comprehension. In *ICLR*, 2017.
- Y. Seo, M. Defferrard, P. Vandergheynst, and X. Bresson. Structured sequence modeling with graph convolutional recurrent networks. *arXiv preprint arXiv:1612.07659*, 2016.
- X. Shi, Z. Gao, L. Lausen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo. Deep learning for precipitation nowcasting: A benchmark and a new model. In *NIPS*, pages 5622–5632, 2017.
- A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, 102(43):15545–15550, 2005.
- A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *NIPS*, pages 6000–6010, 2017.



- P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. In *ICLR*, 2018.
- B. Xu, N. Wang, T. Chen, and M. Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015a.
- K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, pages 2048–2057, 2015b.
- Y. Yuan, X. Liang, X. Wang, D. Yeung, and A. Gupta. Temporal dynamic graph LSTM for action-driven video object detection. In *ICCV*, pages 1819–1828, 2017.
- H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, and A. Agrawal. Context encoding for semantic segmentation. In *CVPR*, 2018.
- J. Zhang, X. Shi, I. King, and D.-Y. Yeung. Dynamic key-value memory networks for knowledge tracing. In *WWW*, pages 765–774, 2017.

---

# Causal Learning for Partially Observed Stochastic Dynamical Systems

---

**Søren Wengel Mogensen**

Department of Mathematical Sciences  
University of Copenhagen  
Copenhagen, Denmark

**Daniel Malinsky**

Department of Computer Science  
Johns Hopkins University  
Baltimore, MD, USA

**Niels Richard Hansen**

Department of Mathematical Sciences  
University of Copenhagen  
Copenhagen, Denmark

## Abstract

Many models of dynamical systems have causal interpretations that support reasoning about the consequences of interventions, suitably defined. Furthermore, local independence has been suggested as a useful independence concept for stochastic dynamical systems. There is, however, no well-developed theoretical framework for causal learning based on this notion of independence. We study independence models induced by directed graphs (DGs) and provide abstract graphoid properties that guarantee that an independence model has the global Markov property w.r.t. a DG. We apply these results to Itô diffusions and event processes. For a partially observed system, directed mixed graphs (DMGs) represent the marginalized local independence model, and we develop, under a faithfulness assumption, a sound and complete learning algorithm of the directed mixed equivalence graph (DMEG) as a summary of all Markov equivalent DMGs.

## 1 INTRODUCTION

Causal learning has been developed extensively using structural causal models and graphical representations of the conditional independence relations that they induce. The Fast Causal Inference (FCI) algorithm and its variations (RFCI, FCI+, ...) can learn a representation of the independence relations induced by a causal model even when the causal system is only partially observed, i.e., the data is “causally insufficient” in the terminology of Spirtes et al. (2000). FCI is, however, not directly applicable for learning causal relations among entire processes in a continuous-time dynamical system. The dy-

namic evolution of such a system cannot be modeled using a finite number of variables related via a structural causal model, and standard probabilistic independence cannot adequately capture infinitesimal conditional independence relationships between processes since such relationships can be asymmetric. The asymmetry can intuitively be explained by the fact that the present of one process may be independent of the past of another process, or the reverse, or both.

Local independence was introduced by Schweder (1970) and is a formalization of how the present of one stochastic process depends on the past of others in a dynamical system. This concept directly lends itself to a causal interpretation as dynamical systems develop as functions of their pasts, see e.g. Aalen (1987). Didelez (2000, 2006a, 2008) considered graphical representations of local independence models using directed graphs (DGs) and  $\delta$ -separation and proved the equivalence of the pairwise and global Markov properties in the case of multivariate counting processes. Nodelman et al. (2002, 2003) and Gunawardana et al. (2011) also considered learning problems in continuous-time models. In this paper, we extend the theory to a broader class of semimartingales, showing the equivalence of pairwise and global Markov properties in DGs. To represent marginalized local independence models, Mogensen and Hansen (2018) introduced directed mixed graphs (DMGs) with  $\mu$ -separation. Bidirected edges in DMGs (roughly) correspond to dependencies induced by latent processes, and in this sense DMGs can represent partially observed dynamical systems. In contrast to the “causally sufficient” setting as represented by a DG, multiple DMGs may represent the same set of (marginal) local independence relations; thus we use the characterization of Markov equivalent DMGs by Mogensen and Hansen (2018) to propose a sound and complete algorithm for selecting a set of DMGs consistent with a given collection of independence relations.

Proofs omitted from the main text can be found in the supplementary material.

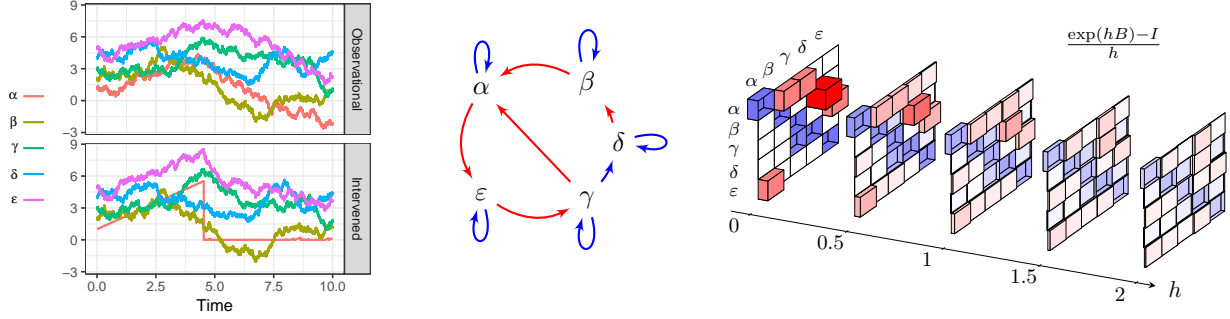


Figure 1: Simulated sample paths (left) for the linear SDE determined by  $B$  in (1). The sample paths are from the observational distribution started in the stationary mean as well as under an intervention regime on  $\alpha$ . For the local independence graph (middle) the color of the edge  $j \rightarrow i$  indicates if the nonzero entry  $B_{ij}$  is positive (red) or negative (blue). The step size  $h$  difference quotient at 0 for the semigroup  $t \mapsto \exp(tB)$  (right) determines the discrete time conditional means for time step  $h$  transitions. It does not directly reflect the local independences except in the limit  $h \rightarrow 0$ , where it converges to the infinitesimal generator  $B$ . Danks and Plis (2013) make a similar point in the case of subsampled time series.

## 2 CAUSAL DYNAMICAL MODELS

The notion of interventions in a continuous-time model of a dynamical system is not new, and has been investigated thoroughly in the context of control theory. Causal models and interventions for event processes and their relation to graphical independence models have been treated in detail (Didelez, 2008, 2015). Relations to structural causal models have been established for ordinary differential equations (ODEs) (Mooij et al., 2013; Rubenstein et al., 2016). Notions of causality and interventions have also been treated for general stochastic processes such as stochastic differential equations (SDEs) (Aalen et al., 2012; Commenges and Gégout-Petit, 2009; Sokol and Hansen, 2014).

To motivate and explain the general results of this paper, we introduce the toy linear SDE model in  $\mathbb{R}^5$  given by  $dX_t = B(X_t - A)dt + dW_t$  with  $A = (1, 2, 3, 4, 5)^T$ ,

$$B = \begin{pmatrix} -1.1 & 1 & 1 & \cdot & \cdot \\ \cdot & -1.1 & \cdot & 2.0 & \cdot \\ \cdot & \cdot & -1.1 & \cdot & 1 \\ \cdot & \cdot & -1 & -1.1 & \cdot \\ 1 & \cdot & \cdot & \cdot & -1.1 \end{pmatrix}, \quad (1)$$

and  $(W_t)$  a five-dimensional standard Brownian motion. The coordinates of this process will be denoted  $\alpha, \beta, \gamma, \delta$ , and  $\epsilon$ . If we assume that this SDE has a causal interpretation, we can obtain predictions under interventions via manipulations of the SDE itself, see e.g. Sokol and Hansen (2014). In Figure 1, for instance, we replace the  $\alpha$  coordinate of the SDE by

$$dX_t^\alpha = 1(X_t^\beta > 1)dt, \quad X_t^\alpha - X_{t-}^\alpha = -X_{t-}^\alpha 1(X_t^\beta \leq 1).$$

The nonzero pattern of the  $B$  matrix defines a directed

graph which we identify as the *local independence graph* below, which in turn is related to the local independence model of the SDE. It is a main result of this paper that the local independence model satisfies the global Markov property w.r.t. this graph. Under a faithfulness assumption we can identify (aspects of) the causal system from observational data even when some processes are unobserved.

It is well known that

$$X_{t+h} - X_t \mid X_t \sim \mathcal{N}((e^{hB} - I)(X_t - A), \Sigma(h))$$

with  $\Sigma(h)$  given in terms of  $B$ . Thus a sample of the process at equidistant time points is a vector autoregressive process with correlated errors. We note that  $e^{hB} - I$  is a dense matrix that will not reveal the local independence graph unless  $h$  is sufficiently small, see Figure 1. The matrix  $B$  is, furthermore, a stable matrix, hence there is a stationary solution to the SDE and for  $h \rightarrow \infty$  we have  $\Sigma(h) \rightarrow \Sigma$ , the invariant covariance matrix. We note that  $\Sigma^{-1}$  is also a dense matrix, thus the invariant distribution does not satisfy the global Markov property w.r.t. to any undirected graph but the complete graph.

In conclusion, the local independence model of the SDE is not encoded directly neither by Markov properties of discrete time samples, nor by Markov properties of the invariant distribution. This is the motivation for our abstract development of local independence models, their relation to continuous-time stochastic processes, and a dedicated learning algorithm.

### 3 INDEPENDENCE MODELS

Consider some finite set  $V$ . An *independence model* over  $V$  is a set of triples  $\langle A, B \mid C \rangle$  such that  $A, B, C \subseteq V$ . We let  $\mathcal{I}$  denote a generic independence model. Following Didelez (2000, 2008) we will consider independence models that are not assumed to be symmetric in  $A$  and  $B$ . The independence models we consider do however satisfy other properties which allow us to deduce some independences from others. We define the following properties, some of which have previously been described as *asymmetric (semi)graphoid properties* (Didelez, 2006b, 2008). Many of them are analogous to properties in the literature on conditional independence models (Lauritzen, 1996), though due to the lack of symmetry, one may define both left and right versions.

- Left redundancy:  $\langle A, B \mid A \rangle \in \mathcal{I}$
- Left decomposition:  
 $\langle A, B \mid C \rangle \in \mathcal{I}, D \subseteq A \Rightarrow \langle D, B \mid C \rangle \in \mathcal{I}$
- Right decomposition:  
 $\langle A, B \mid C \rangle \in \mathcal{I}, D \subseteq B \Rightarrow \langle A, D \mid C \rangle \in \mathcal{I}$
- Left weak union:  
 $\langle A, B \mid C \rangle \in \mathcal{I}, D \subseteq A \Rightarrow \langle A, B \mid C \cup D \rangle \in \mathcal{I}$
- Right weak union:  
 $\langle A, B \mid C \rangle \in \mathcal{I}, D \subseteq B \Rightarrow \langle A, B \mid C \cup D \rangle \in \mathcal{I}$
- Left intersection:  
 $\langle A, B \mid C \rangle \in \mathcal{I}, \langle C, B \mid A \rangle \in \mathcal{I} \Rightarrow$   
 $\langle A \cup C, B \mid A \cap C \rangle \in \mathcal{I}$
- Left composition:  
 $\langle A, B \mid C \rangle \in \mathcal{I}, \langle D, B \mid C \rangle \in \mathcal{I} \Rightarrow$   
 $\langle A \cup D, B \mid C \rangle \in \mathcal{I}$
- Right composition:  
 $\langle A, B \mid C \rangle \in \mathcal{I}, \langle A, D \mid C \rangle \in \mathcal{I} \Rightarrow$   
 $\langle A, B \cup D \mid C \rangle \in \mathcal{I}$
- Left weak composition:  
 $\langle A, B \mid C \rangle \in \mathcal{I}, D \subseteq C \Rightarrow \langle A \cup D, B \mid C \rangle \in \mathcal{I}$

For disjoint sets  $A, C, D \subseteq V$ , we say that  $A$  and  $D$  *factorize* w.r.t.  $C$  if there exists a partition  $C = C_1 \dot{\cup} C_2$  such that (i) and (ii) hold:

- (i)  $\langle A, C_1 \cup D \mid C \cup D \rangle \in \mathcal{I}$
- (ii)  $\langle D, C_2 \cup A \mid C \cup A \rangle \in \mathcal{I}$ .

**Definition 1.** The independence model  $\mathcal{I}$  satisfies *cancellation* if  $\langle A, B \mid C \cup \{\delta\} \rangle \in \mathcal{I}$  implies  $\langle A, B \mid C \rangle \in \mathcal{I}$  whenever  $A$  and  $\{\delta\}$  factorize w.r.t.  $C$ . Such an independence model is called *cancellative*.

Cancellation is related to ordered downward-stability as defined by Sadeghi (2017) for symmetric independence models over a set with a preorder and studied in relation to separation in acyclic graphs.

### 3.1 DIRECTED MIXED GRAPHS

We wish to relate a local independence model, as defined in Section 4, to a graph and therefore we need a notion of graphical separation which allows for asymmetry. *Directed mixed graphs* along with  $\mu$ -separation will provide the means for such graphical modeling of local independence. The subsequent definitions follow Mogensen and Hansen (2018), which we refer to for further details.

**Definition 2** (Directed mixed graph). A *directed mixed graph* (DMG) is an ordered pair  $(V, E)$  where  $V$  is a finite set of vertices (also called nodes) and  $E$  is a finite set of edges of the types  $\rightarrow$  and  $\leftrightarrow$ . A pair of vertices  $\alpha, \beta \in V$  may be joined by any subset of  $\{\alpha \rightarrow \beta, \alpha \leftarrow \beta, \alpha \leftrightarrow \beta\}$ . Note that we allow for loops, i.e., edges  $\alpha \rightarrow \alpha$  and/or  $\alpha \leftrightarrow \alpha$ .

Let  $\mathcal{G}_1 = (V, E_1)$  and  $\mathcal{G}_2 = (V, E_2)$  be DMGs. If  $E_1 \subseteq E_2$ , then we write  $\mathcal{G}_1 \subseteq \mathcal{G}_2$  and say that  $\mathcal{G}_2$  is a *supergraph* of  $\mathcal{G}_1$ . The *complete* DMG on  $V$  is the DMG which is a supergraph of all other DMGs with vertices  $V$ . Throughout this paper,  $\mathcal{G}$  will denote a DMG with node set  $V$  and edge set  $E$ . We will also consider *directed graphs* (DGs) which are DMGs with no bidirected edges. Let  $\alpha, \beta \in V$ . We will say that the edge  $\alpha \rightarrow \beta$  has a *head* at  $\beta$  and a *tail* at  $\alpha$ , and that the edge  $\alpha \leftrightarrow \beta$  has heads at both  $\alpha$  and  $\beta$ . When we write e.g.  $\alpha \rightarrow \beta$  this does not preclude other edges between these nodes. We use  $\alpha * \rightarrow \beta$  to denote any edge between  $\alpha$  and  $\beta$  that has a head at  $\beta$ . A letter over an edge, e.g.  $\alpha \xrightarrow{e} \beta$ , denotes simply that  $e$  refers to that specific edge. If the edge  $\alpha \rightarrow \beta$  is in the graph then we say that  $\alpha$  is a *parent* of  $\beta$  and if  $\alpha \leftrightarrow \beta$  then we say that  $\alpha$  and  $\beta$  are *siblings*. Let  $\text{pa}(\alpha)$  (or  $\text{pa}_{\mathcal{G}}(\alpha)$  to make the graph explicit) denote the set of parents of  $\alpha$  in  $\mathcal{G}$ . Note that due to loops,  $\alpha$  can be both a parent and a sibling of itself.

A *walk* is an alternating, ordered sequence of nodes and edges along with an orientation of the edge such that each edge is between its two adjacent nodes,  $\langle \nu_1, e_1, \nu_2, \dots, e_n, \nu_{n+1} \rangle$ , where  $\nu_i \in V$  and  $e_j \in E$ . We say that the walk is between  $\nu_1$  and  $\nu_{n+1}$  or from  $\nu_1$  to  $\nu_{n+1}$ . The  $\nu_1$  and  $\nu_{n+1}$  are called the *endpoint nodes* of the walk. A non-endpoint node  $\nu_i, i \neq 1, n+1$ , is called a *collider* if the two adjacent edges on the walk both have heads at the node, and otherwise a *non-collider*. Note that the endpoint nodes are neither colliders nor non-colliders. A walk is called *trivial* if it consists of a single node and no edges. A *path* is a walk where no node is repeated. A path from  $\alpha$  to  $\beta$  is *directed* if every edge on the path is directed and points towards  $\beta$ . We say that  $\alpha$  is an ancestor of a set  $C \subseteq V$  if there exists a (possibly trivial) directed path from  $\alpha$  to  $\gamma \in C$ . We let  $\text{an}(C)$  denote the set of nodes that are ancestors to  $C$ . Note that  $C \subseteq \text{an}(C)$ .

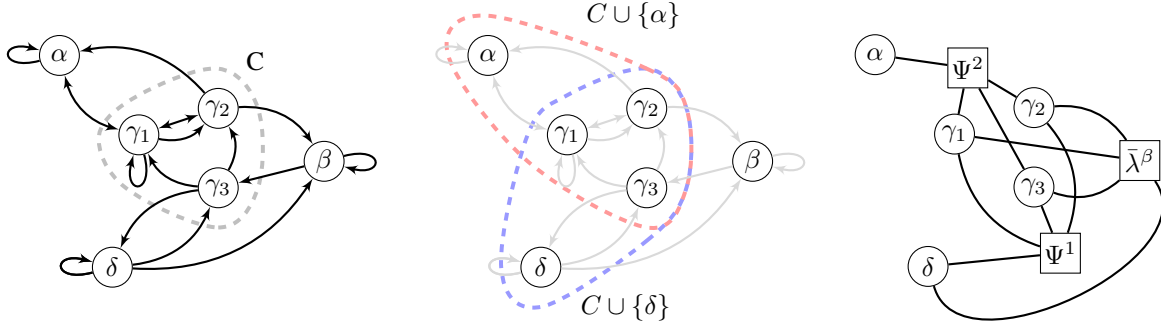


Figure 2: A DMG  $\mathcal{G}$  (left) with sets  $\{\alpha\}$  and  $\{\delta\}$  that factorize w.r.t.  $C = \{\gamma_1, \gamma_2, \gamma_3\}$  such that  $\alpha \perp_{\mu} \beta \mid C \cup \{\delta\}$ . Any node is  $\mu$ -separated from either  $\alpha$  by  $C \cup \{\delta\}$  or  $\delta$  by  $C \cup \{\alpha\}$  (middle), and as  $\mathcal{I}(\mathcal{G})$  is cancellative,  $\alpha \perp_{\mu} \beta \mid C$ . A corresponding factor graph (right) with the three factor nodes  $\Psi^1$ ,  $\Psi^2$  and  $\bar{\lambda}^{\beta}$ , cf. Theorem 14.

### 3.1.1 $\mu$ -separation

**Definition 3** ( $\mu$ -connecting walk). A  $\mu$ -connecting walk from  $\alpha$  to  $\beta$  given  $C$  is a non-trivial walk from  $\alpha$  to  $\beta$  such that  $\alpha \notin C$ , every non-collider is not in  $C$  and every collider is in  $\text{an}(C)$ , and such that the final edge has a head at  $\beta$ .

**Definition 4.** Let  $\alpha, \beta \in V, C \subseteq V$ . We say that  $\beta$  is  $\mu$ -separated from  $\alpha$  given  $C$  in the graph  $\mathcal{G}$  if there is no  $\mu$ -connecting walk from  $\alpha$  to  $\beta$  in  $\mathcal{G}$  given  $C$ . For general sets,  $A, B, C \subseteq V$ , we say that  $B$  is  $\mu$ -separated from  $A$  given  $C$  and write  $A \perp_{\mu} B \mid C$  if  $\beta$  is  $\mu$ -separated from  $\alpha$  given  $C$  for every  $\alpha \in A$  and  $\beta \in B$ . We write  $A \perp_{\mu} B \mid C [\mathcal{G}]$  if we wish to make explicit to which graph the statement applies.

Note that this definition means that  $B$  is separated from  $A$  given  $C$  whenever  $A \subseteq C$ . We associate an independence model  $\mathcal{I}(\mathcal{G})$  with a DMG  $\mathcal{G}$  by

$$\langle A, B \mid C \rangle \in \mathcal{I}(\mathcal{G}) \Leftrightarrow A \perp_{\mu} B \mid C [\mathcal{G}].$$

**Lemma 5.** The independence model  $\mathcal{I}(\mathcal{G})$  satisfies left and right {decomposition, weak union, composition} and left {redundancy, intersection, weak composition}. Furthermore,  $\langle A, B \mid C \rangle \in \mathcal{I}(\mathcal{G})$  whenever  $B = \emptyset$ .

**Lemma 6.**  $\mathcal{I}(\mathcal{G})$  satisfies cancellation.

### 3.1.2 Markov equivalence

We say that DMGs  $\mathcal{G}_1 = (V, E_1)$ ,  $\mathcal{G}_2 = (V, E_2)$  are *Markov equivalent* if  $\mathcal{I}(\mathcal{G}_1) = \mathcal{I}(\mathcal{G}_2)$  and this defines an equivalence relation. We let  $[\mathcal{G}]$  denote the (Markov) equivalence class of  $\mathcal{G}$ . For DMGs, it does not hold that Markov equivalent graphs have the same adjacencies. Note that the same is true for the directed (cyclic) graphs with no loops considered by Richardson (1996,

1997) in another context. We say that a DMG is *maximal* if it is complete or if no edge can be added without changing the associated Markov equivalence class. Mogensen and Hansen (2018) define for every vertex in a DMG a set of *potential parents* and *potential siblings* (both subsets of  $V$ ) using the independence model induced by the graph (these definitions are also included in the supplementary material). We let  $\text{pp}(\alpha, \mathcal{I})$  denote the set of potential parents of  $\alpha$  and  $\text{ps}(\alpha, \mathcal{I})$  denote the set of potential siblings of  $\alpha$  in the independence model  $\mathcal{I}$ . If  $\mathcal{G}_1$  and  $\mathcal{G}_2$  are Markov equivalent we thus have  $\text{pp}(\alpha, \mathcal{I}(\mathcal{G}_1)) = \text{pp}(\alpha, \mathcal{I}(\mathcal{G}_2))$  and  $\text{ps}(\alpha, \mathcal{I}(\mathcal{G}_1)) = \text{ps}(\alpha, \mathcal{I}(\mathcal{G}_2))$  for each  $\alpha \in V$ . Given a DMG  $\mathcal{G}$  and independence model  $\mathcal{I} = \mathcal{I}(\mathcal{G})$ , one can construct another DMG  $\mathcal{N}$  in which  $\alpha$  is a parent of  $\beta$  if and only if  $\alpha \in \text{pp}(\beta, \mathcal{I})$  and  $\alpha$  and  $\beta$  are siblings if and only if  $\alpha \in \text{ps}(\beta, \mathcal{I})$ . Mogensen and Hansen (2018) showed that  $\mathcal{N} \in [\mathcal{G}]$ , that it is a supergraph of all elements of  $[\mathcal{G}]$ , and that  $\mathcal{N}$  is maximal. This allows one to define a *directed mixed equivalence graph* (DMEG) from the (unique) maximal graph  $\mathcal{N}$  in the equivalence class to summarize the entire equivalence class. The DMEG is constructed from  $\mathcal{N}$  by partitioning the edge set into two subsets: one consisting of the edges which are common to all graphs in the Markov equivalence class, and one consisting of edges that are present in some members of the equivalence class but absent in others. One may visualize the DMEG by drawing  $\mathcal{N}$  and making the edges in the latter set dashed. Note that by collapsing the distinction between dashed and solid edges one may straightforwardly apply  $\mu$ -separation to a given DMEG.

## 3.2 MARKOV PROPERTIES

The main result of this section gives conditions on an abstract independence model ensuring equivalence be-

tween the *pairwise* and the *global Markov properties* w.r.t. a directed graph with  $\mu$ -separation. In the next section we give examples of classes of processes that fulfill these conditions, extending results in Didelez (2008) to a broader class of models. We take an axiomatic approach to proving the equivalence in the sense that we describe some abstract properties and use only these to show the equivalence. This is analogous to what Lauritzen and Sadeghi (2017) did in the case of symmetric independence models.

**Definition 7.** A DG and an independence model satisfy the pairwise Markov property if for  $\alpha, \beta \in V$ ,

$$\alpha \notin \text{pa}(\beta) \Rightarrow \langle \alpha, \beta \mid V \setminus \{\alpha\} \rangle \in \mathcal{I}$$

A DMG and an independence model satisfy the global Markov property if for  $A, B, C \subseteq V$ ,

$$A \perp_{\mu} B \mid C \Rightarrow \langle A, B \mid C \rangle \in \mathcal{I}.$$

**Theorem 8.** Assume that  $\mathcal{I}$  is an independence model that satisfies left {redundancy, intersection, decomposition, weak union, weak composition}, right {decomposition, composition}, is cancellative, and furthermore  $\langle A, B \mid C \rangle \in \mathcal{I}$  whenever  $B = \emptyset$ . Let  $\mathcal{D}$  be a DG. Then  $\mathcal{I}$  satisfies the pairwise Markov property with respect to  $\mathcal{D}$  if and only if it satisfies the global Markov property with respect to  $\mathcal{D}$ .

To keep consistency with earlier literature, we define the pairwise Markov condition above as the absence of an edge, which does not directly generalize to DMGs. Therefore, we prove the equivalence of pairwise and global Markov only in the class of DGs. The main purpose of DMGs is to represent Markov properties from marginalized DGs as defined below, in which case the global Markov property w.r.t. a DMG is inherited from the DG.

**Definition 9** (Marginal independence model). Assume that  $\mathcal{I}$  is an independence model over  $V$ . Then the marginal independence model of  $\mathcal{I}$  over  $O \subseteq V$ ,  $\mathcal{I}^O$ , is the independence model,

$$\mathcal{I}^O = \{ \langle A, B \mid C \rangle \mid \langle A, B \mid C \rangle \in \mathcal{I}; A, B, C \subseteq O \}.$$

Mogensen and Hansen (2018) give a marginalization algorithm (a.k.a. a “latent projection”), which outputs a marginal DMG,  $\mathcal{G} = (O, F)$ , from a DG,  $\mathcal{D} = (V, E)$ , such that  $\mathcal{I}(\mathcal{D})^O = \mathcal{I}(\mathcal{G})$ . If  $\mathcal{I}$  satisfies the global Markov property w.r.t.  $\mathcal{D}$  then

$$\mathcal{I}(\mathcal{G}) = \mathcal{I}(\mathcal{D})^O \subseteq \mathcal{I}^O.$$

This shows that the marginalized independence model  $\mathcal{I}^O$  then satisfies the global Markov property w.r.t. the DMG  $\mathcal{G}$ .

## 4 LOCAL INDEPENDENCE

This section introduces local independence models and local independence graphs. The main results of the section provide verifiable conditions that ensure that a local independence model satisfies the global Markov property w.r.t. the local independence graph.

Let  $X = (X_t^1, \dots, X_t^n)$  for  $t \in [0, T]$  be a càdlàg stochastic process defined on the probability space  $(\Omega, \mathcal{F}, P)$ . Introduce for  $A \subseteq V = \{1, \dots, n\}$  the filtration  $\mathcal{F}_t^A$  as the completed and right continuous version of  $\sigma(\{X_s^\alpha, s \leq t, \alpha \in A\})$ . Let also  $\lambda = (\lambda_t^1, \dots, \lambda_t^n)$  be an integrable càdlàg stochastic process. This  $\lambda$ -process need not have any specific relation to  $X$  *a priori*, but for the main Theorem 14 the relation is through the compatibility processes defined below. Note that some computations below technically require that  $E(\cdot \mid \mathcal{F}_t)$  is computed as the optional projection, cf. Theorem VI.7.1 and Lemma VI.7.8 in Rogers and Williams (2000). This is unproblematic, and will not be discussed any further.

**Definition 10.** We say that  $B$  is  $\lambda$ -locally independent of  $A$  given  $C$  if the process

$$t \mapsto E(\lambda_t^\beta \mid \mathcal{F}_t^{A \cup C})$$

has an  $\mathcal{F}_t^C$ -adapted version for all  $\beta \in B$ . In this case we write  $A \not\rightarrow_{\lambda} B \mid C$ .

This is slightly different from the definition in Didelez (2008) in that  $\beta$  is not necessarily in the conditioning set. This change in the definition makes it possible for a process to be locally independent from itself given some separating set. We define the local independence model,  $\mathcal{I}(X, \lambda)$ , determined by  $X$  and  $\lambda$  via

$$\langle A, B \mid C \rangle \in \mathcal{I}(X, \lambda) \Leftrightarrow A \not\rightarrow_{\lambda} B \mid C.$$

When there is no risk of ambiguity we say that  $B$  is locally independent of  $A$  given  $C$ , and we write  $A \not\rightarrow B \mid C$  and  $\mathcal{I} = \mathcal{I}(X, \lambda)$ .

The local independence model satisfies a number of the properties listed in Section 3.

**Lemma 11.** Let  $\mathcal{I}$  be a local independence model. Then it satisfies left {redundancy, decomposition, weak union, weak composition} and right {decomposition, composition} and furthermore  $\langle A, B \mid C \rangle \in \mathcal{I}$  whenever  $B = \emptyset$ . If  $\mathcal{F}_t^A \cap \mathcal{F}_t^C = \mathcal{F}_t^{A \cap C}$  holds for all  $A, C \subseteq V$  and  $t \in [0, T]$ , then left intersection holds.

**Definition 12.** The local independence graph is the directed graph with node set  $V = \{1, \dots, n\}$  such that

$$\alpha \notin \text{pa}(\beta) \Leftrightarrow \alpha \not\rightarrow_{\lambda} \beta \mid V \setminus \{\alpha\}.$$

By Theorem 8 and Lemma 11 a local independence model that satisfies left intersection and is cancellative satisfies the global Markov property w.r.t. the local independence graph. Left intersection holds by Lemma 11 whenever  $\mathcal{F}_t^A \cap \mathcal{F}_t^C = \mathcal{F}_t^{A \cap C}$ . Theorem 14 below gives a general factorization condition on the distribution of the stochastic processes that ensures a local independence model to be cancellative. This condition is satisfied for example by event and Itô processes.

Introduce for  $C \subseteq V$  and  $\beta \in V$  the shorthand notation

$$\lambda_t^{C,\beta} = E(\lambda_t^\beta | \mathcal{F}_t^C).$$

Furthermore, for  $\alpha \in A \subseteq V$  let

$$\Psi_t^{A,\alpha} = \psi_t^\alpha((\lambda_s^{A,\alpha})_{s \leq t}, (X_s^\alpha)_{s \leq t})$$

denote a càdlàg process that is given in terms of a positive functional  $\psi_t^\alpha$  of the history of the  $\lambda^{A,\alpha}$ - and the  $X^\alpha$ -processes up to time  $t$ .

**Definition 13.** We say that  $P$   $\lambda$ -factorizes with compatibility processes  $\Psi^{A,\alpha} > 0$  if for all  $A \subseteq V$

$$P = \frac{1}{Z_t^A} \prod_{\alpha \in A} \Psi_t^{A,\alpha} \cdot Q_t^A$$

with  $Q_t^A$  a probability measure on  $(\Omega, \mathcal{F})$  such that  $(X_s^\alpha)_{0 \leq s \leq t}$  for  $\alpha \in A$  are independent under  $Q_t^A$ . Here,  $Z_t^A$  is a deterministic normalization constant.

**Theorem 14.** The local independence model  $\mathcal{I}(X, \lambda)$  is cancellative if  $P$   $\lambda$ -factorizes.

*Proof.* Assume that  $A, \{\delta\} \subseteq V$  factorize w.r.t.  $C = C_1 \dot{\cup} C_2$ . In this proof, (i) and (ii) refer to the factorization properties, see Definition 1. Let  $F = C \cup A \cup \{\delta\}$ . Then by (i)

$$\Psi_t^{F,\gamma} = \psi_t^\gamma((\lambda_s^{C \cup \{\delta\}, \gamma})_{s \leq t}, (X_s^\gamma)_{s \leq t}) = \Psi_t^{C \cup \{\delta\}, \gamma}$$

for  $\gamma \in C_1 \cup \{\delta\}$ , and by (ii)

$$\Psi_t^{F,\gamma} = \psi_t^\gamma((\lambda_s^{C \cup A, \gamma})_{s \leq t}, (X_s^\gamma)_{s \leq t}) = \Psi_t^{C \cup A, \gamma}$$

for  $\gamma \in C_2 \cup A$ .

It follows that

$$\begin{aligned} \prod_{\gamma \in F} \Psi_t^{F,\gamma} &= \overbrace{\prod_{\gamma \in C_1 \cup \{\delta\}} \Psi_t^{C \cup \{\delta\}, \gamma}}^{\Psi_t^1} \overbrace{\prod_{\gamma \in C_2 \cup A} \Psi_t^{C \cup A, \gamma}}^{\Psi_t^2} \\ &= \Psi_t^1 \Psi_t^2, \end{aligned}$$

cf. Figure 2. Note that  $\Psi_t^2$  is  $\mathcal{F}_t^{C \cup A}$ -adapted. Let  $\beta \in B$ . We have  $\langle A, B \mid C \cup \{\delta\} \rangle \in \mathcal{I}$ , hence with  $\bar{\lambda}_t^\beta =$

$$\lambda_t^{C \cup \{\delta\}, \beta}$$

$$\begin{aligned} E(\lambda_t^\beta | \mathcal{F}_t^{C \cup A}) &= E(E(\lambda_t^\beta | \mathcal{F}_t^{C \cup A \cup \{\delta\}}) | \mathcal{F}_t^{C \cup A}) \\ &= E(\bar{\lambda}_t^\beta | \mathcal{F}_t^{C \cup A}) \\ &= \frac{E_{Q_t^F}(\bar{\lambda}_t^\beta \Psi_t^1 \Psi_t^2 | \mathcal{F}_t^{C \cup A})}{E_{Q_t^F}(\Psi_t^1 \Psi_t^2 | \mathcal{F}_t^{C \cup A})} \\ &= \frac{E_{Q_t^F}(\bar{\lambda}_t^\beta \Psi_t^1 | \mathcal{F}_t^{C \cup A})}{E_{Q_t^F}(\Psi_t^1 | \mathcal{F}_t^{C \cup A})} \\ &= \frac{E_{Q_t^F}(\bar{\lambda}_t^\beta \Psi_t^1 | \mathcal{F}_t^C)}{E_{Q_t^F}(\Psi_t^1 | \mathcal{F}_t^C)} \\ &= \lambda_t^{C,\beta} \end{aligned}$$

where the second last identity follows from  $X^\alpha$  for  $\alpha \in A$  being independent of  $X^\gamma$  for  $\gamma \in C \cup \{\delta\}$  under  $Q_t^F$ . We conclude that  $\langle A, B \mid C \rangle \in \mathcal{I}$ , and this shows that  $\mathcal{I}$  is cancellative.  $\square$

## 4.1 ITÔ PROCESSES

For  $X$  a multivariate Itô process with  $X^\alpha$  fulfilling the equation

$$X_t^\alpha = \int_0^t \lambda_s^\alpha ds + \sigma_t(\alpha) W_t^\alpha$$

with  $W_t$  a standard Brownian motion ( $\sigma_t(\alpha) > 0$  deterministic) we introduce the compatibility processes

$$\Psi_t^{A,\alpha} = \exp\left(\int_0^t \frac{\lambda_s^{A,\alpha}}{\sigma_s^2(\alpha)} dX_s^\alpha - \frac{1}{2} \int_0^t \left(\frac{\lambda_s^{A,\alpha}}{\sigma_s(\alpha)}\right)^2 ds\right).$$

The following result is a consequence of Theorem 7.3 in Liptser and Shirayev (1977) combined with Theorem VI.8.4 in Rogers and Williams (2000).

**Proposition 15.** If for all  $A \subseteq V$

$$E\left(\prod_{\alpha \in A} (\Psi_t^{A,\alpha})^{-1}\right) = 1 \quad (2)$$

then  $P$   $\lambda$ -factorizes.

It can be shown that the linear SDE introduced earlier satisfies the integrability condition (2).

## 4.2 EVENT PROCESSES

For  $X$  a multivariate counting process with  $X^\alpha$  having intensity process  $\lambda^\alpha$  we introduce the compatibility processes

$$\Psi_t^{A,\alpha} = \exp\left(\int_0^t \log(\lambda_{s-}^{A,\alpha}) dX_s^\alpha - \int_0^t \lambda_s^{A,\alpha} ds\right).$$

Here  $\lambda_{s-}^{A,\alpha} = \lim_{r \rightarrow s-} \lambda_r^{A,\alpha}$  denotes the left continuous (and thus predictable) version of the intensity process  $\lambda_t^{A,\alpha} = E(\lambda_t^\alpha \mid \mathcal{F}_t^A)$ . With these compatibility processes, Proposition 15 above holds exactly as formulated for Itô processes, see e.g. Sokol and Hansen (2015) for details and weak conditions ensuring that (2) holds.

## 5 LEARNING ALGORITHMS

In this section, we assume that we have access to a local independence oracle that can answer whether or not some independence statement is in  $\mathcal{I}$ . In applications, the oracle would of course be substituted with statistical tests of local independence. The local independence model,  $\mathcal{I}$ , is assumed to be faithful to some DMG  $\mathcal{G}_0$ , i.e.  $\mathcal{I} = \mathcal{I}(\mathcal{G}_0)$ .

Meek (2014) described a related algorithm for learning local independence graphs which is, however, not complete when the system of stochastic processes is only partially observed. In the FCI algorithm, which learns an equivalence class of MAGs (Maximal Ancestral Graphs), one can exploit the fact that Markov equivalent graphs have the same adjacencies, so the learning algorithm can first find this so-called *skeleton* of the graph and then orient the edges by applying a finite set of rules (Zhang, 2008; Ali et al., 2009). Since Markov equivalent DMGs may have different adjacencies, we cannot straightforwardly copy the FCI strategy here, and our procedure is more complicated.

### 5.1 A THREE-STEP PROCEDURE

As described in Section 3.1.2, we know that there exists a unique graph which is Markov equivalent to  $\mathcal{G}_0$  and a supergraph of all DMGs in  $[\mathcal{G}_0]$  and we denote this graph by  $\mathcal{N}$ . In this section we give a learning algorithm exploiting this fact. Having learned the maximal DMG  $\mathcal{N}$  we can subsequently construct a DMEG to summarize the Markov equivalence class.

The characterization of Markov equivalence of DMGs in Mogensen and Hansen (2018) implies a learning algorithm to construct  $\mathcal{N}$  which is Markov equivalent to  $\mathcal{G}_0$ . For each pair of nodes  $\alpha, \beta$  there exists a well-defined list of independence tests such that  $\alpha \rightarrow \beta$  is in  $\mathcal{N}$  if and only if all requirements in the list is met by  $\mathcal{I}(\mathcal{G}_0)$ , analogously for the edge  $\alpha \leftrightarrow \beta$  (see conditions (p1)-(p4) and (s1)-(s3) in the supplementary material). This means that we can use these lists of tests to construct a maximal graph  $\mathcal{N}$  such that  $\mathcal{I}(\mathcal{N}) = \mathcal{I}(\mathcal{G}_0)$ . However such an algorithm would perform many more independence tests than needed and one can reduce the number of independence tests conducted by a kind of preprocessing. Our proposed algorithm starts from the complete DMG

**input** : a local independence oracle for  $\mathcal{I}$

**output**: a DMG,  $\mathcal{G} = (V, E)$

initialize  $\mathcal{G}$  as the complete DMG, set  $n = 0$ , initialize  $\mathcal{L}_s = \emptyset, \mathcal{L}_n = \emptyset$ ;

```

while  $n \leq \max_{\beta \in V} |\text{pa}_{\mathcal{G}}(\beta)|$  do
  foreach  $\alpha \rightarrow \beta \in E$  do
    foreach  $C \subseteq \text{pa}_{\mathcal{G}}(\beta) \setminus \{\alpha\}, |C| = n$  do
      if  $\alpha \not\rightarrow_{\lambda} \beta \mid C$  then
        delete  $\alpha \rightarrow \beta$  and  $\alpha \leftrightarrow \beta$  from  $\mathcal{G}$ ;
        update  $\mathcal{L}_s = \mathcal{L}_s \cup \{\langle \alpha, \beta \mid C \rangle\}$ ;
      else
        update  $\mathcal{L}_n = \mathcal{L}_n \cup \{\langle \alpha, \beta \mid C \rangle\}$ ;
      end
    end
  end
  end
  update  $n = n + 1$ ;
end
set  $n = 1$ ;
while  $n \leq \max_{\alpha, \beta \in V} |D_{\mathcal{G}}(\alpha, \beta)|$  do
  foreach  $\alpha \rightarrow \beta \in E$  do
    foreach  $C \subseteq D_{\mathcal{G}}(\alpha, \beta), |C| = n$  do
      if  $\alpha \not\rightarrow_{\lambda} \beta \mid C$  then
        delete  $\alpha \rightarrow \beta$  and  $\alpha \leftrightarrow \beta$  from  $\mathcal{G}$ ;
        update  $\mathcal{L}_s = \mathcal{L}_s \cup \{\langle \alpha, \beta \mid C \rangle\}$ ;
      else
        update  $\mathcal{L}_n = \mathcal{L}_n \cup \{\langle \alpha, \beta \mid C \rangle\}$ ;
      end
    end
  end
  update  $n = n + 1$ ;
end
end
return  $\mathcal{G}, \mathcal{L}_s, \mathcal{L}_n$ 

```

#### Subalgorithm 1: Separation step

and removes edges that are not in  $\mathcal{G}_0$  by an FCI-like approach, exploiting properties of DMGs and  $\mu$ -separation, and then in the end applies the potential parents and potential siblings definitions (see the supplementary material), but only if and when needed.

In this section we describe three steps (and three subalgorithms): a *separation*, a *pruning*, and a *potential* step, and then we argue that we can construct a sound and complete algorithm by using these steps. For all three steps, we sequentially remove edges starting from the complete DMG on nodes  $V$ . We will also along the way update a set of triples  $\mathcal{L}_s$  corresponding to independence statements that we know to be in  $\mathcal{I}$  and a set of triples  $\mathcal{L}_n$  corresponding to independence statements that we know to not be in  $\mathcal{I}$ . We keep track of this information as we will reuse some of it to reduce the number of independence tests that we conduct. Figure 3 illustrates what



**input** : a separability graph,  $\mathcal{S}$ , a set of known independencies  $\mathcal{L}_s$   
**output**: a DMG  
initialize  $\mathcal{G} = \mathcal{S}$ ;  
**foreach** *unshielded*  $W$ -structure in  $\mathcal{S}$ ,  $w(\alpha, \beta, \gamma)$  **do**  
    **if**  $\beta \in S_{\alpha, \gamma}$  such that  $\langle \alpha, \gamma \mid S_{\alpha, \gamma} \rangle \in \mathcal{L}_s$  **then**  
        **if**  $\beta \leftrightarrow \gamma$  is in  $\mathcal{G}$  **then**  
            delete  $\beta \leftrightarrow \gamma$  from  $\mathcal{G}$ ;  
        **end**  
    **else**  
        **if**  $\beta \rightarrow \gamma$  is in  $\mathcal{G}$  **then**  
            delete  $\beta \rightarrow \gamma$  from  $\mathcal{G}$ ;  
        **end**  
    **end**  
**end**  
**return**  $\mathcal{G}$

**Subalgorithm 2:** Pruning step

each subalgorithm outputs for an example  $\mathcal{G}_0$ .

### 5.1.1 The separation step

When we have an independence model  $\mathcal{I}$  over  $V$ , we will for  $\alpha, \beta \in V$  say that  $\beta$  is *inseparable* from  $\alpha$  if there exists no  $C \subseteq V \setminus \{\alpha\}$  such that  $\langle \alpha, \beta \mid C \rangle \in \mathcal{I}$ . Let

$$u(\beta, \mathcal{I}) = \{\gamma \in V \mid \beta \text{ is inseparable from } \gamma \text{ in } \mathcal{I}\}.$$

The purpose of the first step is to output a *separability graph*. The separability graph of an independence model  $\mathcal{I}$  is the DMG such that the edge  $\alpha \rightarrow \beta$  is in the DMG if and only if  $\alpha \in u(\beta, \mathcal{I})$  and the edge  $\alpha \leftrightarrow \beta$  is in the DMG if and only if  $\alpha \in u(\beta, \mathcal{I})$  and  $\beta \in u(\alpha, \mathcal{I})$ .

We say that  $\gamma$  is *directedly collider connected* to  $\beta$  if there exists a non-trivial walk from  $\gamma$  to  $\beta$  such that every non-endpoint node on the walk is a collider and such that the final edge has a head at  $\beta$ . As shorthand, we write  $\gamma \twoheadrightarrow \beta$ . We define the separator set of  $\beta$  from  $\alpha$ ,

$$D_{\mathcal{G}}(\alpha, \beta) = \{\gamma \in \text{an}(\alpha, \beta) \mid \gamma \twoheadrightarrow \beta\} \setminus \{\alpha\}.$$

If there exists a subset of  $V \setminus \{\alpha\}$  that separates  $\beta$  from  $\alpha$ , then this set does (Mogensen and Hansen, 2018). This set will play a role analogous to that of the set **Possible-D-Sep** in the FCI algorithm (Spirtes et al., 2000).

In the first part of Subalgorithm 1, we consider pairs of nodes,  $\alpha, \beta$ , and test if they can be separated by larger and larger conditioning sets, though only subsets of  $\text{pa}_{\mathcal{G}}(\beta) \setminus \{\alpha\}$  in the current  $\mathcal{G}$ . In the second part, we use all subsets of the current separator set  $D_{\mathcal{G}}(\alpha, \beta)$  to determine separability of each pair of nodes. Note that separability is not symmetric, hence, one needs to determine separability of  $\beta$  from  $\alpha$  and of  $\alpha$  from  $\beta$ . The

**input** : a local independence oracle for  $\mathcal{I}$ , a DMG  $\mathcal{G} = (V, E)$ , a set of known dependencies  $\mathcal{L}_n$   
**output**: a DMG  
**foreach**  $\alpha \xrightarrow{e} \beta \in E$  **do**  
    **if**  $\mathcal{I}(\mathcal{G} - e) \cap \mathcal{L}_n = \emptyset$  **then**  
        **if**  $\alpha \notin \text{pp}(\beta, \mathcal{I})$  **then**  
            delete  $\alpha \rightarrow \beta$  in  $\mathcal{G}$ ;  
        **end**  
    **end**  
**end**  
**foreach**  $\alpha \xleftrightarrow{e} \beta \in E$  **do**  
    **if**  $\mathcal{I}(\mathcal{G} - e) \cap \mathcal{L}_n = \emptyset$  **then**  
        **if**  $\alpha \notin \text{ps}(\beta, \mathcal{I})$  **then**  
            delete  $\alpha \leftrightarrow \beta$  in  $\mathcal{G}$ ;  
        **end**  
    **end**  
**end**  
**return**  $\mathcal{G}$

**Subalgorithm 3:** Potential step

candidate separator sets may be chosen in more-or-less efficient ways, but we will not discuss this aspect of the algorithm (Colombo et al., 2012; Claassen et al., 2013).

**Lemma 16.** Subalgorithm 1 outputs the separability graph of  $\mathcal{I}$ ,  $\mathcal{S}$ , and furthermore  $\mathcal{N} \subseteq \mathcal{S}$ .

### 5.1.2 The pruning step

Let  $\mathcal{S}$  denote the graph in the output of Subalgorithm 1. One can use some of the information encoded by the graph along with the set  $\mathcal{L}_s$  to further prune the graph. For this purpose, we consider *W-structures* which are triples of nodes  $\alpha, \beta, \gamma$  such that  $\alpha \neq \beta \neq \gamma$ , and  $\alpha \rightarrow \beta \twoheadrightarrow \gamma$ . We denote such a triple by  $w(\alpha, \beta, \gamma)$ . We will say that a *W-structure* is *unshielded* if the edge  $\alpha \rightarrow \gamma$  is not in the graph. For every unshielded *W-structure*  $w(\alpha, \beta, \gamma)$ , there exists exactly one triple  $\langle \alpha, \gamma \mid C \rangle$  in  $\mathcal{L}_s$  (output from Subalgorithm 1) and we let  $S_{\alpha, \gamma}$  denote the separating set  $C$ .

**Lemma 17.** Subalgorithm 2 outputs a supergraph of  $\mathcal{N}$ .

### 5.1.3 Potential step

In the final step, we sequentially consider each edge which is still in the graph. If  $\mathcal{G} = (V, E)$  and  $e \in E$  we let  $\mathcal{G} - e$  denote the DMG  $(V, E \setminus \{e\})$ . We then check if  $\mathcal{I}(\mathcal{G} - e) \cap \mathcal{L}_n = \emptyset$ . If not, we leave this edge in the graph. On the other hand, if the intersection is the empty set, we check if the edge is between a pair of potential parents/siblings using the definition of these sets. That is, in the case of a directed edge we check each of

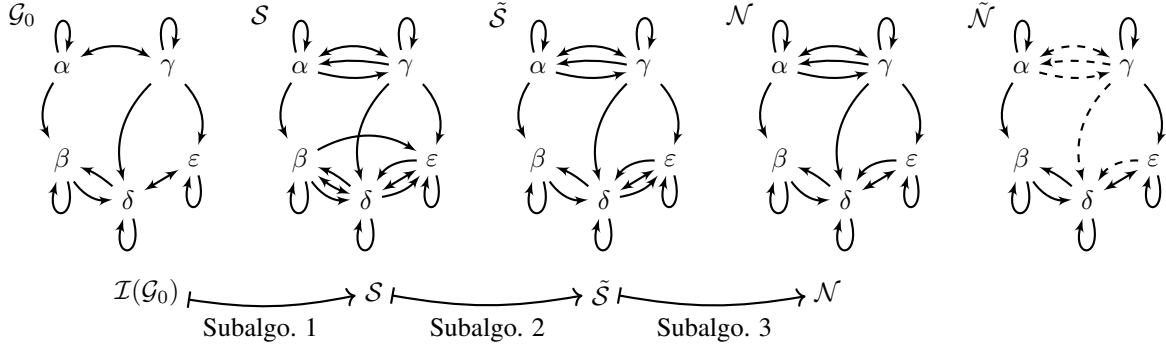


Figure 3: Illustration of the learning algorithm. The DMG  $\mathcal{G}_0$  is the underlying graph and we have access to  $\mathcal{I} = \mathcal{I}(\mathcal{G}_0)$ . Subalgorithm 1 outputs  $\mathcal{S}$ , the separability graph of  $\mathcal{I}(\mathcal{G}_0)$ . Subalgorithm 2 prunes  $\mathcal{S}$  and outputs  $\tilde{\mathcal{S}}$ . Note e.g. the unshielded  $W$ -structure  $\alpha \rightarrow \beta \rightarrow \varepsilon$  in  $\mathcal{S}$ . The DMG  $\mathcal{N}$  is the maximal element in  $[\mathcal{G}_0]$ . Note that  $\delta \rightarrow \varepsilon$  has been removed by Subalgorithm 3 using the potential parent criteria. The final graph  $\tilde{\mathcal{N}}$  is the DMEG constructed from  $\mathcal{N}$ .

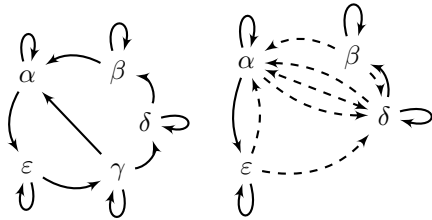


Figure 4: Left: linear SDE example (see Figure 1). Right: the DMEG after marginalization over  $\gamma$ . It is not possible to decide if a loop is directed or bidirected from the independence model only and we choose only to draw the directed loop and to not present it as dashed.

the conditions (p1)-(p4) and in the case of a bidirected edge each of the conditions (s1)-(s3); both sets of conditions are in the supplementary material. Note that if  $\alpha \in \text{ps}(\beta, \mathcal{I})$ , then also  $\beta \in \text{ps}(\alpha, \mathcal{I})$ .

**Theorem 18.** The algorithm defined by first doing the separation step, then the pruning, and finally the potential step outputs  $\mathcal{N}$ , the maximal element of  $[\mathcal{G}_0]$ .

Using properties of maximal DMGs, Mogensen and Hansen (2018) showed how one can construct the DMEG efficiently. The learning algorithm that is defined by first constructing  $\mathcal{N}$  and then constructing the DMEG is sound and complete in the sense that if an edge is absent in the DMEG, then it is also absent in any element of  $[\mathcal{G}_0]$  and therefore also in  $\mathcal{G}_0$ . If it is present and not dashed in the DMEG, then it is present in all elements of  $[\mathcal{G}_0]$  and therefore also in  $\mathcal{G}_0$ . Finally, if it is present and dashed in the DMEG, then there exist  $\mathcal{G}_1, \mathcal{G}_2 \in [\mathcal{G}_0]$  such that the edge is present in  $\mathcal{G}_1$  and absent in  $\mathcal{G}_2$  and therefore it is impossible to determine if the edge is in  $\mathcal{G}_0$  using

knowledge of  $\mathcal{I}(\mathcal{G}_0)$  only.

One could also skip the potential step to reduce the computational requirements. The resulting DMG is then a supergraph of the true graph. A small simulation study (supplementary material) indicates that one could save quite a number of tests and still get close to the true  $\mathcal{N}$ .

## 6 CONCLUSION AND DISCUSSION

We have shown that for a given directed graph with  $\mu$ -separation it is possible to specify abstract properties that ensure equivalence of the pairwise and global Markov properties in asymmetric independence models. We have shown that under certain conditions these properties hold in local independence models of Itô diffusions and event processes, extending known results.

Assuming faithfulness, we have given a sound and complete learning algorithm for the Markov equivalence class of directed mixed graphs representing a marginalized local independence model. Faithfulness is not an innocuous assumption and it remains an open research question how common this property is in different classes of stochastic processes.

### Acknowledgements

SWM and NRH were supported by research grant 13358 from VILLUM FONDEN. DM was supported by research grant R01 AI127271-01A1 from the National Institutes of Health.

## References

- Odd O. Aalen. Dynamic modelling and causality. *Scandinavian Actuarial Journal*, pages 177–190, 1987.
- Odd O. Aalen, Kjetil Røysland, Jon Michael Gran, and Bruno Ledergerber. Causality, mediation and time: a dynamic viewpoint. *Journal of the Royal Statistical Society, Series A*, 175(4):831–861, 2012.
- Ayesha R. Ali, Thomas S. Richardson, and Peter Spirtes. Markov equivalence for ancestral graphs. *The Annals of Statistics*, 37(5B):2808–2837, 2009.
- Tom Claassen, Joris Mooij, and Tom Heskes. Learning sparse causal models is not NP-hard. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence*, pages 172–181, 2013.
- Diego Colombo, Marloes H. Maathuis, Markus Kalisch, and Thomas S. Richardson. Learning high-dimensional directed acyclic graphs with latent and selection variables. *The Annals of Statistics*, 40(1): 294–321, 2012.
- Daniel Commenges and Anne Gégout-Petit. A general dynamical statistical model with causal interpretation. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 71(3):719–736, 2009.
- David Danks and Sergey Plis. Learning causal structure from undersampled time series. In *JMLR: Workshop and Conference Proceedings*, volume 10, pages 1–10, 2013.
- Vanessa Didelez. *Graphical Models for Event History Analysis based on Local Independence*. PhD thesis, Universität Dortmund, 2000.
- Vanessa Didelez. Graphical models for composable finite Markov processes. *Scandinavian Journal of Statistics*, 34(1):169–185, 2006a.
- Vanessa Didelez. Asymmetric separation for local independence graphs. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, 2006b.
- Vanessa Didelez. Graphical models for marked point processes based on local independence. *Journal of the Royal Statistical Society, Series B*, 70(1):245–264, 2008.
- Vanessa Didelez. Causal reasoning for events in continuous time: A decision-theoretic approach. In *Proceedings of the UAI 2015 Workshop on Advances in Causal Inference*, 2015.
- Asela Gunawardana, Christopher Meek, and Puyang Xu. A model for temporal dependencies in event streams. In *Advances in Neural Information Processing Systems 24 (NIPS 2011)*, 2011.
- Steffen Lauritzen. *Graphical Models*. Oxford: Clarendon, 1996.
- Steffen Lauritzen and Kayvan Sadeghi. Unifying Markov properties for graphical models. 2017. URL <https://arxiv.org/abs/1608.05810>.
- R.S. Liptser and A.N. Shiryaev. *Statistics of Random Processes I: General Theory*. Springer-Verlag, 1977.
- Christopher Meek. Toward learning graphical and causal process models. In *Proceedings of the UAI 2014 Workshop on Causal Inference: Learning and Prediction*, 2014.
- Søren Wengel Mogensen and Niels Richard Hansen. Markov equivalence of marginalized local independence graphs. 2018. URL <https://arxiv.org/abs/1802.10163>.
- Joris M. Mooij, Dominik Janzing, and Bernhard Schölkopf. From ordinary differential equations to structural causal models: the deterministic case. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence*, 2013.
- U. Nodelman, C. R. Shelton, and D. Koller. Continuous time Bayesian networks. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence*, pages 378–87, 2002.
- U. Nodelman, C. R. Shelton, and D. Koller. Learning continuous time Bayesian networks. In *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence*, pages 451–8, 2003.
- Thomas S. Richardson. A discovery algorithm for directed cyclic graphs. In *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence*, 1996.
- Thomas S. Richardson. A characterization of Markov equivalence for directed cyclic graphs. *International Journal of Approximate Reasoning*, 17:107–162, 1997.
- L. C. G. Rogers and David Williams. *Diffusions, Markov processes, and martingales. Vol. 2*. Cambridge Mathematical Library. Cambridge University Press, Cambridge, 2000.
- Paul K. Rubenstein, Stephan Bongers, Joris M. Mooij, and Bernhard Schölkopf. From deterministic ODEs to dynamic structural causal models. *arXiv.org preprint*, arXiv:1608.08028 [cs.AI], 2016. URL <http://arxiv.org/abs/1608.08028>.
- Kayvan Sadeghi. Faithfulness of probability distributions and graphs. *Journal of Machine Learning Research*, 18(148):1–29, 2017.
- Tore Schweder. Composable Markov processes. *Journal of Applied Probability*, 7(2):400–410, 1970.
- Alexander Sokol and Niels Richard Hansen. Causal interpretation of stochastic differential equations. *Electronic Journal of Probability*, 19(100):1–24, 2014.

Alexander Sokol and Niels Richard Hansen. Exponential martingales and changes of measure for counting processes. *Stochastic Analysis and Applications*, 33(5): 823–843, 2015.

Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. MIT Press, 2000.

Jiji Zhang. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. *Artificial Intelligence*, 172:1873–1896, 2008.

---

# Variational zero-inflated Gaussian processes with sparse kernels

---

Pashupati Hegde

Markus Heinonen

Samuel Kaski

Helsinki Institute for Information Technology HIIT  
Department of Computer Science, Aalto University

## Abstract

Zero-inflated datasets, which have an excess of zero outputs, are commonly encountered in problems such as climate or rare event modelling. Conventional machine learning approaches tend to overestimate the non-zeros leading to poor performance. We propose a novel model family of *zero-inflated Gaussian processes* (ZiGP) for such zero-inflated datasets, produced by *sparse kernels* through learning a latent probit Gaussian process that can zero out kernel rows and columns whenever the signal is absent. The ZiGPs are particularly useful for making the powerful Gaussian process networks more interpretable. We introduce *sparse GP networks* where variable-order latent modelling is achieved through sparse mixing signals. We derive the non-trivial stochastic variational inference tractably for scalable learning of the sparse kernels in both models. The novel output-sparse approach improves both prediction of zero-inflated data and interpretability of latent mixing models.

## 1 INTRODUCTION

Zero-inflated quantitative datasets with overabundance of zero output observations are common in many domains, such as climate and earth sciences (Enke & Spekat, 1997; Wilby, 1998; Charles et al., 2004), ecology (del Saz-Salazar & Rausell-Köster, 2008; Ancelet et al., 2009), social sciences (Bohning et al., 1997), and in count processes (Barry & Welsh, 2002). Traditional regression modelling of such data tends to underestimate zeros and overestimate nonzeros (Andersen et al., 2014).

A conventional way of forming zero-inflated models is to estimate a mixture of a Bernoulli “on-off” process and a Poisson count distribution (Johnson & Kotz, 1969; Lambert, 1992). In hurdle models a binary “on-off” process determines whether a hurdle is crossed, and the positive responses are governed by a subsequent process (Cragg, 1971; Mullahy, 1986). The hurdle model is analogous to first performing classification and training a continuous predictor on the positive values only, while the zero-inflated model would regress with all observations. Both stages can be combined for simultaneous classification and regression Abraham & Tan (2010).

Gaussian process models have not been proposed for zero-inflated datasets since their posteriors are Gaussian, which are ill-fitted for zero predictions. A suite of Gaussian process models have been proposed for partially related problems, such as mixture models (Tresp, 2001; Rasmussen & Ghahramani, 2002; Lázaro-Gredilla et al., 2012) and change point detection (Herlands et al., 2016). Structured spike-and-slab models place smoothly sparse priors over the structured inputs (Andersen et al., 2014).

In contrast to other approaches, we propose a Bayesian model that learns the underlying latent prediction function, whose covariance is sparsified through another Gaussian process switching between the ‘on’ and ‘off’ states, resulting in an zero-inflated Gaussian process model. This approach introduces a tendency of predicting exact zeros to Gaussian processes, which is directly useful in datasets with excess zeros.

A Gaussian process network (GPRN) is a latent signal framework where multi-output data are explained through a set of latent signals and mixing weight Gaussian processes (Wilson et al., 2012). The standard GPRN tends to have dense mixing that combines all latent signals for all latent outputs. By

applying the zero-predicting Gaussian processes to latent mixture models, we introduce sparse GPRNs where latent signals are mixed with sparse instead of dense mixing weight functions. The sparse model induces variable-order mixtures of latent signals resulting in simpler and more interpretable models. We demonstrate both of these properties in our experiments with spatio-temporal and multi-output datasets.

**Main contributions.** Our contributions include<sup>1</sup>

1. A novel zero-inflated Gaussian process formalism consisting of a latent Gaussian process and a separate ‘on-off’ probit-linked Gaussian process that can zero out rows and columns of the model covariance. The novel sparse kernel adds to GPs the ability to predict zeros.
2. Novel stochastic variational inference (SVI) for such sparse probit covariances, which in general are intractable due to having to compute expectations of GP covariances with respect to probit-linked processes. We derive the SVI for learning both of the underlying processes.
3. A novel sparse GPRN with an on-off process in the mixing matrices leading to sparse and variable-order mixtures of latent signals.
4. A solution to the stochastic variational inference of sparse GPRN where the SVI is derived for the network of full probit-linked covariances.

## 2 GAUSSIAN PROCESSES

We begin by introducing the basics of conventional Gaussian processes. Gaussian processes (GP) are a family of non-parametric, non-linear Bayesian models (Rasmussen & Williams, 2006). Assume a dataset of  $n$  inputs  $X = (\mathbf{x}_1, \dots, \mathbf{x}_n)$  with  $\mathbf{x}_i \in \mathbb{R}^D$  and noisy outputs  $\mathbf{y} = (y_1, \dots, y_n) \in \mathbb{R}^n$ . The observations  $y = f(\mathbf{x}) + \varepsilon$  are assumed to have additive, zero mean noise  $\varepsilon \sim \mathcal{N}(0, \sigma_y^2)$  with a zero-mean GP prior on the latent function  $f(\mathbf{x})$ ,

$$f(\mathbf{x}) \sim \mathcal{GP}(0, K(\mathbf{x}, \mathbf{x}')), \quad (1)$$

which defines a distribution over functions  $f(\mathbf{x})$  whose mean and covariance are

$$\mathbb{E}[f(\mathbf{x})] = 0 \quad (2)$$

$$\text{cov}[f(\mathbf{x}), f(\mathbf{x}')] = K(\mathbf{x}, \mathbf{x}'). \quad (3)$$

<sup>1</sup>The TensorFlow compatible code will be made publicly available at <https://github.com/hegdepashupati/zero-inflated-gp>

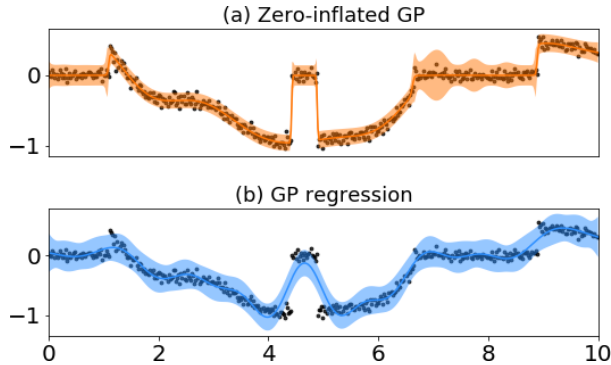


Figure 1: Illustration of a zero-inflated GP (a) and standard GP regression (b). The standard approach is unable to model sudden loss of signal (at 4...5) and signal close to zero (at 0...1 and 7...9).

Then for any collection of inputs  $X$ , the function values follow a multivariate normal distribution  $\mathbf{f} \sim \mathcal{N}(\mathbf{0}, K_{XX})$ , where  $\mathbf{f} = (f(\mathbf{x}_1), \dots, f(\mathbf{x}_N))^T \in \mathbb{R}^n$ , and where  $K_{XX} \in \mathbb{R}^{n \times n}$  with  $[K_{XX}]_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$ . The key property of Gaussian processes is that they encode functions that predict similar output values  $f(\mathbf{x}), f(\mathbf{x}')$  for similar inputs  $\mathbf{x}, \mathbf{x}'$ , with similarity determined by the kernel  $K(\mathbf{x}, \mathbf{x}')$ . In this paper we assume the Gaussian ARD kernel

$$K(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2} \sum_{j=1}^D \frac{(x_j - x'_j)^2}{\ell_j^2}\right), \quad (4)$$

with a signal variance  $\sigma_f^2$  and dimension-specific lengthscales  $\ell_1, \dots, \ell_D$  parameters.

The inference of the hyperparameters  $\theta = (\sigma_y, \sigma_f, \ell_1, \dots, \ell_D)$  is performed commonly by maximizing the marginal likelihood

$$p(\mathbf{y}|\theta) = \int p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\theta)d\mathbf{f}, \quad (5)$$

which results in a convenient marginal likelihood called evidence,  $p(\mathbf{y}|\theta) = N(\mathbf{y}|\mathbf{0}, K_{XX} + \sigma_y^2 I)$  for a Gaussian likelihood.

The Gaussian process defines a univariate normal predictive posterior distribution  $f(\mathbf{x})|\mathbf{y}, X \sim \mathcal{N}(\mu(\mathbf{x}), \sigma^2(\mathbf{x}))$  for an arbitrary input  $\mathbf{x}$  with the prediction mean and variance<sup>2</sup>

$$\mu(\mathbf{x}) = K_{\mathbf{x}X}(K_{XX} + \sigma_y^2 I)^{-1}\mathbf{y}, \quad (6)$$

$$\sigma^2(\mathbf{x}) = K_{\mathbf{x}\mathbf{x}} - K_{\mathbf{x}X}(K_{XX} + \sigma_y^2 I)^{-1}K_{X\mathbf{x}}, \quad (7)$$

<sup>2</sup>In the following we omit the implicit conditioning on data inputs  $X$  for clarity.

where  $K_{X\mathbf{x}} = K_{\mathbf{x}X}^T \in \mathbb{R}^n$  is the kernel column vector over pairs  $X \times \mathbf{x}$ , and  $K_{\mathbf{x}\mathbf{x}} = K(\mathbf{x}, \mathbf{x}) \in \mathbb{R}$  is a scalar. The predictions  $\mu(\mathbf{x}) \pm \sigma(\mathbf{x})$  come with uncertainty estimates in GP regression.

### 3 ZERO-INFLATED GAUSSIAN PROCESSES

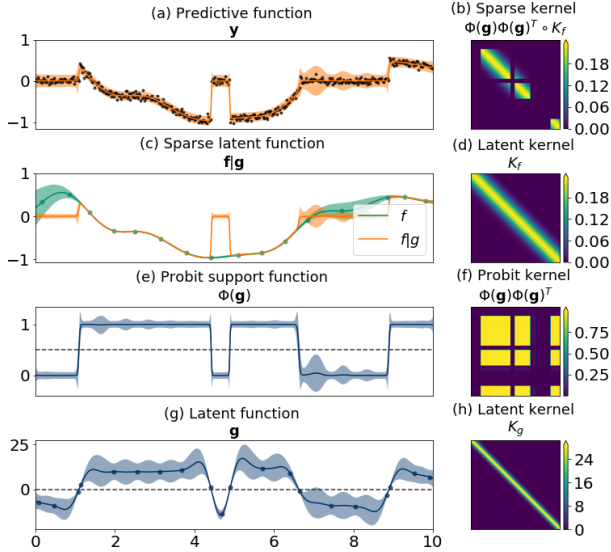


Figure 2: Illustration of the zero-inflated GP (a) and the sparse kernel (b) composed of a smooth latent function (c,d) filtered by a probit support function (e,f), which is induced by the underlying latent sparsity (g,h).

We introduce zero-inflated Gaussian processes that have – in contrast to standard GP’s – a tendency to produce exactly zero predictions (See Figure 1). Let  $g(\mathbf{x})$  denote the latent “on-off” state of a function  $f(\mathbf{x})$ . We assume GP priors for both functions with a joint model

$$p(\mathbf{y}, \mathbf{f}, \mathbf{g}) = p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{g})p(\mathbf{g}), \quad (8)$$

where

$$p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma_y^2 I) \quad (9)$$

$$p(\mathbf{f}|\mathbf{g}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \Phi(\mathbf{g})\Phi(\mathbf{g})^T \circ K_f) \quad (10)$$

$$p(\mathbf{g}) = \mathcal{N}(\mathbf{g}|\beta\mathbf{1}, K_g) \quad (11)$$

The sparsity values  $g(\mathbf{x})$  are squashed between 0 and 1 through a standard Normal cumulative distribution, or a probit link function,  $\Phi: \mathbb{R} \rightarrow [0, 1]$

$$\Phi(g) = \int_{-\infty}^g \phi(\tau) d\tau = \frac{1}{2} \left( 1 + \operatorname{erf} \left( \frac{g}{\sqrt{2}} \right) \right), \quad (12)$$

where  $\phi(\tau) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}\tau^2}$  is the standard normal density function. The structured probit sparsity  $\Phi(\mathbf{g})$  models the “on-off” smoothly due to the latent sparsity function  $\mathbf{g}$  having a GP prior with prior mean  $\beta$ . The latent function  $\mathbf{f}$  is modeled throughout but it is only visible during the “on” states. This masking effect has similarities to both zero-inflated and hurdle models. The underlying latent function  $\mathbf{f}$  is learned from only non-zero data similarly to in hurdle models, but the function  $\mathbf{f}$  is allowed to predict zeros similarly to zero-inflated models.

The key part of our model is the sparse *probit-sparsified covariance*  $\Phi(\mathbf{g})\Phi(\mathbf{g})^T \circ K$  where the “on-off” state  $\Phi(\mathbf{g})$  has the ability to zero out rows and columns of the kernel matrix at the “off” states (See Figure 2f for the probit pattern  $\Phi(\mathbf{g})\Phi(\mathbf{g})^T$  and Figure 2b for the resulting sparse kernel). Since the sparse kernel is represented as Hadamard product between a covariance kernel  $K$  and an outer product kernel  $\Phi(\mathbf{g})\Phi(\mathbf{g})^T$ , Schur product theorem implies that it is a valid kernel. As the sparsity  $g(\mathbf{x})$  converges towards minus infinity, the probit link  $\Phi(g(\mathbf{x}))$  approaches zero, which leads the function distribution approaching  $\mathcal{N}(f_i|0, 0)$ , or  $f_i = 0$ . Numerical problems are avoided since in practice  $\Phi(g) > 0$ , and due to the conditioning noise variance term  $\sigma_y^2 > 0$ .

The marginal likelihood of the zero-inflated Gaussian process is intractable due to the probit-sparsification of the kernel. We derive a stochastic variational Bayes approximation, which we show to be tractable due to the choice of using the probit link function.

#### 3.1 STOCHASTIC VARIATIONAL INFERENCE

Inference for standard Gaussian process models is difficult to scale as complexity grows with  $\mathcal{O}(n^3)$  as a function of the data size  $n$ . Titsias (2009) proposed a variational inference approach for GPs using  $m < n$  inducing variables, with a reduced computational complexity of  $\mathcal{O}(m^3)$  for  $m$  inducing points. The novelty of this approach lies in the idea that the locations and values of inducing points can be treated as variational parameters, and optimized. Hensman et al. (2013, 2015) introduced more efficient stochastic variational inference (SVI) with factorised likelihoods that has been demonstrated with up to billion data points (Salimbeni & Deisenroth, 2017). This approach cannot be directly applied to sparse kernels due to having to compute expectation of the probit product in the covariance. We derive

the SVI bound tractably for the zero-inflated model and its sparse kernel, which is necessary in order to apply the efficient parameter estimation techniques with automatic differentiation with frameworks such as TensorFlow (Abadi et al., 2016).

We begin by applying the inducing point augmentations  $f(\mathbf{z}_f) = \mathbf{u}_f$  and  $g(\mathbf{z}_g) = \mathbf{u}_g$  for both the latent function  $f(\cdot)$  and the sparsity function  $g(\cdot)$ . We place  $m$  inducing points  $\mathbf{u}_{f1}, \dots, \mathbf{u}_{fm}$  and  $\mathbf{u}_{g1}, \dots, \mathbf{u}_{gm}$  for the two functions. The augmented joint distribution is  $p(\mathbf{y}, \mathbf{f}, \mathbf{g}, \mathbf{u}_f, \mathbf{u}_g) = p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{g}, \mathbf{u}_f)p(\mathbf{g}|\mathbf{u}_g)p(\mathbf{u}_f)p(\mathbf{u}_g)$ , where<sup>3</sup>

$$p(\mathbf{f}|\mathbf{g}, \mathbf{u}_f) = \mathcal{N}(\mathbf{f}|\text{diag}(\Phi(\mathbf{g}))Q_f\mathbf{u}_f, \Phi(\mathbf{g})\Phi(\mathbf{g})^T \circ \tilde{K}_f) \quad (13)$$

$$p(\mathbf{g}|\mathbf{u}_g) = \mathcal{N}(\mathbf{g}|Q_g\mathbf{u}_g, \tilde{K}_g) \quad (14)$$

$$p(\mathbf{u}_f) = \mathcal{N}(\mathbf{u}_f|\mathbf{0}, K_{fmm}) \quad (15)$$

$$p(\mathbf{u}_g) = \mathcal{N}(\mathbf{u}_g|\mathbf{0}, K_{gmm}) \quad (16)$$

and where

$$Q_f = K_{fnm}K_{fmm}^{-1} \quad (17)$$

$$Q_g = K_{gnm}K_{gmm}^{-1} \quad (18)$$

$$\tilde{K}_f = K_{fnn} - K_{fnm}K_{fmm}^{-1}K_{fmn} \quad (19)$$

$$\tilde{K}_g = K_{gnn} - K_{gnm}K_{gmm}^{-1}K_{gmn}. \quad (20)$$

We denote the kernels for functions  $f$  and  $g$  by the corresponding subscripts. The kernel  $K_{fnn}$  is between all  $n$  data points, the kernel  $K_{fnm}$  is between all  $n$  datapoints and  $m$  inducing points, and the kernel  $K_{fmm}$  is between all  $m$  inducing points (similarly for  $g$  as well).

The distributions  $p(\mathbf{f}|\mathbf{u}_f)$  and  $p(\mathbf{g}|\mathbf{u}_g)$  can be obtained by conditioning the joint GP prior between respective latent and inducing functions. Further, the conditional distribution  $p(\mathbf{f}|\mathbf{g}, \mathbf{u}_f)$  can be obtained by the sparsity augmentation of latent conditional  $\mathbf{f}|\mathbf{u}_f$  similar to equation (10) (See Supplements).

Next we use the standard variational approach by introducing approximative variational distributions for the inducing points,

$$q(\mathbf{u}_f) = \mathcal{N}(\mathbf{u}_f|\mathbf{m}_f, \mathbf{S}_f) \quad (21)$$

$$q(\mathbf{u}_g) = \mathcal{N}(\mathbf{u}_g|\mathbf{m}_g, \mathbf{S}_g) \quad (22)$$

where  $\mathbf{S}_f, \mathbf{S}_g \in \mathbb{R}^{m \times m}$  are square positive semi-definite matrices. The variational joint posterior is

$$q(\mathbf{f}, \mathbf{g}, \mathbf{u}_f, \mathbf{u}_g) = p(\mathbf{f}|\mathbf{g}, \mathbf{u}_f)p(\mathbf{g}|\mathbf{u}_g)q(\mathbf{u}_f)q(\mathbf{u}_g). \quad (23)$$

<sup>3</sup>We drop the implicit conditioning on  $\mathbf{z}$ 's for clarity.

We minimize the Kullback-Leibler divergence between the true augmented posterior  $p(\mathbf{f}, \mathbf{g}, \mathbf{u}_f, \mathbf{u}_g|\mathbf{y})$  and the variational distribution  $q(\mathbf{f}, \mathbf{g}, \mathbf{u}_f, \mathbf{u}_g)$ , which is equivalent to solving the following evidence lower bound (as shown by e.g. Hensman et al. (2015)):

$$\log p(\mathbf{y}) \geq \mathbb{E}_{q(\mathbf{f})} \log p(\mathbf{y}|\mathbf{f}) - \text{KL}[q(\mathbf{u}_f, \mathbf{u}_g)||p(\mathbf{u}_f, \mathbf{u}_g)], \quad (24)$$

where we define

$$\begin{aligned} q(\mathbf{f}) &= \iiint p(\mathbf{f}|\mathbf{g}, \mathbf{u}_f)q(\mathbf{u}_f)p(\mathbf{g}|\mathbf{u}_g)q(\mathbf{u}_g)d\mathbf{u}_fd\mathbf{u}_gd\mathbf{g} \\ &= \int q(\mathbf{f}|\mathbf{g})q(\mathbf{g})d\mathbf{g}, \end{aligned} \quad (25)$$

where the variational approximations are tractably

$$\begin{aligned} q(\mathbf{g}) &= \int p(\mathbf{g}|\mathbf{u}_g)q(\mathbf{u}_g)d\mathbf{u}_g \\ &= \mathcal{N}(\mathbf{g}|\boldsymbol{\mu}_g, \Sigma_g) \end{aligned} \quad (26)$$

$$\begin{aligned} q(\mathbf{f}|\mathbf{g}) &= \int p(\mathbf{f}|\mathbf{g}, \mathbf{u}_f)q(\mathbf{u}_f)d\mathbf{u}_f \\ &= \mathcal{N}(\mathbf{f}|\text{diag}(\Phi(\mathbf{g}))\boldsymbol{\mu}_f, \Phi(\mathbf{g})\Phi(\mathbf{g})^T \circ \Sigma_f) \end{aligned} \quad (27)$$

with

$$\boldsymbol{\mu}_f = Q_f\mathbf{m}_f \quad (28)$$

$$\boldsymbol{\mu}_g = Q_g\mathbf{m}_g \quad (29)$$

$$\Sigma_f = K_{fnn} + Q_f(\mathbf{S}_f - K_{fmm})Q_f^T \quad (30)$$

$$\Sigma_g = K_{gnn} + Q_g(\mathbf{S}_g - K_{gmm})Q_g^T. \quad (31)$$

We additionally assume the likelihood  $p(\mathbf{y}|\mathbf{f}) = \prod_{i=1}^N p(y_i|f_i)$  factorises.

We solve the final ELBO of equations (24) and (25) as (See Supplements for detailed derivation)

$$\begin{aligned} \mathcal{L}_{\text{ZI}} &= \sum_{i=1}^N \left\{ \log \mathcal{N}(y_i|\langle \Phi(g_i) \rangle_{q(g_i)}\mu_{fi}, \sigma_{fi}^2) \right. \\ &\quad \left. - \frac{1}{2\sigma_{fi}^2} (\text{Var}[\Phi(g_i)]\mu_{fi}^2 + \langle \Phi(g_i)^2 \rangle_{q(g_i)}\sigma_{fi}^2) \right\} \\ &\quad - \text{KL}[q(\mathbf{u}_f)||p(\mathbf{u}_f)] - \text{KL}[q(\mathbf{u}_g)||p(\mathbf{u}_g)], \end{aligned} \quad (32)$$

where  $\mu_{fi}$  is the  $i$ 'th element of  $\boldsymbol{\mu}_f$  and  $\sigma_{fi}^2$  is the  $i$ 'th diagonal element of  $\Sigma_f$  (similarly with  $g$ ).



The expectations are tractable,

$$\langle \Phi(g_i) \rangle_{q(g_i)} = \Phi(\lambda_{gi}), \quad \lambda_{gi} = \frac{\mu_{gi}}{\sqrt{1 + \sigma_{gi}^2}} \quad (33)$$

$$\langle \Phi(g_i)^2 \rangle_{q(g_i)} = \Phi(\lambda_{gi}) - 2T\left(\lambda_{gi}, \frac{\lambda_{gi}}{\mu_{gi}}\right) \quad (34)$$

$$\text{Var}[\Phi(g_i)] = \Phi(\lambda_{gi}) - 2T\left(\lambda_{gi}, \frac{\lambda_{gi}}{\mu_{gi}}\right) - \Phi(\lambda_{gi})^2. \quad (35)$$

The Owen's T function  $T(a, b) = \phi(a) \int_0^b \frac{\phi(a\tau)}{1+\tau^2} d\tau$  (Owen, 1956) has efficient numerical solutions in practise (Patefield & Tandy, 2000).

The ELBO is considerably more complex than the standard stochastic variational bound of a Gaussian process (Hensman et al., 2013), due to the probit-sparsified covariance.

The bound is likely only tractable for the choice of probit link function  $\Phi(\mathbf{g})$ , while other link functions such as the logit would lead to intractable bounds necessitating slower numerical integration (Hensman et al., 2015).

We optimize the  $\mathcal{L}_{\text{ZI}}$  with stochastic gradient ascent techniques with respect to the inducing locations  $\mathbf{z}_g, \mathbf{z}_f$ , inducing value means  $\mathbf{m}_f, \mathbf{m}_g$  and covariances  $\mathbf{S}_f, \mathbf{S}_g$ , the sparsity prior mean  $\beta$ , the noise variance  $\sigma_y^2$ , the signal variances  $\sigma_f, \sigma_g$ , and finally the dimensions-specific lengthscales  $\ell_{f1}, \dots, \ell_{fD}; \ell_{g1}, \dots, \ell_{gD}$  of the Gaussian ARD kernel.

## 4 GAUSSIAN PROCESS NETWORK

The Gaussian Process Regression Networks (GPRN) framework by Wilson et al. (2012) is an efficient model for multi-target regression problems, where each individual output is a linear but non-stationary combination of shared latent functions. Formally, a vector-valued output function  $\mathbf{y}(\mathbf{x}) \in \mathbb{R}^P$  with  $P$  outputs is modeled using vector-valued latent functions  $\mathbf{f}(\mathbf{x}) \in \mathbb{R}^Q$  with  $Q$  latent values and mixing weights  $W(\mathbf{x}) \in \mathbb{R}^{P \times Q}$  as

$$\mathbf{y}(x) = W(x)[\mathbf{f}(x) + \boldsymbol{\epsilon}] + \boldsymbol{\varepsilon}, \quad (36)$$

where for all  $q = 1, \dots, Q$  and  $p = 1, \dots, P$  we assume GP priors and additive zero-mean noises,

$$f_q(\mathbf{x}) \sim \mathcal{GP}(0, K_f(\mathbf{x}, \mathbf{x}')) \quad (37)$$

$$W_{qp}(\mathbf{x}) \sim \mathcal{GP}(0, K_w(\mathbf{x}, \mathbf{x}')) \quad (38)$$

$$\epsilon_q \sim \mathcal{N}(0, \sigma_f^2) \quad (39)$$

$$\varepsilon_p \sim \mathcal{N}(0, \sigma_y^2). \quad (40)$$

The subscripts are used to denote individual components of  $\mathbf{f}$  and  $W$  with  $p$  and  $q$  indicating  $p^{\text{th}}$  output dimension and  $q^{\text{th}}$  latent dimension, respectively. We assume shared latent and output noise variances  $\sigma_f^2, \sigma_y^2$  without loss of generality. The distributions of both functions  $\mathbf{f}$  and  $W$  have been inferred either with variational EM (Wilson et al., 2012) or by variational mean-field approximation with diagonalized latent and mixing functions (Nguyen & Bonilla, 2013).

### 4.1 STOCHASTIC VARIATIONAL INFERENCE

Variational inference for GPRN has been proposed earlier with diagonalized mean-field approximation by (Nguyen & Bonilla, 2013). Further, stochastic variational inference by introducing inducing variables has been proposed for GPRN (Nguyen et al., 2014). In this section we rederive the SVI bound for standard GPRN for completeness and then propose the novel sparse GPRN model, and solve its SVI bounds as well, in the following section.

We begin by introducing the inducing variable augmentation technique for latent functions  $\mathbf{f}(\mathbf{x})$  and mixing weights  $W(\mathbf{x})$  with  $\mathbf{u}_f, \mathbf{z}_f = \{\mathbf{u}_{f_q}, \mathbf{z}_{f_q}\}_{q=1}^Q$  and  $\mathbf{u}_w, \mathbf{z}_w = \{\mathbf{u}_{w_{qp}}, \mathbf{z}_{w_{qp}}\}_{q,p=1}^{Q,P}$ :

$$p(\mathbf{y}, \mathbf{f}, W, \mathbf{u}_f, \mathbf{u}_w) \quad (41)$$

$$= p(\mathbf{y}|\mathbf{f}, W)p(\mathbf{f}|\mathbf{u}_f)p(W|\mathbf{u}_w)p(\mathbf{u}_f)p(\mathbf{u}_w)$$

$$p(\mathbf{f}|\mathbf{u}_f) = \prod_{q=1}^Q \mathcal{N}(\mathbf{f}_q | Q_{f_q} \mathbf{u}_{f_q}, \tilde{K}_{f_q}) \quad (42)$$

$$p(W|\mathbf{u}_w) = \prod_{q,p=1}^{Q,P} \mathcal{N}(\mathbf{w}_{qp} | Q_{w_{qp}} \mathbf{u}_{w_{qp}}, \tilde{K}_{w_{qp}}) \quad (43)$$

$$p(\mathbf{u}_f) = \prod_{q=1}^Q \mathcal{N}(\mathbf{u}_{f_q} | \mathbf{0}, K_{f_q, mm}) \quad (44)$$

$$p(\mathbf{u}_w) = \prod_{q,p=1}^{Q,P} \mathcal{N}(\mathbf{u}_{w_{qp}} | \mathbf{0}, K_{w_{qp}, mm}), \quad (45)$$

where we have separate kernels  $K$  and extrapolation matrices  $Q$  for each component of  $W(\mathbf{x})$  and  $\mathbf{f}(\mathbf{x})$

that are of the same form as in equations (17–20). The  $\mathbf{w}$  is a vectorised form of  $W$ . The variational approximation is then

$$q(\mathbf{f}, W, \mathbf{u}_f, \mathbf{u}_w) = p(\mathbf{f}|\mathbf{u}_f)p(W|\mathbf{u}_w)q(\mathbf{u}_f)q(\mathbf{u}_w) \quad (46)$$

$$q(\mathbf{u}_{f_q}) = \prod_{q=1}^Q \mathcal{N}(\mathbf{u}_{f_q}|\mathbf{m}_{f_q}, \mathbf{S}_{f_q}) \quad (47)$$

$$q(\mathbf{u}_{w_{qp}}) = \prod_{q,p=1}^{Q,P} \mathcal{N}(\mathbf{u}_{w_{qp}}|\mathbf{m}_{w_{qp}}, \mathbf{S}_{w_{qp}}), \quad (48)$$

where  $\mathbf{u}_{w_{qp}}$  and  $\mathbf{u}_{f_q}$  indicate the inducing points for the functions  $W_{qp}(\mathbf{x})$  and  $f_q(\mathbf{x})$ , respectively. The ELBO can be now stated as

$$\log p(\mathbf{y}) \geq \mathbb{E}_{q(\mathbf{f}, W)} \log p(\mathbf{y}|\mathbf{f}, W) - \text{KL}[q(\mathbf{u}_f, \mathbf{u}_w)||p(\mathbf{u}_f, \mathbf{u}_w)], \quad (49)$$

where the variational distributions decompose as  $q(\mathbf{f}, W) = q(\mathbf{f})q(W)$  with marginals of the same form as in equations (28–31),

$$q(\mathbf{f}) = \int q(\mathbf{f}|\mathbf{u}_f)q(\mathbf{u}_f)d\mathbf{u}_f = \mathcal{N}(\mathbf{f}|\boldsymbol{\mu}_f, \Sigma_f) \quad (50)$$

$$q(W) = \int q(W|\mathbf{u}_w)q(\mathbf{u}_w)d\mathbf{u}_w = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_w, \Sigma_w). \quad (51)$$

Since the noise term  $\boldsymbol{\varepsilon}$  is assumed to be isotropic Gaussian, the density  $p(\mathbf{y}|W, \mathbf{f})$  factorises across all target observations and dimensions. The expectation term in equation (49) then reduces to solving the following integral for the  $i^{\text{th}}$  observation and  $p^{\text{th}}$  target dimension,

$$\sum_{i,p=1}^{N,P} \iint \log \mathcal{N}(y_{p,i}|\mathbf{w}_{p,i}^T \mathbf{f}_i, \sigma_y^2) q(\mathbf{f}_i, \mathbf{w}_{p,i}) d\mathbf{w}_{p,i} d\mathbf{f}_i. \quad (52)$$

The above integral has a closed form solution resulting in the final ELBO as (See Supplements)

$$\begin{aligned} \mathcal{L}_{\text{GPRN}} &= \sum_{i=1}^N \left\{ \sum_{p=1}^P \log \mathcal{N}(y_{p,i} | \sum_{q=1}^Q \mu_{w_{qp},i} \mu_{f_q,i}, \sigma_y^2) \right. \\ &\quad \left. - \frac{1}{2\sigma_y^2} \sum_{q,p=1}^{Q,P} \left( \mu_{w_{qp},i}^2 \sigma_{f_q,i}^2 + \mu_{f_q,i}^2 \sigma_{w_{qp},i}^2 + \sigma_{w_{qp},i}^2 \sigma_{f_q,i}^2 \right) \right\} \\ &\quad - \sum_{q,p=1}^{Q,P} \text{KL}[q(\mathbf{u}_{w_{qp}}, \mathbf{u}_{f_q})||p(\mathbf{u}_{w_{qp}}, \mathbf{u}_{f_q})], \quad (53) \end{aligned}$$

where  $\mu_{f_q,i}$  is the  $i^{\text{th}}$  element of  $\boldsymbol{\mu}_{f_q}$  and  $\sigma_{f_q,i}^2$  is the  $i^{\text{th}}$  diagonal element of  $\Sigma_{f_q}$  (similarly for the  $W_{qp}$ 's).

## 5 SPARSE GAUSSIAN PROCESS NETWORK

In this section we demonstrate how zero-inflated GPs can be used as plug-in components in other standard models. In particular, we propose a significant modification to GPRN by adding sparsity to the mixing matrix components. This corresponds to each of the  $p$  outputs being a sparse mixture of the latent  $Q$  functions, i.e. they can effectively use any subset of the  $Q$  latent dimensions by having zeros for the rest in the mixing functions. This makes the mixture more easily interpretable, and induces a variable number of latent functions to explain the output of each input  $\mathbf{x}$ . The latent function  $\mathbf{f}$  can also be sparsified, with a derivation analogous to the derivation below.

We extend the GPRN with probit sparsity for the mixing matrix  $W$ , resulting in a joint model

$$p(\mathbf{y}, \mathbf{f}, W, \mathbf{g}) = p(\mathbf{y}|\mathbf{f}, W)p(\mathbf{f})p(W|\mathbf{g})p(\mathbf{g}), \quad (54)$$

where all individual components of the latent function  $\mathbf{f}$  and mixing matrix  $W$  are given GP priors. We encode the sparsity terms  $\mathbf{g}$  for all the  $Q \times P$  mixing functions  $W_{qp}(\mathbf{x})$  as

$$p(W_{qp}|\mathbf{g}_{qp}) = \mathcal{N}(\mathbf{w}_{qp}|\mathbf{0}, \Phi(\mathbf{g}_{qp})\Phi(\mathbf{g}_{qp})^T \circ K_w). \quad (55)$$

To introduce variational inference, the joint model is augmented with three sets of inducing variables for  $\mathbf{f}$ ,  $W$  and  $\mathbf{g}$ . After marginalizing out the inducing variables as in equations (25–27), the marginal likelihood can be written as

$$\log p(\mathbf{y}) \geq \mathbb{E}_{q(\mathbf{f}, W, \mathbf{g})} \log p(\mathbf{y}|\mathbf{f}, W) - \text{KL}[q(\mathbf{u}_f, \mathbf{u}_w, \mathbf{u}_g)||p(\mathbf{u}_f, \mathbf{u}_w, \mathbf{u}_g)]. \quad (56)$$

The joint distribution in the variational expectation factorizes as  $q(\mathbf{f}, W, \mathbf{g}) = q(\mathbf{f})q(W|\mathbf{g})q(\mathbf{g})$ . Also, with a Gaussian noise assumption, the expectation term factorizes across all the observations and target dimensions. The key step reduces to solving the following integrals:

$$\sum_{i,p=1}^{N,P} \iiint \log \mathcal{N}(y_{p,i} | (\mathbf{w}_{p,i} \circ \mathbf{g}_{p,i})^T \mathbf{f}_i, \sigma_y^2) \cdot q(\mathbf{f}_i, \mathbf{w}_{p,i}, \mathbf{g}_{p,i}) d\mathbf{w}_{p,i} d\mathbf{f}_i d\mathbf{g}_{p,i}. \quad (57)$$

The above integral has a tractable solution leading to the final sparse GPRN evidence lower bound (See Supplements)

$$\begin{aligned}
\mathcal{L}_{\text{sGPRN}} = & \sum_{i=1}^N \left\{ \sum_{p=1}^P \log \mathcal{N}(y_{p,i} | \sum_{q=1}^Q \mu_{w_{qp},i} \mu_{g_{qp},i} \mu_{f_q,i}, \sigma_y^2) \right. \\
& - \frac{1}{2\sigma_y^2} \sum_{q,p=1}^{Q,P} \left( (\mu_{g_{qp},i}^2 + \sigma_{g_{qp},i}^2) \right. \\
& \quad \cdot (\mu_{w_{qp},i}^2 \sigma_{f_q,i}^2 + \mu_{f_q,i}^2 \sigma_{w_{qp},i}^2 + \sigma_{w_{qp},i}^2 \sigma_{f_q,i}^2) \\
& \quad \left. \left. - \frac{1}{2\sigma_y^2} \sum_{q,p=1}^{Q,P} \left( \sigma_{g_{qp},i}^2 \mu_{f_q,i}^2 \mu_{w_{qp},i}^2 \right) \right\} \right. \\
& \left. - \sum_{q,p}^{Q,P} \text{KL}[q(\mathbf{u}_{f_q}, \mathbf{u}_{w_{qp}}, \mathbf{u}_{g_{qp}}) || p(\mathbf{u}_{f_q}, \mathbf{u}_{w_{qp}}, \mathbf{u}_{g_{qp}})], \right.
\end{aligned} \tag{58}$$

where  $\mu_{f_q,i}, \mu_{w_{qp},i}$  are the variational expectation means for  $\mathbf{f}(\cdot), W(\cdot)$  as in equations (28, 29),  $\mu_{g_{qp},i}$  is the variational expectation mean of  $g(\cdot)$  as in equation (33), and analogously for the variances.

## 6 EXPERIMENTS

First we demonstrate how the proposed method can be used for regression problems with zero-inflated targets. We do that both on a simulated dataset and for real-world climate modeling scenarios on a Finnish rain precipitation dataset with approximately 90% zeros. Finally, we demonstrate the GPRN model and how it improves both the interpretability and predictive performance in the JURA geological dataset.

We use the squared exponential kernel with ARD in all experiments. All the parameters including inducing locations, values and variances and kernel parameters were learned through stochastic Adam optimization (Kingma & Ba, 2014) on the TensorFlow (Abadi et al., 2016) platform.

We compare our approach ZIGP to baseline ZERO voting, to conventional Gaussian process regression (GPR) and classification (GPC) with SVI approximations from the GPflow package (Matthews et al., 2017). Finally, we also compare to first classifying the non-zeros, and successively applying regression either to all data points (GPCR), or to only predicted non-zeros (GPCR<sub>≠0</sub>, hurdle model).

We record the predictive performance by considering mean squared error and mean absolute error. We also compare the models' ability to predict true zeros with F1, accuracy, precision, and recall of the optimal models.

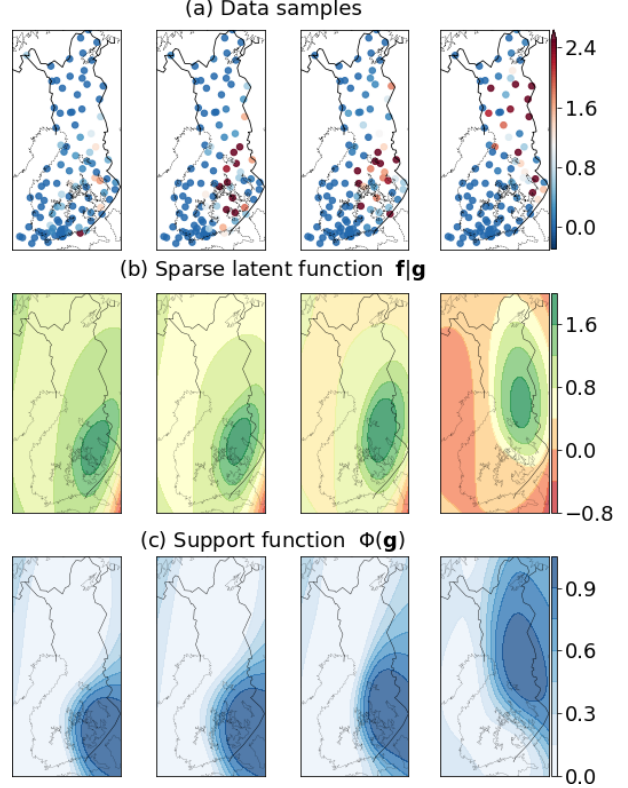


Figure 3: ZIGP model fit on the precipitation dataset. Sample of the actual data (a) against the sparse rain function estimate (b), with the probit support function (c) showing the rain progress.

### 6.1 SPATIO-TEMPORAL DATASET

Zero-inflated cases are commonly found in climatology and ecology domains. In this experiment we demonstrate the proposed method by modeling precipitation in Finland<sup>4</sup>. The dataset consists of hourly quantitative non-negative observations of precipitation amount across 105 observatory locations in Finland for the month of July 2018. The dataset contains 113015 datapoints with approximately 90% zero precipitation observations. The data inputs are three-dimensional: latitude, longitude and time. Due to the size of the data, this experiment illustrates the scalability of the variational inference.

We randomly split 80% of the data for training and the rest 20% for testing purposes. We split across time only, such that at a single measurement time, all locations are simultaneously either in the training set, or in the test set.

<sup>4</sup>Data can be found at <http://en.ilmatieteenlaitos.fi/>

Table 1: Results for the precipitation dataset over baseline (Zero; majority voting), four competing methods and the proposed method ZiGP on test data. The columns list both quantitative and qualitative performance criteria, best performance is boldfaced.

MODEL	RMSE	MAE	F1	ACC.	PREC.	RECALL
ZERO	0.615	0.104	0.000	0.898	0.000	0.000
GPC	-	-	0.367	<b>0.911</b>	0.675	0.252
GPR	0.569	0.159	0.401	0.750	0.266	<b>0.817</b>
GPCR	0.589	<b>0.102</b>	0.366	<b>0.911</b>	0.679	0.251
GPCR <sub>≠0</sub>	0.575	<b>0.101</b>	0.358	<b>0.912</b>	<b>0.712</b>	0.240
ZiGP	<b>0.561</b>	0.121	<b>0.448</b>	0.861	0.381	0.558

We further utilize the underlying spatio-temporal grid structure of the data to perform inference in an efficient manner by Kronecker techniques (Saatchi, 2011). All the kernels for latent processes are assumed to factorise as  $\mathbf{K} = \mathbf{K}_{space} \otimes \mathbf{K}_{time}$  which allows placing inducing points independently on spatial and temporal grids.

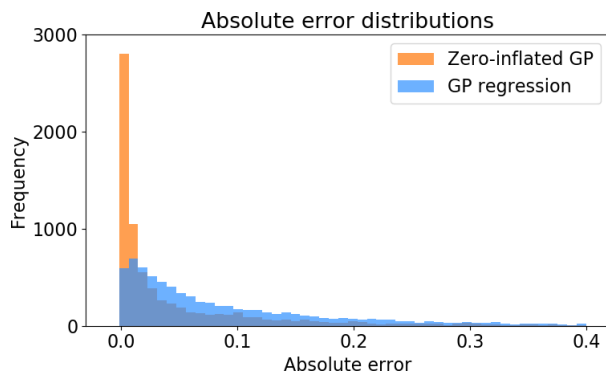


Figure 4: The distribution of errors with the rain dataset with the ZiGP and the GPR. The zero-inflated GP achieves much higher number of perfect (zero) predictions.

Figure 3 depicts the components of the zero-inflated GP model on the precipitation dataset. As shown in panel (c), the latent support function models the presence or absence of rainfall. It smoothly follows the change in rain patterns across hourly observations. The amount of precipitation is modeled by the other latent process and the combination of these two results in sparse predictions. Figure 4 shows that the absolute error distribution is remarkably better with the ZiGP model due to it identifying the absence of rain exactly. While both models fit the high rainfall regions well, for zero and near-zero regions GPR does not refine its small errors. Table 1 indicates that the ZiGP model achieves the lowest mean square error, while also achieving the highest F1 score that takes into account the class imbalance,

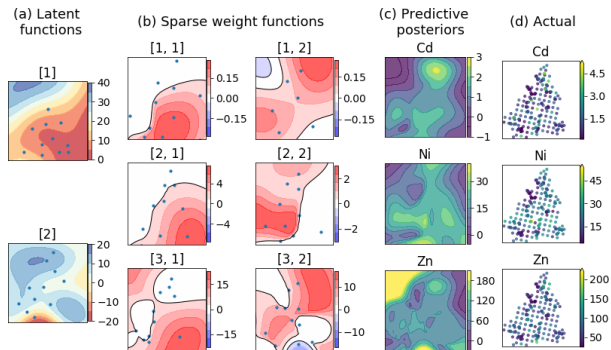


Figure 5: The sparse GPRN model fit on the Jura dataset with 11 inducing points. The  $Q = 2$  (dense) latent functions (a) are combined with the  $3 \times 2$  sparse mixing functions (b) into the  $P = 3$  output predictions (c). The real data are shown in (d). The white mixing regions are estimated ‘off’.

which biases the elementary accuracy, precision and recall quantities towards the majority class.

## 6.2 MULTI-OUTPUT PREDICTION - JURA

In this experiment we model the multi-response Jura dataset with the sparse Gaussian process regression network sGPRN model and compare it with standard GPRN as baseline. Jura contains concentration measurements of cadmium, nickel and zinc metals in the region of Swiss Jura. We follow the experimental procedure of Wilson et al. (2012) and Nguyen & Bonilla (2013). The training set consists of  $n = 259$  observations across  $D = 2$  dimensional geo-spatial locations, and the test set consists of 100 separate locations. For both models we use  $Q = 2$  latent functions with the stochastic variational inference techniques proposed in this paper. Sparse GPRN uses a sparsity inducing kernel in the mixing weights. The locations of inducing points for the weights  $W(\mathbf{x})$  and the support  $g(\mathbf{x})$  are shared. The kernel length-scales are given a gamma prior with the shape parameter  $\alpha = 0.3$  and rate parameter  $\beta = 1.0$  to induce smoothness. We train both the models 30 times with random initialization.

Table 2 shows that our model performs better than the state-of-the-art SVI-GPRN, both with  $m = 5$  and  $m = 10$  inducing points. Figure 5 visualises the optimized sparse GPRN model, while Figure 6 indicates the sparsity pattern in the mixing weights. The weights have considerable smooth ‘on’ regions (black) and ‘off’ regions (white). The ‘off’ regions indicate that for certain locations, only one of the two latent functions is adaptively utilised.

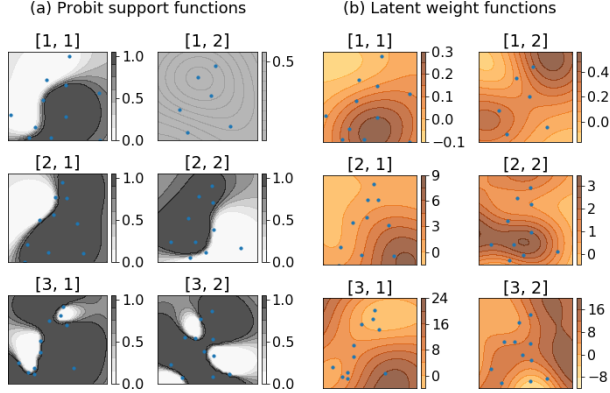


Figure 6: The sparse probit support (a) and latent functions (b) of the weight function  $W(\mathbf{x})$  of the optimized sparse GPRN model. The black regions of (a) show regional activations, while the white regions show where the latent functions are ‘off’. The elementwise product of the support and weight functions is indicated in the Figure 5b).

Table 2: Results for the Jura dataset for sparse GPRN and vanilla GPRN models with test data. Best performance is with boldface. We do not report RMSE and MAE values GPC, since its a classification method.

MODEL	$m$	CADMIUM		NICKEL		ZINC	
		RMSE	MAE	RMSE	MAE	RMSE	MAE
GPRN	5	0.724	0.566	<b>6.469</b>	<b>4.958</b>	33.729	21.959
	10	0.736	0.574	6.626	5.109	34.923	22.544
	15	0.749	0.590	6.526	5.033	35.033	22.670
sGPRN	5	<b>0.719</b>	<b>0.565</b>	6.553	5.054	<b>33.475</b>	<b>21.774</b>
	10	<b>0.727</b>	<b>0.567</b>	<b>6.520</b>	<b>5.062</b>	<b>34.225</b>	<b>22.114</b>
	15	<b>0.725</b>	<b>0.569</b>	<b>6.479</b>	<b>5.033</b>	<b>34.308</b>	<b>22.288</b>

### 6.3 MULTI-OUTPUT PREDICTION - SARCOS

In this experiment we tackle the problem of learning inverse dynamics for seven degrees of freedom of SARCOS anthropomorphic robot arm (Vijayakumar et al., 2005). The dataset consists of 48,933 observations with an input space of 21 dimensions (7 joints positions, 7 joint velocities, 7 joint accelerations). The multi-output prediction task is to learn a mapping from these input variables to the corresponding 7 joint torques of the robot. Multi-output GP has been previously used for inverse dynamics modeling (Williams et al., 2009), but in a different model setting and on a smaller dataset. GPRN with stochastic inference framework has also been explored to model SARCOS dataset (Nguyen et al., 2014), however, they use a different experimental setting and consider 2 of the 7 joint torques as multi-output targets.

Table 3: Normalized MSE results on the SARCOS test data for sparse GPRN and standard GPRN models. Best performance is mentioned with boldface.

MODEL		$m = 50$	$m = 100$	$m = 150$
GPRN	$Q = 2$	0.0167	0.0145	0.0127
	$Q = 3$	0.0146	0.0121	0.0108
sGPRN	$Q = 2$	<b>0.0159</b>	<b>0.0131</b>	<b>0.0125</b>
	$Q = 3$	<b>0.0140</b>	<b>0.0117</b>	<b>0.0096</b>

We consider 80%+20% random split of the full dataset for training and testing respectively. Both GPRN and sGPRN model are trained with  $m = 50, 100$  and  $150$  inducing points and  $Q = 2$  and  $3$  latent functions. We repeat the experiment 20 times and report normalized-MSE in Table 3. Sparse GPRN gives better results than standard GPRN in all our experimental settings. Moreover, sparse model (nMSE= 0.0096) gives gives 12% improvement over the standard model (nMSE= 0.0108) for the best test performance with  $Q = 3$  latent functions and  $m = 150$ .

## 7 DISCUSSION

We proposed a novel paradigm of zero-inflated Gaussian processes with a novel sparse kernel. The sparsity in the kernel is modeled with smooth probit filtering of the covariance rows and columns. This model induces zeros in the prediction function outputs, which is highly useful for zero-inflated datasets with excess of zero observations. Furthermore, we showed how the zero-inflated GP can be used to model sparse mixtures of latent signals with the proposed sparse Gaussian process network. The latent mixture model with sparse mixing coefficients leads to locally using only a subset of the latent functions, which improves interpretability and reduces model complexity. We demonstrated tractable solutions to stochastic variational inference of the sparse probit kernel for the zero-inflated GP, conventional GPRN, and sparse GPRN models, which leads to efficient exploration of the parameter space of the model.

## Acknowledgements

We would like to thank anonymous reviewers for their helpful suggestions and comments. This work has been supported by the Academy of Finland grant no. 294238 and 292334.

## References

- Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pp. 265–283, 2016.
- Abraham, Z. and Tan, P-N. An integrated framework for simultaneous classification and regression of time-series data. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, pp. 653–664. SIAM, 2010.
- Ancelet, S., Etienne, M-P., Benot, H., and Parent, E. Modelling spatial zero-inflated continuous data with an exponentially compound Poisson process. *Environmental and Ecological Statistics*, 2009.
- Andersen, M., Winther, O., and Hansen, L. Bayesian inference for structured spike and slab priors. In *NIPS*, pp. 1745–1753, 2014.
- Barry, S. and Welsh, A. H. Generalized additive modelling and zero inflated count data. *Ecological Modelling*, 157:179–188, 2002.
- Bohning, D., Dierz, E., and Schlattmann, P. Zero-inflated count models and their applications in public health and social science. In Rost, J. and Langeheine, R. (eds.), *Applications of Latent Trait and Latent Class Models in the Social Sciences*. Waxman Publishing Co, 1997.
- Charles, S., Bates, B., Smith, I., and Hughes, J. Statistical downscaling of daily precipitation from observed and modelled atmospheric fields. *Hydrological Processes*, pp. 1373–1394, 2004.
- Cragg, J.G. Some statistical models for limited dependent variables with application to the demand for durable goods. *Econometrica*, 39:829–844, 1971.
- del Saz-Salazar, S. and Rausell-Köster, P. A double-hurdle model of urban green areas valuation: dealing with zero responses. *Landscape and urban planning*, 84(3-4):241–251, 2008.
- Enke, W. and Spekat, A. Downscaling climate model outputs into local and regional weather elements by classification and regression. *Climate Research*, 8:195–207, 1997.
- Hensman, J., Fusi, N., and Lawrence, N. Gaussian processes for big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, pp. 282–290. AUAI Press, 2013.
- Hensman, J., Matthews, A., and Ghahramani, Z. Scalable variational Gaussian process classification. In *Artificial Intelligence and Statistics*, pp. 351–360, 2015.
- Herlands, W., Wilson, A., Nickisch, H., Flaxman, S., Neill, D., van Panhuis, W., and Xing, E. Scalable Gaussian processes for characterizing multidimensional change surfaces. In *AISTATS*, volume 51 of *PMLR*, pp. 1013–1021, 2016.
- Johnson, N. and Kotz, S. *Distributions in Statistics: Discrete Distributions*. Houghton MiZin, Boston, 1969.
- Kingma, D. and Ba, J. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014.
- Lambert, D. Zero-inflated Poisson regression with an application to defects in manufacturing. *Technometrics*, 34:1–14, 1992.
- Lázaro-Gredilla, M., Van Vaerenbergh, S., and Lawrence, N. Overlapping mixtures of Gaussian processes for the data association problem. *Pattern Recognition*, 45(4):1386–1395, 2012.
- Matthews, A., van der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., Leon-Villagra, P., Ghahramani, Z., and Hensman, J. GPflow: A Gaussian process library using TensorFlow. *Journal of Machine Learning Research*, 18(40):1–6, 2017.
- Mullahy, J. Specification and testing of some modified count data models. *Journal of Econometrics*, 33:341–365, 1986.
- Nguyen, T. and Bonilla, E. Efficient variational inference for Gaussian process regression networks. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, volume 31 of *Proceedings of Machine Learning Research*, pp. 472–480. PMLR, 2013.
- Nguyen, Trung V, Bonilla, Edwin V, et al. Collaborative multi-output gaussian processes. In *UAI*, pp. 643–652, 2014.
- Owen, D.B. Tables for computing bivariate normal probabilities. *Annals of Mathematical Statistics*, 27:1075–1090, 1956.
- Patefield, M. and Tandy, D. Fast and accurate calculation of owens t-function. *Journal of Statistical Software*, 5:1–25, 2000.
- Rasmussen, C. and Ghahramani, Z. Infinite mixtures of Gaussian process experts. In *NIPS*, pp. 881–888, 2002.
- Rasmussen, C.E. and Williams, K.I. *Gaussian processes for machine learning*. MIT Press, 2006.
- Saatchi, Y. *Scalable Inference for Structured Gaussian Process Models*. PhD thesis, University of Cambridge, 2011.

- Salimbeni, H. and Deisenroth, M. Doubly stochastic variational inference for deep Gaussian processes. In *NIPS*, volume 30, 2017.
- Titsias, M. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial Intelligence and Statistics*, pp. 567–574, 2009.
- Tresp, V. Mixtures of Gaussian processes. In *NIPS*, pp. 654–660, 2001.
- Vijayakumar, Sethu, D’Souza, Aaron, and Schaal, Stefan. Lwpr: A scalable method for incremental online learning in high dimensions. 2005.
- Wilby, R.L. Statistical downscaling of daily precipitation using daily airflow and seasonal teleconnection. *Climate Research*, 10:163–178, 1998.
- Williams, Christopher, Klanke, Stefan, Vijayakumar, Sethu, and Chai, Kian M. Multi-task gaussian process learning of robot inverse dynamics. In *Advances in Neural Information Processing Systems*, pp. 265–272, 2009.
- Wilson, A. G., Knowles, D., and Ghahramani, Z. Gaussian process regression networks. In *ICML*, 2012.

---

# KBLRN: End-to-End Learning of Knowledge Base Representations with Latent, Relational, and Numerical Features

---

**Alberto García-Durán**  
NEC Labs Europe  
Heidelberg, Germany  
alberto.duran@neclab.eu

**Mathias Niepert**  
NEC Labs Europe  
Heidelberg, Germany  
mathias.niepert@neclab.eu

## Abstract

We present KBLRN, a framework for end-to-end learning of knowledge base representations from latent, relational, and numerical features. KBLRN integrates feature types with a novel combination of neural representation learning and probabilistic product of experts models. To the best of our knowledge, KBLRN is the first approach that learns representations of knowledge bases by integrating latent, relational, and numerical features. We show that instances of KBLRN outperform existing methods on a range of knowledge base completion tasks. We contribute a novel data set enriching commonly used knowledge base completion benchmarks with numerical features. The data sets are available under a permissive BSD-3 license<sup>1</sup>. We also investigate the impact numerical features have on the KB completion performance of KBLRN.

## 1 INTRODUCTION

The importance of knowledge bases (KBs) for AI systems has been demonstrated in numerous application domains. KBs provide ways to organize, manage, and retrieve structured data and allow AI system to perform reasoning. In recent years, KBs have been playing an increasingly crucial role in AI applications. Purely logical representations of knowledge bases have a long history in AI [27]. However, they suffer from being inefficient and brittle. Inefficient because the computational complexity of reasoning is exponential in the worst case and, therefore, the time required by a reasoner highly unpredictable. Brittle because a purely logical KB requires a large set of logical rules that are handcrafted and/or

<sup>1</sup><https://github.com/nle-ml/mmkb>

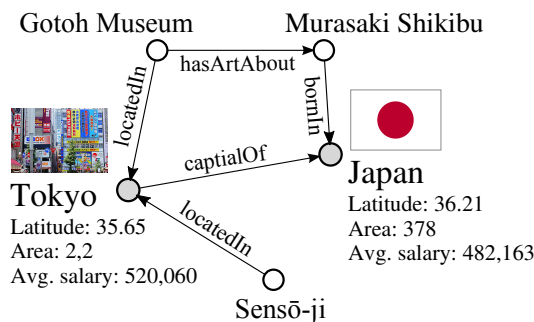


Figure 1: A small part of a knowledge base.

mined. These problems are even more pressing in applications whose environments are changing over time.

Motivated by these shortcomings, there has been a flurry of work on combining logical and statistical approaches to build systems capable of reasoning over and learning from incomplete structured data. Most notably, the statistical relational learning community has proposed numerous formalisms that combine logic and probability [24]. These formalisms are able to address the learning problem and make the resulting AI systems more robust to missing data and missing rules. Intuitively, logical formulas act as relational features and the probability of a possible world is determined by a sufficient statistic for the values of these features. These approaches, however, are in most cases even less efficient because logical inference is substituted with probabilistic inference.

More recently, the research community has focused on efficient machine learning models that perform well on restricted tasks such as link prediction in KBs. Examples are knowledge base factorization and embedding approaches [3, 21, 11, 20] and random-walk based ML models [15, 7]. The former learn latent features for the entities and relations in the knowledge base and use those to perform link prediction. The latter explore specific re-



lational features such as path types between two entities and train a machine learning model for link prediction.

With this work, we propose KBLRN, a novel approach to combining relational, latent (learned), and numerical features, that is, features that can take on a large or infinite number of real values. The combination of various features types is achieved by integrating embedding-based learning with probabilistic models in two ways. First, we show that modeling numerical features with radial basis functions is beneficial and can be integrated in an end-to-end differentiable learning system. Second, we propose a probabilistic product of experts (PoE) [13] approach to combine the feature types. Instead of training the PoE with contrastive divergence, we approximate the partition function with a negative sampling strategy. The PoE approach has the advantage of being able to train the model jointly and end-to-end.

The paper is organized as follows. First, we discuss relational, latent, and numerical features. Second, we describe KBLRN. Third, we present empirical evidence that instances of KBLRN outperform state of the art methods for KB completion. We also investigate in detail under what conditions numerical features are beneficial.

## 2 RELATIONAL, LATENT, AND NUMERICAL FEATURES

We assume that the facts of a knowledge base (KB) are given as a set of triples of the form  $(h, r, t)$  where  $h$  and  $t$  are the head and tail entities and  $r$  is a relation type. Figure 1 depicts a small fragment of a KB with relations and numerical features. KB completion is the problem of answering queries of the form  $(?, r, t)$  or  $(h, r, ?)$ . While the proposed approach can be generalized to more complex queries, we focus on completion queries for the sake of simplicity. We now discuss the three feature types used in KBLRN and motivate their utility for knowledge base completion. How exactly we extract features from a given KB is described in the experimental section.

### 2.1 Relational Features

Each relational feature is given as a logical formula which is evaluated in the KB to determine the feature’s value. For instance, the formula  $\exists x (A, \text{bornIn}, x) \wedge (x, \text{capitalOf}, B)$  corresponds to a binary feature which is 1 if there exists a path of that type from entity A to entity B, and 0 otherwise. These features are often used in relational models [30, 22] and random-walk based models such as PRA and SFE [15, 7]. In this work, we use relational paths of length one and two and use the rule mining approach AMIE+ [5]. We detail the generation of

the relational features in the experimental section. For a pair of entities  $(h, t)$ , we denote the feature vector computed based on a set of relational features by  $r_{(h,t)}$ .

### 2.2 Latent Features

Numerous embedding methods for KBs have been proposed in recent years [21, 3, 11, 20]. Embedding methods provide fixed-size vector representations (embeddings) for all entities in the KB. In the simplest of cases, relations are modeled as translations in the entity embedding space [3]. We incorporate typical embedding learning objectives into KBLRN and write  $e_h$  and  $e_t$  to refer to an embedding of a head entity and a tail entity, respectively. The advantages of latent feature models are their computational efficiency and their ability to learn latent entity types suitable for downstream ML tasks without hand-crafted or mined logical rules.

### 2.3 Numerical Features

Numerical features are entity features whose values can take on a very large or infinite number of real values. To the best of our knowledge, there does not exist a principled approach that integrates numerical features into a relational ML model for KB completion. This is surprising, considering that numerical data is available in almost all existing large-scale KBs. The assumption that numerical data is helpful for KB completion tasks is reasonable. For several relations types the *differences* between the *head* and *tail* are characteristic of the relation itself. For example, while the mean difference of birth years is 0.4 for the Freebase relation `/people/marriage/spouse`, it is 32.4 for the relation `/person/children`. These observations motivate specifically the use of *differences* of numerical feature values. Taking the difference has the advantage that even if a numerical feature is not distributed according to a normal distribution (e.g., birth years in a KB), the difference is often normally distributed. This is important as we need to fit simple parametric distributions to the sparse numerical data. We detail the fully automated extraction and generation of the numerical features in the experimental section.

## 3 KBLRN: LEARNING END-TO-END JOINT REPRESENTATIONS FOR KNOWLEDGE BASES

With KBLRN we aim to provide a framework for end-to-end learning of KB representations. Since we want to combine different feature types (relational, latent or learned, and numerical) we need to find a suitable

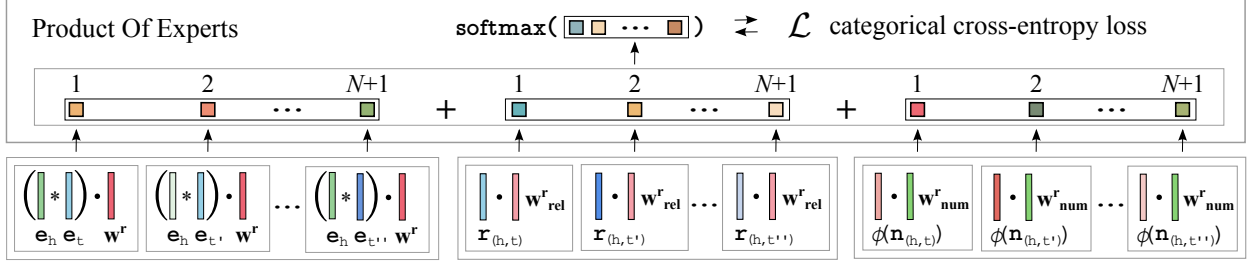


Figure 2: An illustration of an instance of KBLRN implemented with standard deep learning framework components. For every relation type, there is a separate expert for each of the different feature types. The entities  $\tau'$  and  $\tau''$  are two of  $N$  randomly sampled entities. The scores of the various submodels are added and normalized with a softmax function. A categorical cross-entropy loss is applied to the normalized scores.

method for integrating the respective submodels, one per feature type. We propose a product of experts (PoE) approach where one expert is trained for each (relation type, feature type) pair. We extend the product of experts approach in two novel ways. First, we create dependencies between the experts by sharing the parameters of the entity embedding model across relation types. By doing this, we combine a probabilistic model with a model that learns vector representations from discrete and numerical data. Second, while product of experts are commonly trained with contrastive divergence [13], we train it with negative sampling and a cross-entropy loss.

In general, a PoE's probability distribution is

$$p(\mathbf{d} \mid \theta_1, \dots, \theta_n) = \frac{\prod_m f_m(\mathbf{d} \mid \theta_m)}{\sum_c \prod_m f_m(\mathbf{c} \mid \theta_m)},$$

where  $\mathbf{d}$  is a data vector in a discrete space,  $\theta_m$  are the parameters of individual model  $m$ ,  $f_m(\mathbf{d} \mid \theta_m)$  is the value of  $d$  under model  $m$ , and the  $\mathbf{c}$ 's index all possible vectors in the data space. The PoE model is now trained to assign high probability to observed data vectors.

In the KB context, the data vector  $\mathbf{d}$  is always a triple  $\mathbf{d} = (\mathbf{h}, \mathbf{r}, \mathbf{t})$  and the objective is to learn a PoE that assigns high probability to true triples and low probabilities to triples assumed to be false. If  $(\mathbf{h}, \mathbf{r}, \mathbf{t})$  holds in the KB, the pair's vector representations are used as positive training examples. Let  $\mathbf{d} = (\mathbf{h}, \mathbf{r}, \mathbf{t})$ . We can now define one individual expert  $f_{(\mathbf{r}, \mathbf{F})}(\mathbf{d} \mid \theta_{(\mathbf{r}, \mathbf{F})})$  for each (relation type  $\mathbf{r}$ , feature type  $\mathbf{F}$ ) pair.

The specific experts we utilize here are based on simple linear models and the DistMult embedding method.

$$\begin{aligned} f_{(\mathbf{r}, \mathbf{L})}(\mathbf{d} \mid \theta_{(\mathbf{r}, \mathbf{L})}) &= \exp((\mathbf{e}_h * \mathbf{e}_t) \cdot \mathbf{w}^{\mathbf{r}}) \\ f_{(\mathbf{r}, \mathbf{R})}(\mathbf{d} \mid \theta_{(\mathbf{r}, \mathbf{R})}) &= \exp(\mathbf{r}_{(\mathbf{h}, \mathbf{t})} \cdot \mathbf{w}_{\text{rel}}^{\mathbf{r}}) \\ f_{(\mathbf{r}, \mathbf{N})}(\mathbf{d} \mid \theta_{(\mathbf{r}, \mathbf{N})}) &= \exp(\phi(\mathbf{n}_{(\mathbf{h}, \mathbf{t})}) \cdot \mathbf{w}_{\text{num}}^{\mathbf{r}}) \text{ and} \\ f_{(\mathbf{r}', \mathbf{F})}(\mathbf{d} \mid \theta_{(\mathbf{r}', \mathbf{F})}) &= 1 \text{ for all } \mathbf{r}' \neq \mathbf{r} \text{ and } \mathbf{F} \in \{\mathbf{L}, \mathbf{R}, \mathbf{N}\}, \end{aligned}$$

where  $*$  is the element-wise product,  $\cdot$  is the dot product,  $\mathbf{w}^{\mathbf{r}}$ ,  $\mathbf{w}_{\text{rel}}^{\mathbf{r}}$ ,  $\mathbf{w}_{\text{num}}^{\mathbf{r}}$  are the parameter vectors for the latent, relational, and numerical features corresponding to the relation  $\mathbf{r}$ , and  $\phi$  is the radial basis function (RBF) applied element-wise to  $\mathbf{n}_{(\mathbf{h}, \mathbf{t})}$ .  $f_{(\mathbf{r}, \mathbf{L})}(\mathbf{d} \mid \theta_{(\mathbf{r}, \mathbf{L})})$  is equivalent to the exponential of the DISTMULT [35] scoring function but with KBLRN we can use any of the existing KB embedding scoring functions.

The probability for triple  $\mathbf{d} = (\mathbf{h}, \mathbf{r}, \mathbf{t})$  of the PoE model is now

$$p(\mathbf{d} \mid \theta_1, \dots, \theta_n) = \frac{\prod_{\mathbf{F} \in \{\mathbf{R}, \mathbf{L}, \mathbf{N}\}} f_{(\mathbf{r}, \mathbf{F})}(\mathbf{d} \mid \theta_{(\mathbf{r}, \mathbf{F})})}{\sum_c \prod_{\mathbf{F} \in \{\mathbf{R}, \mathbf{L}, \mathbf{N}\}} f_{(\mathbf{r}, \mathbf{F})}(\mathbf{c} \mid \theta_{(\mathbf{r}, \mathbf{F})})},$$

where  $\mathbf{c}$  indexes all possible triples.

For numerical features, an activation function should fire when the difference of values is in a specific *range*. For example, we want the activation to be high when the difference of the birth years between a parent and its child is close to 32.4 years. Commonly used activation functions such as sigmoid or tanh are not suitable here, since they saturate whenever they exceed a certain threshold. For each relation  $\mathbf{r}$  and the  $d_n$  corresponding relevant numerical features, therefore, we apply a radial basis function over the differences of values  $\phi(\mathbf{n}_{(\mathbf{h}, \mathbf{t})}) = [\phi(\mathbf{n}_{(\mathbf{h}, \mathbf{t})}^{(1)}), \dots, \phi(\mathbf{n}_{(\mathbf{h}, \mathbf{t})}^{(d_n)})]$ , where

$$\phi(\mathbf{n}_{(\mathbf{h}, \mathbf{t})}^{(i)}) = \exp\left(\frac{-\|\mathbf{n}_{(\mathbf{h}, \mathbf{t})}^{(i)} - c_i\|_2^2}{\sigma_i^2}\right).$$

This results in the RBF kernel being activated whenever the difference of values is close to the expected value  $c_i$ . We discuss and evaluate several alternative strategies for incorporating numerical features in the experimental section.

### 3.1 Learning

Product of experts are usually trained with contrastive divergence (CD) [13] which relies on an approximation of the gradient of the log-likelihood using a short Markov chain started at the current seen example. The advantage of CD is that the partition function, that is, the denominator of the probability  $p$ , which is intractable to compute, does not need to be approximated. Due to the parameterization of the PoE we have defined here, however, it is not possible to perform CD since there is no way to sample a hidden state given a triple  $\mathbf{d}$ . Hence, instead of using CD, we approximate the partition function by performing negative sampling.

The logarithmic loss for the given training triples  $\mathbf{T}$  is defined as

$$\mathcal{L} = - \sum_{\mathbf{t} \in \mathbf{T}} \log p(\mathbf{t} \mid \theta_1, \dots, \theta_n).$$

To fit the PoE to the training triples, we follow the derivative of the log likelihood of each observed triple  $\mathbf{d} \in \mathbf{T}$  under the PoE

$$\frac{\partial \log p(\mathbf{d} \mid \theta_1, \dots, \theta_n)}{\partial \theta_m} = \frac{\partial \log f_m(\mathbf{d} \mid \theta_m)}{\partial \theta_m} - \frac{\partial \log \sum_c \prod_m f_m(\mathbf{c} \mid \theta_m)}{\partial \theta_m}$$

Now, to approximate the intractable second term of the right hand side of the above equation, we generate for each triple  $\mathbf{d} = (\mathbf{h}, \mathbf{r}, \mathbf{t})$  a set  $\mathbf{E}$  consisting of  $N$  triples  $(\mathbf{h}, \mathbf{r}, \mathbf{t}')$  by sampling exactly  $N$  entities  $\mathbf{t}'$  uniformly at random from the set of all entities. The term

$$\frac{\partial \log \sum_c \prod_m f_m(\mathbf{c} \mid \theta_m)}{\partial \theta_m}$$

is then approximated by the term

$$\frac{\partial \log \sum_{\mathbf{c} \in \mathbf{E}} \prod_m f_m(\mathbf{c} \mid \theta_m)}{\partial \theta_m}.$$

Analogously for the head of the triple. This is often referred to as negative sampling. Figure 2 illustrates the KBLRN framework.

## 4 RELATED WORK

A combination of latent and relational features has been explored by Toutanova et al. [30, 31]. There, a weighted combination of two independently learned models, a latent feature model [35] and a model fed with a binary vector reflecting the presence of paths of length one between the *head* and *tail*, is proposed. These simple relational features aim at capturing association

strengths between pairs of relationships (e.g. contains and contained\_by). Riedel et al. [25] proposed a method that learns implicit associations between pairs of relations in addition to a latent feature model in the context of relation extraction. Gardner et al. [8] modifies the path ranking algorithm (PRA) [15] to incorporate latent representations into models based on random walks in KBs. Gardner et al. [7] extracted relational features other than paths to better capture entity type information. There are a number of recent approaches that combine relational and latent representations by incorporating known logical rules into the embedding learning formulation [26, 10, 18]. Despite its simplicity, KBLRN’s combination of relational and latent representations significantly outperforms all these approaches.

There exists additional work on combining various types of KB features. Nickel et al. [19] proposed a modification of the well-known tensor factorization method RESCAL [21], called ARE, which adds a learnable matrix that weighs a set of metrics (e.g. Common Neighbors) between pairs of entities; Garcia-Duran et al. [6] proposed a combination of latent features, aiming to take advantage of the different interaction patterns between the elements of a triple. The integration of different feature types into relational machine learning models has been previously addressed [1] [17], but not in the context of link prediction in multi-relational graphs.

KBLRN is different to these approaches in that (i) we incorporate numerical features for KB completion, (ii) we propose a unifying end-to-end learning framework that integrates arbitrary relational, latent, and numerical features.

More recent work [23] combines numerical, visual, and textual features by learning feature type specific encoders and using the vector representations in an off-the-shelf scoring function such as DISTMULT. In contrast to this approach, KBLRN combines experts that are specialized to a specific feature type with a product of expert approach. Moreover, by taking the *difference* between numerical features and explicitly modeling relational features that hold *between* head and tail entities, KBLRN incorporates dependencies between modalities of the head and tail entities. These dependencies cannot be captured with a model that only includes modalities of either the head or the tail entity but not both at the same time.

## 5 EXPERIMENTS

We conducted experiments on six different knowledge base completion data sets. Primarily, we wanted to understand for what type of relations numerical features are helpful and what input representation of numerical fea-

Data set	FB15k	FB15k-num	FB15k-237	FB15k-237-num	WN18	FB122
Entities	14,951	14,951	14,541	14,541	40,943	9,738
Relation types	1,345	1,345	237	237	18	122
Training triples	483,142	483,142	272,115	272,115	141,442	91,638
Validation triples	50,000	5,156	17,535	1,058	5,000	9,595
Test triples	59,071	6,012	20,466	1,215	5,000	11,243
Relational features	90,318	90,318	7,834	7,834	14	47

Table 1: Statistics of the data sets.

tures achieves the best results. An additional objective was the comparison to state of the art methods.

## 5.1 Datasets

We conducted experiments on six different data sets: FB15k, FB15k-237, FB15k-num, FB15k-237-num, WN18, and FB122. FB15k [3] and Wordnet (WN) [2] are knowledge base completion data sets commonly used in the literature. The FB15k data set is a representative subset of the Freebase knowledge base. WN18 represents lexical relations between word senses. The two data sets are being increasingly criticized for the frequent occurrence of reverse relations causing simple relational baselines to outperform most embedding-based methods [30]. For these reasons, we also conducted experiments with FB15k-237 a variant of FB15k where reverse relations have been removed [30]. FB122 is a subset of FB15k focusing on relations pertaining to the topics of “people”, “location”, and “sports.” In previous work, a set of 47 logical rules was created for FB122 and subsequently used in experiments for methods that take logical rules into account [10, 18].

The main objective of this paper is to investigate the impact of incorporating numerical features. Hence, we created two additional data set by removing those triples from FB15k’s and FB15k-237’s validation and test sets where numerical features are never used for the triples’ relation type. Hence, the remaining test and validation triples lead to completion queries where the numerical features under consideration are potentially used. We refer to these new data sets as FB15k-num and FB15k-237-num. A similar methodology can be followed to evaluate the performance on a different set of numerical features.

We extracted numerical data from the 1.9 billion triple Freebase RDF dump by mining triples that associate entities to literals of some numerical type. For example, the relation `/location/geocode/latitude` maps entities to their latitude. We performed these extractions for all entities in FB15k but only kept a numerical feature if at least 5 entities had values for it. This resulted in 116 different numerical features and 12,826 entities for which at least one of the numerical features had a value. On average each entity had 2.3 numerical features with a

$n_{(h,t)}$	MR	MRR	Hits@1	Hits@10
sign	231	29.7	20.1	50.1
RBF	<b>121</b>	<b>31.4</b>	<b>21.2</b>	<b>52.3</b>

Table 2: KBLRN for two possible input representations of numerical features for FB15k-237-num.

value. Since numerical data is not available for Wordnet, we do not perform experiments with numerical features for variants of this KB.

Each data set contains a set of triples that are known to be true (usually referred to as positive triples). Statistics of the data sets are provided in Table 1. Since the identifiers for entities and relations have been changed in FB13 [29], we could not extract numerical features for the data set and excluded it from the experiments.

## 5.2 General Set-up

We evaluated the different methods by their ability to answer completion queries of the form  $(h, r, ?)$  and  $(?, r, t)$ . For queries of the form  $(h, r, ?)$ , we replaced the tail by each of the KB’s entities in turn, sorted the triples based on the scores or probabilities returned by the different methods, and computed the rank of the correct entity. We repeated the same process for the queries of type  $(?, r, t)$ . We follow the filtered setting described in [3] which removes correct triples that are different to the target triple from the ranked list. The mean of all computed ranks is the Mean Rank (lower is better) and the fraction of correct entities ranked in the top  $n$  is called hits@ $n$  (higher is better). We also compute the Mean Reciprocal Rank (higher is better) which is an evaluation measure for rankings that is less susceptible to outliers.

We conduct experiments with the scoring function of DISTMULT [35] which is an application of parallel factor analysis to multi-relational graphs. For a review on parallel factor analysis we refer the reader to [12]. We validated the embedding size of KBLRN from the values  $\{100, 200\}$  for all experiments. These values are used in most of the literature on KB embedding methods. For all other embedding methods, we report the original results from the literature or run the authors’ original implementation. For FB15k and FB15k-237, the results for Dist-

	FB15k				FB15k-237			
	MR	MRR	Hits@1	Hits@10	MR	MRR	Hits@1	Hits@10
TRANSE	51	44.3	25.1	76.3	214	25.1	14.5	46.3
DISTMULT	-	65.4	54.6	82.4	-	19.1	10.6	37.6
COMPLEX	-	69.2	59.9	84.0	-	20.1	11.2	38.8
NODE+LINKFEAT	-	<b>82.2</b>	-	87.0	-	23.7	-	36.0
R-GCN+	-	69.6	60.1	84.2	-	24.8	15.3	41.7
CONVE	64	74.5	67.0	87.3	330	30.1	<b>22.0</b>	45.8
without numerical features								
KBL	69	77.4	71.2	87.6	231	30.1	21.4	47.5
KBR	628	78.7	<b>75.6</b>	84.3	2518	18.0	12.8	28.5
KBLR	45	79.0	74.2	87.3	231	30.6	<b>22.0</b>	48.2
with numerical features								
KBLN	66	78.3	72.6	<b>87.8</b>	229	30.4	<b>22.0</b>	47.0
KBRN	598	78.7	<b>75.6</b>	84.2	3303	18.2	13.0	28.7
KBLRN	<b>44</b>	79.4	74.8	87.5	<b>209</b>	<b>30.9</b>	21.9	<b>49.3</b>

Table 3: Results (filtered setting) for KBLRN and state of the art approaches.

	FB15k-num				FB15k-237-num			
	MR	MRR	Hits@1	Hits@10	MR	MRR	Hits@1	Hits@10
TRANSE	25	34.7	5.5	79.9	158	21.8	10.41	45.6
DISTMULT	39	72.6	62.1	89.7	195	26.4	16.4	47.3
without numerical features								
KBL	39	72.6	62.1	89.7	195	26.4	16.4	47.3
KBR	399	84.7	<b>81.6</b>	90.1	3595	23.6	17.8	36.1
KBLR	28	85.3	80.3	92.4	232	29.3	19.7	49.2
with numerical features								
KBLN	32	73.6	63.0	90.7	122	28.6	17.9	51.6
KBRN	68	84.0	80.6	90.0	600	26.1	19.3	39.7
KBLRN	<b>25</b>	<b>85.9</b>	81.0	<b>92.9</b>	<b>121</b>	<b>31.4</b>	<b>21.2</b>	<b>52.3</b>

Table 4: Results (filtered) on the data sets where the test and validation sets are comprised of those triples whose type could potentially benefit from numerical features.

	WN18+rules[10]					FB122-all[10]				
	MR	MRR	Hits@3	Hits@5	Hits@10	MR	MRR	Hits@3	Hits@5	Hits@10
TRANSE	-	45.3	79.1	89.1	93.6	-	48.0	58.9	64.2	70.2
TRANSH	-	56.0	80.0	86.1	90.0	-	46.0	53.7	59.1	66.0
TRANSR	-	51.4	69.7	77.5	84.3	-	40.1	46.4	52.4	59.3
KALE-PRE	-	53.2	86.4	91.9	94.4	-	52.3	61.7	66.2	71.8
KALE-JOINT	-	66.2	85.5	90.1	93.0	-	52.3	61.2	66.4	72.8
COMPLEX	-	<b>94.2</b>	<b>94.7</b>	<b>95.0</b>	<b>95.1</b>	-	64.1	67.3	69.5	71.9
ASR-COMPLEX	-	<b>94.2</b>	<b>94.7</b>	<b>95.0</b>	<b>95.1</b>	-	69.8	71.7	73.6	75.7
KBL	<b>537</b>	80.8	92.5	93.7	94.7	117	69.5	<b>74.6</b>	<b>77.2</b>	<b>80.0</b>
KBR	7113	72.0	72.1	72.1	72.1	2018	54.7	54.7	54.7	54.7
KBLR	588	93.6	94.5	94.8	<b>95.1</b>	<b>113</b>	<b>70.2</b>	74.0	77.0	79.7

Table 5: Results (filtered setting) for KB completion benchmarks where logical rules are provided.

Mult, Complex, and R-GCN+ are taken from [28]; results for the relational baseline Node+LinkFeat are taken from [30]; results for ConvE are taken from [4] and results for TransE were obtained by running the authors’ implementation. For WN18-rules and FB122-all, the results for TransE, TransH, TransR, and KALE are taken from [10], and results for Complex and ASR-Complex are taken from [18]. All methods were tuned for each of the respective data sets.

For KBLRN we used ADAM [14] for parameter learning in a mini-batch setting with a learning rate of 0.001, the categorical cross-entropy as loss function and the number of epochs was set to 100. We validated every 5 epochs and stopped learning whenever the MRR (Mean Reciprocal Rank) values on the validation set decreased. The batch size was set to 512 and the number  $N$  of negative samples to 500 for all experiments. We use the abbreviations  $KB_{suffix}$  to refer to the different instances of

KBLRN. *suffix* is a combination of the letters L (Latent), R (Relational) and N (Numerical) to indicate the inclusion of each of the three feature types.

### 5.3 Automated Generation of Relational and Numerical Features

For the data sets FB15k, FB15k-237, and their numerical versions, we used all relational paths of length one and two found in the training data as relational features. These correspond to the formula types  $(h, r, t)$  (1-hop) and  $\exists x (h, r_1, x) \wedge (x, r_2, t)$  (2-hops). We computed these relational paths with AMIE+ [5], a highly efficient system for mining logical rules from knowledge bases. We used the standard settings of AMIE+ with the exception that the minimal head support was set to 1. With these settings, AMIE+ returns horn rules of the form  $body \Rightarrow (x, r, y)$  that are present for at least 1% of the triples of the form  $(x, r, y)$ . For each relation  $r$ , we used the body of those rules where  $r$  occurs in the head as  $r$ 's relational path features. For instance, given a rule such as  $(x, r_1, z), (z, r_2, y) \Rightarrow (x, r, y)$ , we introduce the relational feature  $\exists x (h, r_1, x) \wedge (x, r_2, t)$  for the relation  $r$ . Table 7 lists a sample of relational features that contributed positively to the performance of KBLRN for specific relation types. For the data sets WN18 and FB122, we used the set of logical formulas previously used in the literature [10]. Using the same set of relational features allows us to compare KBLRN with existing approaches that incorporate logical rules into the embedding learning objective [10, 18].

For each relation  $r$  we only included a numerical feature if, in at least  $\tau = 90\%$  of training triples, both the head and the tail had a value for it. This increases the likelihood that the feature is usable during test time. For  $\tau = 90\%$  there were 105 relations in FB15k for which at least one numerical feature was included during learning, and 33 relations in FB15k-237. With the exception of the RBF parameters, all network weights are initialized following [9]. The parameters of KBLRN's RBF kernels are initialized and fixed to  $c_i = \frac{1}{|\mathbf{T}|} \sum_{(h,r,t) \in \mathbf{T}} \mathbf{n}_{(h,t)}^{(i)}$ , where  $\mathbf{T}$  is the set of training triples  $(h, r, t)$  for the relation  $r$  for which both  $\mathbf{n}_h^{(i)}$  and  $\mathbf{n}_t^{(i)}$  have a value; and  $\sigma_i = \sqrt{\frac{1}{|\mathbf{T}|} \sum_{(h,r,t) \in \mathbf{T}} (\mathbf{n}_{(h,t)}^{(i)} - c_i)^2}$ .

### 5.4 Representations of Numerical Features

We experimented with different strategies for incorporating raw numerical features. For the difference of feature values the simplest method is the application of the sign function. For a numerical attribute  $i$ , the activation is either 1 or  $-1$  depending on whether the difference  $\mathbf{n}_h^{(i)} - \mathbf{n}_t^{(i)}$  is positive or negative. For a more

Relation	KBLR		KBLRN	
	MRR	H@10	MRR	H@10
capital_of	5.7	13.6	<b>14.6</b>	<b>18.2</b>
spouse_of	4.4	<b>0.0</b>	<b>7.9</b>	<b>0.0</b>
influenced_by	7.3	20.9	<b>9.9</b>	<b>26.8</b>

Table 6: MRR and hits@10 results (filtered) for KBLRN with and without numerical features in FB15k-237. Results improve for relations where the difference of the relevant features is approximately normal (see Figure 3).

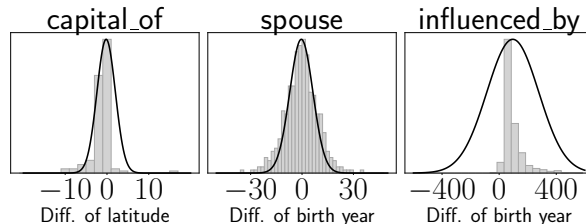


Figure 3: Histograms and fitted RBFs for three representative relations and numerical features.

nuanced representation of differences of numerical features, a layer of RBF kernels is a suitable choice since the activation is here highest in a particular range of input values. The RBF kernel might not be appropriate, however, in cases where the underlying distribution is not normal.

To evaluate different input representations, we conducted experiments with KBLRN on the FB15k-237-num data set. Table 2 depicts the KB completion performance of two representation strategies for the difference of head and tail values. Each row corresponds to one evaluated strategy. “sign” stands for applying the sign function to the difference of numerical feature values. RBF stands for using an RBF kernel layer for the differences of numerical feature values. All results are for the FB15k-237-num test triples.

The RBF kernels outperform the sign functions significantly. This indicates that the difference of feature values is often distributed normally and that having a region of activation is beneficial. Given these results, we use the RBF input layer for  $\mathbf{n}_{(h,t)}$  for the remainder of the experiments.

### 5.5 Comparison to State of the Art

For the standard benchmark data sets FB15k and FB15k-237, we compare KBLRN with TRANSE, DISTMULT, COMPLEX [32], R-GCN+ [28], and ConvE [4].

Table 3 lists the KB completion results. KBLRN is competitive with state of the art knowledge base comple-

Relational Feature	Triple (h, r, t)
$\exists x (h, \text{containedby}, x) \wedge (t, \text{locations\_in\_this\_time\_zone}, x)$	(h, time\_zone, t)
$\exists x (t, \text{prequel}, x) \wedge (x, \text{character}, h)$	(h, character\_in\_film, t)
$\exists x (x, \text{cause\_of\_death}, h) \wedge (x, \text{cause\_of\_death}, t)$	(h, includes\_causes\_of\_death, t)

Table 7: Relational features found by AMIE+ that positively contributed to the performance of KBLRN for the particular relation type holding between a head entity h and tail entity t.

	MR		MRR	
	ONE	MANY	ONE	MANY
KBLR	<b>42</b>	201	<b>74.9</b>	21.0
KBLRN	60	<b>135</b>	74.2	<b>22.8</b>

Table 8: MR and MRR results (filtered) on FB15k-237-num based on the cardinality of the test queries.

tion methods on FB15k and significantly outperforms all other methods on the more challenging FB15k-237 data set. Since the fraction of triples that can potentially benefit from numerical features is very small for FB15k, the inclusion of numerical features is only slightly beneficial. For FB15k-237, however, the numerical features significantly improve the results.

For the numerical versions of FB15k and FB15k-237, we compared KBLRN to TransE and DistMult. Table 4 lists the results for the KB completion task on these data sets. KBLRN significantly outperforms the KB embedding approaches. The positive impact of including the numerical features is significant.

For the data sets WN18-rules and FB122-all we compared KBLRN to KB embedding methods TransE, TransR [16], TransH [34], and ComplEx [32] as well as state of the art approaches for incorporating logical rules into the learning process. The experimental set-up is consistent with that of previous work. Table 5 lists the results for the KB completion task on these data sets. KBLRN combining relational and latent representations significantly outperforms existing approaches on FB122 with exactly the same set of rules. This provides evidence that KBLRN’s strategy to combine latent and relational features is effective despite its simplicity relative to existing approaches. For WN18+rules, KBLRN is competitive with ComplEx, the best performing method on this data set. In addition, KBLRN’s performance improves significantly when relational and latent representations are combined. In contrast, ASR-COMPLEX is not able to improve the results of ComplEx, its underlying latent representation.

## 5.6 The Impact of Numerical Features

The integration of numerical features improves KBLRN’s performance significantly. We performed

	AUC-PR	MR	MRR
TRANSE	0.837	231	26.5
KBLR	0.913	94	66.8
KBLRN	<b>0.958</b>	<b>43</b>	<b>70.8</b>

Table 9: AUC-PR, MR, and MRR results for the completion query (USA, /location/contains, ?).

several additional experiments so as to gain a deeper understanding of the impact numerical features have.

Table 6 compares KBLRN’s performance with and without integrating numerical features on three relations. The performance of the model with numerical features is clearly superior for all three relationships (capital\_of, spouse and influenced\_by). Figure 3 shows the normalized histograms for the values  $n_{(h,t)}$  for these relations. We observe the differences of feature values are approximately normal.

Following previous work [3], we have classified each test query of FB15k-237-num as either ONE or MANY, depending on the whether one or many entities can complete that query. For the queries labeled ONE the model without numerical features shows a slightly worse performance with respect to the model that makes use of them, whereas for the queries labeled MANY, KBLRN significantly outperforms KBLR in both MR and MRR.

A well-known finding is the lack of completeness of FB15k and FB15k-237. This results in numerous cases where the correct entity for a completion query is not contained in the ground truth (neither in the training, nor test, nor validation data set). This is especially problematic for queries where a large number of entities are correct completions. To investigate the actual benefits of the numerical features we carried out the following experiment: We manually determined all correct completions for the query (USA, /location/contains, ?). We ended up with 1619 entities that correctly complete the query. FB15k-237 contains only 954 of these correct completions. With a complete ground truth, we can now use the precision-recall area under the curve (PR-AUC) metric to evaluate KB completion methods [21, 19, 6]. A high PR-AUC represents both high recall and high precision. Table 9 lists the results for the different methods. KBLRN with numerical features consistently and significantly outperformed all other approaches.

## 6 CONCLUSIONS

We introduced KBLRN, a class of machine learning models that, to the best of our knowledge, is the first proposal aiming at integrating relational, latent, and continuous features of head and tail entities in KBs into a single end-to-end differentiable framework. KBLRN outperforms state of the art KB completion methods on a range of data sets. We show that the inclusion of numerical features is beneficial for KB completion tasks.

Future work will primarily study instances of experts that can combine different numerical features such as life expectancy and latitude. Furthermore, combinations of multiple different embedding methods, which was shown to be beneficial in recent work [33], is also possible with our PoE approach.

## References

- [1] D. Andrzejewski, X. Zhu, M. Craven, and B. Recht. A framework for incorporating general domain knowledge into latent dirichlet allocation using first-order logic. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 1171, 2011.
- [2] A. Bordes, X. Glorot, J. Weston, and Y. Bengio. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2):233–259, 2014.
- [3] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*, pages 2787–2795, 2013.
- [4] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel. Convolutional 2d knowledge graph embeddings. *arXiv preprint arXiv:1707.01476*, 2017.
- [5] L. Galárraga, C. Teflioudi, K. Hose, and F. M. Suchanek. Fast rule mining in ontological knowledge bases with amie+. *The VLDB Journal*, 24(6):707–730, 2015.
- [6] A. Garcia-Duran, A. Bordes, N. Usunier, and Y. Grandvalet. Combining two and three-way embeddings models for link prediction in knowledge bases. *arXiv preprint arXiv:1506.00999*, 2015.
- [7] M. Gardner and T. M. Mitchell. Efficient and expressive knowledge base completion using sub-graph feature extraction. In *EMNLP*, pages 1488–1498, 2015.
- [8] M. Gardner, P. P. Talukdar, J. Krishnamurthy, and T. Mitchell. Incorporating vector space similarity in random walk inference over knowledge bases. In *EMNLP*, 2014.
- [9] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, volume 9, pages 249–256, 2010.
- [10] S. Guo, Q. Wang, L. Wang, B. Wang, and L. Guo. Jointly embedding knowledge graphs and logical rules. In *EMNLP*, pages 192–202, 2016.
- [11] K. Guu, J. Miller, and P. Liang. Traversing knowledge graphs in vector space. In *EMNLP*, 2015.
- [12] R. A. Harshman and M. E. Lundy. Parafac: Parallel factor analysis. *Computational Statistics & Data Analysis*, 18(1):39–72, 1994.
- [13] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.
- [14] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [15] N. Lao, T. Mitchell, and W. W. Cohen. Random walk inference and learning in a large scale knowledge base. In *EMNLP*, pages 529–539, 2011.
- [16] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, pages 2181–2187, 2015.
- [17] X. Liu and J. S. Baras. Trust-aware crowdsourcing with domain knowledge. In *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*, pages 2913–2918. IEEE, 2015.
- [18] P. Minervini, T. Demeester, T. Rocktäschel, and S. Riedel. Adversarial sets for regularised neural link predictors. In *UAI*, 2017.
- [19] M. Nickel, X. Jiang, and V. Tresp. Reducing the rank in relational factorization models by including observable patterns. In *NIPS*, pages 1179–1187, 2014.
- [20] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2016.
- [21] M. Nickel, V. Tresp, and H.-P. Kriegel. A three-way model for collective learning on multi-relational data. In *ICML*, pages 809–816, 2011.



- [22] M. Niepert. Discriminative gaifman models. In *NIPS*, pages 3405–3413, 2016.
- [23] P. Pezeshkpour, L. Chen, and S. Singh. Embedding multimodal relational data. In *Workshop on Automated Knowledge Base Construction (AKBC)*, 2017.
- [24] L. D. Raedt, K. Kersting, S. Natarajan, and D. Poole. Statistical relational artificial intelligence: Logic, probability, and computation. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 10(2):1–189, 2016.
- [25] S. Riedel, L. Yao, A. McCallum, and B. M. Marlin. Relation extraction with matrix factorization and universal schemas. 2013.
- [26] T. Rocktäschel, S. Singh, and S. Riedel. Injecting logical background knowledge into embeddings for relation extraction. In *HLT-NAACL*, pages 1119–1129, 2015.
- [27] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, 3rd edition, 2009.
- [28] M. Schlichtkrull, T. Kipf, P. Bloem, R. v. d. Berg, I. Titov, and M. Welling. Modeling relational data with graph convolutional networks. *arXiv preprint arXiv:1703.06103*, 2017.
- [29] R. Socher, D. Chen, C. D. Manning, and A. Y. Ng. Reasoning with neural tensor networks for knowledge base completion. In *NIPS*, pages 926–934, 2013.
- [30] K. Toutanova and D. Chen. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, 2015.
- [31] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, and M. Gamon. Representing text for joint embedding of text and knowledge bases. In *EMNLP*, pages 1499–1509, 2015.
- [32] T. Trouillon, J. Welbl, S. Riedel, E. Gaussier, and G. Bouchard. Complex embeddings for simple link prediction. In *ICML*, pages 2071–2080, 2016.
- [33] Y. Wang, R. Gemulla, and H. Li. On multi-relational link prediction with bilinear models. *arXiv preprint arXiv:1709.04808*, 2017.
- [34] Z. Wang, J. Zhang, J. Feng, and Z. Chen. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, pages 1112–1119, 2014.
- [35] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng. Learning multi-relational semantics using neural-embedding models. *arXiv preprint arXiv:1411.4072*, 2014.

# Probabilistic AND-OR Attribute Grouping for Zero-Shot Learning

Yuval Atzmon\*

\*Gonda Brain Research Center,  
Bar-Ilan University, Israel  
yuval.atzmon@biu.ac.il

Gal Chechik\*,\*\*

\*\*Google AI,  
Mountain View, CA  
gal.chechik@biu.ac.il

## Abstract

In zero-shot learning (ZSL), a classifier is trained to recognize visual classes without any image samples. Instead, it is given semantic information about the class, like a textual description or a set of attributes. Learning from attributes could benefit from explicitly modeling structure of the attribute space. Unfortunately, learning of general structure from empirical samples is hard with typical dataset sizes.

Here we describe LAGO<sup>1</sup>, a probabilistic model designed to capture natural soft *and-or* relations across *groups of attributes*. We show how this model can be learned end-to-end with a deep attribute-detection model. The soft group structure can be learned from data jointly as part of the model, and can also readily incorporate prior knowledge about groups if available. The soft and-or structure succeeds to capture meaningful and predictive structures, improving the accuracy of zero-shot learning on two of three benchmarks.

Finally, LAGO reveals a unified formulation over two ZSL approaches: DAP (Lampert et al., 2009) and ESZSL (Romera-Paredes & Torr, 2015). Interestingly, taking only one singleton group for each attribute, introduces a *new* soft-relaxation of DAP, that outperforms DAP by ~40%.



The *Mourning Warbler* has an olive-green wing, a grey-blue head and pink feet.

**A "hard" logical expression**  
AND [OR(wing\_color:olive, wing\_color:green),  
OR (forehead\_color:grey, forehead\_color:blue),  
OR (leg\_color:pink)]

**A "soft" model:**  
[ weight<sub>wing:olive</sub> P(wing:olive|image) +  
weight<sub>wing:green</sub> P(wing:green|image) +  
weight<sub>wing:red</sub> P(wing:red|image) + ... ] x  
...  
x [weight<sub>leg:pink</sub> P(leg:pink|image) + ...]

Figure 1: Classifying a bird species based on attributes from (Wah et al., 2011). The *Mourning Warbler* can be distinguished from other species by a combination of a grey head and olive-green underparts. Both human raters and machine learning models may confuse semantically-similar attributes like olive or green wings. These attributes naturally cluster into "OR" groups, where we aim to recognize this species if the wing is labeled as either green or olive. The LAGO model (Eq. 4) weighs attributes detection, inferring classes based on within-group soft-OR and across-groups soft-AND. In general, OR-groups include alternative choices of a property ( $wing\_color:\{red, olive, green\}$ ) and soft-OR allows to weigh down class-irrelevant attributes (here,  $wing:red$ ).

## 1 INTRODUCTION

People can easily learn to recognize visual entities based on a handful of semantic attributes. For example, we can recognize a bird based by its visual features (long beak, red crown), or find a location based on a language description (a 2-stories brick town house). Unfortunately, when training models that use such semantic features, it is typically very hard to leverage semantic information effectively. With semantic features, the input space has rich and complex structure, due to nontrivial interactions and logical relations among attributes. For example, the color of petals may be red or blue but rarely both, while the size of a bird is often not indicative of its color.

Taking into account the semantics of features or at-

<sup>1</sup> A video of the highlights, and code is available at: <http://chechiklab.biu.ac.il/~yuvval/LAGO/>

tributes becomes crucial when no training samples are available. This learning setup, called zero-shot learning (ZSL) is the task of learning to recognize objects from classes without any image samples to train on. (Lampert et al., 2009; Farhadi et al., 2009; Palatucci et al., 2009; Xian et al., 2017b). Instead, learning is based on semantic knowledge about the classes (Socher et al., 2013; Elhoseiny et al., 2013; Berg et al., 2010), like in the case of attribute sharing (Lampert et al., 2009, 2014). Here, the training and test classes are accompanied by a set of predefined attributes, like "A Zebra is striped" or "A Hummingbird has a long bill", provided by human experts. Then, a classifier is trained to detect these attributes in images (Ferrari & Zisserman, 2008), and test images are classified by detecting attributes and mapping to test classes based on the expert knowledge.

Broadly speaking, approaches to ZSL with attributes can be viewed as learning a compatibility function  $f(Attr(image), Attr(class))$  between an attribute-based representation of an image and an attribute-based representation of classes (Romera-Paredes & Torr, 2015; Akata et al., 2016; Frome et al., 2013; Akata et al., 2015). Here, the attributes of a class are provided by (possibly several) experts, the image attributes are automatically detected, and one aims to learn a scoring function that can find the class whose attributes are most compatible with an image. Most ZSL approaches represent attributes as embedded in a "flat" space, Euclidean or Simplex, but flat embedding may miss important semantic structures. Other studies aimed to learn a structured scoring function, for example using a structured graphical model over the attributes (Wang & Ji, 2013). Unfortunately, learning complex structures of probabilistic models from data requires large datasets, which are rarely available.

Here we put forward an intermediate approach: We use training classes to learn a simple structure that can capture simple (soft) and-or logical relations among attributes. More concretely, after mapping an image to attributes, we aggregate attributes into groups using a soft OR (weighted-sum), and then score a class by taking a soft AND (product of probabilities) over group activations (Figure 2). While the attributes are predefined and provided by experts, the soft groups are learned from the training data.

The motivation for learning the and-or structure becomes clear when observing how attributes tend to cluster naturally into semantically-related groups. For example, descriptions of bird species in the CUB dataset include attributes like  $\{wing-color:green, wing-color:olive, wing-color:red\}$  (Wah et al., 2011). As another example, animal attributes in (Lampert et al., 2009) include  $\{texture:hairless, texture:tough-skin\}$ . In these two ex-

amples, the attributes are semantically related, and raters (or a classifier) may mistakenly interchange them, as evident by how Wikipedia describes the Mourning Warbler (Figure 1) as having "olive-green underparts". In such cases, it is natural to model attribute structure as a soft OR relation over attributes ("olive" or "green") in a group ("underparts"). It is also natural to apply a soft AND relation across groups, since a class is often recognized by a set of necessary properties.

We describe LAGO, "*Learning Attribute Grouping for 0-shot learning*", a new zero-shot probabilistic model that leverages and-or semantic structure in attribute space. LAGO achieves new state-of-the-art result on CUB and AWA2 (Lampert et al., 2009), and competitive performance on SUN (Patterson & Hays, 2012). Interestingly, when considering two extremes of attribute grouping, LAGO becomes closely related to two important ZSL approaches. First, in the case of a single group (all OR), LAGO is closely related to ESZSL (Romera-Paredes & Torr, 2015). At the opposite extreme where each attribute forms a singleton group, (all AND), LAGO is closely related to DAP (Lampert et al., 2009). LAGO therefore reveals an interesting unified formulation over seemingly unrelated ZSL approaches.

Our paper makes the following novel contributions. We develop a new probabilistic model that captures soft logical relations over semantic attributes, and can be trained end-to-end jointly with deep attribute detectors. The model learns attribute grouping from data, and can effectively use domain knowledge about semantic grouping of attributes. We further show that it outperforms competing methods on two ZSL benchmarks, CUB and AWA2, and obtain comparable performance on another benchmark (SUN). Finally, LAGO provides a unified probabilistic framework, where two previous important ZSL methods approximate extreme cases of LAGO.

## 2 RELATED WORK

Zero-shot-learning with attributes attracted significant interest recently (Xian et al., 2017b; Fu et al., 2017). One influential early works is *Direct Attribute Prediction* (DAP), which takes a Bayesian approach to predict unseen classes from binary attributes (Lampert et al., 2009). In DAP, a class is predicted by the product of attribute-classifier scores, using a hard-threshold over the semantic information of attribute-to-class mapping. DAP is related in an interesting way to LAGO. We show below that DAP can be viewed as a hard-threshold special case of LAGO where each group consists of a single attribute.

Going beyond a flat representation of attributes, several studies modeled structure among attributes. Wang & Ji (2013) learned a Bayesian network over attribute space

that captures object-dependent and object-independent relationships. Jiang et al. (2017) learned latent attributes that preserve semantics and also provide discriminative combinations of given semantic attributes. Structure in attribute space was also used to improve attribute prediction: Jayaraman et al. (2014) leveraged side information about semantic relatedness of attributes in given groups and proposed a multi-task learning framework, where same-group attributes are encouraged to share low-level features. In Park & Zhu (2015); Park et al. (2017), the authors propose an AND-OR grammar model (Zhu & Mumford, 2006), to jointly represent both the object parts and their semantic attributes within a unified compositional hierarchy. For that, they decompose an object to its constituent parts with a parse tree. In their model, the tree nodes (the parts) constitute an AND relation, and each OR-node points to alternative sub-configurations

Our approach resonates with *Markov Logic Networks* (MLN), (Richardson & Domingos, 2006) and *Probabilistic Soft Logic* (PSL), (Kimmig et al., 2012; Bach et al., 2017). It shares the idea of modeling domain knowledge using soft logical relations. Yet, LAGO formulation provides a complementary point of view: (1) LAGO derivation reveals the probabilistic meaning of every soft weight of the logical relation Eq. (3), and offers a principled way to set priors when deriving the soft logical expression. (2) The step-by-step derivation reveals which approximations are taken when mapping features to classes with soft logical relations. (3) Logical relations are incorporated in LAGO as part of an end-to-end deep network. PSL and MLN use Markov random field with logical relations as constraints or potentials.

The study of ZSL goes beyond learning with attributes (Changpinyo et al., 2016; Tsai et al., 2017b; Morgado & Vasconcelos, 2017; Rohrbach et al., 2011; Al-Halah & Stiefelhagen, 2015; Zhang et al., 2017; Ye & Guo, 2017; Tsai et al., 2017a; Xu et al., 2017; Li et al., 2017; Zhang & Koniusz, 2018). Recently, Zhang & Koniusz (2018) described a kernel alignment approach, mapping images to attribute space such that projected samples match the distribution of attributes in terms of a nonlinear kernel. Another popular approach to ZSL learns a bi-linear compatibility function  $F(\mathbf{x}, y; W) = \theta(\mathbf{x})^\top W \phi(y)$  to match visual information  $\theta(\mathbf{x})$  with semantic information  $\phi(y)$  (Romera-Paredes & Torr, 2015; Akata et al., 2016; Frome et al., 2013; Akata et al., 2015). In this context, most related to our work is ESZSL (Romera-Paredes & Torr, 2015), which uses a one-hot encoding of class labels to define a mean-squared-error loss function. This allows ESZSL to have a closed-form solution where reaching the optimum is guaranteed. We show below that ESZSL is closely related to a special case of LAGO where all attributes are assigned to a single group.

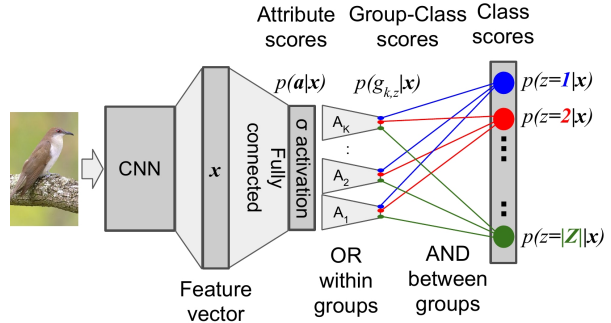


Figure 2: LAGO network architecture (Sec 3). Image features are extracted by a deep ConvNet and fed to a FC sigmoid layer ( $\sigma$ ) that outputs a prediction score  $p(a_m|x)$  for each binary attribute  $a_m$ . Attribute scores are grouped into  $K$  soft groups, and mapped to a set  $\{g_{k,z}\}$  of  $K \times |Z|$  binary classifiers, according to a soft-OR Eq. (3). Finally, a class is computed by the soft product of all group-scores for that class, approximating a conjunction (AND). In the diagram, each colored circle represents a classifier score for a separate class.

The current work focuses on a new architecture for ZSL with attributes. Other aspects of ZSL, including feature selection (Guo et al., 2018) and data augmentation (Mishra et al., 2018; Arora et al., 2018; Xian et al., 2018), can improve accuracy significantly, but are orthogonal to the current work.

### 3 A PROBABILISTIC AND-OR MODEL

**The Problem Setup:** Following the notations of (Lampert et al., 2009), we are given a set of labeled training images  $(\mathbf{x}_i \in \mathcal{X}, z_i \in \mathcal{Z})$  drawn from a distribution  $D$ . Each image  $\mathbf{x}$  is accompanied by a vector of binary attributes  $\mathbf{a} \in \{True, False\}^{|A|}$ ,  $\mathbf{a} = (a_1, \dots, a_m, \dots, a_{|A|})$ , where  $a_m = True$  if the image has attribute  $a_m$ . We are also given a set of class "descriptions" in the form class-conditioned attribute distribution  $p(\mathbf{a}|z)$ . In practice, the descriptions are often collected separately per attribute ( $m$ ), and only the marginals  $p(a_m|z)$ ,  $\forall m$  are available.

At training time, we aim to learn a classifier that predicts a class  $z$  of an image  $\mathbf{x}$  by first learning to predict the attributes  $\mathbf{a}$ , and then use  $p(a_m|z)$ ,  $\forall m$  to predict a class  $z$  based on the attributes.

At inference time (the zero-shot phase), we are given images  $\mathbf{x}$  from new unseen classes with labels  $y \in \mathcal{Y}$ , and together with their class descriptions  $p(a_m|y)$ ,  $\forall m$ . We similarly predict the class  $y$  by first predicting attributes  $\mathbf{a}$  and then use  $p(a_m|z)$ ,  $\forall m$  to predict a class  $y$  based on the attributes.

**Model Overview:** The LAGO model (Figures 1, 2) learns a soft logical AND-OR structure over semantic attributes. It can be viewed as a concatenation of three

mapping steps  $\mathcal{X} \rightarrow \mathcal{A} \rightarrow \mathcal{G} \rightarrow \mathcal{Z}$ . First, **attribute predictions**: an image  $\mathbf{x}$  is mapped to a vector of attribute detection probabilities  $f_W^1 : \mathcal{X} \rightarrow \mathcal{A}$ ,  $\mathcal{A} = [0, 1]^{|A|}$ . The mapping parameters  $W$  determine the weights of the attribute detectors and are learned from labeled training data. Second, **weighted-OR group scores**: Attribute probabilities are mapped to  $K$  groups. Each group calculates a class-dependent weighted-OR over the  $|Z|$  classes  $f_{U,V}^2 : \mathcal{A} \rightarrow \mathcal{G}$ ,  $\mathcal{G} = [0, 1]^{K \times |Z|}$ . The mapping parameters  $U$  are the distributions  $p(\mathbf{a}|z)$  provided with each class; The mapping parameters  $V$  determine how attributes are grouped and are learned from data. Last, **soft-AND group conjunction**: Per-group scores are mapped to class detection probabilities by a soft-AND, approximating group conjunction.  $f^3 : \mathcal{G} \rightarrow \mathcal{Z}$ ,  $\mathcal{Z} = [0, 1]^{|Z|}$ . The parameters  $W, V$  are learned jointly to minimize a regularized loss with a regularizer  $R$ :

$$\min_{W,V} \text{loss}(f^3(f_{U,V}^2(f_W^1(\mathbf{x}_i))), z_i) + R(W, V) \quad . \quad (1)$$

The key idea in the proposed approach is to define a layer of binary classifiers  $g_{k,z}$ , each evaluating a class  $z$  based only on a subset  $G_k \subset \mathcal{A}$  of attributes. For example, for bird-species recognition, one classifier may detect a *Mourning Warbler* based on wing colors and another based on bill shapes. In this example, each of the classifiers output the probability  $p(g_{k,z} = \text{True}|\mathbf{a})$  that the image has a *Mourning Warbler*, but based on different subsets of attributes. The partition of attributes to subsets is shared across classes, hence with  $K$  subsets we have  $K \times |Z|$  binary classifiers. We also define  $\mathbf{a}_k$  to be the vector of attributes detections for  $G_k$ ,  $\mathbf{a}_k \in \{T, F\}^{|G_k|}$ . For clarity, we first derive the algorithm for groups that are fixed (not learned) and hard (non-overlapping). Section 3.1 then generalizes the derivation to soft learned groups.

Consider now how we compute  $p(g_{k,z} = T|\mathbf{x})$ . According to the Markov property, it equals  $\sum_{\mathbf{a}_k} p(g_{k,z}|\mathbf{a}_k)p(\mathbf{a}_k|\mathbf{x})$ , but computing this sum raises several challenges. First, since the number of possible patterns in a group grows exponentially with its size, the summation becomes prohibitively large when attribute groups are large. Second, estimating  $p(g_{k,z}|\mathbf{a}_k)$  from data may also be hard because the number of samples is often limited. Finally, description information is often available for the marginals only,  $(z, a_m)$ , rather than the full distribution  $(z, \mathbf{a}_k)$ . We now discuss a model that addresses these constraints.

**The Within-Group Model  $\mathcal{A} \rightarrow \mathcal{G}$ :** We now show how one can compute  $p(g_{k,z}|\mathbf{x})$  efficiently by treating attributes within group as obeying a soft OR relation. As discussed above, OR relations are in good agreement with how real-world classes are described using hard

semantic attributes, because a single property (a group like *beak-shape*) may be mapped to several semantically-similar attributes (*pointy, long*).

Formally, we first define a complementary attribute per group,  $\tilde{a}_k = (\bigcup_{m \in G_k} a_m)^c$ , handling the case where no attributes are detected or described, and accordingly define  $G'_k = G_k \cup \tilde{a}_k$ . We then use the identity  $p(A) = p(A, B) + p(A, B^c)$  to partition  $p(g_{k,z} = T|\mathbf{x})$  to a union (OR) of its contributions from each of its attributes. Specifically,  $p(g_{k,z} = T|\mathbf{x}) = p(g_{k,z} = T, \bigcup_{m \in G_k} a_m = T|\mathbf{x}) + p(g_{k,z} = T, \tilde{a}_k = T|\mathbf{x}) = p(\bigcup_{m \in G'_k} (g_{k,z} = T, a_m = T)|\mathbf{x})$ . Using this equality and approximating attributes within a group as being mutually exclusive, we have

$$p(g_{k,z} = T|\mathbf{x}) \approx \sum_{m \in G'_k} p(g_{k,z} = T, a_m = T|\mathbf{x}). \quad (2)$$

To rewrite this expression in terms of class descriptions  $p(a_m|z)$  we take the following steps. First, the Markov chain  $\mathcal{X} \rightarrow \mathcal{A} \rightarrow \mathcal{G}$  gives  $p(g_{k,z} = T, a_m|\mathbf{x}) = p(g_{k,z} = T|a_m)p(a_m|\mathbf{x})$ . Second, we note that by the definition of  $g_{k,z}$ ,  $p(g_{k,z}|\mathbf{a}_k) = p(z|\mathbf{a}_k)$ , because  $g_{k,z}$  is the classifier of  $z$  based on  $\mathbf{a}_k$ . This yields  $p(g_{k,z} = T, a_m) = p(z, a_m)$  by marginalization. Applying Bayes to the last identity gives  $p(g_{k,z} = T|a_m) = p(a_m|z)p(g_{k,z} = T)/p(a_m)$  (more details in Supplementary Material D.1). Finally, combining it with the expression for  $p(g_{k,z} = T, a_m|\mathbf{x})$  and with Eq. (2) we can express  $p(g_{k,z} = T|\mathbf{x})$  as

$$p(g_{k,z} = T|\mathbf{x}) \approx p(g_{k,z} = T) \sum_{m \in G'_k} \frac{p(a_m = T|z)}{p(a_m = T)} p(a_m = T|\mathbf{x}). \quad (3)$$

**Conjunction of Groups  $\mathcal{G} \rightarrow \mathcal{Z}$ :** Next, we derive an expression of the conditional probability of classes  $p(z|\mathbf{x})$  using soft-conjunction of group-class classifiers  $g_{k,z}$ . Using the Markov property  $\mathcal{X} \rightarrow \mathcal{A} \rightarrow \mathcal{G} \rightarrow \mathcal{Z}$ , and denoting  $g_{1,z} \dots g_{K,z}$  by  $\mathbf{g}_z$ , we can write  $p(z|\mathbf{x}) = \sum_{\mathbf{g}_z} p(z|\mathbf{g}_z)p(\mathbf{g}_z|\mathbf{x})$ . We show on Supplementary D.2, that making a similar approximation as in DAP (Lampert et al., 2009), for groups instead of attributes, yields Eq. (A.10):  $p(z|\mathbf{x}) \approx p(z) \prod_{k=1}^K \frac{p(g_{k,z} = T|\mathbf{x})}{p(g_{k,z} = T)}$ . Combining it with Eq. (3), we conclude

$$p(z|\mathbf{x}) \approx p(z) \prod_{k=1}^K \left[ \sum_{m \in G'_k} \frac{p(a_m = T|z)}{p(a_m = T)} p(a_m = T|\mathbf{x}) \right]. \quad (4)$$

### 3.1 SOFT GROUPS:

The above derivation treated attribute groups as hard: deterministic and non-overlapping. We now discuss the

more general case where attributes are *probabilistically* assigned to groups.

We introduce a soft group-membership variable  $\Gamma_{m,k} = p(m \in G'_k)$ , yielding a soft version of Eq. (3)

$$p(g_{k,z} = T|\mathbf{x}) \approx p(g_{k,z} = T) \sum_{m=1}^{|\mathcal{A}|} \Gamma_{m,k} \frac{p(a_m = T|z)}{p(a_m = T)} p(a_m = T|\mathbf{x}), \quad (5)$$

where each row of  $\Gamma$  represents a distribution over  $K$  groups per attribute in the simplex  $\Delta^K$ . Hard grouping is a special case of this model where all probability mass is assigned to a single group for each row of  $\Gamma$ . The full derivation is detailed in Supplementary Material (D.3).

### 3.2 LEARNING

LAGO has three sets of parameters learned from data.

First, a matrix  $W$  parametrizes the mapping  $f_W^1 : \mathbf{x} \rightarrow [0, 1]^{|\mathcal{A}|}$  from image features to attribute detection probabilities  $p(a_m|\mathbf{x})$ . This mapping is implemented as a fully-connected layer with sigmoid activation over image features extracted from ResNet-101.

Second, a matrix  $U$ , where its entry  $U_{m,z}$  parametrizes the class-level description  $p(a_m|z)$ . When attribute ratings are given per image, we estimate  $U$  using maximum likelihood from co-occurrence data over attributes and classes.

Third, a matrix  $V_{|\mathcal{A}| \times K}$  parametrizes the soft group assignments  $\Gamma_{|\mathcal{A}| \times K}$ , such that each row  $m$  maintains  $\Gamma_{(m,:)} = \text{softmax}(\zeta V_{(m,:)})$ , where  $\zeta \in \mathbb{R}^+$  is a smoothing coefficient. This parametrization allows taking arbitrary gradient steps over  $V$ , while guaranteeing that each row of  $\Gamma$  corresponds to a probability distribution in the simplex  $\Delta^K$ .

Since  $W$  and  $V$  are shared across all classes, they are learned over the training classes and transferred to the test classes at (zero-shot) inference time. They are learned end-to-end by applying cross-entropy loss over the outputs of Eq. (4) normalized by their sum across classes (forcing a unit sum of class predictions). As in (Romera-Paredes & Torr, 2015), the objective includes two regularization terms over  $W$ : A standard  $L_2$  regularizer  $\|W\|_{Fro}^2$  and a term  $\|WU\|_{Fro}^2$ , which is equivalent for an ellipsoid Gaussian prior for  $W$ . For the "LAGO-Semantic-Soft" learning-setup (Section 4) we introduce an additional regularization term  $\|\Gamma_{(V)} - \Gamma_{(V_{SEM})}\|_{Fro}^2$ , pushing the solutions closer to known semantic hard-grouping  $\Gamma_{(V_{SEM})}$ . Finally, we optimize the loss:

$$L(W, U, V, Z, A, X) = \text{CXE}_{p(z|\mathbf{x}; W, U, V)}(X, Z) + \alpha \text{BXE}_{p(a|\mathbf{x}; W)}(X, A) + \beta \|W\|_{Fro}^2 + \lambda \|WS\|_{Fro}^2 + \psi \|\Gamma_{(V)} - \Gamma_{(V_{SEM})}\|_{Fro}^2, \quad (6)$$

where CXE is the categorical cross-entropy loss for  $p(z|\mathbf{x})$ , BXE is the binary cross-entropy loss for  $p(a|\mathbf{x})$ ,  $X, Z$  and  $A$  denote the training samples, labels and attribute-labels. Per-sample attribute labels are provided as their empirical mean per class. In practice, we set  $\alpha = 0$  (See Section 4.2) and cross-validate to select the values of  $\beta$ ,  $\lambda$  and  $\psi$  when relevant.

### 3.3 INFERENCE

At inference time, we are given images from new classes  $y \in \mathcal{Y}$ . As with the training data, we are given semantic information about the classes in the form of the distribution  $p(a_m|y)$ . In practice, we are often not given that distribution directly, but instead estimate it using maximum likelihood from a set of labeled attribute vectors.

To infer the class of a given test image  $\mathbf{x}$ , we plug  $p(a_m|y)$  estimates instead of  $p(a_m|z)$  in Eq. (4), and select the class  $y$  that maximizes Eq. (4).

### 3.4 DAP, ESZSL AS SPECIAL CASES OF LAGO

LAGO encapsulates similar versions of two other zero-shot learning approaches as extreme cases: DAP (Lampert et al., 2009), when having each attribute in its own singleton group ( $K = |\mathcal{A}|$ ), and ESZSL (Romera-Paredes & Torr, 2015), when having one big group over all attributes ( $K = 1$ ).

Assigning each single attribute  $a_m$  to its own singleton group reduces Eq. (4) to Eq. (A.20) (details in Supplementary D.4). This formulation is closely related to DAP. When expert annotations  $p(a_m = T|z)$  are thresholded to  $\{0, 1\}$  and denoted by  $a_m^z$ , Eq. (A.20) become the DAP posterior Eq. (A.22). This makes the singletons variant a **new** soft relaxation of DAP.

At the second extreme (details in Supplementary D.4), all attributes are assigned to a single group,  $K = 1$ . Taking a uniform prior for  $p(z)$  and  $p(a_m)$ , and replacing  $p(a_m = T|\mathbf{x})$  with the network model  $\sigma(\mathbf{x}^\top W)$ , transforms Eq. (4) to  $p(z|\mathbf{x}) \propto \sum_{m=1}^{|\mathcal{A}|} \sigma(\mathbf{x}^\top W) p(a_m = T|z)$ . Denoting  $U_{m,z} = p(a_m = T|z)$ , this formulation reveals that at the extreme case of  $K = 1$ , LAGO can be viewed as a non-linear variant that is closely related to ESZSL:  $\text{Score}(z|\mathbf{x}) = \mathbf{x}^\top WU$ , with same entries  $U_{m,z}$ .

## 4 EXPERIMENTS

Fair comparisons across ZSL studies tends to be tricky, since not all papers use a unified evaluation protocol. To guarantee an "apple-to-apple" comparison, we follow the protocol of a recent meta-analysis by Xian et al. (2017b) and compare to the leading methods evaluated with that protocol: **DAP** (Lampert et al., 2009), **ESZSL** (Romera-Paredes & Torr, 2015), **ALE** (Akata et al., 2016), **SYNC**

(Changpinyo et al., 2016), **SJE** (Akata et al., 2015), **DE-VICE** (Frome et al., 2013), **Zhang2018** (Zhang & Koniusz, 2018). Recent work showed that data augmentation and feature selection can be very useful for ZSL (Mishra et al., 2018; Arora et al., 2018; Xian et al., 2018; Guo et al., 2018). Since such augmentation are orthogonal to the modelling part, which is the focus of this paper, we do not use them here.

#### 4.1 DATASETS

We tested LAGO on three datasets: CUB, AWA2 and SUN. First, we tested LAGO in a fine-grained classification task of bird-species recognition using CUB-2011 (Wah et al., 2011). CUB has 11,788 images of 200 bird species and a vocabulary of 312 binary attributes (wing-color:olive), derived from 28 attribute groups (wing-color). Each image is annotated with attributes generated by one rater. We used the class description  $p(a_m|z)$  provided in the data. The names of the CUB attributes provide a strong prior for grouping (wing-color:olive, wing-color:red, ...  $\rightarrow$  wing-color:{olive, red, ...}).

The second dataset, Animals with Attributes2 (AWA2), (Xian et al., 2017a) consists of 37,322 images of 50 animal classes with pre-extracted feature representations for each image. Classes and attributes are aligned with the class-attribute matrix of (Osherson et al., 1991; Kemp et al., 2006). We use the class-attribute matrix as a proxy for the class description  $p(a_m|z)$ , since human subjects in (Osherson et al., 1991) did not see any image samples during the data-collection process. As a prior over attribute groups, we used the 9 groups proposed by (Lampert, 2011; Jayaraman et al., 2014) for 82 of 85 attributes, like *texture*:{furry, hairless, ...} and *shape*:{big, bulbous, ...}. We added two groups for remaining attributes: *world*:{new-world, old-world}, *smelly*:{smelly}.

As the third dataset, we used SUN (Patterson & Hays, 2012), a dataset of complex visual scenes, having 14,340 images from 717 scene types and 102 binary attributes from four groups.

#### 4.2 EXPERIMENTAL SETUP

We tested four variants of LAGO:

- (1) **LAGO-Singletons**: The model of Eq. (4) for the extreme case using  $K = |A|$  groups, where each attribute forms its own hard group.
- (2) **LAGO-Semantic-Hard**: The model of Eq. (4) with hard groups determined by attribute names. As explained in Section 4.1.
- (3) **LAGO-K-Soft**: The soft model of Eqs. 4-5, learning  $K$  soft group assignments with  $\Gamma$  initialized uniformly up to a small random perturbation.  $K$  is a hyper parameter with a value between 1 and the number of attributes. It is chosen by cross-validation.

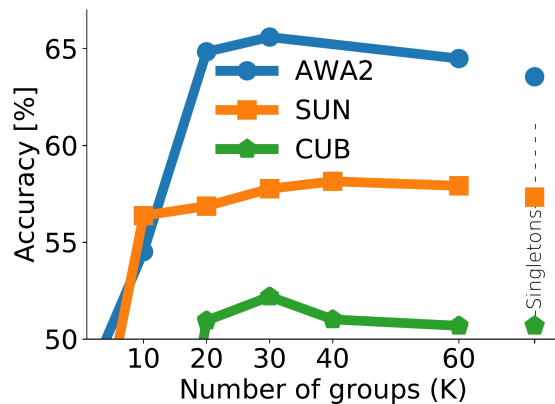


Figure 3: Learning  $K$  soft-group assignments: Validation-set accuracy for different number of groups ( $K$ ) for the LAGO-K-Soft variant and of the LAGO-Singletons baseline. Here, no prior information is given about the groups, and the model successfully learns groups assignments from data, and better than the group-naive LAGO-Singletons baseline.

(4) **LAGO-Semantic-Soft**: The model as in LAGO-K-Soft, but the soft groups  $\Gamma$  are initialized using the dataset-specific semantic groups assignments. These are also used as the prior  $\Gamma_{SEM}$  in Eq. (6).

Importantly, to avoid implicit overfitting to the test set, we used the validation set to select a single best variant, so we can report only a single prediction accuracy for the test set. For reference, we provide detailed test results of all variants in the Supplementary Material, Table A.1.

To learn the parameters  $W, V$ , we trained the weights with cross entropy-loss over outputs (and regularization terms) described in section 3.2. **In the hard-group case**, we only train  $W$ , while keeping  $V$  fixed. We sparsely initialize  $V$  with ones on every intersection of an attribute and its hard-assigned group and choose a high constant value for  $\zeta$  ( $\zeta = 10$ ). Since the rows of  $\Gamma$  correspond to attributes, it renders each row of  $\Gamma$  as a unit mass probability on a certain group. **In the soft-group case**, we train  $W, V$  alternately per epoch, allowing us to choose different learning rate for  $W$  and  $V$ . For LAGO-K-Soft,  $V$  was initialized with uniform random weights in  $[0, 1e-3]$ , inducing a uniform distribution over  $\Gamma$  up to a small random perturbation. **For LAGO-Semantic-Soft**, we initialized  $V$  as in the hard-group case, and we also used this initialization for the prior  $V_{SEM}$  Eq. (6).

**Design decisions:** (1) We use a uniform prior for  $p(z)$  as in (Xian et al., 2017b; Lampert et al., 2009; Romera-Paredes & Torr, 2015).  $p(a_m)$  can be estimated by marginalizing over  $p(a_m, z)$ , but as in ESZSL and DAP, we found that uniform priors performed better empirically. (2) To approximate the complementary attribute terms we used a De-Morgan based approximation for

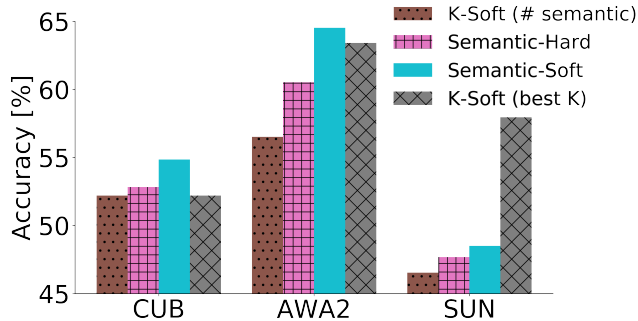


Figure 4: Validation accuracy (in %) of LAGO variants for three ZSL benchmark datasets. (1) On all the datasets, using semantic grouping information improves the performance relative to LAGO-K-Soft with  $K =$  number of semantic hard-groups. (2) We used these results to select which variant to use for the test set. Explicitly, when training the model on train+validation data, we used the *LAGO-Semantic-Soft* variant for CUB & AWA2, and *LAGO-K-Soft* variant for SUN.

$p(\tilde{a}_k = T|z) \approx \prod_{m \in G_k \setminus \{\tilde{a}_k\}} p(a_m^c|z)$ . For  $p(\tilde{a}_k = T|\mathbf{x})$ , we found that setting a constant value was empirically better than using a De-Morgan based approximation. (3) Our model does not use an explicit supervision signal for learning the weights of the attributes-prediction layer. Experiments showed that usage of explicit attributes supervision, by setting a non-zero value for  $\alpha$  in Eq. (6), results in deteriorated performance.

In Section 4.4, we demonstrate the above design decision with ablation experiments on the validation sets of CUB and AWA2.

**Implementation and training details:** The Supplementary Material (A) describes the training protocol, including the cross-validation procedure, optimization and tuning of hyper parameters.

### 4.3 RESULTS

Our experiments first compare variants of LAGO, and then compare the best variant to baseline methods. We then study in more depth the properties of the learned models.

Figure 3 shows validation-set accuracy of LAGO-K-Soft variants as a function of the number of groups ( $K$ ) and for the LAGO-Singletons baseline. We used these results to select the optimal number of groups  $K$ . In these experiments, even-though no prior information is provided about grouping, LAGO successfully learns group assignments from data, performing better than the group-naïve LAGO-Singletons baseline. The performance degrades largely when the number of groups is small.

Figure 4 shows validation-set accuracy for main variants of LAGO for three benchmark datasets. We used these results to select the variant of LAGO applied to

	CUB	AWA2	SUN
<b>DAP</b>	40.0	46.2	39.9
<b>ALE</b>	54.9	62.5	<b>58.1</b>
<b>ESZSL</b>	53.9	58.6	54.5
<b>SYNC</b>	55.6	46.6	56.3
<b>SJE</b>	53.9	61.9	53.7
<b>DEVISE</b>	52.0	59.7	56.5
<b>ZHANG2018*</b>	48.7-57.1	58.3- <b>70.5</b>	57.8- <b>61.7</b>
<b>LAGO (OURS)</b>	<b>57.8</b>	<b>64.8</b>	57.5

Table 1: Test accuracy (in %) of LAGO and compared methods on three ZSL benchmark datasets. We follow the protocol of a meta-analysis by (Xian et al., 2017b) and compare to leading methods evaluated with it. Only one LAGO variant is shown, selected using a validation set. See Table A.1 in the supplementary for more results. LAGO outperforms previous baselines on CUB and AWA2 by a significant margin. On SUN, LAGO loses by a small margin. (\*) Comparison with *Zhang2018* is inconclusive. Zhang & Koniusz (2018) report the results for 7 kernel types on the test set, but results on a validation set were not published, hence taking the best kernel over the test set may be optimistic.

the test split of each dataset. Specifically, when training the model on train+validation data, we used *LAGO-Semantic-Soft* for CUB & AWA2, and *LAGO-K-Soft* ( $K = 40$ ) for SUN. This demonstrates that LAGO is useful even when the semantic-grouping prior is of low quality as in SUN. Figure 4 also shows that semantic grouping, significantly improves performance, relative to LAGO-K-Soft with a similar number of groups.

We draw three conclusions from Figures 3-4. (1) The prior grouping based on attribute semantics contains very valuable information that LAGO can effectively use. (2) LAGO succeeds even when no prior group information is given, effectively learning group assignments from data. (3) Using the semantic hard-groups as a prior, allows us to soften the semantic hard-groups and optimize the grouping structure from data.

Table 1 details the main empirical results, comparing test accuracy of LAGO with the competing methods. Importantly, to guarantee "apple-to-apple" comparison, evaluations are made based on the standard evaluation protocol from Xian et al. (2017b), using the same underlying image features, data splits and metrics. Results are averaged over 5 random initializations (seeds) of the model weights ( $W, V$ ). Standard-error-of-the-mean (S.E.M) is  $\sim 0.4\%$ . On CUB and AWA2, LAGO outperform all competing approaches by a significant margin. On CUB, reaching 57.8% versus 55.6% for SYNC (Changpinyo et al., 2016). On AWA2, reaching 64.8% versus 62.5% for ALE (Akata et al., 2016). On SUN, LAGO loses by a small margin (57.5% versus 58.1%). Note that comparison with "Zhang2018" (Zhang & Koniusz, 2018) is



inconclusive. "Zhang2018" reports the results for 7 kernel types on the test set, but results on a validation set were not published, hence taking the best kernel over the test set may be optimistic.

**LAGO-Singletons versus DAP:** LAGO-Singletons is a reminiscent of DAP, but unlike DAP, it applies a soft relaxation that balances between appearance of an attribute and its negation. Interestingly, this minor change allows LAGO-Singletons to outperform DAP by  $\sim 40\%$  on average over all three datasets, while keeping an appealing simplicity as of DAP (Supplementary Table A.1).

**LAGO with few groups:** When the number of groups is small, the accuracy of LAGO is poor (Fig 3, 4) This happens because when groups have too many attributes, the AND-OR structure of LAGO becomes too permissive. For example, when all attributes are grouped into a single group, an OR is applied over all attributes and no AND, leading to many spurious matches when partial attributes are observed. A similar effect is observed when applying LAGO to SUN data which has only 4 semantic hard groups for 102 attributes. Indeed applying LAGO-Semantic-Hard to SUN performs poorly since it is too permissive. Another interesting effect arises when comparing the the poor performance of the single-group case with ESZSL. ESZSL is convex and with a closed-form solution, hence reaching the optimum is guaranteed. Single-group LAGO is non-convex (due to sigmoidal activation) making it harder to find the optimum. Indeed, we observed a worse training accuracy for single-group LAGO compared with ESZSL (61% vs 84% on CUB), suggesting that single-group LAGO tends to underfit the data.

**Learned Soft Group Assignments  $\Gamma$ :** We analyzed the structure of learned soft group assignments ( $\Gamma$ ) for LAGO-K-Soft (details in Supplementary B). We found two interesting observations: First, we find that the learned  $\Gamma$  tends to be sparse: with 2.5% non-zero values on SUN, 8.7% on AWA2 and 3.3% on CUB. Second, we observed that the model tends to group anti-correlated attributes. This is consistent with human-based grouping, whose attribute are also often anti correlated (red foot, blue foot). In SUN, 45% of attribute-pairs that are grouped together were anti-correlated, versus 23% of all attribute-pairs. In AWA2, 38% vs 5% baseline, CUB 16% vs 10% baseline (p-value  $\leq 0.003$ , KS-test).

**Qualitative Results:** To gain insight into why and when attribute grouping can reduce false positives and false negatives, we discuss in more depth two examples shown on Figure 5, predicted by LAGO-Semantic-Hard on CUB. The analysis demonstrates an interpretable quality of LAGO, allowing to "look under the hood" and explain class-predictions based on seen attributes.

**The effect of within-group disjunction (OR):** Image 5a is correctly classified by LAGO as a *Black-billed Cuckoo*, even-though a detector misses its brown primary-color. In more detail, for this class, raters disagreed whether the primary-color is mostly brown ( $p(\text{brown}|z) = 0.6$ ) or white (0.5), because this property largely depends on the point-of view. Somewhat surprisingly, the primary color in this photo was detected to be mostly white ( $p(\text{white}|\mathbf{x}) = 0.7$ ), and hardly brown (0.1), perhaps because of a brown branch that interferes with segmenting out the bird. Missing the brown color hurts any classifier that requires both brown and white, like DAP. LAGO treats the detected primary color as a good match because it takes a soft OR relation over the two primary colors, hence avoids missing the right class.

**The effect of group conjunction (AND):** Image 5b.1 was correctly classified by LAGO as a *White-Breasted Nuthatch*, even-though a detector incorrectly detects a yellow primary color ( $p(\text{yellow}|\mathbf{x}) = 0.6$ ) together with white and grey primary colors (0.7). As a comparison, the perceived yellow primary color confused ESZSL to mistake this image for a *Cape-May Warbler*, shown in image (b.2). Since ESZSL treats attributes as "flat", it does not use the fact that the breast pattern does not match a Warbler, and adheres to other attributes that produce a false positive detection of the Warbler. Yet, LAGO successfully avoids being confused by the yellow primary color, since the Nuthatch is expected to have a solid breast pattern, which is correctly detected  $p(\text{breast} : \text{solid}|\mathbf{x}) = 0.6$ . The Warbler is ranked lower because it is expected to have a striped breast pattern, which does not satisfy the AND condition because stripes are not detected  $p(\text{breast} : \text{striped}|x) = 0$ .

#### 4.4 ABLATION EXPERIMENTS

We carried empirical ablation experiments with the semantic hard-grouping of LAGO. Specifically, we tested three design decisions we made, as described above. **(1) Uniform** relates to taking a uniform prior for  $p(a_m)$ , which is the average of the estimated  $p(a_m)$ . "Per-attribute" relates to using the estimated  $p(a_m)$  directly. **(2) Const** relates to setting a constant value for the approximation of the complementary attribute  $p(\tilde{a}_k|\mathbf{x})$ . "DeMorgan" relates to approximating it from predictions of other attributes with De-Morgan's rule. **(3) Implicit** relates to setting a zero weight ( $\alpha = 0$ ) for the loss term of the attribute supervision. I.e. attributes are learned implicitly, because only class-level supervision is given. "Explicit" related to setting a non-zero  $\alpha$  respectively.

Table A.2 (in Supplementary) shows contributions of each combination of the design decisions to prediction accuracy, on the validation set of CUB and AWA2. The results are consistent for both CUB and AWA2. The

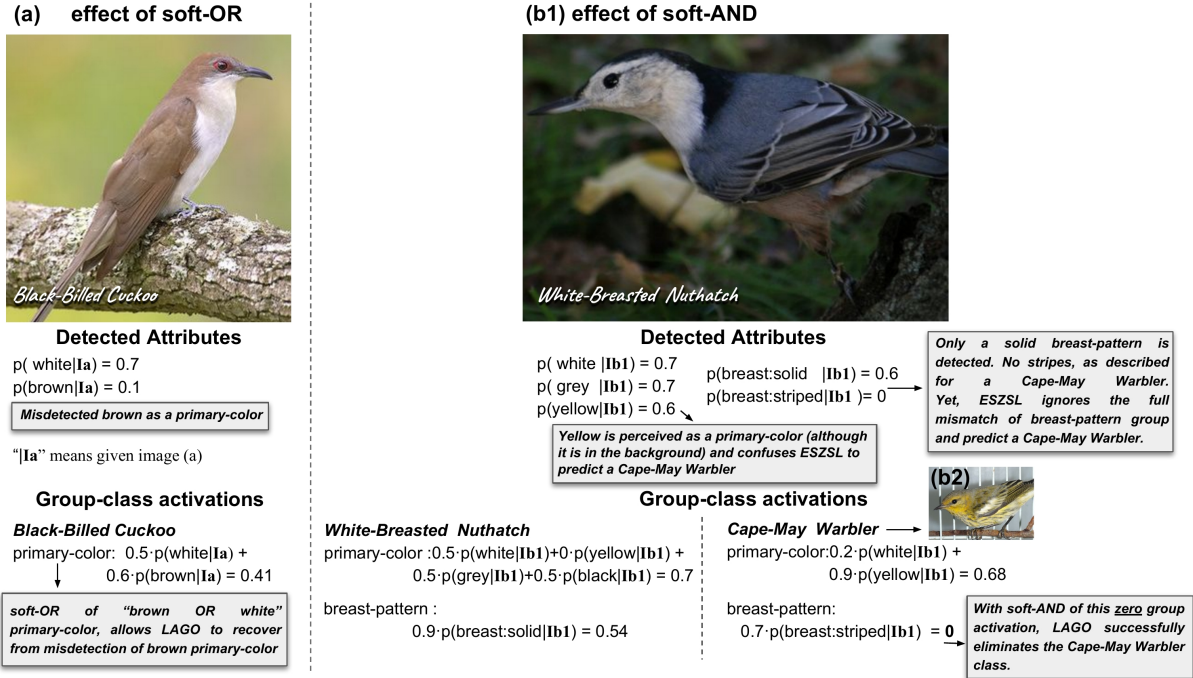


Figure 5: Qualitative examples. **(a)** LAGO correctly classifies a Black-billed Cuckoo, due to soft-OR of “brown or white” primary-color, although a detector misses its brown primary-color. **(b1)** LAGO correctly classifies a White-Breasted Nuthatch: The soft-intersection across groups prevents incorrect classification. ESZSL incorrectly classified Image (b1) as class of (b2), despite irrelevant attribute groups like its breast pattern **(b2)** A typical Cape-May Warbler. ESZSL mistakes (b1) image for this class

most major effect is contributed for the uniform prior of  $p(a_m)$ . All experiments that use the uniform prior yield better accuracy. We observe that taking a uniform prior also reduces variability due to the other approximations we take. Specifically, on CUB there is  $\approx 4.5\%$  best-to-worst gap with a uniform prior, vs  $\approx 12.5\%$  without ( $\approx 11\%$  vs  $\approx 16\%$  for AWA2 respectively). Next we observe that in the uniform case, approximating  $p(\tilde{a}_k|\mathbf{x})$  by a constant, is superior to approximating it with DeMorgan’s rule, and similarly, reduces the impact of the variability of the implicit/explicit condition. Last, the contribution of attributes supervision condition mostly depends on selection of the previous two conditions.

## 5 DISCUSSION

Three interesting future research directions can be followed. First, since LAGO is probabilistic, one can plug measures for model uncertainty (Gal & Ghahramani, 2016), to improve model prediction and increase robustness to adversarial attacks. Second, descriptions of fine-grained categories often provide richer logical expressions to describe and differentiate classes. It will be interesting to study how LAGO may be extended to incorporate richer relations that could be explicitly discriminative (Vedantam et al., 2017). For example, Wikipedia describes *White-Breasted-Nuthatch* to make it distinct from other, commonly confused, Nuthatches

by: “Three other, significantly smaller, nuthatches have ranges which overlap that of white-breasted, but none has white plumage completely surrounding the eye.”.

Third, when people describe classes, they often use a handful of attributes instead of listing all values for the full set of attributes. The complementary attribute used in LAGO allows to model a “don’t-care” about a group, when no description is provided for a group. Such an approach could enable to recognize visual entities based on a handful and partial indication of semantic properties.

## 6 SUMMARY

We presented LAGO, a new probabilistic zero-shot-learning approach that can be trained end-to-end. LAGO approximates  $p(\text{class} = z | \text{image} = \mathbf{x})$  by capturing natural soft *and-or* logical relations among *groups of attributes*, unlike most ZSL approaches that represent attributes as embedded in a “flat” space. LAGO learns the grouping structure from data, and can effectively incorporate prior domain knowledge about the grouping of attributes when available. We find that LAGO achieves new state-of-the-art result on CUB (Wah et al., 2011), AWA2 (Lampert et al., 2009), and is competitive on SUN (Patterson & Hays, 2012). Finally, LAGO reveals an interesting unified formulation over seemingly-unrelated ZSL approaches, DAP (Lampert et al., 2009) and ESZSL (Romera-Paredes & Torr, 2015).

## References

- Z. Akata, S. Reed, D. Walter, Honglak Lee, and B. Schiele. Evaluation of output embeddings for fine-grained image classification. In *CVPR*, 2015.
- Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-embedding for image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2016.
- Z. Al-Halah and R. Stiefelhagen. How to transfer? zero-shot object recognition via hierarchical transfer of semantic attributes. In *WACV*, 2015.
- G. Arora, V-K. Verma, A. Mishra, and P. Rai. Generalized zero-shot learning via synthesized examples. In *CVPR*, 2018.
- S. Bach, M. Broecheler, B. Huang, and L. Getoor. Hinge-loss markov random fields and probabilistic soft logic. *Journal of Machine Learning Research*, 2017.
- T. Berg, A.C. Berg, and J. Shih. Automatic attribute discovery and characterization from noisy web data. In *ECCV*. Springer, 2010.
- S. Changpinyo, W. L. Chao, B. Gong, and F. Sha. Synthesized classifiers for zero-shot learning. In *CVPR*, 2016.
- M. Elhoseiny, B. Saleh, and A. Elgammal. Write a classifier: Zero-shot learning using purely textual descriptions. In *ICCV*, 2013.
- A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *CVPR. IEEE*, 2009.
- V. Ferrari and A. Zisserman. Learning visual attributes. In *NIPS*, pp. 433–440, 2008.
- A. Frome, G. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov. Devise: A deep visual-semantic embedding model. In *NIPS*, 2013.
- Y. Fu, T. Xiang, Y-G Jiang, X. Xue, L. Sigal, and S. Gong. Recent advances in zero-shot recognition. *arXiv preprint arXiv:1710.04837*, 2017.
- Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016.
- Y. Guo, G. Ding, J. Han, and S. Tang. Zero-shot learning with attribute selection. In *AAAI*, 2018.
- D. Jayaraman, F. Sha, and K. Grauman. Decorrelating semantic visual attributes by resisting the urge to share. In *CVPR, CVPR '14*, 2014.
- H. Jiang, R. Wang, S. Shan, Y. Yang, and X. Chen. Learning discriminative latent attributes for zero-shot classification. In *ICCV*, 2017.
- C. Kemp, J.B. Tenenbaum, T.L. Griffiths, T. Yamada, and N. Ueda. Learning systems of concepts with an infinite relational model. In *AAAI*, volume 1, 2006.
- A. Kimmig, S. Bach, M. Broecheler, B. Huang, and L. Getoor. A short introduction to probabilistic soft logic. In *NIPS Workshop on Probabilistic Programming*, 2012.
- C.H. Lampert. Semantic attributes for object categorization (slides). *ist.ac.at/chl/talks/lampert-vrml2011b.pdf*, 2011.
- C.H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR. IEEE*, 2009.
- C.H. Lampert, H. Nickisch, and S. Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 36(3), 2014.
- Y. Li, D. Wang, H. Hu, Y. Lin, and Y. Zhuang. Zero-shot recognition using dual visual-semantic mapping paths. In *CVPR*, 2017.
- A. Mishra, M. Reddy, A. Mittal, and H. A. Murthy. A generative model for zero shot learning using conditional variational autoencoders. In *WACV*, 2018.
- P. Morgado and N. Vasconcelos. Semantically consistent regularization for zero-shot recognition. In *CVPR. IEEE*, 2017.
- D. N. Osherson, J. Stern, O. Wilkie, M. Stob, and E. Smith. Default probability. *Cognitive Science*, 15(2), 1991.
- M. Palatucci, D. Pomerleau, G. E. Hinton, and T.M. Mitchell. Zero-shot learning with semantic output codes. In *NIPS*, 2009.
- S. Park and S. C. Zhu. Attributed grammars for joint estimation of human attributes, part and pose. In *ICCV*, 2015.
- S. Park, X. Nie, and S. C. Zhu. Attribute and-or grammar for joint parsing of human pose, parts and attributes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2017.
- G. Patterson and J. Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *CVPR*, 2012.
- M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1), 2006.
- M. Rohrbach, M. Stark, and B. Schiele. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In *CVPR*, 2011.
- B. Romera-Paredes and P. Torr. An embarrassingly simple approach to zero-shot learning. In *ICML*, 2015.
- R. Socher, M. Ganjoo, C.D. Manning, and A. Ng. Zero-shot learning through cross-modal transfer. In *NIPS*, 2013.
- Y-H. Tsai, L-K. Huang, and R. Salakhutdinov. Learning robust visual-semantic embeddings. In *ICCV*, 2017a.
- Y-H H. Tsai, L-K Huang, and R. Salakhutdinov. Learning robust visual-semantic embeddings. In *ICCV*, 2017b.
- R. Vedantam, S. Bengio, K. Murphy, D. Parikh, and G. Chechik. Context-aware captions from context-agnostic supervision. In *CVPR*, 2017.
- C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.
- X. Wang and Q. Ji. A unified probabilistic approach modeling relationships between attributes and objects. In *ICCV*, 2013.
- Y. Xian, C.H. Lampert, B. Schiele, and Z. Akata. Zero-shot learning - A comprehensive evaluation of the good, the bad and the ugly. *arXiv preprint arXiv:1707.00600*, 2017a.
- Y. Xian, B. Schiele, and Z. Akata. Zero-shot learning - the good, the bad and the ugly. In *CVPR*, 2017b.
- Y. Xian, T. Lorenz, B. Schiele, and Z. Akata. Feature generating networks for zero-shot learning. In *CVPR*, 2018.

- X. Xu, F. Shen, Y. Yang, D. Zhang, H. T. Shen, and J. Song. Matrix tri-factorization with manifold regularizations for zero-shot learning. In *CVPR*, 2017.
- M. Ye and Y. Guo. Zero-shot classification with discriminative semantic representation learning. In *CVPR*, 2017.
- Hongguang Zhang and Piotr Koniusz. Zero-shot kernel learning. In *CVPR*, 2018.
- L. Zhang, T. Xiang, and S. Gong. Learning a deep embedding model for zero-shot learning. In *CVPR*, 2017.
- S-C Zhu and D. Mumford. A stochastic grammar of images. *Found. Trends. Comput. Graph. Vis.*, 2(4), 2006.

---

# Sylvester Normalizing Flows for Variational Inference

---

**Rianne van den Berg\***  
Informatics Institute  
University of Amsterdam

**Leonard Hasenclever\***  
Department of statistics  
University of Oxford

**Jakub M. Tomczak**  
Informatics Institute  
University of Amsterdam

**Max Welling<sup>†</sup>**  
Informatics Institute  
University of Amsterdam

## Abstract

Variational inference relies on flexible approximate posterior distributions. Normalizing flows provide a general recipe to construct flexible variational posteriors. We introduce Sylvester normalizing flows, which can be seen as a generalization of planar flows. Sylvester normalizing flows remove the well-known single-unit bottleneck from planar flows, making a single transformation much more flexible. We compare the performance of Sylvester normalizing flows against planar flows and inverse autoregressive flows and demonstrate that they compare favorably on several datasets.

## 1 INTRODUCTION

Stochastic variational inference [Hoffman et al., 2013] allows for posterior inference in increasingly large and complex problems using stochastic gradient ascent. In continuous latent variable models, variational inference can be made particularly efficient through the amortized inference, in which inference networks amortize the cost of calculating the variational posterior for a data point [Gershman and Goodman, 2014]. A particularly successful class of models is the variational autoencoder (VAE) in which both the generative model and the inference network are given by neural networks, and sampling from the variational posterior is efficient through the non-centered parameterization [Kingma and Welling, 2014], also known as the reparameterization trick [Kingma and Welling, 2013, Rezende et al., 2014].

Despite its success, variational inference has drawbacks compared to other inference methods such as MCMC. Variational inference searches for the best posterior approximation within a parametric family of distributions. Hence, the true posterior distribution can only be recovered exactly if it happens to be in the chosen family. In particular, with widely used simple variational families such as diagonal covariance Gaussian distributions, the variational approximation is likely to be insufficient. More complex variational families enable better posterior approximations, resulting in improved model performance. Therefore, designing tractable and more expressive variational families is an important problem in variational inference [Nalisnick et al., 2016, Salimans et al., 2015, Tran et al., 2015].

Rezende and Mohamed [2015] introduced a general framework for constructing more flexible variational distributions, called normalizing flows. Normalizing flows transform a base density through a number of invertible parametric transformations with tractable Jacobians into more complicated distributions. They proposed two classes of normalizing flows: planar flows and radial flows. While effective for small problems, these can be hard to train and often many transformations are required to get good performance. For planar flows, Kingma et al. [2016] argue that this is due to the fact that the transformation used acts as a bottleneck, warping one direction at a time. Having a large number of flows makes the inference network very deep and harder to train, empirically resulting in suboptimal performance. Kingma et al. [2016] proposed inverse auto-regressive flows (IAF), achieving state of the art results on dynamically binarized MNIST at the time of publication. While very successful, IAFs require a very large number of parameters. Due to the large number of parameters IAFs cannot amortize all flow parameters. Instead amortization is achieved through an additional context vector that is fed into each flow step.

---

\*Equal contribution

<sup>†</sup> Also affiliated with the Canadian Institute for Advanced Research (CIFAR)

**Paper contribution** In this paper, we use Sylvester’s determinant identity to introduce Sylvester normalizing flows (SNFs). This family of flows is a generalization of planar flows, removing the bottleneck. We compare a number of different variants of SNFs and show that they compare favorably against planar flows and IAFs. We show that one specific variant of SNFs is related to IAFs, while requiring many fewer parameters due to direct amortization of all flow parameters.

## 2 VARIATIONAL INFERENCE

Consider a probabilistic model with observations  $\mathbf{x}$  and continuous latent variables  $\mathbf{z}$  and model parameters  $\theta$ . In generative modeling we are often interested in performing maximum (marginal) likelihood learning of the parameters  $\theta$  of the latent-variable model  $p_\theta(\mathbf{x}, \mathbf{z})$ . This requires marginalization over the unobserved latent variables  $\mathbf{z}$ . Unfortunately, this integration is generally intractable. Variational inference [Jordan et al., 1999] instead introduces a variational approximation  $q_\phi(\mathbf{z}|\mathbf{x})$  to the posterior with learnable parameters  $\phi$ , to construct a lower bound on the log marginal likelihood:

$$\log p_\theta(\mathbf{x}) \geq \log p_\theta(\mathbf{x}) - \text{KL}(q_\phi(\mathbf{z}|\mathbf{x}) || p(\mathbf{z}|\mathbf{x})) \quad (1)$$

$$= \mathbb{E}_{q_\phi}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \text{KL}(q_\phi(\mathbf{z}|\mathbf{x}) || p(\mathbf{z})) \quad (2)$$

$$=: -\mathcal{F}(\theta, \phi) \quad (3)$$

This bound is known as the evidence lower bound (ELBO) and  $\mathcal{F}$  is referred to as the variational free energy. In equation (2), the first term represents the reconstruction error, and the second term is the Kullback-Leibler (KL) divergence from the approximate posterior to the prior distribution, which acts as a regularizer. In this paper we consider variational autoencoders (VAEs), where both  $p_\theta(\mathbf{x}|\mathbf{z})$  and  $q_\phi(\mathbf{z}|\mathbf{x})$  are distributions whose parameters are given by neural networks. The parameters  $\theta$  and  $\phi$  of the generative model and inference model, respectively, are trained jointly through stochastic minimisation of  $\mathcal{F}$  which can be made efficient through the reparameterization trick [Kingma and Welling, 2013, Rezende et al., 2014].

From equation (1) we see that the better the variational approximation to the posterior the tighter the ELBO. The simplest, but probably most widely used choice of variation distribution  $q_\phi(\mathbf{z}|\mathbf{x})$  is diagonal-covariance Gaussians of the form  $\mathcal{N}(\boldsymbol{\mu}(\mathbf{x}), \boldsymbol{\sigma}^2(\mathbf{x}))$ . However, with such simple variational distributions the ELBO will be fairly loose, resulting in biased maximum likelihood estimates of the model parameters  $\theta$  (see Fig. 2) and harming generative performance. Thus, for variational inference to work well, more flexible approximate posterior distributions are needed.

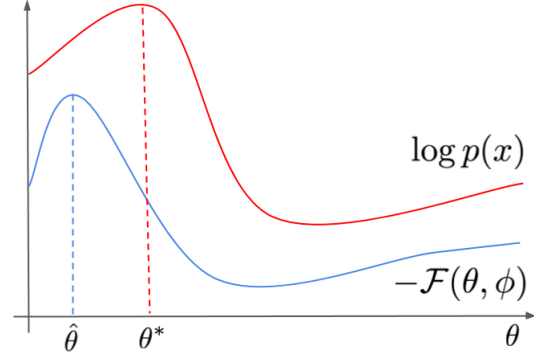


Figure 1: Since the ELBO is only a lower bound on the log marginal likelihood, they do not share the same local maxima. The looser the ELBO is the more this can bias maximum likelihood estimates of the model parameters.

### 2.1 NORMALIZING FLOWS

Rezende and Mohamed [2015] propose a way to construct more flexible posteriors by transforming a simple base distribution with a series of invertible transformations (known as normalizing flows) with easily computable Jacobians. The resulting transformed density after one such transformation  $f$  is as follows [Tabak and Turner, 2013, Tabak and Vanden-Eijnden, 2010]:

$$p_1(\mathbf{z}') = p_0(\mathbf{z}) \left| \det \left( \frac{\partial f(\mathbf{z})}{\partial \mathbf{z}} \right) \right|^{-1}, \quad (4)$$

where  $\mathbf{z}' = f(\mathbf{z})$ ,  $\mathbf{z}, \mathbf{z}' \in \mathbb{R}^D$  and  $f : \mathbb{R}^D \mapsto \mathbb{R}^D$  is an invertible function. In general the cost of computing the Jacobian will be  $\mathcal{O}(D^3)$ . However, it is possible to design transformations with more efficiently computable Jacobians.

This strategy is used in variational inference as follows: first, a stochastic variable is drawn from a simple base posterior distribution such as a diagonal Gaussian  $\mathcal{N}(\mathbf{z}_0|\boldsymbol{\mu}(\mathbf{x}), \boldsymbol{\sigma}^2(\mathbf{x}))$ . The sample is then transformed with a number of flows. After applying  $K$  flows, the final latent stochastic variables are given by  $\mathbf{z}_K = f_K \circ \dots \circ f_2 \circ f_1(\mathbf{z}_0)$ . The corresponding log-density is then given by:

$$\log q_K(\mathbf{z}_K|\mathbf{x}) = \log q_0(\mathbf{z}_0|\mathbf{x}) - \sum_{k=1}^K \log \left| \det \left( \frac{\partial f_k(\mathbf{z}_{k-1}; \lambda_k(\mathbf{x}))}{\partial \mathbf{z}_{k-1}} \right) \right|, \quad (5)$$

where  $\lambda_k$  are the parameters of the  $k$ -th transformation. Given variational posterior  $q_\phi(\mathbf{z}|\mathbf{x}) = q_K(\mathbf{z}|\mathbf{x})$  parametrized by a normalizing flow of length  $K$ , the vari-

ational objective can be rewritten as:

$$\begin{aligned} \mathcal{F}(\theta, \phi) &= \mathbb{E}_{q_\phi} [\log q_\phi(\mathbf{z}|\mathbf{x}) - \log p_\theta(\mathbf{x}, \mathbf{z})] \\ &= \mathbb{E}_{q_0} [\log q_0(\mathbf{z}_0|\mathbf{x}) - \log p_\theta(\mathbf{x}, \mathbf{z})] \\ &\quad - \mathbb{E}_{q_0} \left[ \sum_{k=1}^K \log \left| \det \left( \frac{\partial f_k(\mathbf{z}_{k-1}; \lambda_k(\mathbf{x}))}{\partial \mathbf{z}_{k-1}} \right) \right| \right]. \end{aligned} \quad (6)$$

Normalizing flows are normally used with amortized variational inference. Instead of learning variational parameters for each data point, both  $\boldsymbol{\mu}$  and  $\boldsymbol{\sigma}$ , as well as all the flow parameters are outputs of a deep neural network conditioned on  $\mathbf{x}$ . This is referred to as the inference network.

Rezende and Mohamed [2015] introduced a normalizing flow, called planar flow, for which the Jacobian determinant could be computed efficiently. A single transformation of the planar flow is given by:

$$\mathbf{z}' = \mathbf{z} + \mathbf{u}h(\mathbf{w}^T \mathbf{z} + b). \quad (8)$$

Here,  $\mathbf{u}, \mathbf{w} \in \mathbb{R}^D$ ,  $b \in \mathbb{R}$  and  $h$  is a suitable smooth activation function. Rezende and Mohamed [2015] show that for  $h = \tanh$ , transformations of this kind are invertible as long as  $\mathbf{u}^T \mathbf{w} \geq -1$ .

By the *Matrix determinant lemma* the Jacobian of this transformation is given by:

$$\begin{aligned} \det \frac{\partial \mathbf{z}'}{\partial \mathbf{z}} &= \det (\mathbf{I} + \mathbf{u}h'(\mathbf{w}^T \mathbf{z} + b)\mathbf{w}^T) \\ &= 1 + \mathbf{u}^T h'(\mathbf{w}^T \mathbf{z} + b)\mathbf{w}, \end{aligned} \quad (9)$$

where  $h'$  denotes the derivative of  $h$  and which can be computed in  $O(D)$  time.

In practice, many planar flow transformations are required to transform a simple base distribution into a flexible distribution, especially for high dimensional latent spaces. Kingma et al. [2016] argue that this is related to the term  $\mathbf{u}h(\mathbf{w}^T \mathbf{z} + b)$  in Eq. (8), which effectively acts as a single-neuron MLP. In the next section we will derive a generalization of planar flows, which does not have a single-neuron bottleneck, while still maintaining the property of an efficiently computable Jacobian determinant.

### 3 SYLVESTER NORMALIZING FLOWS

Consider the following more general transformation similar to a single layer MLP with  $M$  hidden units and a residual connection:

$$\mathbf{z}' = \mathbf{z} + \mathbf{A}h(\mathbf{B}\mathbf{z} + \mathbf{b}), \quad (10)$$

with  $\mathbf{A} \in \mathbb{R}^{D \times M}$ ,  $\mathbf{B} \in \mathbb{R}^{M \times D}$ ,  $\mathbf{b} \in \mathbb{R}^M$ , and  $M \leq D$ . The Jacobian determinant of this transformation can be obtained using Sylvester's determinant identity, which is a generalization of the matrix determinant lemma.

**Theorem 1** (Sylvester's determinant identity). *For all  $\mathbf{A} \in \mathbb{R}^{D \times M}$ ,  $\mathbf{B} \in \mathbb{R}^{M \times D}$ ,*

$$\det(\mathbf{I}_D + \mathbf{A}\mathbf{B}) = \det(\mathbf{I}_M + \mathbf{B}\mathbf{A}), \quad (11)$$

where  $\mathbf{I}_M$  and  $\mathbf{I}_D$  are  $M$  and  $D$ -dimensional identity matrices, respectively.

When  $M < D$ , the computation of the determinant of a  $D \times D$  matrix is thus reduced to the computation of the determinant of an  $M \times M$  matrix.

Using Sylvester's determinant identity, the Jacobian determinant of the transformation in Eq. (10) is given by:

$$\det \left( \frac{\partial \mathbf{z}'}{\partial \mathbf{z}} \right) = \det (\mathbf{I}_M + \text{diag} (h'(\mathbf{B}\mathbf{z} + \mathbf{b})) \mathbf{B}\mathbf{A}). \quad (12)$$

Since Sylvester's determinant identity plays a crucial role in the proposed family of normalizing flows, we will refer to them as *Sylvester normalizing flows*.

#### 3.1 PARAMETERIZATION OF A AND B

In general, the transformation in (10) will not be invertible. Therefore, we propose the following special case of the above transformation:

$$\mathbf{z}' = \mathbf{z} + \mathbf{Q}\mathbf{R}h(\tilde{\mathbf{R}}\mathbf{Q}^T \mathbf{z} + \mathbf{b}) = \phi(\mathbf{z}), \quad (13)$$

where  $\mathbf{R}$  and  $\tilde{\mathbf{R}}$  are upper triangular  $M \times M$  matrices, and

$$\mathbf{Q} = (\mathbf{q}_1 \dots \mathbf{q}_M)$$

with the columns  $\mathbf{q}_m \in \mathbb{R}^D$  forming an orthonormal set of vectors. By theorem 1, the determinant of the Jacobian  $\mathbf{J}$  of this transformation reduces to:

$$\begin{aligned} \det \mathbf{J} &= \det \left( \mathbf{I}_M + \text{diag} \left( h'(\tilde{\mathbf{R}}\mathbf{Q}^T \mathbf{z} + \mathbf{b}) \right) \tilde{\mathbf{R}}\mathbf{Q}^T \mathbf{Q}\mathbf{R} \right) \\ &= \det \left( \mathbf{I}_M + \text{diag} \left( h'(\tilde{\mathbf{R}}\mathbf{Q}^T \mathbf{z} + \mathbf{b}) \right) \tilde{\mathbf{R}}\mathbf{R} \right), \end{aligned} \quad (14)$$

which can be computed in  $O(M)$ , since  $\tilde{\mathbf{R}}\mathbf{R}$  is also upper triangular. The following theorem gives a sufficient condition for this transformation to be invertible.

**Theorem 2.** *Let  $\mathbf{R}$  and  $\tilde{\mathbf{R}}$  be upper triangular matrices. Let  $h : \mathbb{R} \rightarrow \mathbb{R}$  be a smooth function with bounded, positive derivative. Then, if the diagonal entries of  $\mathbf{R}$  and  $\tilde{\mathbf{R}}$  satisfy  $r_{ii}\tilde{r}_{ii} > -1/\|h'\|_\infty$  and  $\tilde{\mathbf{R}}$  is invertible, the transformation given by (13) is invertible.*

*Proof. Case 1: R and R-tilde diagonal*

Recall that one-dimensional real functions with strictly positive derivatives are invertible. The columns of  $\mathbf{Q}$  are orthonormal and span a subspace  $\mathcal{W} = \text{span}\{\mathbf{q}_1, \dots, \mathbf{q}_M\}$  of  $\mathbb{R}^D$ . Let  $\mathcal{W}^\perp$  denote its orthogonal complement. We can decompose  $\mathbf{z} = \mathbf{z}_\parallel + \mathbf{z}_\perp$ , where  $\mathbf{z}_\parallel \in \mathcal{W}$  and  $\mathbf{z}_\perp \in \mathcal{W}^\perp$ . Similarly we can decompose  $\mathbf{z}' = \mathbf{z}'_\parallel + \mathbf{z}'_\perp$ . Clearly,  $\mathbf{Q}\mathbf{R}h(\tilde{\mathbf{R}}\mathbf{Q}^T\mathbf{z} + \mathbf{b}) \in \mathcal{W}$ . Hence  $\phi$  only acts on  $\mathbf{z}_\parallel$  and  $\mathbf{z}_\perp = \phi(\mathbf{z})_\perp = \mathbf{z}'_\perp$ . Thus, it suffices to consider the effect of  $\phi$  on  $\mathbf{z}_\parallel$ . Multiplying (13) by  $\mathbf{Q}^T$  from the left gives:

$$\underbrace{\mathbf{Q}^T\mathbf{z}'}_{\mathbf{v}'} = \underbrace{\mathbf{Q}^T\mathbf{z}}_{\mathbf{v}} + \mathbf{R}h(\underbrace{\tilde{\mathbf{R}}\mathbf{Q}^T\mathbf{z} + \mathbf{b}}_{\mathbf{v}}) \quad (15)$$

$$= (f_1(v_1), \dots, f_M(v_M))^T,$$

where the vectors  $\mathbf{v}$  and  $\mathbf{v}'$  are the respective coordinates of  $\mathbf{z}_\parallel$  and  $\mathbf{z}'_\parallel$  w.r.t.  $\mathbf{q}_1, \dots, \mathbf{q}_M$ . The dimensions in (15) are completely independent and each dimension is transformed by a real function  $f_i(v) = v + r_{ii}h(\tilde{r}_{ii}v + b_i)$ . Consider a single dimension  $i$  of (15). Since  $\|h'\|_\infty r_{ii}\tilde{r}_{ii} > -1$ , we have  $f'_i(v) > 0$  and thus  $f_i$  is invertible. Since all dimensions are independent and the transformation is invertible in each dimension we can find  $f^{-1} : \mathcal{W} \rightarrow \mathcal{W}$  such that  $\mathbf{z}_\parallel = f^{-1}(\mathbf{z}'_\parallel)$ . Hence we can write the inverse of  $\phi$  as:

$$\phi^{-1}(\mathbf{z}') = \underbrace{\mathbf{z}'_\perp}_{\mathbf{z}_\perp} + \underbrace{f^{-1}(\mathbf{z}'_\parallel)}_{\mathbf{z}_\parallel} = \mathbf{z}, \quad (16)$$

**Case 2: R triangular, R-tilde diagonal**

Let us now consider the case when  $\mathbf{R}$  is an upper triangular matrix. By the argument for the diagonal case above, it suffices to consider the effect of the transformation in  $\mathcal{W}$ . Multiplying (13) by  $\mathbf{Q}^T$  from the left gives:

$$\underbrace{\mathbf{Q}^T\mathbf{z}'}_{\mathbf{v}'} = \underbrace{\mathbf{Q}^T\mathbf{z}}_{\mathbf{v}} + \mathbf{R}h(\underbrace{\tilde{\mathbf{R}}\mathbf{Q}^T\mathbf{z} + \mathbf{b}}_{\mathbf{v}}) \quad (17)$$

where the vectors  $\mathbf{v}$  and  $\mathbf{v}'$  contain the respective coordinates of  $\mathbf{z}_\parallel$  and  $\mathbf{z}'_\parallel$  w.r.t.  $\mathbf{q}_1, \dots, \mathbf{q}_M$ . As in the diagonal case consider the functions  $f_i(v) = v + r_{ii}h(\tilde{r}_{ii}v + b_i)$ . Since  $\|h'\|_\infty r_{ii}\tilde{r}_{ii} > -1$ , we have  $f'_i(v) > 0$  and thus  $f_i$  is invertible. Let us rewrite (17) in terms of  $f_i$ :

$$v'_1 = f_1(v_1) + \sum_{j=2}^M r_{1j}h(\tilde{r}_{jj}v_j + b_j) \quad (18)$$

...

$$v'_k = f_k(v_k) + \sum_{j=k+1}^M r_{kj}h(\tilde{r}_{jj}v_j + b_j) \quad (19)$$

...

$$v'_M = f_M(v_M) \quad (20)$$

Since  $f_M$  is invertible we can write  $v_M = f_M^{-1}(v'_M)$ . Now suppose we have expressed  $\{v_j, \forall j > k\}$  in terms of  $\{v'_j, \forall j > k\}$ . Then

$$f_k(v_k) = v'_k - \underbrace{\sum_{j=k+1}^M r_{kj}h(\tilde{r}_{jj}v_j + b_j)}_{\text{some function of } \{v'_j, \forall j > k\}}$$

$$=: g_k(v'_k, v'_{k+1}, \dots, v'_M) \quad (21)$$

$$v_k = f_k^{-1}(g_k(v'_k, v'_{k+1}, \dots, v'_M)).$$

Thus we have expressed  $\{v_j, \forall j \geq k\}$  in terms of  $\{v'_j, \forall j \geq k\}$ . By induction, we can express  $\{v_j, \forall j\}$  in terms of  $\{v'_j, \forall j\}$  and hence the transformation is invertible.

**Case 3: R and R-tilde triangular**

Now consider the general case when  $\tilde{\mathbf{R}}$  is triangular. As before we only need to consider the effect of the transformation in  $\mathcal{W}$ .

$$\underbrace{\mathbf{Q}^T\mathbf{z}'}_{\mathbf{v}'} = \underbrace{\mathbf{Q}^T\mathbf{z}}_{\mathbf{v}} + \mathbf{R}h(\underbrace{\tilde{\mathbf{R}}\mathbf{Q}^T\mathbf{z} + \mathbf{b}}_{\mathbf{v}}) \quad (22)$$

Let  $g$  be the function  $g(\mathbf{v}) = \tilde{\mathbf{R}}\mathbf{v}$ . By assumption,  $g$  is invertible with inverse  $g^{-1}$ . Multiplying (22) by  $\tilde{\mathbf{R}}$  gives:

$$g(\mathbf{v}') = \underbrace{g(\mathbf{v}) + \tilde{\mathbf{R}}\mathbf{R}h(g(\mathbf{v}) + \mathbf{b})}_{=: f(g(\mathbf{v}))} \quad (23)$$

Since  $\tilde{\mathbf{R}}\mathbf{R}$  is upper triangular with diagonal entries  $\tilde{r}_{jj}r_{jj}$ ,  $f$  is covered by case 2 considered before and is invertible. Thus,  $\mathbf{v}$  can be written as:

$$\mathbf{v} = g^{-1}(f^{-1}(g(\mathbf{v}'))). \quad (24)$$

Hence the transformation in (22) is invertible.  $\square$

**3.2 PRESERVING ORTHOGONALITY OF Q**

Orthogonality is a convenient property, mathematically, but hard to achieve in practice. In this paper we consider three different flows based on the theorem above and various ways to preserve the orthogonality of  $\mathbf{Q}$ . The first two use explicit differentiable constructions of orthogonal matrices, while the third variant assumes a specific fixed permutation matrix as the orthogonal matrix.

**Orthogonal Sylvester flows.** First, we consider a Sylvester flow using matrices with  $M$  orthogonal columns (O-SNF). In this flow we can choose  $M < D$ , and thus introduce a flexible bottleneck. Similar to [Hasenclever et al., 2017], we ensure orthogonality of



$\mathbf{Q}$  by applying the following differentiable iterative procedure proposed by [Björck and Bowie, 1971, Kovarik, 1970]:

$$\mathbf{Q}^{(k+1)} = \mathbf{Q}^{(k)} \left( \mathbf{I} + \frac{1}{2} \left( \mathbf{I} - \mathbf{Q}^{(k)\top} \mathbf{Q}^{(k)} \right) \right). \quad (25)$$

with a sufficient condition for convergence given by  $\|\mathbf{Q}^{(0)\top} \mathbf{Q}^{(0)} - \mathbf{I}\|_2 < 1$ . Here, the 2-norm of a matrix  $\mathbf{X}$  refers to  $\|\mathbf{X}\|_2 = \lambda_{\max}(\mathbf{X})$ , with  $\lambda_{\max}(\mathbf{X})$  representing the largest singular value of  $\mathbf{X}$ . In our experimental evaluations we ran the iterative procedure until  $\|\mathbf{Q}^{(k)\top} \mathbf{Q}^{(k)} - \mathbf{I}\|_F \leq \epsilon$ , with  $\|\mathbf{X}\|_F$  the Frobenius norm, and  $\epsilon$  a small convergence threshold. We observed that running this procedure up to 30 steps was sufficient to ensure convergence with respect to this threshold. To minimize the computational overhead introduced by orthogonalization we perform this orthogonalization in parallel for all flows.

Since this orthogonalization procedure is differentiable, it allows for the calculation of gradients with respect to  $\mathbf{Q}^{(0)}$  by backpropagation, allowing for any standard optimization scheme such as stochastic gradient descent to be used for updating the flow parameters.

**Householder Sylvester flows.** Second, we study Householder Sylvester flows (H-SNF) where the orthogonal matrices are constructed by products of Householder reflections. Householder transformations are reflections about hyperplanes. Let  $\mathbf{v} \in \mathbb{R}^D$ , then the reflection about the hyperplane orthogonal to  $\mathbf{v}$  is given by:

$$H(\mathbf{z}) = \mathbf{z} - \frac{\mathbf{v}\mathbf{v}^T}{\|\mathbf{v}\|^2} \mathbf{z} \quad (26)$$

It is worth noting that performing a single Householder transformation is very cheap to compute, as it only requires  $D$  parameters. Chaining together several Householder transformations results in more general orthogonal matrices, and it can be shown [Bischof and Sun, 1997, Sun and Bischof, 1995] that any  $M \times M$  orthogonal matrix can be written as the product of  $M - 1$  Householder transformations. In our Householder Sylvester flow, the number of Householder transformations  $H$  is a hyperparameter that trades off the number of parameters and the generality of the orthogonal transformation. Note that the use of Householder transformations forces us to use  $M = D$ , since Householder transformation result in square matrices.

**Triangular Sylvester flows.** Third, we consider a triangular Sylvester flow (T-SNF), in which all orthogonal matrices  $\mathbf{Q}$  alternate per transformation between the

identity matrix and the permutation matrix corresponding to reversing the order of  $\mathbf{z}$ . This is equivalent to alternating between lower and upper triangular  $\tilde{\mathbf{R}}$  and  $\mathbf{R}$  for each flow.

### 3.3 AMORTIZING FLOW PARAMETERS

When using normalizing flows in an amortized inference setting, the parameters of the base distribution as well as the flow parameters can be functions of the data point  $\mathbf{x}$  [Rezende and Mohamed, 2015]. Figure 2 (left) shows a diagram of one SNF step and the amortization procedure. The inference network takes datapoints  $\mathbf{x}$  as input, and provides as an output the mean and variance of  $\mathbf{z}^0$  such that  $\mathbf{z}^0 \sim \mathcal{N}(\mathbf{z}|\mu^0, \sigma^0)$ . Several SNF transformations are then applied to  $\mathbf{z}^0 \rightarrow \mathbf{z}^1 \rightarrow \dots \rightarrow \mathbf{z}^K$ , producing a flexible posterior distribution for  $\mathbf{z}^K$ . All of the flow parameters ( $\mathbf{R}$ ,  $\tilde{\mathbf{R}}$  and  $\mathbf{Q}$  for each transformation) are produced as an output by the inference network, and are thus fully amortized.

## 4 RELATED WORK

### 4.1 NORMALIZING FLOWS FOR VARIATIONAL INFERENCE

A number of invertible transformations with tractable Jacobians have been proposed in recent years. Rezende and Mohamed [2015] first discussed such transformations in the context of stochastic variation inference, coining the term normalizing flows.

Rezende and Mohamed [2015] proposed two different parametric families of transformations with tractable Jacobians: planar and radial flows. While effective for small problems, these transformations are hard to scale to large latent spaces and often require a large number of transformations. The transformation corresponding to planar flows is given in Eq. (8).

More recently, a successful class of flows called Inverse Autoregressive Flows was introduced in [Kingma et al., 2016]. As the name suggests, one IAF transformation can be seen as the inverse of an autoregressive transformation. Consider the following autoregressive transformation:

$$\begin{aligned} z_0 &= \bar{\mu}_0 + \bar{\sigma}_0 \cdot \epsilon_0 \\ z_i &= \bar{\mu}_i(\mathbf{z}_{1:i-1}) + \bar{\sigma}_i(\mathbf{z}_{1:i-1}) \cdot \epsilon_i, \quad i = 1, \dots, D \end{aligned} \quad (27)$$

with  $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ . This transformation models the distribution over the variable  $\mathbf{z}$  with an autoregressive factorization  $p(\mathbf{z}) = p(z_0) \prod_{i=1}^D p(z_i|z_{i-1}, \dots, z_0)$ . Since the parameters of transformation for  $z_i$  are dependent on  $\mathbf{z}_{1:i-1}$ , this procedure requires  $D$  sequential steps to sam-

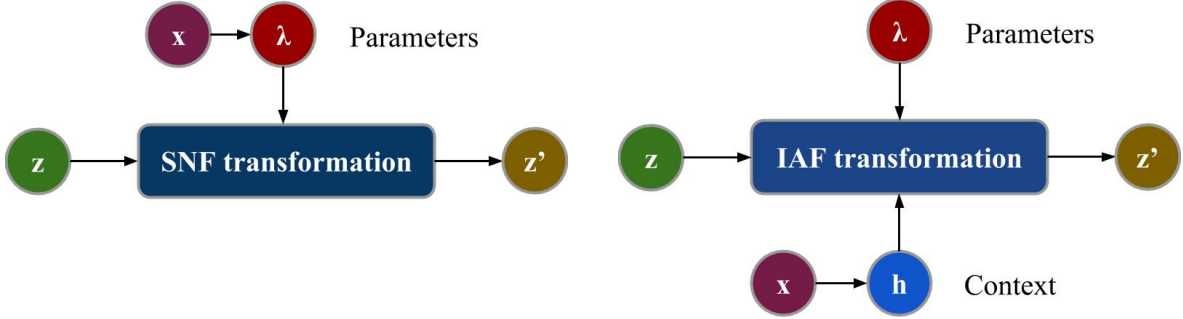


Figure 2: Different amortization strategies for Sylvester normalizing flows and Inverse Autoregressive Flows. Left: our inference network produces amortized flow parameters. This strategy is also employed by planar flows. Right: IAF has a large number of parameters, and introduces a measure of  $\mathbf{x}$  dependence through a context  $\mathbf{h}(\mathbf{x})$ . This context acts as an additional input for each transformation. The flow parameters themselves are independent of  $\mathbf{x}$ .

ple a single vector  $\mathbf{z}$ . This is undesirable for variational inference, where sampling occurs for every forward pass.

However, the inverse transformation (which exists if  $\bar{\sigma}_i > 0 \forall i$ ) is easy to sample from:

$$\epsilon_i = \frac{z_i - \bar{\mu}_i(\mathbf{z}_{1:i-1})}{\bar{\sigma}_i(\mathbf{z}_{1:i-1})}. \quad (28)$$

For this inverse transformation,  $\epsilon_i$  is no longer dependent on the transformation of  $\epsilon_j$  for  $j \neq i$ . Hence, this transformation can be computed in parallel:  $\epsilon = (\mathbf{z} - \bar{\boldsymbol{\mu}}(\mathbf{z})) / \bar{\boldsymbol{\sigma}}(\mathbf{z})$ . Rewriting  $\sigma_i(z_{1:i-1}) = 1/\bar{\sigma}_i(z_{1:i-1})$  and  $\mu_i(z_{1:i-1}) = -\bar{\mu}(z_{1:i-1})/\bar{\sigma}_i(z_{1:i-1})$ , yields the IAF transformation:

$$z_i^t = \mu_i^t(\mathbf{z}_{1:i-1}^{t-1}) + \sigma_i^t(\mathbf{z}_{1:i-1}^{t-1}) \cdot z_i^{t-1}, \quad i = 1, \dots, D. \quad (29)$$

Starting from  $\mathbf{z}^0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ , multiple IAF transformations can be stacked on top of each other to produce flexible probability distributions.

If  $\boldsymbol{\mu}^t$  and  $\boldsymbol{\sigma}^t$  depend on  $\mathbf{z}^{t-1}$  linearly, IAF can model full covariance Gaussian distributions. In order to move away from Gaussian distributions to more flexible distributions, it is important that  $\boldsymbol{\mu}^t$  and  $\boldsymbol{\sigma}^t$  are nonlinear functions of  $\mathbf{z}^{t-1}$ .

In practice, wide MADEs [Germain et al., 2015] or deep PixelCNN layers [van den Oord et al., 2016] are needed to increase the flexibility of IAF transformations. This results in transformations with a large number of parameters. As shown in Figure 2 (right), amortization is achieved through a context  $\mathbf{h}(\mathbf{x})$  that is fed into the autoregressive networks as an additional input at every IAF step.

Our Triangular Sylvester flows are strongly related to mean-only IAF transformations ( $\boldsymbol{\sigma}^t = 1$ ). As mentioned

in Kingma et al. [2016], between every IAF transformation the order of  $\mathbf{z}$  is reversed, in order to ensure that on average all dimensions get warped equally. In T-SNF, the same effect is achieved by using the permutation matrix that reverses the order of  $\mathbf{z}$  in every other transformation as the orthogonal matrix. However, mean-only IAF is a volume-preserving transformation, i.e. the determinant of the Jacobian has absolute value one. T-SNF is not volume preserving due to the nonzero elements on the diagonals of  $\mathbf{R}$  and  $\tilde{\mathbf{R}}$ . Note, that in Kingma et al. [2016] it was shown that the empirical difference in performance between mean-only IAF and the general IAF transformation is negligible.

The most important difference between IAF and T-SNF is the way parameters are amortized. In T-SNF,  $\mathbf{R}$  and  $\tilde{\mathbf{R}}$  are directly amortized functions of the input  $\mathbf{x}$  (see Fig. 2). This is equivalent to amortizing the MADE parameters in mean-only IAF. Having input dependent MADE parameters allows for flexible transformations with fewer parameters.

Householder Sylvester flows can also be seen as a non-linear extension of Householder flows [Tomczak and Welling, 2016]. Householder flows are volume-preserving flows, which transform the variational posterior with a diagonal covariance matrix to a full-covariance posterior. Householder flows are a special case of H-SNF if  $h(\mathbf{z}) = \mathbf{z}$ ,  $\mathbf{R}$  is the identity matrix, and the residual connection in Eq. (13) is left out.

## 4.2 NORMALIZING FLOWS FOR DENSITY ESTIMATION

A number of invertible transformations have been proposed in the context of density estimation. Note that density estimation requires the inverse of the flow to be tractable. Having a provably invertible transformation is

not the same as being able to compute the inverse.

For density estimation with normalizing flows, we are interested maximizing the log-likelihood of the data:

$$\log p(\mathbf{x}) = \log p_0(f^{-1}(\mathbf{x})) + \log \left| \det \left( \frac{\partial f^{-1}(\mathbf{x})}{\partial \mathbf{x}} \right) \right|. \quad (30)$$

Thus, the goal is to transform a complicated data distribution back to a simple distribution. In general, both directions of an invertible transformations need not be tractable. Hence, methods developed for density estimation are generally not directly applicable to variational inference.

Non-linear independent component estimation (NICE, Dinh et al. [2014]) and the related Real NVP [Dinh et al., 2016], and Masked Autoregressive Flow (MAF, Papamakarios et al. [2017]) are recent examples of normalizing flows for density estimation.

In NICE, each transformation splits the variables into two disjoint subsets  $\mathbf{z}_A, \mathbf{z}_B$ . One of the subsets is transformed as  $\mathbf{z}'_A = \mathbf{z}_A + f(\mathbf{z}_B)$ , while  $\mathbf{z}_B$  is left unchanged. In the next transformation a different subset of variables is transformed. This results in a transformation which is trivially invertible and has a tractable Jacobian. Real NVP uses the same fundamental idea. Appealingly, because of the tractable inverse, NICE and real NVP can generate data and estimate density with one forward pass. However due to fact that only a subset of variables is updated in each transformation many transformations are needed in practice. Rezende and Mohamed [2015] compared NICE to planar flows in the context of variational inference and found that planar flows empirically perform better.

Finally, Papamakarios et al. [2017] showed that fitting an MAF can be seen as fitting an implicit IAF from the data distribution to the base distribution. However, generating data from an MAF density model requires  $D$  passes, making it unappealing for variational inference.

## 5 NUMBER OF PARAMETERS

Here, we briefly compare the number of parameters needed by planar flows, IAF and the three Sylvester normalizing flows. We denote the size of the stochastic variables  $z$  with  $D$ , and the number of output units of the inference network with  $E$ .

Planar flows use amortized parameters  $\mathbf{u}, \mathbf{w} \in \mathbb{R}^D$  and  $b \in \mathbb{R}$  for each flow transformation. Therefore, the number of parameters related to  $K$  flow transformations is equal to  $2EDK + EK$ .

For the implementation of IAF as described in Section 6, the inference network needs to produce a context of size  $C$ , where  $C$  denotes the width of the MADE layers. The total number of flow related learnable parameters then comes down to  $EC + K \times (C^2 + 3CD)$ .

In the case of Orthogonal Sylvester flows with a bottleneck of size  $M$ , we require  $KE \times (MD + 2M^2 + M)$  parameters. For Householder Sylvester flows with  $H$  Householder reflections per flow transformation,  $KE \times (HD + 2D^2 + D)$  parameters are needed. Finally, for triangular Sylvester flows  $KE \times (2D^2 + D)$  parameters require optimization.

Planar flows require the smallest number of parameters but generally result in worse results. IAFs on the other hand require a number of parameters that is quadratic in the width of the MADE layers. For good results this has to be quite large. In contrast, for SNFs the number of parameters is quadratic in the dimension of the latent space and while large, this can still be amortized.

## 6 EXPERIMENTS

We perform empirical studies of the performance of Sylvester flows on four datasets: statically binarized MNIST, Freyfaces, Omniglot and Caltech 101 Silhouettes. The baseline model is a plain VAE with a fully factorized Gaussian distribution. We furthermore compare against planar flows and Inverse Autoregressive Flows of different sizes.

We use annealing to optimize the lower bound, where the prefactor of the KL divergence is linearly increased from 0 to 1 during 100 epochs as suggested by Bowman et al. [2015] and Sønderby et al. [2016]. A learning rate of 0.0005 was used in all experiments. In order to obtain estimates for the negative log likelihood we used importance sampling (as proposed in [Rezende et al., 2014]). Unless otherwise stated, 5000 importance samples were used.

In order to assess the performance of the different flows properly, we use the same base encoder and decoder architecture for all models. We use gated convolutions and transposed convolutions as base layers for the encoder and decoder architecture respectively. The inference network consists of several gated convolution layers that produce a hidden unit vector. After being flattened, these hidden units act as an input to two fully connected layers that predict the mean and variance of  $\mathbf{z}^0$ .

For planar and Sylvester flows, the flattened hidden units are passed to a separate linear layer that output the amortized flow parameters. For IAF, the flattened hidden units are also passed to a linear layer to produce the context

vector  $\mathbf{h}_{\text{context}}(\mathbf{x})$ . For details of the architecture see Section A of the appendix. In all models the latent space is of dimension 64.

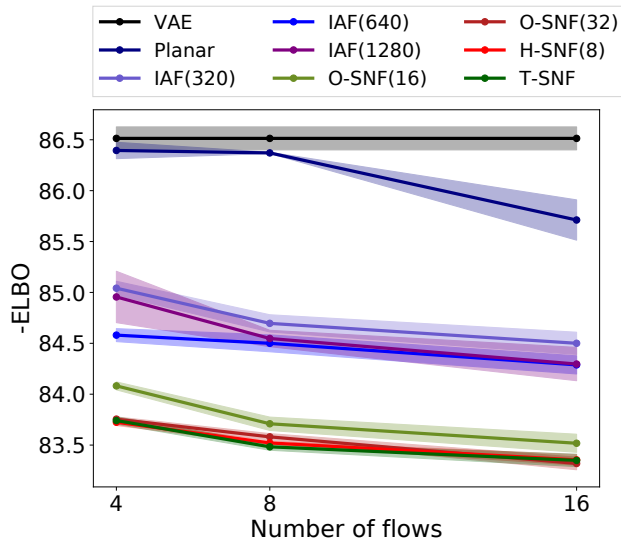


Figure 3: The negative evidence lower bound for static MNIST. The results for H-SNF with 4 reflections per orthogonal matrix are left out for clarity, as they are very similar to the results with 8 reflections. Each model is evaluated 3 times. The shaded areas indicate  $\pm$  one standard deviation.

We use the following implementation for each IAF transformation<sup>1</sup>: one IAF transformation first applies one MADE Layer (denoted as MaskedLinear) followed by a nonlinearity to the input  $z$ , upscaling it to a hidden variable of size  $M$ . At this point the context vector  $\mathbf{h}_{\text{context}}(\mathbf{x})$  is added to the hidden units, after which two more masked layers are applied to produce the mean and scale of the IAF transformation:

$$\begin{aligned}
 \mathbf{h}_z &\leftarrow \text{ELU}(\text{MaskedLinear}(\mathbf{z})) \\
 \mathbf{h} &\leftarrow \mathbf{h}_z + \mathbf{h}_{\text{context}}(\mathbf{x}) \\
 \mathbf{h} &\leftarrow \text{ELU}(\text{MaskedLinear}(\mathbf{h})) \\
 \boldsymbol{\mu} &\leftarrow \text{MaskedLinear}(\mathbf{h}), \quad \mathbf{s} \leftarrow \text{MaskedLinear}(\mathbf{h}) \\
 \mathbf{z}' &\leftarrow \sigma(\mathbf{s}) \odot \mathbf{z} + (1 - \sigma(\mathbf{s})) \odot \boldsymbol{\mu}. \tag{31}
 \end{aligned}$$

Here,  $\sigma(\cdot)$  denotes the sigmoid activation function. In Kingma et al. [2016] it was mentioned that the gated form of IAF in Eq. (31) is more stable than the form of Eq. (29). Note that the size of  $\mathbf{h}_{\text{context}}(\mathbf{x})$  scales with the width of the MADE layers  $C$ .

<sup>1</sup>This implementation is based on the open source code for IAF available at <https://github.com/openai/iaf>

Table 1: Negative log-likelihood and free energy (negative evidence lower bound) for static MNIST. Numbers are produced with 3 runs per model with different random initializations. Standard deviations over the 3 different runs are also shown.

Model	-ELBO	NLL
VAE	$86.55 \pm 0.06$	$82.14 \pm 0.07$
Planar	$86.06 \pm 0.31$	$81.91 \pm 0.22$
IAF	$84.20 \pm 0.17$	$80.79 \pm 0.12$
O-SNF	<b><math>83.32 \pm 0.06</math></b>	<b><math>80.22 \pm 0.03</math></b>
H-SNF	$83.40 \pm 0.01$	$80.29 \pm 0.02$
T-SNF	$83.40 \pm 0.10$	$80.28 \pm 0.06$

## 6.1 MNIST

Figure 3 shows the dependence of the negative evidence lower bound (or free energy) on the number of flows and the type of flow for static MNIST. The exact numbers corresponding to the figure are shown in Section C in the appendix.

For all models the performance improves as a functions of the number of flows. For 4 flows the difference between the baseline VAE and planar flows is very small. However, planar flows clearly benefit from more flow transformations.

For IAF three different widths of the MADE layers were used:  $C = 320, 640$  and  $1280$ . Surprisingly, for 4 flows the widest IAF with 1280 hidden units is outperformed by an IAF with 640 hidden units in the MADE layers. We expect this to be due to the fact that this model has more parameters and can therefore be harder to train, as indicated by the larger standard deviation for this model.

All three Sylvester flows outperform IAF and planar flows. For Orthogonal Sylvester flows, we show results for  $M = 16$  and  $M = 32$  orthogonal vectors per orthogonal matrix, thus corresponding to bottlenecks of size 16 and 32 respectively for a latent space of size  $D = 64$ . Clearly, a larger bottleneck improves performance. For Householder Sylvester flows we experimented with  $H = 4$  and  $H = 8$  Householder reflections per orthogonal matrix. Since the results were nearly indistinguishable between these two variants, we have left out the curve for  $H = 4$  to avoid clutter. O-SNF with  $M = 32$ , H-SNF and T-SNF seem to perform on par.

In Table 1, the negative evidence lower bound and the estimated negative log-likelihood are shown for the baseline VAE, together with all flow models for 16 flows. The reported result for IAF is for a MADE width of 1280. The O-SNF model has a bottleneck of  $M = 32$ , and

Table 2: Results for Freyfaces, Omniglot and Caltech 101 Silhouettes datasets. For the Freyfaces dataset the results are reported in bits per dim. For the other datasets the results are reported in nats. For each flow model 16 flows are used. For IAF a MADE width of 1280 was used, and for O-SNF flow a bottleneck of  $M = 32$  was used. For H-SNF 8 householder reflections were used to construct orthogonal matrices. For all datasets 3 runs per model were performed.

Model	Freyfaces		Omniglot		Caltech 101	
	-ELBO	NLL	-ELBO	NLL	-ELBO	NLL
VAE	$4.53 \pm 0.02$	$4.40 \pm 0.03$	$104.28 \pm 0.39$	$97.25 \pm 0.23$	$110.80 \pm 0.46$	$99.62 \pm 0.74$
Planar	<b><math>4.40 \pm 0.06</math></b>	<b><math>4.31 \pm 0.06</math></b>	$102.65 \pm 0.42$	$96.04 \pm 0.28$	$109.66 \pm 0.42$	$98.53 \pm 0.68$
IAF	$4.47 \pm 0.05$	$4.38 \pm 0.04$	$102.41 \pm 0.04$	$96.08 \pm 0.16$	$111.58 \pm 0.38$	$99.92 \pm 0.30$
O-SNF	$4.51 \pm 0.04$	$4.39 \pm 0.05$	$99.00 \pm 0.29$	$93.82 \pm 0.21$	$106.08 \pm 0.39$	$94.61 \pm 0.83$
H-SNF	$4.46 \pm 0.05$	$4.35 \pm 0.05$	<b><math>99.00 \pm 0.04</math></b>	<b><math>93.77 \pm 0.03</math></b>	<b><math>104.62 \pm 0.29</math></b>	<b><math>93.82 \pm 0.62</math></b>
T-SNF	$4.45 \pm 0.04$	$4.35 \pm 0.04$	$99.33 \pm 0.23$	$93.97 \pm 0.13$	$105.29 \pm 0.64$	$94.92 \pm 0.73$

H-SNF contains 8 Householder reflections per orthogonal matrix. Again, all Sylvester flows outperform planar flows and IAF, both in terms of the free energy and the negative log-likelihood.

As discussed in Section 4, T-SNF is closely related to mean-only IAF, but with the MADE parameters directly amortized. The fact that T-SNF outperforms IAF indicates that amortizing the parameters directly leads to a more flexible transformation compared to taking a very wide MADE with a data dependent context as an additional input.

## 6.2 FREYFACES, OMNIGLOT AND CALTECH 101 SILHOUETTES

We further assess the performance of the different models on Freyfaces, Omniglot and Caltech 101 Silhouettes. The results are shown in Table 2. The model settings are the same<sup>2</sup> as those used for Table 1.

Freyfaces is a very small dataset of around 2000 faces. All normalizing flows increase the performance, with planar flows yielding the best result, closely followed by Triangular and Householder Sylvester flows. We expect planar flows to perform the best in this case since it is the least sensitive to overfitting.

For Omniglot and Caltech 101 Silhouettes the results are clearer, with the Sylvester normalizing flows family resulting in the best performance. Both H-SNF and T-SNF perform better than O-SNF. This could be attributed to the fact that O-SNF has a bottleneck of  $M = 32$  for a latent space size of  $D = 64$ . The IAF scores for Caltech 101 are surprisingly bad. We expect this could be the case due to the large number of parameters that need to be trained for IAF(1280). Therefore we also evaluated

<sup>2</sup>For Caltech 101 Silhouettes we used 2000 importance samples for the estimation of the negative log-likelihood.

the result for MADEs of width 320 for 16 flows. The resulting free energy and estimated negative log-likelihood are  $111.23 \pm 0.45$  and  $99.74 \pm 0.28$  respectively, only slightly improving on the results of 1280 wide IAFs.

## 7 CONCLUSION

We present a new family of normalizing flows: Sylvester normalizing flows. These flows generalize planar flows, while maintaining an efficiently computable Jacobian determinant through the use of Sylvester’s determinant identity. We ensure invertibility of the flows through the use of orthogonal and triangular parameter matrices. Three variants of Sylvester flows are investigated. First, orthogonal Sylvester flows use an iterative procedure to maintain orthogonality of parameter matrices. Second, Householder Sylvester flows use Householder reflections to construct orthogonal matrices. Third, triangular Sylvester flows alternate between fixed permutation and identity matrices for the orthogonal matrices. We show that the triangular Sylvester flows are closely related to mean-only IAF, with directly amortized MADE parameters. While performing comparably with planar flows and IAF for the Freyfaces dataset, our proposed family of flows improve significantly upon planar flows and IAF on the three other datasets.

### Acknowledgements

We would like to thank Christos Louizos for useful discussions and helping with the implementation of inverse autoregressive flows. LH is funded by the UK EPSRC OxWaSP CDT through grant EP/L016710/1. JMT is funded by the European Commission within the MSC-IF (Grant No. 702666). RvdB is funded by SAP SE.

## References

- Christian Bischof and Xiaobai Sun. On orthogonal block elimination. Technical Report MCS-P450-0794, Argonne National Laboratory, Argonne, IL, 10 1997.
- Åke Björck and Clazett Bowie. An iterative algorithm for computing the best estimate of an orthogonal matrix. *SIAM Journal on Numerical Analysis*, 8(2):358–364, 1971.
- Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. Generating Sentences from a Continuous Space. nov 2015. URL <http://arxiv.org/abs/1511.06349>.
- Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: non-linear independent components estimation. [abs/1410.8516](https://arxiv.org/abs/1410.8516), 2014.
- Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP. *arXiv preprint arXiv:1605.08803*, 2016.
- Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. MADE: Masked Autoencoder for Distribution Estimation. *ICML*, pages 881–889, 2015.
- Samuel Gershman and Noah Goodman. Amortized inference in probabilistic reasoning. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 36, 2014.
- Leonard Hasenclever, Jakub Tomczak, Rianne van den Berg, and Max Welling. Variational inference with orthogonal normalizing flows. 2017.
- Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14:1303–1347, 2013.
- Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.
- Diederik Kingma and Max Welling. Efficient gradient-based inference through transformations between bayes nets and neural nets. *ICML*, pages 1782–1790, 2014.
- Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved Variational Inference with Inverse Autoregressive Flow. *NIPS*, pages 4743–4751, 2016.
- Zdislav Kovarik. Some iterative methods for improving orthonormality. *SIAM Journal on Numerical Analysis*, 7(3):386–389, 1970.
- Eric Nalisnick, Lars Hertel, and Padhraic Smyth. Approximate inference for deep latent gaussian mixtures. In *NIPS Workshop on Bayesian Deep Learning*, 2016.
- George Papamakarios, Iain Murray, and Theo Pavlakou. Masked Autoregressive Flow for Density Estimation. *NIPS*, pages 2335–2344, 2017.
- Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. *ICML*, pages 1530–1538, 2015.
- Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.
- Tim Salimans, Diederik Kingma, and Max Welling. Markov Chain Monte Carlo and variational inference: Bridging the gap. *ICML*, pages 1218–1226, 2015.
- Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder Variational Autoencoders. feb 2016. URL <http://arxiv.org/abs/1602.02282>.
- Xiaobai Sun and Christian Bischof. A basis-kernel representation of orthogonal matrices. *SIAM Journal on Matrix Analysis and Applications*, 16(4):1184–1196, 1995.
- EG Tabak and Cristina V Turner. A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164, 2013.
- Esteban G Tabak and Eric Vanden-Eijnden. Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, 8(1):217–233, 2010.
- Jakub M Tomczak and Max Welling. Improving Variational Auto-encoders using Householder Flow. *arXiv preprint arXiv:1611.09630*, 2016.
- Dustin Tran, Rajesh Ranganath, and David M Blei. The variational Gaussian process. *arXiv preprint arXiv:1511.06499*, 2015.
- Aaron van den Oord, Nal Kalchbrenner, Lasse Espeholt, koray kavukcuoglu, Oriol Vinyals, and Alex Graves. Conditional image generation with pixelcnn decoders. *NIPS*, pages 4790–4798, 2016.

---

# Holistic Representations for Memorization and Inference

---

**Yunpu Ma\***  
LMU  
Siemens AG  
Otto-Hahn-Ring 6  
81739 Munich

**Marcel Hildebrandt**  
LMU  
Siemens AG  
Otto-Hahn-Ring 6  
81739 Munich

**Stephan Baier**  
LMU  
Oettingenstr. 67  
80538 Munich

**Volker Tresp**  
LMU  
Siemens AG  
Otto-Hahn-Ring 6  
81739 Munich

## Abstract

In this paper we introduce a novel holographic memory model for the distributed storage of complex association patterns and apply it to knowledge graphs. In a knowledge graph, a labelled link connects a subject node with an object node, jointly forming a subject-predicate-objects triple. In the presented work, nodes and links have initial random representations, plus *holistic representations* derived from the initial representations of nodes and links in their local neighbourhoods. A memory trace is represented in the same vector space as the holistic representations themselves. To reduce the interference between stored information, it is required that the initial random vectors should be pairwise quasi-orthogonal. We show that pairwise quasi-orthogonality can be improved by drawing vectors from heavy-tailed distributions, e.g., a Cauchy distribution, and, thus, memory capacity of holistic representations can significantly be improved. Furthermore, we show that, in combination with a simple neural network, the presented holistic representation approach is superior to other methods for link predictions on knowledge graphs.

## 1 INTRODUCTION

An associative memory is a key concept in artificial intelligence and cognitive neuroscience for learning and memorizing relationships between entities and concepts. Various computational models of associative memory have been proposed, see, e.g., [Hopfield 1982; Gentner 1983]. One important family of associative memory

models is the holographic associative memory (HAM), which was first proposed in [Gabor 1969]. HAMS can store a large number of stimulus-response pairs as additive superpositions of memory traces. It has been suggested that this holographic storage is related to the working principle of the human brain [Westlake 1970].

An important extension to the HAM is based on holographic reduced representations (HRR) [Plate 1995]. In HRR, each entity or symbol is represented as a vector defined in a continuous space. Associations between two entities are compressed in the same vector space via a vector binding operation; the resulting vector is a memory trace. Two associated entities are referred to as a *cue-filler* pair, since a noisy version of the *filler* can be recovered from the memory trace and the *cue* vector via a decoding operation. Multiple *cue-filler* pairs can be compressed in a single memory trace through superposition. Associations can be read out from this single trace, however with large distortions. Thus, a clean-up mechanism was introduced into HRR, such that associations can be *retrieved* with high probability.

The number of associations which can be compressed in a single trace is referred to as *memory capacity*. It has been shown in [Plate 1995] that the memory capacity of the HRR depends on the degree of the pairwise orthogonality of initial random vectors associated with the entities.

Quasi-orthogonality was put forward in [Diaconis et al. 1984; Hall et al. 2005]. They informally stated that “most independent high-dimensional random vectors are nearly orthogonal to each other”. A rigorous mathematical justification to this statement has only recently been given in [Cai et al. 2012; Cai et al. 2013], where the density function of pairwise angles among a large number of Gaussian random vectors was derived. To the best of our knowledge, density functions for other distributions have not been derived, so far. As a first contribution, we will derive a significantly improved quasi-orthogonality, and

---

\*yunpu.ma@siemens.com

we show that memory capacity of holographic representations can significantly be improved. Our result could potentially have numerous applications, e.g., in sparse random projections or random geometric graphs [Penrose 2003].

After the HRR had been proposed, it had mainly been tested on small toy datasets. Quasi-orthogonality becomes exceedingly important when a large amount of entities needs to be initialized with random vectors, as in applications involving large-scale knowledge graphs.

Modern knowledge graphs (KGs), such as FREEBASE [Bollacker et al. 2008], YAGO [Suchanek et al. 2007], and GDELTA [Leetaru et al. 2013], are relational knowledge bases, where nodes represent entities and directed labelled links represent predicates. An existing labelled link between a head node (or subject) and a tail node (or object) is a triple and represents a fact, e.g. (*California, locatedIn, USA*).

As a second contribution, we demonstrate how the holographic representations can be applied to KGs. First, one needs to define association pairs (or *cue-filler* pairs). We propose that the representation of a *subject* should encode all *predicate-object* pairs, such that given the *predicate* representation as a *cue*, the *object* should be recovered or at least recognized. Similarly, the representation of an *object* should encode all *predicate-subject* pairs, such that the *subject* can be retrieved after decoding with the *predicate* representation. We call those representations *holistic*, since they are inspired by the semantic holism in the philosophy of language, in the sense that an abstract entity can only be comprehended through its relationships to other abstract entities.

So far we have discussed memory formation and memory retrieval. Another important function is the generalization of stored memory to novel facts. This has technical applications and there are interesting links to human memory. From a cognitive neuroscientist point of view, the brain requires a dual learning system: one is the hippocampus for rapid memorization, and the other is the neocortex for gradual consolidation and comprehension. This hypothesis is the basis for the *Complementary Learning System* (CLS) which was first proposed in [McClelland et al. 1995]. Connections between KGs and long-term declarative memories has recently been stated in [Tresp et al. 2017a; Ma et al. 2018; Tresp et al. 2017b].

As a third contribution of this paper, we propose a model which not only memorizes patterns in the training datasets through holistic representations, but also is able to infer missing links in the KG, by a simple neural network that uses the holistic representations as input representations. Thus, our model realizes a form of

a *complementary learning system*. We compare our results on multiple datasets with other state-of-the-art link prediction models, such as RESCAL [Nickel et al. 2011; Nickel et al. 2012], DISTMULT [Yang et al. 2014], COMPLEX [Trouillon et al. 2016], and R-GCN [Schlichtkrull et al. 2018].

The above mentioned learning-based methods model the KGs by optimizing the latent representations of entities and predicates through minimizing the loss function. It had been observed that latent embeddings are suitable for capturing global connectivity patterns and generalization [Nickel et al. 2016a; Toutanova et al. 2015], but are not as good in memorizing unusual patterns, such as patterns associated with locally and sparsely connected entities. This motivates us to *separate* the memorization and inference tasks. As we will show in our experiments, our approach can, on the one hand, memorize local graph structures, but, on the other hand, also generalizes well to global connectivity patterns, as required by complementary learning systems.

Note, that in our approach holistic representations are derived from random vectors and are **not** learned from data via backpropagation, as in most learning-based approaches to representation learning on knowledge graphs. One might consider representations derived from random vectors to be biologically more plausible, if compared to representations which are learned via complex gradient based update rules [Nickel et al. 2016a]. Thus, in addition to its very competitive technical performance, one of the interesting aspects of our approach is its biological plausibility.

In Section 2 we introduce notations for KGs and embedding learning. In Section 3 we discuss improved quasi-orthogonality by using heavy-tailed distributions. In Section 4 we propose our own algorithm for holistic representations, and test it on various datasets. We also discuss how the memory capacity can be improved. In Section 5 we propose a model which can infer implicit links on KGs through holistic representations. Section 6 contains our conclusions.

## 2 REPRESENTATION LEARNING

In this section we provide a brief introduction to representation learning in KGs, where we adapt the notation of [Nickel et al. 2016b]. Let  $\mathcal{E}$  denotes the set of entities, and  $\mathcal{P}$  the set of predicates. Let  $N_e$  be the number of entities in  $\mathcal{E}$ , and  $N_p$  the number of predicates in  $\mathcal{P}$ .

Given a predicate  $p \in \mathcal{P}$ , the characteristic function  $\phi_p : \mathcal{E} \times \mathcal{E} \rightarrow \{1, 0\}$  indicates whether a triple  $(\cdot, p, \cdot)$  is true or false. Moreover,  $\mathcal{R}_p$  denotes the set of all subject-object pairs, such that  $\phi_p = 1$ . The entire KG can be



written as  $\chi = \{(i, j, k)\}$ , with  $i = 1, \dots, N_e$ ,  $j = 1, \dots, N_p$ , and  $k = 1, \dots, N_e$ .

We assume that each entity and predicate has a unique latent representation. Let  $\mathbf{a}_{e_i}$ ,  $i = 1, \dots, N_e$ , be the representations of entities, and  $\mathbf{a}_{p_i}$ ,  $i = 1, \dots, N_p$ , be the representations of predicates. Note that  $\mathbf{a}_{e_i}$  and  $\mathbf{a}_{p_i}$  could be real- or complex-valued vectors/matrices.

A probabilistic model for the KG  $\chi$  is defined as  $\Pr(\phi_p(s, o) = 1 | \mathcal{A}) = \sigma(\eta_{spo})$  for all  $(s, p, o)$ -triples in  $\chi$ , where  $\mathcal{A} = \{\mathbf{a}_{e_i}\}_i^{N_e} \cup \{\mathbf{a}_{p_i}\}_i^{N_p}$  denotes the collection of all embeddings;  $\sigma(\cdot)$  denotes the sigmoid function; and  $\eta_{spo}$  is the a function of latent representations,  $\mathbf{a}_s$ ,  $\mathbf{a}_p$  and  $\mathbf{a}_o$ . Given a labeled dataset containing both true and false triples  $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^m$ , with  $x_i \in \chi$ , and  $y_i \in \{1, 0\}$ , latent representations can be learned. Commonly, one minimizes a binary cross-entropy loss

$$-\frac{1}{m} \sum_{i=1}^m (y_i \log(p_i) + (1 - y_i) \log(1 - p_i)) + \lambda \|\mathcal{A}\|_2^2, \quad (1)$$

where  $m$  is the number of training samples, and  $\lambda$  is the regularization parameter;  $p_i := \sigma(\eta_{x_i})$  with  $\sigma(\cdot)$  being the sigmoid function.  $\eta_{spo}$  is defined differently in various models.

For instance, for RESCAL entities are represented as  $r$ -dimensional vectors,  $\mathbf{a}_{e_i} \in \mathbb{R}^r$ ,  $i = 1, \dots, N_e$ , and predicates are represented as matrices,  $\mathbf{a}_{p_i} \in \mathbb{R}^{r \times r}$ ,  $i = 1, \dots, N_p$ . Moreover, one uses  $\eta_{spo} = \mathbf{a}_s^\top \mathbf{a}_p \mathbf{a}_o$ .

For DISTMULT,  $\mathbf{a}_{e_i}, \mathbf{a}_{p_j} \in \mathbb{R}^r$ , with  $i = 1, \dots, N_e, j = 1, \dots, N_p$ ;  $\eta_{spo} = \langle \mathbf{a}_s, \mathbf{a}_p, \mathbf{a}_o \rangle$ , where  $\langle \cdot, \cdot, \cdot \rangle$  denotes the tri-linear dot product.

For COMPLEX,  $\mathbf{a}_{e_i}, \mathbf{a}_{p_j} \in \mathbb{C}^r$ , with  $i = 1, \dots, N_e, j = 1, \dots, N_p$ ;  $\eta_{spo} = \Re(\langle \mathbf{a}_s, \mathbf{a}_p, \bar{\mathbf{a}}_o \rangle)$ , where the bar denotes complex conjugate, and  $\Re$  denotes the real part.

### 3 DERIVATION OF $\epsilon$ -ORTHOGONALITY

As we have discussed in the introduction, quasi-orthogonality of the random vectors representing the entities and the predicates is required for low interference memory retrieval. In this section we investigate the asymptotic distribution of pairwise angles in a set of independently and identically drawn random vectors. In particular, we study random vectors drawn from either a Gaussian or a heavy-tailed Cauchy distribution distribution. A brief summary of notations is referred to the A.7. First we define the term “ $\epsilon$ -orthogonality”.

**Definition 1.** A set of  $n$  vectors  $\mathbf{x}_1, \dots, \mathbf{x}_n$  is said to be pairwise  $\epsilon$ -orthogonal, if  $|\langle \mathbf{x}_i, \mathbf{x}_j \rangle| < \epsilon$  for  $i, j = 1, \dots, n, i \neq j$ .

Here,  $\epsilon > 0$  is a small positive number, and  $\langle \cdot, \cdot \rangle$  denotes the inner product in the vector space.

#### 3.1 $\epsilon$ -ORTHOGONALITY FOR A GAUSSIAN DISTRIBUTION

In this section we revisit the empirical distribution of pairwise angles among a set of random vectors. More specifically, let  $\mathbf{X}_1, \dots, \mathbf{X}_n$  be independent  $q$ -dimensional Gaussian variables with distribution  $\mathcal{N}(\mathbf{0}, \mathbf{I}_q)$ . Denote with  $\Theta_{ij}$  the angle between  $\mathbf{X}_i$  and  $\mathbf{X}_j$ , and  $\rho_{ij} := \cos \Theta_{ij} \in [-1, 1]$ . [Cai et al. 2012; Muirhead 2009] derived the density function of  $\rho_{ij}$  in the following Lemma.

**Lemma 1.** Consider  $\rho_{ij}$  as defined above. Then  $\{\rho_{ij} | 1 < i < j \leq n\}$  are pairwise i.i.d. random variables with the following asymptotic probability density function

$$g(\rho_G) = \frac{1}{\sqrt{\pi}} \frac{\Gamma(\frac{q}{2})}{\Gamma(\frac{q-1}{2})} (1 - \rho_G^2)^{\frac{q-3}{2}}, \quad |\rho_G| < 1, \quad (2)$$

with fixed dimensionality  $q$ .

[Cai et al. 2013] also derived the following Theorem 1.

**Theorem 1.** Let the empirical distribution  $\mu_n$  of pairwise angles  $\Theta_{ij}, 1 \leq i < j \leq n$  be defined as  $\mu_n := \frac{1}{\binom{n}{2}} \sum_{1 \leq i < j \leq n} \delta_{\Theta_{ij}}$ . With fixed dimension  $q$ , as  $n \rightarrow \infty$ ,  $\mu_n$  converges weakly to the distribution with density

$$h(\theta) = \frac{1}{\sqrt{\pi}} \frac{\Gamma(\frac{q}{2})}{\Gamma(\frac{q-1}{2})} (\sin \theta)^{q-2}, \quad \theta \in [0, \pi]. \quad (3)$$

From the above distribution function we can derive the upper bound of quasi-orthogonal random vectors with pairwise  $\epsilon$ -orthogonality in the Euclidean space  $\mathbb{R}^q$ .

**Corollary 1.** Consider a set of independent  $q$ -dimensional Gaussian random vectors which are pairwise  $\epsilon$ -orthogonal with probability  $1 - \nu$ , then the number of such Gaussian random vectors is bounded by

$$N \leq \sqrt[4]{\frac{\pi}{2q}} e^{\frac{\epsilon^2 q}{4}} \left[ \log \left( \frac{1}{1 - \nu} \right) \right]^{\frac{1}{2}}. \quad (4)$$

The derivation is given in A.1. Due to the symmetry of density function  $g(\rho_G)$ , we immediately have  $\mathbb{E}[\rho_G] = 0$ , moreover,  $\mathbb{E}[\theta] = \frac{\pi}{2}$ . However, for the later use, it is important to consider the expected absolute value of  $\rho_G$ :

**Corollary 2.** Consider a set of  $n$   $q$ -dimensional random Gaussian vectors, we have

$$\lambda_G := \mathbb{E}[|\rho_G|] = \sqrt{\frac{2}{\pi q}}. \quad (5)$$

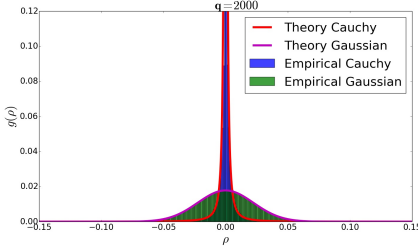


Figure 1: Empirical pairwise angle distribution in a set of Gaussian random vectors (green) is compared with theoretical prediction Eq. 2 (magenta); Empirical pairwise angle distribution in a set of Cauchy random vectors (blue) is compared with prediction Eq. 6 (red)

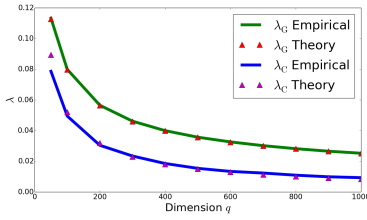


Figure 2: Compare  $\lambda_G$  and  $\lambda_C$  from simulation and theory, see Eq. 5 and Eq. 9.

Note, that the quantity  $\frac{\pi}{2} - \arccos \mathbb{E}[|\rho_G|]$  has a clear geometrical meaning: It indicates the expected deviation from  $\frac{\pi}{2}$  of pairwise angles. In fact, in the extreme case when  $q \rightarrow \infty$ , the deviation converges to 0 with the rate  $\sqrt{q}$ .

### 3.2 $\epsilon$ -ORTHOGONALITY FOR A CAUCHY DISTRIBUTION

In this subsection, we show that the set of random vectors whose elements are initialized with a heavy-tailed distribution, e.g., a Cauchy distribution  $\mathcal{C}(0, 1)$ , has improved  $\epsilon$ -orthogonality. The intuition is as follows: Consider a set of  $q$ -dimensional random vectors initialized with a heavy-tailed distribution. After normalization, each random vector can be approximated by only the elements which significantly deviate from zero and were drawn from the heavy tails. If the number of those elements is  $k$  with  $k \ll q$ , then there are at most  $\binom{q}{k}$  orthogonal random vectors.

Moreover,  $\binom{q}{k} \approx \frac{q^k}{k\Gamma(k)}$  could be much larger than  $\sqrt[4]{\frac{\pi}{2q}} e^{\frac{\epsilon^2 q}{4}}$  from Eq. 4, when  $q$  is sufficiently large,  $k \ll q$ , and  $\epsilon \rightarrow 0$ . In other words, under stricter quasi-orthogonality condition with smaller  $\epsilon$ , random vectors drawn from a heavy-tailed distribution could have more pairs satisfying the quasi-orthogonality condition.

Consider a set of  $q$ -dimensional Cauchy random vectors. As  $q \rightarrow \infty$  the approximate density function of  $\rho_{ij}$ , with  $1 \leq i < j \leq n$  is described in the following conjecture.

**Conjecture 1.** Let  $\mathbf{X}_1, \dots, \mathbf{X}_n$  be independent  $q$ -dimensional random vectors whose elements are independently and identically drawn from Cauchy a distribution  $\mathcal{C}(0, 1)$ . Moreover, consider the angle  $\Theta_{ij}$  between  $\mathbf{X}_i$ , and  $\mathbf{X}_j$ . Then, as  $q \rightarrow \infty$ ,  $\rho_{ij} := \cos \Theta_{ij} \in [-1, 1]$ ,  $1 \leq i < j \leq n$  are pairwise i.i.d. with a density function approximated by

$$g(\rho_C) = -\frac{2}{\pi^2 q^2 \rho_C^3} \cdot \frac{1}{z^{\frac{3}{2}}} \left[ e^{\frac{1}{\pi z}} \text{Ei} \left( -\frac{1}{\pi z} \right) \right], \quad (6)$$

where  $z := \frac{1}{q^2} \left( \frac{1}{\rho_C^2} - 1 \right)$ , and the exponential integral

$$\text{Ei}(x) \text{ is defined as } \text{Ei}(x) = -\int_{-x}^{\infty} \frac{e^{-t}}{t} dt.$$

The intuition behind the conjecture is as follows. Suppose  $\mathbf{X} = (X_1, \dots, X_q)$  and  $\mathbf{Y} = (Y_1, \dots, Y_q)$  are random vector variables, and assume that elements of  $\mathbf{X}$  and  $\mathbf{Y}$  are independently Gaussian distributed. In order to derive  $g(\rho_{\mathbf{X}, \mathbf{Y}})$  in Lemma 1, [Cai et al. 2012; Muirhead 2009] compute the distribution function for  $\frac{\alpha^T \mathbf{X}}{\|\mathbf{X}\|}$  instead, where  $\alpha^T \alpha = 1$ . In particular, they assume that  $\alpha = (1, 0, \dots, 0)$ . The underlying reason for this assumption is that the random vector  $\frac{\mathbf{X}}{\|\mathbf{X}\|}$  is uniformly distributed on the  $(q-1)$ -dimensional sphere.

Here, elements of  $\mathbf{X}$  and  $\mathbf{Y}$  are independently Cauchy distributed. We derive the approximation in Eq. 6 under the same assumption by taking  $g(\rho_{\mathbf{X}, \mathbf{Y}}) \approx \frac{X_1}{\sqrt{X_1^2 + \dots + X_q^2}}$ . Furthermore, we introduce a new variable  $z_{\mathbf{X}, \mathbf{Y}} := \frac{1}{q^2} \left( \frac{1}{\rho_{\mathbf{X}, \mathbf{Y}}^2} - 1 \right) = \frac{1}{q^2} \frac{X_2^2 + \dots + X_q^2}{X_1^2}$ , and derive the density function  $\hat{g}(z_{\mathbf{X}, \mathbf{Y}})$  by using the generalized central limit theorem [Gnedenko et al. 1954] and properties of quotient distributions of two independent random variables.  $g(\rho_{\mathbf{X}, \mathbf{Y}})$  can be directly obtained from  $\hat{g}(z_{\mathbf{X}, \mathbf{Y}})$  by a variable transform. More details and derivation are referred to the A.2.

We turn to study the limiting behaviour of the density function when  $\rho$  approaches zero. In this case, the variable  $z$  defined in in Conjecture 1 can be approximated by  $z \approx \frac{1}{q^2 \rho_C^2}$ . Using properties of the exponential integral, as  $q \rightarrow \infty$ , the density function in Eq. 6 can be approximated by its Laurent series,

$$g(\rho_C) \approx \frac{2}{\pi q \rho_C^2} - \frac{2}{q^3 \rho_C^4} + \frac{4\pi}{q^5 \rho_C^6} + \mathcal{O} \left( \frac{1}{q^7 \rho_C^8} \right) \quad (7)$$

In the following corollary we give the upper bound of the number of pairwise  $\epsilon$ -orthogonal Cauchy random vectors using Eq. 6.

**Corollary 3.** Consider a set of independent  $q$ -dimensional Cauchy random vectors which are pairwise  $\epsilon$ -orthogonal with probability  $1 - \nu$ , then the number of such Cauchy random vectors is bounded by

$$N \leq \sqrt{\frac{\pi \epsilon q}{4}} \left[ \log \left( \frac{1}{1 - \nu} \right) \right]^{\frac{1}{2}}. \quad (8)$$

Let us compare the prefactor of this upper bound for two distributions: That is  $\sqrt[4]{\frac{\pi}{2q}} e^{\frac{\epsilon^2 q}{4}}$  for the Gaussian distribution, and  $\sqrt{\frac{\pi \epsilon q}{4}}$  for the Cauchy distribution. Under strict quasi-orthogonal conditions with arbitrarily small but fixed  $\epsilon > 0$ , for the dimension  $q \gg 2 \sqrt[3]{\frac{1}{\pi \epsilon^2}}$  we have that  $\sqrt{\frac{\pi \epsilon q}{4}} \gg \sqrt[4]{\frac{\pi}{2q}} e^{\frac{\epsilon^2 q}{4}} \approx \sqrt[4]{\frac{\pi}{2q}}$ . It implies that in sufficiently high-dimensional spaces, random vectors which are independently drawn from a Cauchy distribution are more likely to satisfy the pairwise  $\epsilon$ -orthogonality condition - particularly when  $\epsilon \ll 1$ .

**Remark 1.** For the later use, we define  $\lambda_C$  as  $\lambda_C := \mathbb{E}[|\rho_C|]$  for the case of Cauchy distribution. However, no simple analytic form is known for this integral. Thus we use the following numerically stable and non-divergent equation to approximate  $\lambda_C$ ,

$$\lambda_C \approx -\frac{4q}{\pi^2} \int_0^1 \rho \left[ e^{\frac{q^2 \rho^2}{\pi}} \text{Ei} \left( -\frac{q^2 \rho^2}{\pi} \right) \right] d\rho. \quad (9)$$

This simpler form is derived from Eq. 6 using the approximation  $z \approx \frac{1}{q^2 \rho^2}$ .

Fig. 1 shows the empirical distribution of  $\rho_C$  in a set of Gaussian random vectors (green) compared with theoretical prediction in Eq.2 (magenta); and the empirical distribution of  $\rho_C$  in a set of Cauchy random vectors (blue) compared with theoretical prediction (red). In the case of Cauchy random vectors, the leading orders of the Laurent expansion of Eq. 6 are used, see Eq. 7. For the empirical simulation, 10000 random vectors with dimensionality  $q = 2000$  were drawn independently from either a Gaussian or a Cauchy distribution.

In addition, in Fig. 2 we plot  $\lambda_G$  and  $\lambda_C$  as a function of  $q$  in comparison with the theoretical predictions from Eq. 5 and Eq. 9, respectively, under the same simulation condition. It is necessary to emphasize that  $\lambda_C(q) < \lambda_G(q)$  for all the dimensions  $q$ ; this fact will be used to explain the relatively high memory capacity encoded from Cauchy random vectors.

In the Appendix, see Remark A 2, the distribution of elements from the normalized random variable  $\frac{\mathbf{x}}{\|\mathbf{x}\|}$  is also considered. In particular, for normalized Cauchy random vector most of its elements are nearly zero, and it realizes a **sparse** representation.

## 4 HOLISTIC REPRESENTATIONS FOR KGS

### 4.1 HRR MODEL

First, we briefly review HRR. Three operations are defined in HRR to model associative memories: *encoding*, *decoding*, and *composition*.

Let  $\mathbf{a}$ ,  $\mathbf{b}$ ,  $\mathbf{c}$ , and  $\mathbf{d}$  be random vectors representing different entities. The encoding phase stores the association between  $\mathbf{a}$  and  $\mathbf{b}$  in a memory trace  $\mathbf{a} * \mathbf{b}$ , where  $*$  :  $\mathbb{R}^q \times \mathbb{R}^q \rightarrow \mathbb{R}^q$  denotes circular convolution, which is defined as  $[\mathbf{a} * \mathbf{b}]_k = \sum_{i=0}^{q-1} a_i b_{(k-i) \bmod q}$ .

A noisy version of  $\mathbf{b}$  can be retrieved from the memory trace, using the item  $\mathbf{a}$  as a cue, with:  $\mathbf{b} \approx \mathbf{a} \star (\mathbf{a} * \mathbf{b})$ , where  $\star$  :  $\mathbb{R}^q \times \mathbb{R}^q \rightarrow \mathbb{R}^q$  denotes the circular correlation<sup>1</sup>. It is defined as  $[\mathbf{a} \star \mathbf{b}]_k = \sum_{i=0}^{q-1} a_i b_{(k+i) \bmod q}$ . In addition, several associations can be superimposed in a single trace via the addition operation:  $(\mathbf{a} * \mathbf{b}) + (\mathbf{c} * \mathbf{d}) + \dots$ .

### 4.2 HOLISTIC MODEL

Initially, each entity and predicate in a KG is associated with a  $q$ -dimensional normalized random vector, which is then normalized. We denote them as  $\mathbf{r}_{e_i}^{G/C}$ ,  $i = 1, \dots, N_e$ , and  $\mathbf{r}_{p_i}^{G/C}$ ,  $i = 1, \dots, N_p$ , respectively. The superscript indicates from which distribution vector elements are independently drawn, either the Gaussian or Cauchy distribution. If there is no confusion, we may omit the superscript.

Consider an entity  $e_i$ . Let  $\mathcal{S}^s(e_i) = \{(p, o) | \phi_p(e_i, o) = 1\}$  be the set of all predicate-object pairs for which triples  $(e_i, p, o)$  is true and where  $e_i$  is the subject. We store these multiple associations in a single memory trace via circular correlation and superposition:

$$\mathbf{h}_{e_i}^s = \sum_{(p, o) \in \mathcal{S}^s(e_i)} [\text{Norm}(\mathbf{r}_p \star \mathbf{r}_o) + \xi \mathbf{r}_{e_i}], \quad (10)$$

where  $\text{Norm} : \mathbb{R}^q \rightarrow \mathbb{R}^q$  represents the normalization operation<sup>2</sup>, which is defined as  $\text{Norm}(\mathbf{r}) := \frac{\mathbf{r}}{\|\mathbf{r}\|}$ . Moreover, the hyper-parameter  $\xi > 0$  determines the contribution of the individual initial representation  $\mathbf{r}$ .

<sup>1</sup>It uses the fact that  $\mathbf{a} \star \mathbf{a} \approx \delta$ , where  $\delta$  is the identity operation of convolution.

<sup>2</sup>In other sections, we may obviate Norm operator in the equation for the sake of simplicity, since it can be shown that the circular correlation of two normalized high-dimensional random vectors are almost normalized.

Note, that the same entity  $e_i$  could also play the role of an object. For instance, the entity *California* could be the subject in the triple (*California, locatedIn, USA*), or the object in another triple (*Paul, livesIn, California*). Thus, it is necessary to have another representation to specify its role in the triples. Consider the set of subject-predicate pairs  $\mathcal{S}^o(e_i) = \{(s, p) | \phi_p(s, e_i) = 1\}$  for which triples  $(s, p, e_i)$  are true. These pairs are stored in a single trace via  $\mathbf{h}_{e_i}^o = \sum_{(s,p) \in \mathcal{S}^o(e_i)} [\text{Norm}(\mathbf{r}_p \star \mathbf{r}_s) + \xi \mathbf{r}_{e_i}]$ , where  $\mathbf{h}_{e_i}^o$  is the representation of the entity  $e_i$  when it acts as an object.

For the later generalization task, the overall holistic representation for the entity  $e_i$  is defined as the summation of both representations, namely

$$\mathbf{h}_{e_i} = \mathbf{h}_{e_i}^s + \mathbf{h}_{e_i}^o. \quad (11)$$

In this way, the complete neighbourhood information of an entity can be used for generalization.

Furthermore, given a predicate  $p_i$ , the holistic representation  $\mathbf{h}_{p_i}$  encodes all the subject-object pairs in the set  $\mathcal{S}(p_i) = \{(s, o) | \phi_{p_i}(s, o) = 1\}$  via

$$\mathbf{h}_{p_i} = \sum_{(s,o) \in \mathcal{S}(p_i)} [\text{Norm}(\mathbf{r}_s \star \mathbf{r}_o) + \xi \mathbf{r}_{p_i}]. \quad (12)$$

After storing all the association pairs into holistic features of entities and predicates, the initial randomly assigned representations are not required anymore and can be deleted. These representations are then fixed and not trainable unlike other embedding models.

After encoding, entity retrieval is performed via a circular convolution. Consider a concrete triple  $(e_1, p_1, e_2)$  with unknown  $e_2$ . The identity of  $e_2$  could be revealed with the holistic representation of  $p_1$  and the holistic representation of  $e_1$  as a subject, namely  $\mathbf{h}_{p_1}$  and  $\mathbf{h}_{e_1}^s$ . Then retrieval is performed as  $\mathbf{h}_{p_1} \star \mathbf{h}_{e_1}^s$ . The associations can be retrieved from the holography memory with low fidelity due to interference. Therefore, after decoding, a clean-up operation is employed, as in the HRR model. Specifically, a nearest neighbour is determined using cosine similarity. The pseudo-code for encoding holistic representations is provided in A.6.

### 4.3 EXPERIMENTS ON MEMORIZATION

We test the memorization of complex structure on different datasets and compare the performance of different models. Recall that  $\mathcal{R}_p$  is the set of all true triples with respect to a given predicate  $p$ . Consider a possible triple  $(s, p, o) \in \mathcal{R}_p$ . The task is now to retrieve the object entity from holistic vectors  $\mathbf{h}_s$  and  $\mathbf{h}_p$ , and to retrieve the subject entity from holistic vectors  $\mathbf{h}_p$  and  $\mathbf{h}_o$ .

As discussed, in retrieval, the noisy vector  $\mathbf{r}'_o = \mathbf{h}_p \star \mathbf{h}_s$  is compared to the holistic representations of all entities using cosine similarity, according to which the entities are then ranked. In general, multiple objects could be connected to a single subject-predicate pair. Thus, we employ the *filtered mean rank* introduced in [Bordes et al. 2013] to evaluate the memorization task.

We have discussed that the number of pairwise quasi-orthogonal vectors crucially depends on the random initialization. Now we analyse, if the memory capacity depends on the quasi-orthogonality of the initial representation vectors, as well. We perform memorization task on three different KGs, which are FB15k-237 [Toutanova et al. 2015], YAGO3 [Mahdisoltani et al. 2013], and a subset of GDEL T [Leetaru et al. 2013]. The exact statistics of the datasets are given in Table. 1.

Table 1: Statistics of KGs

	$\#\mathcal{D}$	$N_a$	$N_e$	$N_p$
<b>GDEL T</b>	497, 605	$\approx 73$	6786	231
<b>FB15k-237</b>	301, 080	$\approx 20$	14505	237
<b>YAGO3</b>	1, 089, 000	$\approx 9$	123143	37

Recall that  $N_e$  and  $N_p$  denote the number of entities and predicates, respectively. Moreover,  $\#\mathcal{D}$  denotes the total number of triples in a KG, and  $N_a$  is the average number of association pairs compressed into holistic feature vectors of entities, which can be estimated as  $\frac{\#\mathcal{D}}{N_e}$ . After encoding triples in a dataset into holistic features, filtered mean rank is evaluated by ranking retrieved subjects and objects of all triples. Filtered mean ranks on three datasets with holistic representations encoded from Gaussian and Cauchy distributions are displayed in Fig. 3 (a)-(c).

Cauchy holistic representations outperform Gaussian holistic representations significantly when the total number of entities is large (see, Fig. 3(c) for YAGO3), or the average number of encoded associations is large (see, Fig. 3(a) for GDEL T). This implies that quasi-orthogonality plays an important role in holographic memory. Improved quasi-orthogonality allows for more entities to be initialized with quasi-orthogonal representations, which is very important for memorizing huge KGs. In addition, it reduces the interference between associations. Moreover, Cauchy holistic features are intrinsically very sparse, making them an attractive candidate for modeling biologically plausible memory systems.

### 4.4 CORRELATION VERSUS CONVOLUTION

One of the main differences between *holistic representation* and the *holographic reduced representation* is the binding operation. In HRR, two vectors are composed

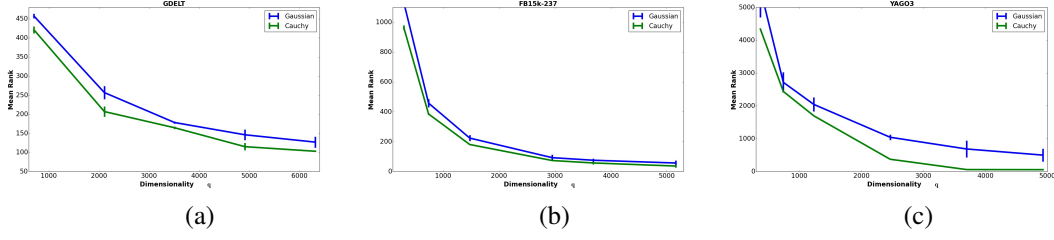


Figure 3: Filtered MR vs. the dimensionality of holistic representations evaluated on dataset: (a) GDEL, (b) FB15k-237, and (c) YAGO3. Blues lines denote holistic representations encoded from Gaussian random vectors, and green lines denote holistic representations encoded from Cauchy random vectors. Lower values are preferred.

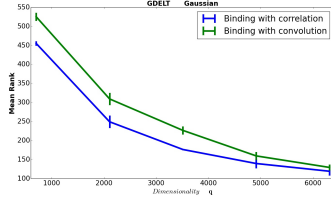


Figure 4: Filtered MR vs. the dimensionality of holistic representations evaluated on the GDEL dataset with Gaussian initialization.

via circular convolution, while in holistic representation, they are composed via circular correlation.

Binding with convolution and correlation is compared in Fig. 4. We report the filtered MR scores on the GDEL dataset versus the dimensionality of holistic representations. It can be seen that binding with circular correlation is significantly superior to convolution. Therefore, a non-commutative compositional operator is essential for storing the directed structures of KG into holographic memory. A theoretical explanation is given in the A.4, along with experimental results on other datasets.

#### 4.5 HYPER-PARAMETER $\xi$

In the experiments so far, the optimal hyper-parameter  $\xi$  is found via grid search. However, it is possible to roughly estimate the range of the optimal hyper-parameter  $\xi$ . Indeed,  $\xi$  strongly depends on  $\lambda_G$  or  $\lambda_C$  and the average number of encoded association pairs  $N_a$ .

So far, the deep relation between holographic memory capacity and quasi-orthogonality has not been discussed in the literature. In the original work on HRR, memory capacity and information retrieval quality are estimated from the distribution of elements in random vectors. In this section we give a plausible explanation from the point of view of the pairwise angle distribution.

Consider a subject  $s$ . The predicate-object pair  $(p, o)$

is stored in the holistic representation  $\mathbf{h}_s$  along with the other  $N_a - 1$  pairs, such that

$$\mathbf{h}_s = \xi N_a \mathbf{r}_s + \mathbf{r}_p \star \mathbf{r}_o + \sum_{i=2}^{N_a} \mathbf{r}_{p_i} \star \mathbf{r}_{o_i}.$$

Suppose we try to identify the object in the triple  $(s, p, \cdot)$  via  $\mathbf{h}_s$  and  $\mathbf{h}_p$ . After decoding, the noisy vector  $\mathbf{r}'_o = \mathbf{h}_p \star \mathbf{h}_s$  should be recalled with  $\mathbf{h}_o$ , which is the holistic representation of  $o$ . Let  $\theta_{\mathbf{r}'_o, \mathbf{h}_o}$  denote the angle between  $\mathbf{r}'_o$  and  $\mathbf{h}_o$ . The cosine function of this angle is again defined as  $\rho_{\mathbf{r}'_o, \mathbf{h}_o} := \cos \theta_{\mathbf{r}'_o, \mathbf{h}_o}$ .

In order to recall the object successfully, the angle between  $\mathbf{r}'_o$  and  $\mathbf{h}_o$  should be smaller than the expected absolute angle between two arbitrary vectors, namely

$$\theta_{\mathbf{r}'_o, \mathbf{h}_o} < \mathbb{E}[\|\theta_{G/C}\|], \quad (13)$$

This inequality first implies that the optimal  $\xi$  should be a positive number. Given the definition of  $\lambda_{G/C}$  in Eq. 5 and 9, equivalently, Eq. 13 requires

$$\rho_{\mathbf{r}'_o, \mathbf{h}_o} > \lambda_{G/C}. \quad (14)$$

After some manipulations, a sufficient condition to recognize the object correctly is given by (see A.5)

$$\begin{aligned} \rho_{\mathbf{r}'_o, \mathbf{h}_o} > & \frac{\xi^2 N_a^2 - (\xi^3 N_a^3 + 2\xi^2 N_a^3 - \xi^2 N_a^2 + \xi N_a^2 + \xi N_a^3) \lambda_{G/C}}{\xi^2 N_a^2 + N_a + 2\xi N_a^2 \lambda_{G/C} + N_a(N_a - 1) \lambda_{G/C}} \\ & > \lambda_{G/C}. \end{aligned} \quad (15)$$

In the following, we verify this condition on the FB15k-237 dataset. We consider one of the experimental settings employed in the memorization task. The dimension of holistic features is  $q = 5200$ , with  $\lambda_G = 0.0111$  computed from Eq. 5, and  $\lambda_C = 0.00204$  from Eq. 9. For Gaussian initialization, the optimum is found at  $\xi = 0.14$  via grid search, while for Cauchy initialization, the optimum is found at  $\xi = 0.05$ .

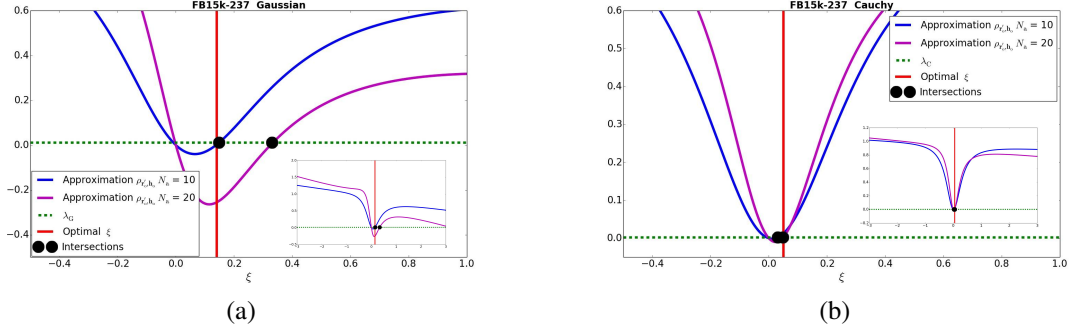


Figure 5: Analysis of the hyper-parameter  $\xi$  on the FB15k-237 dataset. (a): Approximation of  $\rho_{r'_o, h_o}$  for Gaussian initialization. Curves with  $N_a = 10$  (blue),  $N_a = 20$  (magenta) and their intersections with the retrieval threshold  $\lambda_G$  are displayed. The red vertical line denotes the experimentally determined optimal  $\xi$ . Insert shows the curves with  $\xi \in [-3, 3]$ . (b): Approximation of  $\rho_{r'_o, h_o}$  for Cauchy initialization with  $N_a = 10$  (blue), and  $N_a = 20$  (magenta). Rest remains the same.

To verify these optima, Fig. 5 (a) and (b) display the approximation of  $\rho_{r'_o, h_o}(\xi, N_a)$  as a function of  $\xi$ .<sup>3</sup> Its intersection with  $\lambda_{G/C}$  is marked with a black dot. In FB15k-237,  $N_a$  is estimated to be 20, while, in general, a KG could be quite imbalanced. Thus,  $\rho_{r'_o, h_o}(\xi, N_a)$  with  $N_a = 10$ , and 20 are shown together for comparison.

In Fig. 5 (a) for Gaussian initialization, experimentally determined optimal  $\xi$  (red vertical line) is found close to the intersection of  $\rho_{r'_o, h_o}(\xi, N_a = 10)$  and threshold  $\lambda_G$ , meaning that Gaussian holistic features tend to memorize fewer association pairs. They can only map sparsely connected graph structures into meaningful representations.

In Fig. 5 (b) for Cauchy initialization, however, the optimal  $\xi$  is close to the intersection of  $\rho_{r'_o, h_o}(\xi, N_a = 20)$  and  $\lambda_C$ . Thus, Cauchy holistic features are more suitable to memorize a larger chunk of associations, meaning that they are capable of mapping densely connected graph structures into meaningful representations. All optima are found near the intersection points instead of the local maximum with  $\xi > 0$ . It indicates that, to maximize the memory capacity, the holistic features can only store information with very low fidelity.

Table 2: Filtered recall scores on FB15k-237

Methods	MR	MRR	Hits		
			@10	@3	@1
RESCAL	996	0.221	0.363	0.237	0.156
DISTMULT	254	0.241	0.419	0.263	0.155
COMPLEX	339	0.247	0.428	0.275	0.158
R-GCN <sup>4</sup>	-	0.248	0.414	0.258	0.153
HOLNN <sub>G</sub> <sup>5</sup>	235	0.285	0.455	0.315	0.207
HOLNN <sub>C</sub>	<b>228</b>	<b>0.295</b>	<b>0.465</b>	<b>0.320</b>	<b>0.212</b>

<sup>3</sup>The approximation of  $\rho_{r'_o, h_o}$  is the second term of Eq. 15

## 5 INFERENCE ON KG

### 5.1 INFERENCE VIA HOLISTIC REPRESENTATION

In this section, we describe the model for inferring the missing links in the KG. Recall the scoring function  $\eta_{spo}$  defined in Sec. 2. Our model uses holistic representations as input and generalizes them to implicit facts, by a two-layer neural network<sup>6</sup>. Formally, the scoring function is given as follow:

$$\eta_{spo} = \langle \text{ReLU}(\mathbf{h}_s \mathbf{W}_1^e) \mathbf{W}_2^e, \text{ReLU}(\mathbf{h}_p \mathbf{W}_1^p) \mathbf{W}_2^p, \text{ReLU}(\mathbf{h}_o \mathbf{W}_1^e) \mathbf{W}_2^e \rangle, \quad (16)$$

where  $\langle \cdot, \cdot, \cdot \rangle$  denotes tri-linear dot product;  $\mathbf{h}_s, \mathbf{h}_o$  are the holistic representations for entities defined in Eq. 11,  $\mathbf{h}_p$  is defined in Eq. 12.

Suppose that the holistic representations are defined in  $\mathbb{R}^q$ . Then  $\mathbf{W}_1^e \in \mathbb{R}^{q \times h_1}$  and  $\mathbf{W}_2^e \in \mathbb{R}^{h_1 \times h_2}$  are shared weights for entities;  $\mathbf{W}_1^p \in \mathbb{R}^{q \times h_1}$  and  $\mathbf{W}_2^p \in \mathbb{R}^{h_1 \times h_2}$  are shared weights for predicates. We refer Eq. 16 as HOLNN, a combination of holistic representations and a simple neural network.

As an example, for training on FB15k-237, we take  $q = 3600$ ,  $h_1 = 64$ , and  $h_2 = 256$ . Note that only weight matrices in the neural network are trainable parameters, holistic representations are fixed after encoding. Thus, the total number of trainable parameters in HOLNN is  $0.48M$ , which is much smaller than COM-

<sup>4</sup>see [Schlichtkrull et al. 2018]

<sup>5</sup>G stands for Gaussian holistic features, and C for Cauchy holistic features.

<sup>6</sup>Further experimental details are referred to A.8

PLEX with 5.9M parameters, by assuming that the dimension of embeddings in the COMPLEX is 200.

To evaluate the performance of HOLNN for missing links prediction, we compare it to the state-of-the-art models on two datasets: FB15k-237, and GDELTA. They were split randomly in training, validation, and test sets. We implement all models with the identical loss function Eq. 1, and minimize the loss on the training set using Adam as the optimization method. Hyper-parameters, e.g., the learning rate, and  $l2$  regularization, are optimized based on the validation set.

We use filtered MR, filtered mean reciprocal rank (MRR), and filtered Hits at  $n$  (Hits@ $n$ ) as evaluation metrics [Bordes et al. 2013]. Table 2 and Table 3 report different metrics on the FB15k-237, and GDELTA dataset, respectively. It can be seen that HOLNN is superior to all the baseline methods on both datasets with considerably less trainable parameters. Moreover, HOLNN<sub>C</sub> consistently outperforms HOLNN<sub>G</sub>, indicating that the memory capacity of holistic representations is important for generalization.

Table 3: Filtered recall scores on GDELTA

Methods	MR	MRR	Hits		
			@10	@3	@1
RESCAL	212	0.202	0.396	0.225	0.107
DISTMULT	181	0.232	0.451	0.268	0.124
COMPLEX	158	0.256	0.460	0.295	0.146
HOLNN <sub>G</sub>	105	0.284	0.457	0.301	0.198
HOLNN <sub>C</sub>	<b>102</b>	<b>0.296</b>	<b>0.471</b>	<b>0.315</b>	<b>0.210</b>

## 5.2 INFERENCE ON NEW ENTITIES

In additional experiments, we show that HOLNN is capable of inferring implicit facts on new entities without re-training the neural network. Experiments are performed on FB15k-237 as follows. We split the entire FB15k-237 dataset  $\mathcal{D}$  into  $\mathcal{D}_{old}$  and  $\mathcal{D}_{new}$ . In  $\mathcal{D}_{new}$ , the subjects of triples are new entities which do not show up in  $\mathcal{D}_{old}$ , while objects and predicates are already seen in the  $\mathcal{D}_{old}$ . Suppose our task is to predict implicit links between new entities (subjects in  $\mathcal{D}_{new}$ ) and old entities (entities in  $\mathcal{D}_{old}$ ). Thus, we further split  $\mathcal{D}_{new}$  into  $\mathcal{D}_{new}^{train}$ ,  $\mathcal{D}_{new}^{valid}$ , and  $\mathcal{D}_{new}^{test}$  sets.

For embedding models, e.g., COMPLEX, after training on  $\mathcal{D}_{old}$ , the most efficient way to solve this task is to adapt the embeddings of new entities on  $\mathcal{D}_{new}^{train}$ , with fixed embeddings of old entities. On the other hand, for the HOLNN model, new entities obtain their holistic representations via triples in the  $\mathcal{D}_{new}^{train}$  set. These holistic features are then fed into the trained two-layer neural network. Table 4 shows filtered recall scores for predict-

ing links between new entities and old entities on  $\mathcal{D}_{new}^{test}$ , with the number of new entities in  $\mathcal{D}_{new}$  being 300, 600, or 900. COMPLEX and HOLNN with Cauchy holistic features are compared.

There are two settings for the HOLNN<sub>C</sub> model. New entities could be encoded either from holistic features of old entities, or from random initializations of old entities<sup>7</sup>. We denote these two cases as HOLNN<sub>C</sub>(**h**) and HOLNN<sub>C</sub>(**r**), respectively. It can be seen that HOLNN<sub>C</sub>(**r**) outperforms HOLNN<sub>C</sub>(**h**) only to some degree. It indicates that HOLNN<sub>C</sub> is robust to the noise, making it generalizes well.

Table 4: Inference of new entities on FB15k-237

Methods	Number of New Entities					
	300		600		900	
	MR	MRR	MR	MRR	MR	MRR
COMPLEX	262	0.291	<b>265</b>	0.266	<b>286</b>	0.243
HOLNN <sub>C</sub> ( <b>h</b> )	345	0.274	415	0.242	510	0.222
HOLNN <sub>C</sub> ( <b>r</b> )	<b>252</b>	<b>0.315</b>	302	<b>0.281</b>	395	<b>0.265</b>

## 6 CONCLUSION

We have introduced the holistic representation for the distributed storage of complex association patterns and have applied it to knowledge graphs. We have shown that interference between stored information is reduced with initial random vectors which are pairwise quasi-orthogonal and that pairwise quasi-orthogonality can be improved by drawing vectors from heavy-tailed distributions, e.g., a Cauchy distribution. The experiments demonstrated excellent performance on memory retrieval and competitive results on link prediction.

In our approach, latent representations are derived from random vectors and are not learned from data, as in most modern approaches to representation learning on knowledge graphs. One might consider representations derived from random vectors to be biologically more plausible, if compared to representations which are learned via complex gradient based update rules. Thus in addition to its very competitive technical performance, one of the interesting aspects of our approach is its biological plausibility.

*Outlook:* Potential applications could be applying the holistic encoding algorithm to Lexical Functional for modeling distributional semantics [Coecke et al. 2010], or graph convolutional network [Kipf et al. 2017] for semi-supervised learning using holistic representations as feature vectors of nodes on a graph.

<sup>7</sup>Recall that random initializations are actually deleted after encoding. Here we use them just for comparison.

## References

- Bollacker, Kurt, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor (2008). “Freebase: a collaboratively created graph database for structuring human knowledge”. *Proceedings of the 208 ACM SIGMOD*. AcM, pp. 1247–1250.
- Bordes, Antoine, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko (2013). “Translating embeddings for modeling multi-relational data”. *NIPS*, pp. 2787–2795.
- Cai, Tony and Tiejun Jiang (2012). “Phase transition in limiting distributions of coherence of high-dimensional random matrices”. *Journal of Multivariate Analysis* 107, pp. 24–39.
- Cai, Tony, Jianqing Fan, and Tiejun Jiang (2013). “Distributions of angles in random packing on spheres”. *The Journal of Machine Learning Research* 14.1, pp. 1837–1864.
- Coecke, Bob, Mehrnoosh Sadrzadeh, and Stephen Clark (2010). “Mathematical foundations for a compositional distributional model of meaning”. *Linguistic Analysis* 36.
- Diaconis, Persi and David Freedman (1984). “Asymptotics of graphical projection pursuit”. *The annals of statistics*, pp. 793–815.
- Gabor, D. (1969). “Associative holographic memories”. *IBM Journal of Research and Development* 13.2, pp. 156–159.
- Gentner, Dedre (1983). “Structure-mapping: A theoretical framework for analogy”. *Cognitive science* 7.2, pp. 155–170.
- Gnedenko, B.V. and A.N. Kolmogorov (1954). *Limit distributions for sums of independent random variables*. Addison-Wesley.
- Hall, Peter, James Stephen Marron, and Amnon Neeman (2005). “Geometric representation of high dimension, low sample size data”. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67.3, pp. 427–444.
- Hopfield, John J. (1982). “Neural networks and physical systems with emergent collective computational abilities”. *Proceedings of the national academy of sciences* 79.8, pp. 2554–2558.
- Kipf, Thomas N and Max Welling (2017). “Semi-supervised classification with graph convolutional networks”. *ICLR*.
- Leetaru, Kalev and Philip A. Schrodt (2013). “GDELT: Global data on events, location, and tone”. *ISA Annual Convention*.
- Ma, Yunpu, Volker Tresp, and Erik Daxberger (2018). “Embedding models for episodic memory”. *arXiv preprint arXiv:1807.00228*.
- Mahdisoltani, Farzaneh, Joanna Biega, and Fabian M. Suchanek (2013). “Yago3: A knowledge base from multilingual wikipedias”. *CIDR*.
- McClelland, James L., Bruce L. McNaughton, and Randall C. O’reilly (1995). “Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory.” *Psychological review* 102.3, p. 419.
- Muirhead, Robb J. (2009). *Aspects of multivariate statistical theory*. Vol. 197. John Wiley & Sons.
- Nickel, Maximilian, Volker Tresp, and Hans-Peter Kriegel (2011). “A Three-Way Model for Collective Learning on Multi-Relational Data”. *ICML*. Vol. 11, pp. 809–816.
- Nickel, Maximilian, Volker Tresp, and Hans-Peter Kriegel (2012). “Factorizing yago: scalable machine learning for linked data”. *Proceedings of the 21st international conference on World Wide Web*. ACM, pp. 271–280.
- Nickel, Maximilian, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich (2016a). “A review of relational machine learning for knowledge graphs”. *Proceedings of the IEEE* 104.1, pp. 11–33.
- Nickel, Maximilian, Lorenzo Rosasco, and Tomaso Poggio (2016b). “Holographic Embeddings of Knowledge Graphs”. *AAAI*, pp. 1955–1961.
- Penrose, Mathew (2003). *Random geometric graphs*. 5. Oxford university press.
- Plate, Tony A. (1995). “Holographic reduced representations”. *IEEE Transactions on Neural networks* 6.3, pp. 623–641.
- Schlichtkrull, Michael, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling (2018). “Modeling relational data with graph convolutional networks”. *European Semantic Web Conference*. Springer, pp. 593–607.
- Suchanek, Fabian M., Gjergji Kasneci, and Gerhard Weikum (2007). “Yago: a core of semantic knowledge”. *Proceedings of the 16th international conference on World Wide Web*. ACM, pp. 697–706.
- Toutanova, Kristina and Danqi Chen (2015). “Observed versus latent features for knowledge base and text inference”. *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pp. 57–66.
- Tresp, Volker, Yunpu Ma, Stephan Baier, and Yinchong Yang (2017a). “Embedding learning for declarative memories”. *ESWC*. Springer, pp. 202–216.
- Tresp, Volker and Yunpu Ma (2017b). “The Tensor Memory Hypothesis”. *arXiv preprint arXiv:1708.02918*.
- Trouillon, Théo, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard (2016). “Com-



- plex embeddings for simple link prediction”. *ICML*, pp. 2071–2080.
- Westlake, Philip R. (1970). “The possibilities of neural holographic processes within the brain”. *Kybernetik* 7.4, pp. 129–153.
- Yang, Bishan, Wentau Yih, Xiaodong He, Jianfeng Gao, and Li Deng (2014). “Embedding entities and relations for learning and inference in knowledge bases”. *ICLR 2015*.

---

# Simple and practical algorithms for $\ell_p$ -norm low-rank approximation

---

Anastasios Kyrillidis

IBM T.J. Watson Research Center

Rice University

anastasios@rice.edu

## Abstract

We propose practical algorithms for entrywise  $\ell_p$ -norm low-rank approximation, for  $p = 1$  or  $p = \infty$ . The proposed framework, which is non-convex and gradient-based, is easy to implement and typically attains better approximations, faster, than state of the art.

From a theoretical standpoint, we show that the proposed scheme can attain  $(1 + \varepsilon)$ -OPT approximations. Our algorithms are not hyperparameter-free: they achieve the desiderata only assuming algorithm’s hyperparameters are known *a priori*—or are at least approximable. *I.e.*, our theory indicates what problem quantities need to be known, in order to get a good solution within polynomial time, and does not contradict to recent inapproximability results, as in [46].

## 1 INTRODUCTION

We focus on the following optimization problem:

$$\min_{U \in \mathbb{R}^{m \times r}, V \in \mathbb{R}^{n \times r}} \|M - UV^\top\|_p, \quad p \in \{1, \infty\}. \quad (1)$$

Here,  $M \in \mathbb{R}^{m \times n}$  is a given input matrix of arbitrary rank,  $r \leq \{m, n\}$  is the target rank,  $(U, V)$  represent the variables such that  $\text{rank}(UV^\top) \leq r$ , and  $\|\cdot\|_p$  denotes the  $p$ -th, *entrywise*, matrix norm. In words, (1) is described as “finding the factors of the best rank- $r$  approximation of  $M$ , with respect to the  $\ell_p$ -norm”. We denote such optimal factors  $U^*$  and  $V^*$ , and their product  $X^* = U^*V^{*\top}$ . We focus on  $p \in \{1, \infty\}$ , since these instances are the most common found in practice, beyond the classic  $p = 2$  (Frobenius) norm; we will use the terms “Frobenius” and “ $\ell_2$ ” norm, interchangeably.

There are numerous applications where  $\ell_1$ - /  $\ell_\infty$ -norm low rank approximations are useful in practice. First, the  $\ell_1$ -norm is more robust than the  $\ell_2$ -norm, and is suited in problem settings where Gaussian assumptions for noise models may not apply.  $\ell_1$ -norm low rank applications include robust PCA applications [56, 6, 31, 32, 24, 57], computer vision tasks such as background subtraction and motion detection [52, 1, 38], detection of brain activation patterns [44], and detection of anomalous behavior in dynamic networks [44].<sup>1</sup>

For the  $\ell_\infty$ -norm version of (1), the problem cases are only a few. [43] considers the special case of  $m = n$  and  $r = \min\{m, n\} - 1$  as the problem of distance to robust non-singularity. [22, 23] use the notion of  $\ell_\infty$ -norm low rank approximation for the maximal-volume concept in approximation, as well as for the skeleton approximation of a matrix. Finally, [17] identifies that (1) with  $p = \infty$  can be used for the recovery of a low-rank matrix from a quantized  $M$ .

Despite the utility of (1), its solution is not straightforward. While (1) with  $\ell_2$ -norm has a closed-form solution via the Singular Value Decomposition (SVD), the same does not hold for  $p \in \{1, \infty\}$ . Additionally, it has been proved that actually finding the exact solution to (1) can be exponentially complex:

---

<sup>1</sup>Closely related to the  $\ell_1$ -norm low-rank approximation is the problem of  $\ell_1$ -norm subspace recovery [30]. Briefly, it is well-known that, for  $p = 2$  in (1), the SVD solution is also the solution to the dual problem:  $U^* = \text{argmax}_{U \in \mathbb{R}^{m \times r}} |U^\top M|_2$ , subject to  $U^\top U = I$ .  $V^*$  is then set as  $V^* = U^{*\top} M$ ; this can be easily proved due to the orthogonality of  $U^*$  [20]. Motivated by this dual formulation,  $\ell_1$ -norm subspace recovery is defined as

$$U^* = \text{argmax}_{U \in \mathbb{R}^{m \times r}} |U^\top M|_1, \quad \text{subject to } U^\top U = I.$$

Algorithmic solutions to this criterion are usually greedy [30], even combinatorial [36, 37]. However, in this case,  $U^*$  does not necessarily resemble with that of (1) with  $p = 1$  (up to orthogonal rotations).

[19] show that  $\ell_1$ -norm low rank matrix approximation is NP-hard, even for  $r = 1$ ; further, under the exponential time hypothesis for 3SAT problems, [46] provide a  $\left(1 + \frac{1}{\log^{1+\gamma}(\max\{m,n\})}\right)$ -inapproximability result for some hard instances  $M$ , where  $\gamma > 0$  is an arbitrary small constant. [17] proves the NP-completeness of (1) for  $p = \infty$ , using a reduction from not-all-equal-3SAT.

The above restrict research to only approximations of (1). To the best of our knowledge only the works in [9, 46] present polynomial and provably good approximation schemes: [46] focuses mostly on the case of  $\ell_1$ -norm, and proves the existence of a  $O(\log(\min\{m,n\}) \cdot \text{poly}(r))$ -approximation scheme with  $O(\text{nnz}(M) + (m+n)\text{poly}(r))$  computational complexity. [9] extends the ideas in [46] for  $\ell_p$ -norms, where  $p \in [1, \infty]$ : there, the authors describe a  $\text{poly}(r)$ -approximation with  $O(\text{poly}(m,n)(r \log \max\{m,n\})^r)$  computational complexity. Both approaches are based on numerical linear algebra and sketching techniques.

Apart from the above provable schemes, there are numerous heuristics proposed for (1), with no rigorous approximation guarantees. Starting with  $\ell_1$ -norm, [38] propose a coordinate descent algorithm for (1), where a sequence of alternating scalar minimization sub-problems are solved using a (weighted) median filter; see also [29]. Previously to that work, [26, 27] follow a similar approach, where each sub-problem is solved using linear or quadratic programming<sup>2</sup>. Inspired by [55], [13] propose a  $\ell_1$ -norm version of the Wiberg method; the resulting algorithm involves several matrix-matrix multiplications (even of size greater than the input matrix), and the solution of linear programming criteria, per iteration. Cabral et al. use Augmented Lagrange Multipliers (ALM) method and handle the weighted  $\ell_1$ -norm low rank approximation problem in [5]; however, no non-asymptotic convergence guarantees are provided. We note that most of the above heuristics are designed to handle *missing data* in  $M$  or the case of *weighted* factorization; we plan to consider such cases for our future research directions. For the  $\ell_\infty$ -norm case, we mention the recent work of Gillis et al. [17] that proposes a block coordinate descent method that operates in an alternating minimization fashion over subsets of variables in (1).

**Our approach and main contributions:** Inspired by the recent advances on smooth non-convex optimization for matrix factorization [47, 58, 51, 4, 42, 16, 40, 41, 35, 34,

<sup>2</sup>In [26, 27], there are some convergence guarantees for the alternating optimization scheme; however, there are no results w.r.t. whether we converge to a saddle point or local minimum, nor results on the convergence rate.

50, 54, 15, 33], we study the application of alternating gradient descent in (1). Despite its NP-hardness, this paper follows a more optimistic course and works towards deciphering the components/quantities that, if known a priori, could lead to a  $(1 + \varepsilon)$ -approximation for (1).

Our approach is based on two techniques from optimization theory: (i) the smoothing technique for non-smooth convex optimization by Nesterov [39, 12] (Section 4), and (ii) the recent theoretical results on finding the global minimum of matrix factorization problems using non-convex smooth methods (Section 3); see also references above. Our theory relies on provably bounding the objective function in  $\ell_1$ - or  $\ell_\infty$ -norm by its smoothing counterpart (Sections 4), using the provable performance of the non-convex algorithm (Section 3), and properly setting up the input parameters (Section 5). Our guarantees assume that we can at least approximate the optimal function value of (1), and that the optimal low-rank solution of the smoothed problem is well-conditioned; the latter assumption is required for a good initialization to be easily found. The above are summarized as:

- Under assumptions, we provide a polynomial approximation algorithm for  $p = \{1, \infty\}$  in (1) that achieves a  $(1 + \varepsilon)$ -approximation guarantee.
- We experimentally show that our scheme outperforms in practice state-of-the-art approaches.

There are several questions that remain open and need further investigation. In Section 7, we discuss what are the advantages and disadvantages of our approach and point to possible future research directions.

## 2 NOTATION AND ASSUMPTIONS

*Notation.* For matrices  $X, Y \in \mathbb{R}^{m \times n}$ ,  $\langle X, Y \rangle = \text{Tr}(X^\top Y)$  represents their inner product and  $X \odot Y$  their Hadamard product. We represent matrix norms as follows:  $|X|_2 = \sqrt{\sum_{i=1}^m \sum_{j=1}^n |X_{ij}|^2}$  denotes the Frobenius (or  $\ell_2$ -) norm,  $|X|_1 = \sum_{i=1}^m \sum_{j=1}^n |X_{ij}|$  denotes the entrywise  $\ell_1$ -norm, and  $|X|_\infty = \max_{i,j} |X_{ij}|$  denotes the entrywise  $\ell_\infty$ -norm. For the spectral norm, we use  $\sigma_1(X)$ ; this also denotes the largest singular value of  $X$ . For vectors, we use  $\|x\|_2$  to denote its Euclidean  $\ell_2$ -norm. For a differentiable function  $f(X)$  with  $X = UV^\top$ , the gradient of  $f$  w.r.t.  $U$  and  $V$  is  $\nabla f(X)V$  and  $\nabla f(X)^\top U$ , respectively.

*Assumptions.* For our discussion, we will need two well-known notions of convex analysis: (restricted) *strong* convexity and (restricted) Lipschitz gradient continuity.

**Definition 2.1.** *Let  $f : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}$  be a convex differentiable function. Then,  $f$  is (resp. restricted) gradient Lip-*

schitz continuous with parameter  $L$  if  $\forall X, Y \in \mathbb{R}^{m \times n}$  (resp.  $\forall X, Y \in \mathbb{R}^{m \times n}$  that are at most rank- $r$ ):

$$f(Y) \leq f(X) + \langle \nabla f(X), Y - X \rangle + \frac{L}{2} \|Y - X\|_2^2. \quad (2)$$

**Definition 2.2.** Let  $f : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}$  be convex and differentiable. Then,  $f$  is (resp. restricted)  $\mu$ -strongly convex if  $\forall X, Y \in \mathbb{R}^{m \times n}$  (resp.  $\forall X, Y \in \mathbb{R}^{m \times n}$  that are at most rank- $r$ ):

$$f(Y) \geq f(X) + \langle \nabla f(X), Y - X \rangle + \frac{\mu}{2} \|Y - X\|_2^2. \quad (3)$$

### 3 BFGD FOR SMOOTH OBJECTIVES

Let us first succinctly describe the Bi-Factored Gradient Descent (BFGD) algorithm [41], upon which our proposal is based. BFGD is a non-convex gradient descent scheme for *smooth* problems such as:

$$\min_{U \in \mathbb{R}^{m \times r}, V \in \mathbb{R}^{n \times r}} f(UV^\top), \quad (4)$$

where  $f$  is assumed to be convex, *differentiable*, and at least have Lipschitz continuous gradients. Observe that while  $f$  is convex w.r.t. to any input  $\in \mathbb{R}^{m \times n}$ , motions over  $U$  and  $V$  jointly lead to non-convex optimization. Such approaches have a long history and different variants have been proposed for (4).

For the rest of this section, we denote  $X = UV^\top$  as the result of the factorization. Also, let  $\hat{X}^*$  be the optimal point of (4): if  $\text{rank}(X^*) = r$ , then  $\hat{X}^* = X_r^*$ ; otherwise, denote its best rank- $r$  approximation (w.r.t. the  $\ell_2$ -norm) as  $\hat{X}_r^*$ .

---

#### Algorithm 1 Bi-factored gradient descent (BFGD)

---

- 1: **Input:**  $r, T, \gamma$  (e.g.,  $\frac{1}{4}$ ),  $C > 0$  (e.g.,  $C = 1$ ),  $\hat{L}$ .
- 2: Compute  $X_0 := \frac{1}{\hat{L}} \cdot (-\nabla f(0_{m \times n}))$ .
- 3: Set  $U_0 \in \mathbb{R}^{m \times r}, V_0 \in \mathbb{R}^{n \times r}$  s.t.  $X_0 = U_0 V_0^\top$ , via SVD.
- 4: **for**  $i = 0$  to  $T - 1$  **do**
- 5:   Set  $\eta$  such that:  $\eta \leq \frac{C}{15\hat{L} \left( \| [U_i \ V_i]^\top \|^2 + 3 \|\nabla f(U_i V_i^\top)\|_2 \right)}$ .
- 6:   • If  $f$  satisfies Definition 2.1: **Rule 1**

$$\begin{bmatrix} U_{i+1} \\ V_{i+1} \end{bmatrix} = \begin{bmatrix} U_i \\ V_i \end{bmatrix} - \eta \begin{bmatrix} \nabla f(U_i V_i^\top) \cdot V_i \\ \nabla f(U_i V_i^\top)^\top \cdot U_i \end{bmatrix}$$

- If  $f$  satisfies Definitions 2.1-2.2: **Rule 2**

$$\begin{bmatrix} U_{i+1} \\ V_{i+1} \end{bmatrix} = \begin{bmatrix} U_i \\ V_i \end{bmatrix} - \eta \begin{bmatrix} \nabla f(U_i V_i^\top) V_i + \gamma U_i (U_i^\top U_i - V_i^\top V_i) \\ \nabla f(U_i V_i^\top)^\top U_i - \gamma V_i (U_i^\top U_i - V_i^\top V_i) \end{bmatrix}$$

7: **end for**

8: **Output:**  $\hat{X} = U_T V_T^\top$ .

---

The pseudocode for BFGD is provided in Algorithm 1 and obeys the following motions: (i) given a proper ini-

tialization  $X_0 = U_0 V_0^\top$ , and (ii) a proper step size  $\eta$ ,<sup>3</sup> BFGD applies iteratively **Rule 1** if  $f$  satisfies only Definition 2.1, or **Rule 2** if  $f$  also satisfies Definition 2.2. The algorithm assumes an approximation of  $L$ —say  $\hat{L}$  and see [4]—and a good initialization point  $(U_0, V_0)$ . For a more complete discussion of initialization  $(U_0, V_0)$ , we refer the reader to [4, 41]; we briefly discuss this issue in Section 5.

An important issue in optimizing  $f$  over  $(U, V)$  is the existence of non-unique possible factorizations for a given  $X$ . We need a notion of distance to the low-rank solution  $X_r^*$  over the factors. Similar to [51, 41], we focus on the set of “equally-footed” factorizations:

$$\hat{\mathcal{X}}_r^* = \left\{ (\hat{U}^*, \hat{V}^*) : \hat{U}^* \in \mathbb{R}^{m \times r}, \hat{V}^* \in \mathbb{R}^{n \times r}, \hat{U}^* \hat{V}^{*\top} = \hat{X}_r^*, \sigma_i(\hat{U}^*) = \sigma_i(\hat{V}^*) = \sigma_i(\hat{X}_r^*)^{1/2}, \forall i \in [r] \right\}. \quad (5)$$

Given a pair  $(U, V)$ , we define the distance to  $\hat{\mathcal{X}}_r^*$  as:

$$\text{DIST}(U, V; \hat{\mathcal{X}}_r^*) = \min_{(\hat{U}^*, \hat{V}^*) \in \hat{\mathcal{X}}_r^*} \left\| \begin{bmatrix} U \\ V \end{bmatrix} - \begin{bmatrix} \hat{U}^* \\ \hat{V}^* \end{bmatrix} \right\|_2.$$

Algorithm 1 has local convergence guarantees, when  $f$  is  $\mu$ -strongly convex and has  $L$ -Lipschitz continuous gradients, according to the following theorem:<sup>4</sup>

**Theorem 3.1** (Theorem 4.4 in [41]). Let  $\kappa = L/\mu$ . If the initial point  $X_0 = U_0 V_0^\top$ , satisfies  $\text{DIST}(U_0, V_0; X_r^*) \leq \frac{\sqrt{2} \cdot \sigma_r(X_r^*)^{1/2}}{10}$ , then BFGD converges with rate  $O(1/T)$ :

$$f(U_T V_T^\top) - f(\hat{U}^* \hat{V}^{*\top}) \leq \frac{10 \cdot \text{DIST}(U_0, V_0; \hat{\mathcal{X}}_r^*)^2}{\eta T}$$

### 4 CHARBONNIER APPROXIMATION AND THE $\text{logsumexp}$ FUNCTION

A key assumption in BFGD is that  $f$  is at least once *differentiable* and has Lipschitz continuous gradients. Therefore, to connect BFGD with our original objective in (1), we will first approximate both the  $\ell_1$  and  $\ell_\infty$  entrywise matrix norms by smooth functions that have derivatives at least in two degrees. For similar approaches in optimization where non-smooth functions are substituted by smooth ones, we refer to the seminal paper of Nesterov [39] and follow-up works [12, 28].

<sup>3</sup>In this work, we do not focus on the most efficient step size selections: e.g., the step size considered in this work varies per iteration, and it is less efficient than a constant step size selection as in [4, 41]. However, in all cases, we could bound the varying step size with one that is constant.

<sup>4</sup>In this work, we will borrow only the sublinear rate results in [41], since that result alone is sufficient to lead to polynomial algorithms for (1). Using the linear convergence rate result in [41] is left for the extension of this work.

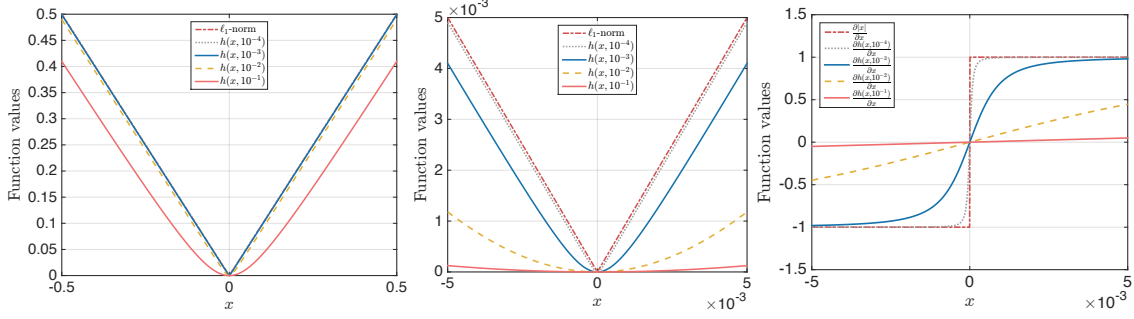


Figure 1:  $\ell_1$ -norm and its Charbonnier smooth approximations. *Left and middle*: Function values vs. input variable. *Right*: Gradient approximation.

**Approximating the entrywise  $\ell_1$ -norm.** For the approximation of the  $\ell_1$ -norm, we will use the Charbonnier loss function [7, 3], parameterized as follows:

$$h(x, \tau) = \tau \cdot \left( \sqrt{\left(\frac{x}{\tau}\right)^2 + 1} - 1 \right). \quad (6)$$

To illustrate how a good approximation is (6) to the  $\ell_1$ -norm, see Figure 1.

We now discuss about the matrix form of (6) and its properties. With a slight overload of notation, we define the matrix version of (6) as follows:

$$\begin{aligned} h(X, \tau) &= \sum_{i=1}^m \sum_{j=1}^n h(X_{ij}, \tau) \\ &:= \tau \cdot \sum_{i=1}^m \sum_{j=1}^n \left( \sqrt{\left(\frac{X_{ij}}{\tau}\right)^2 + 1} - 1 \right). \end{aligned} \quad (7)$$

The distinction between scalar and matrix  $h$  will be apparent from the text. Gradient and Hessian information of  $h$  satisfy the following lemma; the proof is deferred to the supp. material:

**Lemma 4.1.** For any  $X \in \mathbb{R}^{m \times n}$ :

- $\nabla h(X, \tau) = \frac{1}{\tau} X \odot S \in \mathbb{R}^{m \times n}$ , where  $S \in \mathbb{R}^{m \times n}$  and  $S_{ij} := \frac{2}{\sqrt{(X_{ij}/\tau)^2 + 1}}$ ,
- $\nabla^2 h(X, \tau) = \frac{1}{\tau} I \odot Q \in \mathbb{R}^{mn \times mn}$ , where  $Q \in \mathbb{R}^{mn \times mn}$  and  $Q_{ij} := \frac{2}{((X_{ij}/\tau)^2 + 1)^{3/2}}$ .

The above lead to the following lemma; the proof is provided in the supp. material:

**Lemma 4.2.** Function  $h$  is a convex continuously differentiable function and it has Lipschitz continuous gradients with constant  $\frac{2}{\tau}$ . Moreover:

$$|X|_1 - mn\tau \leq h(X, \tau) \leq |X|_1.$$

An alternative to the Charbonnier approximation is the Huber loss function with parameter  $\tau$  [25]:

$$h(x, \tau) = \begin{cases} x^2/2\tau, & \text{if } |x| \leq \tau \\ |x| - \tau/2, & \text{otherwise.} \end{cases} \quad (8)$$

Huber loss combines a  $\ell_2$ -norm measure for small values of  $x$  and a  $\ell_1$ -norm like measure for large  $x$ . Observe in (8) that it is only first-order differentiable; thus any computations involving second order derivatives cannot be applied. On the other hand, the Charbonnier loss function, which is also known as the ‘‘pseudo-Huber loss function’’, is a smooth approximation of the Huber loss that ensures that derivatives are continuous for all degrees. W.l.o.g., we focus on the Charbonnier function.

**Approximating the entrywise  $\ell_\infty$ -norm.** Following similar procedure for the entrywise matrix  $\ell_\infty$ -norm, we will use the `logsumexp` function, defined as follows:

$$\sigma(X, \tau) = \tau \cdot \log \left( \frac{\sum_{i=1}^m \sum_{j=1}^n e^{X_{ij}/\tau} + e^{-X_{ij}/\tau}}{2mn} \right) \quad (9)$$

Define matrices  $P, N \in \mathbb{R}^{m \times n}$  such that:  $P_{ij} = e^{X_{ij}/\tau} + e^{-X_{ij}/\tau}$  and  $N_{ij} = e^{X_{ij}/\tau} - e^{-X_{ij}/\tau}$ . Then, the following lemma defines the gradient and Hessian information of the `logsumexp` function; see also the supp. material:

**Lemma 4.3.** For any  $X \in \mathbb{R}^{m \times n}$ :

- $\nabla \sigma(X, \tau) = \frac{1}{\text{Tr}(\mathbb{1} \cdot P)} \cdot N \in \mathbb{R}^{m \times n}$ ,
- $\nabla^2 \sigma(X, \tau) = \frac{\left( \text{diag}(\text{vec}(P)) - \frac{\text{vec}(N)\text{vec}(N)^\top}{\text{Tr}(\mathbb{1} \cdot P)} \right)}{\tau \cdot \text{Tr}(\mathbb{1} \cdot P)} \in \mathbb{R}^{mn \times mn}$

where  $\text{diag}(\cdot) : \mathbb{R}^{mn} \rightarrow \mathbb{R}^{mn \times mn}$  turns the vector input to a diagonal matrix output,  $\text{vec}(\cdot) : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^{mn}$  turns a matrix to a vector by ‘‘stacking’’ its columns, and  $\mathbb{1}$  denotes the all-ones matrix.

Similar to the Charbonnier approximation, we get the following lemma; the proof is in the supp. material:

**Lemma 4.4.** *The  $\text{logsumexp}$  function  $\sigma$  is a convex continuously differentiable function and it has Lipschitz continuous gradients with constant  $\frac{1}{\tau}$ . Moreover:*

$$|X|_\infty - \tau \log(2mn) \leq \sigma(X, \tau) \leq |X|_\infty.$$

## 5 AN APPROXIMATE SOLVER FOR $\ell_p$ -NORM LOW RANK APPROXIMATION

The proposed schemes are provided in Algorithms 2-3, and are based on Algorithm 1 as a sub-solver. In order to hope for a good initialization, we consider the smooth versions of (1), as described in Section 4, with the added twist that we regularize further the objective with a strongly convex component. *I.e.*, we approximate (1) for  $p = 1$  with:

$$\min_{U \in \mathbb{R}^{m \times r}, V \in \mathbb{R}^{n \times r}} h(M - UV^\top, \tau) + \frac{\lambda}{2} |UV^\top|_2^2, \quad (10)$$

and the case  $p = \infty$  with

$$\min_{U \in \mathbb{R}^{m \times r}, V \in \mathbb{R}^{n \times r}} \sigma(M - UV^\top, \tau) + \frac{\lambda}{2} |UV^\top|_2^2. \quad (11)$$

This modification asserts that both (10)-(11) are strongly convex w.r.t.  $X$  with parameter  $\lambda$ ; see also the proof of Lemma 4.2. Observe that the smaller the  $\lambda$  parameter is, the less the “drift” from the original problem. We remind that the optimal factors of (1) are  $U^*$  and  $V^*$ , and their product is denoted as  $X^* = U^*V^{*\top}$ .

---

### Algorithm 2 $\ell_1$ -norm low rank approximation solver

- 1: **Parameters:**  $r$ , OPT, values of  $|X^*|_2^2$  and  $\sigma_r(\widehat{X}^*)$ ,  $\varepsilon > 0$ .
  - 2: Set  $\tau = \frac{\varepsilon \cdot \text{OPT}}{3mn}$ .
  - 3: Set function  $T = O\left(\frac{\sigma_r(\widehat{X}_r^*)}{\varepsilon \text{OPT}}\right)$ .
  - 4: Set  $\lambda = \frac{2\varepsilon \cdot \text{OPT}}{3|X^*|_2^2}$ .
  - 5: Compute  $\widehat{L} = \left(\frac{1}{\tau} + \lambda\right)$ .
  - 6: Set  $f(UV^\top) := h(M - UV^\top, \tau) + \frac{\lambda}{2} |UV^\top|_2^2$ .
  - 7: Run Algorithm 1  $(U_T, V_T) = \text{BFGD}(r, T, \frac{1}{4}, 1, \widehat{L})$ .
- 

Let us first focus on the case of  $\ell_1$ -norm and Algorithm 2. The following theorem states that, under proper configuration of algorithm’s hyperparameters, one can achieve  $(1 + \varepsilon)$ -OPT approximation guarantee.

**Theorem 5.1.** *Let  $\widehat{X} = U_T V_T^\top \in \mathbb{R}^{m \times n}$  be the solution of Algorithm 2. Let the optimal function value of (1) for  $p = 1$  be denoted as  $\text{OPT} := \min_{U, V} |M - UV^\top|_1$  and assumed known, or at least be approximable. Also, assume we know  $\sigma_r(\widehat{X}^*)$  and  $|X^*|_2^2$ . For user defined*

*parameter  $\varepsilon > 0$  and setting the Charbonnier parameter  $\tau = \frac{\varepsilon \cdot \text{OPT}}{3mn}$ , and the strong convexity parameter as  $\lambda = \frac{2\varepsilon \cdot \text{OPT}}{3|X^*|_2^2}$ , the pair  $(U_T, V_T)$  of Algorithm 2 satisfies:*

$$|M - U_T V_T^\top|_1 \leq (1 + \varepsilon) \cdot \text{OPT},$$

*after  $T = O\left(\sigma_r(\widehat{X}_r^*) \left(\frac{mn}{(\varepsilon \text{OPT})^2} + \frac{1}{\|X^*\|_2^2}\right)\right)$  iterations.*

The proof is provided in the appendix. In the case where OPT is only approximable, straightforward modifications lead to similar performance (where higher number of iterations required).

*Analytical complexity:* Let us denote the time to compute  $\nabla f(\cdot)$  as  $t_{\text{grad}}$ . The initialization complexity of Algorithm 1, as well as its per iteration complexity, is  $O(t_{\text{grad}} + mn r)$ , where the last term is due to either low-rank SVD calculation or matrix-matrix multiplication. Running Algorithm 1 for  $T = O\left(\frac{\sigma_r(\widehat{X}_r^*)}{\varepsilon \text{OPT}}\right)$  iterations leads to an overall  $O\left(\frac{\sigma_r(\widehat{X}_r^*)}{\varepsilon \text{OPT}} \cdot (t_{\text{grad}} + mn r)\right)$  time complexity.

Similarly for the case of  $p = \infty$ , we use the  $\text{logsumexp}$  function in Algorithm 3 to smooth the objective, and we obtain the following guarantees:

---

### Algorithm 3 $\ell_\infty$ -norm low rank approximation solver

- 1: **Parameters:**  $r$ , OPT, values of  $|X^*|_2^2$  and  $\sigma_r(\widehat{X}^*)$ ,  $\varepsilon > 0$ .
  - 2: Set  $\tau = \frac{\varepsilon \cdot \text{OPT}}{3 \log(2mn)}$ .
  - 3: Set function  $T = O\left(\frac{\sigma_r(\widehat{X}_r^*)}{\varepsilon \text{OPT}}\right)$ .
  - 4: Set  $\lambda = \frac{2\varepsilon \cdot \text{OPT}}{3|X^*|_2^2}$ .
  - 5: Compute  $\widehat{L} = \left(\frac{1}{\tau} + \lambda\right)$ .
  - 6: Set  $f(UV^\top) := \sigma(M - UV^\top, \tau) + \frac{\lambda}{2} |UV^\top|_2^2$ .
  - 7: Run Algorithm 1  $(U_T, V_T) = \text{BFGD}(r, T, \frac{1}{4}, 1, \widehat{L})$ .
- 

**Corollary 5.2.** *Let  $\widehat{X} = U_T V_T^\top \in \mathbb{R}^{m \times n}$  be the solution of Algorithm 2. Let the optimal function value of (1) for  $p = \infty$  be denoted as  $\text{OPT} := \min_{U, V} |M - UV^\top|_\infty$ , and assumed known, or be at least approximable. Also, assume we know  $\sigma_r(\widehat{X}^*)$  and  $|X^*|_2^2$ . For user defined approximation parameter  $\varepsilon > 0$  and setting the  $\text{logsumexp}$  parameter  $\tau = \frac{\varepsilon \cdot \text{OPT}}{3 \log(2mn)}$ , and the strong convexity parameter as  $\lambda = \frac{2\varepsilon \cdot \text{OPT}}{3|X^*|_2^2}$ , the pair  $(U_T, V_T)$  of Algorithm 2 satisfies:*

$$|M - U_T V_T^\top|_\infty \leq (1 + \varepsilon) \cdot \text{OPT},$$

*after  $T = O\left(\sigma_r(\widehat{X}_r^*) \left(\frac{\log(mn)}{(\varepsilon \text{OPT})^2} + \frac{1}{\|X^*\|_2^2}\right)\right)$  iterations.*

Similar analytical complexity can be derived for Algorithm 3 and is omitted due to lack of space.

Results of similar flavor (and under similar assumptions) can be found in [28] for the problem of maximum flow. There, the authors consider non-Euclidean gradient descent algorithms for the minimization of  $\ell_\infty$ -norm over vectors, where the gradient step takes into consideration the geometry of the non-smooth objective with the use of *sharp* operators. We applied a similar approach for both  $p \in \{1, \infty\}$  in our setting; however, the empirical performance was prohibitive to consider a similar approach here (despite the fact that one can still achieve  $(1 + \varepsilon)$ -optimal approximation guarantees).

Some remarks regarding the above results.

**Remark 1.** *Both algorithms require the knowledge of three quantities: OPT,  $|X^*|_2^2$  and  $\sigma_r(\hat{X}^*)$ . While finding these values could be as difficult as the original problem (1), these values do not need to be known exactly: in particular, the algorithms imply that “for sufficiently small  $\tau$  and  $\lambda$  parameters, and for a sufficiently large number of iterations  $T$ , we can find a good approximation”.*

**Remark 2.** *While finding the exact value of OPT is difficult, there are problem cases where this value could be easily upper bounded. E.g., consider the problem of low-rank matrix approximation from quantization, as noted in [17]: there, we know from structure that  $|M - X^*|_\infty = \text{OPT} \leq 0.5$ .*

**Remark 3.** *Finding a good initialization is a key assumption for Theorem 5.1 and its corollary. Such assumptions are made also in other non-convex matrix factorization results; see [47, 58, 51, 4, 42, 16, 40, 41, 35, 34, 54, 15]. From [41], it is known that we can easily compute such an initialization as the best rank- $r$  approximation of  $M$  w.r.t. the  $\ell_2$ -norm, via SVD. In particular, such an initialization satisfies  $\text{DIST}(U_0, V_0; \hat{X}^*) \leq \frac{\sqrt{2} \cdot \sigma_r(\hat{X}^*)^{1/2}}{10\sqrt{\kappa}}$ , as long as  $f$  is strongly convex with condition number  $\kappa \leq 1 + \frac{\sigma_r(\hat{X}^*)^2}{4608 \cdot |X^*|_2^2}$ . While this condition is not easily met in theory (i.e., since  $\kappa = \frac{1+\lambda}{\lambda}$ , this means that  $\tau$  should be large enough compared to  $\lambda$ ), our experiments show that such an initialization performs well.*

**Remark 4.** *As a continuation of the above remark, the reason we use the regularizer  $\frac{\lambda}{2}|UV^\top|_2^2$  is to turn the smooth approximations into strongly convex functions (and thus borrow results for initialization). In practice, the proposed schemes work as well without the addition of the regularizer; and thus, knowing a priori the quantity  $|X^*|_2^2$  is not necessary in practice.*

**Remark 5.** *The approach we follow somewhat resembles with the approach proposed in [27]. There, the authors consider (1) for  $p = 1$  and propose an alternating minimization scheme. Despite the similarities, there are differences with our approach: among which, we perform a single gradient descent step on  $U$  and  $V$  per itera-*

*tion, for a smoothed version of (1), instead of minimizing a quadratic programming formulation per each column of  $U$  and  $V$ . On the contrary, [27] handles empirically missing values and weighted low-rank matrix factorization cases; we leave this direction for future research.*

## 6 EXPERIMENTS

Our experiments include synthesized applications, in order to highlight the empirical performance of the proposed framework. We compare the algorithms in Section 5 (i) with the algorithms for  $\ell_p$ -low rank approximation in [9], and (ii) with the recent heuristic in [17] for  $\ell_\infty$ -low rank approximation.

Similarly to [9, 17] and in order to guarantee fair comparison, we follow in practice the “folklore” advice for getting an initial estimate for the  $\ell_p$ -norm problem in (1) by beginning with the optimum  $\ell_2$ -norm solution (i.e., with the low-rank SVD solution).

### 6.1 $\ell_1$ -norm approximation

We perform experiments on both real and synthetic datasets. At first, we generate data according to the recent ICML paper [9]: We use  $20 \times 30$  random matrices  $M$ , where each entry is a uniformly random value in  $[0, 1]$ . Such constructions lead to full rank matrices with high-probability. We also construct matrices  $M$  of the same size with  $\{\pm 1\}$  entries, each selected with 0.5 probability. For real datasets, similar to [9], we use the FIDAP dataset<sup>5</sup> and a word frequency dataset from UC Irvine<sup>6</sup>. The FIDAP matrix  $M$  is  $27 \times 27$  with 279 real asymmetric non-zero entries. The word frequency matrix  $M$  is  $3430 \times 6906$  with 353, 160 non-zero entries.

For the synthesized datasets, we perform 10 Monte Carlo instantiations and take the median error reported. For all datasets, we are interested in computing the best rank- $r$  approximation of each  $M$  above, w.r.t. the  $\ell_1$ -norm and for  $r \in \{1, \dots, 10\}$ . To compare with [9], we use their suggestion and run a simplified version of Algorithm 2 in [9], where we repeatedly sample  $r$  columns, uniformly at random. We then run the  $\ell_p$ -projection (see Lemma 1 in [9]) on each sampled set and finally select the solution with the smallest  $\ell_p$ -error. For a fair contrast between the algorithms, we first run our algorithm and measure the required time; for approximately the same amount of time, we run [9].<sup>7</sup> To perform the  $\ell_p$ -projection, we use

<sup>5</sup><http://math.nist.gov/MatrixMarket/data/SPARSKIT/fidap/fidap005.html>

<sup>6</sup><https://archive.ics.uci.edu/ml/datasets/Bag+of+Words>

<sup>7</sup>In all our experiments, we make sure the algorithm in [9] runs at least the same time with our scheme.

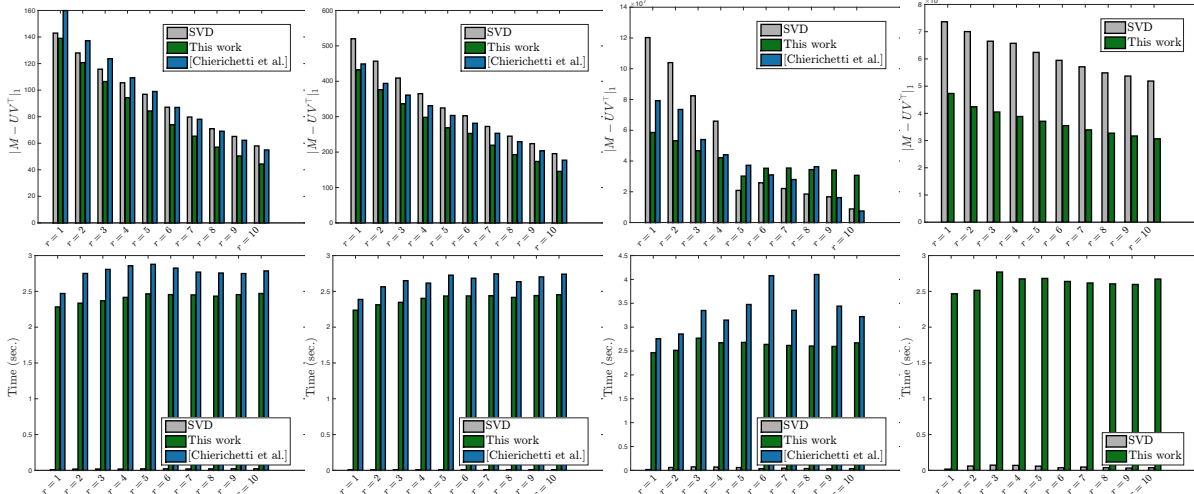


Figure 2: Top row: function value performance  $|M - UV^T|_1$ ; Bottom row: corresponding execution time. In all settings, we set problem (1) for  $r = \{1, \dots, 10\}$ . First column:  $M \in [0, 1]^{20 \times 30}$  where each entry is randomly and independently generated. Second column:  $M \in [-1, 1]^{20 \times 30}$  where each entry is randomly and independently generated. Third column:  $M \in \mathbb{R}^{27 \times 27}$  is the FIDAP matrix. Fourth column:  $M \in \mathbb{R}^{3430 \times 6906}$  is the word-frequency matrix. In the latter case, the sub-solver for  $\ell_p$ -projection was not able to complete the task, and thus the algorithm in [9] is omitted.

CVX package [14].<sup>8</sup>

In our algorithm, we set  $\tau = \lambda = 10^{-3}$ , and the maximum number of iterations as  $T = 4 \cdot 10^4$ . As mentioned above, we use the SVD initialization, and the step size is set according to Algorithm 1.

The results are provided in Figure 2. Some remarks: (i) for the synthetic cases (two leftmost columns), we observe that our approach attains a better objective function, faster, compared to [9]. Both our work and [9] is much slower than plain SVD; however, the latter gives a worse solution. (ii) for the real case (two rightmost columns), our approach is overall better in terms of objective function values; however, this is not universal; there are cases where [9] (or even SVD) gets to a better result within the same time, especially when  $r$  increases. For the large matrix case, [9] with CVX do not scale well; thus omitted.

## 6.2 $\ell_\infty$ -norm approximation

In this experiment, we follow the experimental setting in [17]. We generate matrices  $M \in \mathbb{R}^{100 \times 75}$  as follows: We generate  $\tilde{M} = UV^T$  where  $U \in \mathbb{R}^{100 \times r}$  and  $V \in \mathbb{R}^{75 \times r}$ . Each  $U$  and  $V$  is generated i.i.d. from  $N(0, 1)$ . Given  $\tilde{M}$ , we compute the rounded version of

<sup>8</sup>We are not aware of another standardized package for  $\ell_p$ -regression. To accelerate the execution of SeDuMi, we use the lowest precision set up in CVX.

$\tilde{M}$  such as  $M = \text{round}(\tilde{M})$ . This procedure guarantees that, given  $M$ , there is a low-rank matrix  $\tilde{M}$  that satisfies  $|M - \tilde{M}|_\infty \leq 0.5$  (since this is an hard problem, this construction gives an idea how far/close we are to a good solution).

We repeat the above procedure for  $r = \{1, \dots, 10\}$  and for 10 Monte Carlo instances. We report the minimum, mean and median values of the objective function attained and the time required. We compare our algorithms with plain SVD and the heuristics in [17].

The results are reported in Table 1. Our findings show that both our work and the algorithm in [17] perform much better (in terms of quality of solution) than plain SVD (the full set of results can be found in the appendix). Further, the algorithm in [17] has time comparable to the implementation of SVD in Matlab, while our proposed algorithm is much slower; accelerating our proposed algorithm is considered future research direction. However, while our algorithm does not succeed to find solutions with small objective value (see minimum value in table and compare our work with [17]), the median value of objective function values over 10 problem instances is lower than that of [17]. *I.e.*, the “typical” achieved objective value is lower than that of [17].<sup>9</sup>

<sup>9</sup>We ran the algorithm in [17] for more time (repeatedly within allowed time) and picked the best minimum result. However, this did not improve the results of [17].



[17]		
Rank $r$	Time (sec.)	Error
	[min, mean, median]	
1	[6.81e-02, 2.24e-01, 2.28e-01]	[4.91e-01, 4.93e-01, 4.93e-01]
2	[1.55e-02, 2.75e-02, 2.31e-02]	[5.33e-01, 6.00e-01, 5.96e-01]
3	[2.42e-02, 5.89e-02, 4.59e-02]	[5.22e-01, 5.63e-01, 5.44e-01]
4	[2.69e-02, 4.61e-02, 4.04e-02]	[5.24e-01, 5.66e-01, 5.42e-01]
5	[4.67e-02, 3.36e-01, 1.48e-01]	[5.04e-01, 5.36e-01, 5.26e-01]
6	[6.72e-02, 6.24e-01, 1.34e-01]	[4.98e-01, 5.20e-01, 5.22e-01]
7	[5.46e-02, 8.91e-01, 5.47e-01]	[4.90e-01, 5.14e-01, 5.11e-01]
8	[1.36e-01, 1.66e+00, 5.39e-01]	[4.81e-01, 5.15e-01, 5.02e-01]
9	[1.90e-01, 2.91e+00, 2.56e+00]	[4.73e-01, 4.98e-01, 4.89e-01]
10	[2.30e-01, 9.60e+00, 4.25e+00]	[4.59e-01, 4.97e-01, 4.79e-01]

This work		
Rank $r$	Time (sec.)	Error
	[min, mean, median]	
1	[2.57e-02, 4.32e+01, 5.44e+01]	[4.99e-01, 5.82e-01, 5.01e-01]
2	[2.60e-02, 4.95e+01, 5.44e+01]	[5.04e-01, 5.49e-01, 5.07e-01]
3	[5.20e+01, 5.43e+01, 5.42e+01]	[5.06e-01, 5.10e-01, 5.10e-01]
4	[1.55e-02, 3.67e+01, 5.15e+01]	[5.05e-01, 5.90e-01, 5.10e-01]
5	[4.17e-02, 7.92e+01, 8.93e+01]	[5.07e-01, 5.33e-01, 5.13e-01]
6	[7.27e+01, 8.03e+01, 7.76e+01]	[5.02e-01, 5.08e-01, 5.09e-01]
7	[1.62e-02, 5.11e+01, 6.52e+01]	[5.08e-01, 5.84e-01, 5.08e-01]
8	[5.51e+01, 6.55e+01, 6.73e+01]	[4.95e-01, 5.09e-01, 5.02e-01]
9	[5.36e+01, 5.89e+01, 5.77e+01]	[4.78e-01, 5.06e-01, 5.06e-01]
10	[1.69e-02, 3.86e+01, 5.23e+01]	[4.69e-01, 5.94e-01, 4.75e-01]

Table 1: Attained objective function values and execution time. Table includes minimum, mean and median values for 10 Monte Carlo instances.

## 7 CONCLUSION AND FUTURE WORK

We consider the problem of low-rank matrix approximation, w.r.t. (entrywise)  $\ell_p$ -norms, and proposed two algorithms that lead to  $(1 + \varepsilon)$ -OPT approximations. Our schemes combine ideas from smoothing techniques in convex optimization, as well as recent non-convex gradient descent algorithms. Key assumption is that problem-related quantities are known or at least are approximable. Our experiments show that our scheme performs (at least) competitively with state of the art.

We have provided several possible extensions of this work. A particularly interesting open problem is that of weighted low-rank matrix approximation:

$$\min_{U \in \mathbb{R}^{m \times r}, V \in \mathbb{R}^{n \times r}} |W \odot (M - UV^T)|_p, \quad p \in \{1, \infty\},$$

where different assumptions on  $W$  lead to different open research questions.

## References

- [1] H. Aanas, R. Fisker, K. Astrom, and J. Carstensen. Robust factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1215–1225, 2002.
- [2] M. Asteris, A. Kyriallidis, D. Papailiopoulos, and A. Dimakis. Bipartite correlation clustering: Maximizing agreements. In *Artificial Intelligence and Statistics*, pages 121–129, 2016.
- [3] J. Barron. A more general robust loss function. *arXiv preprint arXiv:1701.03077*, 2017.
- [4] S. Bhojanapalli, A. Kyriallidis, and S. Sanghavi. Dropping convexity for faster semi-definite optimization. In *29th Annual Conference on Learning Theory*, pages 530–582, 2016.
- [5] R. Cabral, F. De la Torre, J. Costeira, and A. Bernardino. Unifying nuclear norm and bilinear factorization approaches for low-rank matrix decomposition. In *Proceedings of the IEEE International Conference on Computer Vision*, 2013.
- [6] E. Candes, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.
- [7] P. Charbonnier, L. Blanc-Feraud, G. Aubert, and M. Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *Image Processing, 1994. Proceedings. ICIP-94., IEEE International Conference*, volume 2, pages 168–172. IEEE, 1994.
- [8] K.-Y. Chiang, C.-J. Hsieh, and I. Dhillon. Robust principal component analysis with side information. In *International Conference on Machine Learning*, pages 2291–2299, 2016.
- [9] F. Chierichetti, S. Gollapudi, R. Kumar, S. Lattanzi, R. Panigrahy, and D. Woodruff. Algorithms for  $\ell_p$  low rank approximation. *arXiv preprint arXiv:1705.06730*, 2017.
- [10] M. Collins, S. Dasgupta, and R. Schapire. A generalization of principal components analysis to the exponential family. In *Advances in neural information processing systems*, pages 617–624, 2002.
- [11] I. Csiszar and G. Tusnady. Information geometry and alternating minimization procedures. *Statistics and decisions*, 1984.
- [12] A. d’Aspremont. Smooth optimization with approximate gradient. *SIAM Journal on Optimization*, 19(3):1171–1183, 2008.
- [13] A. Eriksson and A. Van Den Hengel. Efficient computation of robust low-rank matrix approximations in the presence of missing data using the  $\ell_1$ -norm. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010.
- [14] Michael G. and Stephen B. CVX: Matlab software for disciplined convex programming, version 2.1. <http://cvxr.com/cvx>, March 2014.
- [15] R. Ge, C. Jin, and Y. Zheng. No spurious local minima in nonconvex low rank problems: A unified geometric analysis. *arXiv preprint arXiv:1704.00708*, 2017.
- [16] R. Ge, J. Lee, and T. Ma. Matrix completion has no spurious local minimum. *To appear in NIPS-16, arXiv preprint arXiv:1605.07272*, 2016.
- [17] N. Gillis and Y. Shitov. Low-rank matrix approximation in the infinity norm. *arXiv preprint arXiv:1706.00078*, 2017.
- [18] N. Gillis and S. Vavasis. Fast and robust recursive algorithms for separable nonnegative matrix factorization. *IEEE transactions on pattern analysis and machine intelligence*, 36(4):698–714, 2014.
- [19] N. Gillis and S. Vavasis. On the complexity of robust PCA and  $\ell_1$ -norm low-rank matrix approximation. *arXiv preprint arXiv:1509.09236*, 2015.
- [20] G. Golub and C. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.
- [21] G. Gordon. Generalized<sup>2</sup> linear<sup>2</sup> models. In *Advances in neural information processing systems*, pages 593–600, 2003.
- [22] S. Goreinov and E. Tyrtshnikov. The maximal-volume concept in approximation by low-rank matrices. *Contemporary Mathematics*, 280:47–52, 2001.
- [23] S. Goreinov and E. Tyrtshnikov. Quasioptimality of skeleton approximation of a matrix in the Chebyshev norm. In *Doklady Mathematics*, volume 83, pages 374–375. Springer, 2011.
- [24] Q. Gu, Z. W. Wang, and H. Liu. Low-rank and sparse structure pursuit via alternating minimization. In *Artificial Intelligence and Statistics*, pages 600–609, 2016.
- [25] P. Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.
- [26] Q. Ke and T. Kanade. Robust subspace computation using  $\ell_1$ -norm. 2003.
- [27] Q. Ke and T. Kanade. Robust  $\ell_1$  factorization in the presence of outliers and missing data by alternative convex programming. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 739–746. IEEE, 2005.

- [28] J. Kelner, Y. T. Lee, L. Orecchia, and A. Sidford. An almost-linear-time algorithm for approximate max flow in undirected graphs, and its multi-commodity generalizations. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 217–226. SIAM, 2014.
- [29] E. Kim, M. Lee, C.-H. Choi, N. Kwak, and S. Oh. Efficient  $\ell_1$ -norm-based low-rank matrix approximations for large-scale problems using alternating rectified gradient method. *IEEE transactions on neural networks and learning systems*, 26(2):237–251, 2015.
- [30] N. Kwak. Principal component analysis based on  $\ell_1$ -norm maximization. *IEEE transactions on pattern analysis and machine intelligence*, 30(9):1672–1680, 2008.
- [31] A. Kyrillidis and V. Cevher. Matrix ALPS: Accelerated low rank and sparse matrix reconstruction. In *Statistical Signal Processing Workshop (SSP), 2012 IEEE*, pages 185–188. IEEE, 2012.
- [32] A. Kyrillidis and V. Cevher. Matrix recipes for hard thresholding methods. *Journal of mathematical imaging and vision*, 48(2):235–265, 2014.
- [33] A. Kyrillidis, A. Kalev, D. Park, S. Bhojanapalli, C. Caramanis, and S. Sanghavi. Provable quantum state tomography via non-convex methods. *arXiv preprint arXiv:1711.02524*, 2017.
- [34] X. Li, Z. Wang, J. Lu, R. Arora, J. Haupt, H. Liu, and T. Zhao. Symmetry, saddle points, and global geometry of nonconvex matrix factorization. *arXiv preprint arXiv:1612.09296*, 2016.
- [35] Y. Li, Y. Liang, and A. Risteski. Recovery guarantee of non-negative matrix factorization via alternating updates. In *Advances in Neural Information Processing Systems*, pages 4987–4995, 2016.
- [36] P. Markopoulos, G. Karystinos, and D. Pados. Some options for  $\ell_1$ -subspace signal processing. In *Wireless Communication Systems (ISWCS 2013), Proceedings of the Tenth International Symposium on*, pages 1–5. VDE, 2013.
- [37] P. Markopoulos, G. Karystinos, and D. Pados. Optimal algorithms for  $\ell_1$ -subspace signal processing. *IEEE Transactions on Signal Processing*, 62(19):5046–5058, 2014.
- [38] D. Meng, Z. Xu, L. Zhang, and J. Zhao. A cyclic weighted median method for  $\ell_1$  low-rank matrix factorization with missing entries. In *AAAI*, volume 4, page 6, 2013.
- [39] Y. Nesterov. Smoothing technique and its applications in semidefinite optimization. *Mathematical Programming*, 110(2):245–259, 2007.
- [40] D. Park, A. Kyrillidis, S. Bhojanapalli, C. Caramanis, and S. Sanghavi. Provable Burer-Monteiro factorization for a class of norm-constrained matrix problems. *arXiv preprint arXiv:1606.01316*, 2016.
- [41] D. Park, A. Kyrillidis, C. Caramanis, and S. Sanghavi. Finding low-rank solutions to matrix problems, efficiently and provably. *arXiv preprint arXiv:1606.03168*, 2016.
- [42] D. Park, A. Kyrillidis, C. Caramanis, and S. Sanghavi. Non-square matrix sensing without spurious local minima via the Burer-Monteiro approach. *arXiv preprint arXiv:1609.03240*, 2016.
- [43] S. Poljak and J. Rohn. Checking robust nonsingularity is NP-hard. *Mathematics of Control, Signals, and Systems (MCSS)*, 6(1):1–9, 1993.
- [44] C. Qiu, N. Vaswani, B. Lois, and L. Hogben. Recursive robust PCA or recursive sparse recovery in large but structured noise. *IEEE Transactions on Information Theory*, 60(8):5007–5039, 2014.
- [45] A. Singh and G. Gordon. A unified view of matrix factorization models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 358–373. Springer, 2008.
- [46] Z. Song, D. Woodruff, and P. Zhong. Low rank approximation with entrywise  $\ell_1$ -norm error. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 688–701. ACM, 2017.
- [47] R. Sun and Z.-Q. Luo. Guaranteed matrix completion via nonconvex factorization. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015*, pages 270–289, 2015.
- [48] M. Tipping. Probabilistic visualisation of high-dimensional binary data. In *Advances in neural information processing systems*, pages 592–598, 1999.
- [49] M. Tipping and C. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.
- [50] Q. Tran-Dinh and Z. Zhang. Extended Gauss-Newton and Gauss-Newton-ADMM algorithms for low-rank matrix optimization. *arXiv preprint arXiv:1606.03358*, 2016.
- [51] S. Tu, R. Boczar, M. Simchowitz, M. Soltanolkotabi, and B. Recht. Low-rank solutions of linear matrix equations via Procrustes flow. *arXiv preprint arXiv:1507.03566*, 2015.
- [52] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86, 1991.

- [53] N. Veldt, A. Wirth, and D. Gleich. Correlation clustering with low-rank matrices. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1025–1034. International World Wide Web Conferences Steering Committee, 2017.
- [54] L. Wang, X. Zhang, and Q. Gu. A universal variance reduction-based catalyst for nonconvex low-rank matrix recovery. *arXiv preprint arXiv:1701.02301*, 2017.
- [55] T. Wiberg. Computation of principal components when data are missing. In *Proc. of Second Symp. Computational Statistics*, pages 229–236, 1976.
- [56] H. Xu, C. Caramanis, and S. Sanghavi. Robust PCA via outlier pursuit. In *Advances in Neural Information Processing Systems*, pages 2496–2504, 2010.
- [57] X. Yi, D. Park, Y. Chen, and C. Caramanis. Fast algorithms for robust PCA via gradient descent. In *Advances in neural information processing systems*, pages 4152–4160, 2016.
- [58] T. Zhao, Z. Wang, and H. Liu. A nonconvex optimization framework for low rank matrix estimation. In *Advances in Neural Information Processing Systems*, pages 559–567, 2015.
- [59] T. Zhou and D. Tao. GoDec: Randomized low-rank & sparse matrix decomposition in noisy case. In *International conference on machine learning*. Omnipress, 2011.

---

# Quantile-Regret Minimisation in Infinitely Many-Armed Bandits

---

Arghya Roy Chaudhuri and Shivaram Kalyanakrishnan

Department of Computer Science and Engineering  
Indian Institute of Technology Bombay, Mumbai 400076, India  
{arghya, shivaram}@cse.iitb.ac.in

## Abstract

The stochastic multi-armed bandit is a well-studied abstraction of decision making in the face of uncertainty. We consider the setting in which the number of bandit arms is much larger than the possible number of pulls, and can even be infinite. With the aim of minimising regret with respect to an *optimal* arm, existing methods for this setting either assume some structure over the set of arms (Kleinberg et al., 2008, Ray Chowdhury and Gopalan, 2017), or some property of the reward distribution (Wang et al., 2008). Invariably, the validity of such assumptions—and therefore the performance of the corresponding methods—depends on instance-specific parameters, which might not be known beforehand.

We propose a conceptually simple, parameter-free, and practically effective alternative. Specifically we introduce a notion of regret with respect to the top *quantile* of a probability distribution over the expected reward of randomly drawn arms. Our main contribution is an algorithm that achieves sublinear “quantile-regret”, both (1) when it is specified a quantile, and (2) when the quantile can be any (unknown) positive value. The algorithm needs no side information about the arms or about the structure of their reward distributions: it relies on random sampling to reach arms in the top quantile. Experiments show that our algorithm outperforms several previous methods (in terms of conventional regret) when the latter are not tuned well, and often even when they are.

## 1 INTRODUCTION

The stochastic multi-armed bandit (Berry and Fristedt, 1985) is a well-studied abstraction of on-line learning. Each bandit *arm* represents a slot-machine with a fixed (but unknown) real-valued reward distribution. An experimenter is allowed

to *pull* an arm at every time instant and observe its reward. The experimenter aims to maximise the total expected reward obtained over a horizon, or equivalently, to minimise the *regret* with respect to a strategy that always plays an optimal arm. Side information regarding the bandit instance may or may not be available to the experimenter.

Regret minimisation algorithms have to achieve a balance between exploration (gathering information about the reward distributions of arms) and exploitation (pulling seemingly-good arms). For a  $K$ -armed bandit, the optimal regret that can be achieved after  $T$  pulls is  $\Omega(\sqrt{KT})$  (Auer et al., 2003). To achieve a regret of  $O(\sqrt{KT})$  (Audibert and Bubeck, 2009), algorithms invariably have to maintain separate statistics for the pulls coming from each arm (since any of them could be the sole optimal arm).

In many modern applications of bandits, the set of arms that can be pulled is very large, often even infinite. Examples include (1) sensor networks, in which a central controller must learn to deploy the most accurate sensor from among a large number of noisy sensors (Kadono and Fukuta, 2014); (2) crowd-sourcing tasks, in which a periodic task should ideally be assigned to the most skilled worker in a large pool (Tran-Thanh et al., 2014); (3) on-line advertising, in which a layout for an ad should be chosen from among a large set so as to maximise the click-through rate (Tang et al., 2013). Clearly, the  $\Theta(\sqrt{KT})$  bound on the regret is not helpful when  $K \gg T$  or  $K = \infty$ . Perhaps the most common approach to deal with infinitely-many armed bandits is to utilise some sort of side information about the arms: for example, to assume that the arms are embedded in a metric space in which the reward function is Lipschitz continuous (Kleinberg, 2005) or even linear (Auer, 2003, Chu et al., 2011). Unfortunately such side information is not always available; even if available, it is often not accurate (Ghosh et al., 2017).

In this paper, we propose an approach for exploring the arms of infinitely many-armed bandits with no recourse to side information. Naturally, we cannot always guarantee sublinear regret with respect to optimal arms (which may never get pulled in any finite horizon). Instead, assuming that the

set of arms  $\mathcal{A}$  is being accessed through a given sampling distribution  $P_{\mathcal{A}}$ , we benchmark regret against *quantiles* of the reward distribution induced by  $P_{\mathcal{A}}$ . By fixing the quantile,  $P_{\mathcal{A}}$  will eventually sample “good enough” arms. Using a doubling trick, we can also ensure that our algorithm will eventually sample every arm whose expected reward is bounded away from the optimal reward. Below we formalise the notion of *quantile regret* and outline our contributions.

**Problem Definition.** A *bandit instance*  $\mathcal{B} = (\mathcal{A}, M)$  comprises a (possibly infinite) set of *arms*  $\mathcal{A}$ , and a reward function  $M$  that gives a bounded, real-valued, reward distribution  $M(a)$  for each arm  $a \in \mathcal{A}$ . When pulled, arm  $a$  produces a reward drawn i.i.d. from  $M(a)$ . We denote the expected reward from arm  $a$  as  $\mu_a \stackrel{\text{def}}{=} \mathbb{E}[M(a)]$ . Without loss of generality, we assume that all rewards lie in  $[0, 1]$ .

An *algorithm* is a mapping from the set of histories (of arms pulled and rewards obtained) to the set of probability distributions over  $\mathcal{A}$ : thus, given a history, an algorithm specifies a probability for sampling each arm. Let  $\mu^* \stackrel{\text{def}}{=} \min\{y \in [0, 1] : \forall a \in \mathcal{A}, \mu_a \leq y\}$ . Then, for a given horizon of pulls  $T$ , the *regret* of an algorithm is

$$\mathbb{R}_T^* = T\mu^* - \sum_{t=1}^T \mathbb{E}[\mu_{a_t}], \quad (1)$$

where  $a_t$  denotes the arm pulled by the algorithm at time  $t$ . The expectation is taken over the random outcomes of pulls, as well as random choices (if any) made by the algorithm.

For us, a *problem instance*  $\mathcal{I} = (\mathcal{B}, P_{\mathcal{A}})$  contains a bandit instance  $\mathcal{B}$  with arms  $\mathcal{A}$ , and a sampling distribution  $P_{\mathcal{A}}$  to choose arms from  $\mathcal{A}$ . We apply the concept of “ $(\epsilon, \rho)$ -optimality” introduced by Roy Chaudhuri and Kalyan Krishnan (2017). For a *quantile fraction*  $\rho \in [0, 1]$  and a *tolerance*  $\epsilon \in [0, 1]$ , an arm  $a \in \mathcal{A}$  is said to be  $(\epsilon, \rho)$ -optimal if  $\Pr_{a' \sim P_{\mathcal{A}}} \{\mu_a \geq \mu_{a'} - \epsilon\} \geq 1 - \rho$ . Let  $\mathcal{TOP}_{\rho}$  be the set of all  $(0, \rho)$ -optimal arms. Also let  $\mu_{\rho} \in [0, 1]$  be a quantity such that,  $\forall a' \in \mathcal{A} \setminus \mathcal{TOP}_{\rho}$ ,  $\mu_{\rho} > \mu_{a'}$  and  $\forall a \in \mathcal{TOP}_{\rho}$ ,  $\mu_a \geq \mu_{\rho}$ . In other words, if  $\mu$  denotes the mean of an arm drawn according to  $P_{\mathcal{A}}$ , then  $\mu_{\rho}$  is the  $(1 - \rho)$ -th quantile of the distribution of  $\mu$ .

Figure 1 shows the example of an infinite set of arms  $\mathcal{A}$  whose means lie between 0.15 and 0.95. When sampled according to  $P_{\mathcal{A}}^1$ , the resulting probability density function of the mean  $\mu$  is  $D^1(\mu)$ . If an algorithm is constrained to access arms using  $P_{\mathcal{A}}^1$ , it is only natural for us to evaluate it against a baseline that is also constrained by  $P_{\mathcal{A}}^1$ . For example, we could aim for the algorithm to eventually surpass  $q^1$ , which is the 94-th percentile of the distribution induced by  $P_{\mathcal{A}}^1$ . Without additional information, such an algorithm cannot hope to surpass  $q^2$ , the 94-th percentile of a different sampling distribution  $P_{\mathcal{A}}^2$ . In general we expect that there is some “natural” way to sample the unknown set of arms—and this is given to us as  $P_{\mathcal{A}}$ . For example, if  $\mathcal{A}$  is finite, it is a reasonable choice to assign an equal probability to

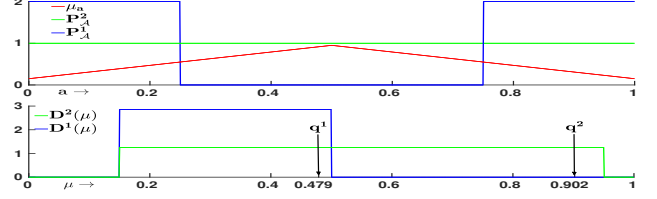


Figure 1: Two problem instances that share the same set of arms  $\mathcal{A}$ , varying continuously in  $[0, 1]$ . The top panel shows the expected rewards  $\mu_a$  of each arm  $a \in \mathcal{A}$ , as well as probability density functions  $P_{\mathcal{A}}^1$  and  $P_{\mathcal{A}}^2$  for sampling  $\mathcal{A}$ . The bottom panel shows the reward distributions  $D_{\mathcal{A}}^1$  and  $D_{\mathcal{A}}^2$  induced by each of the sampling distributions, along with 94-th percentiles  $q^1$  and  $q^2$ , respectively.

sampling each arm. If  $\mathcal{A}$  corresponds to a parameterised set,  $P_{\mathcal{A}}$  could implement some distribution over the parameters.

We are now ready to define *quantile-regret* with respect to a given quantile fraction  $\rho$ . If  $a_t$  is the arm drawn by an algorithm in its  $t$ th pull, we define the (cumulative) quantile-regret (or the “ $\rho$ -regret”) after  $T$  pulls as

$$\mathbb{R}_T(\rho) = T\mu_{\rho} - \sum_{t=1}^T \mathbb{E}[\mu_{a_t}]. \quad (2)$$

Our aim is to devise algorithms to minimise  $\mathbb{R}_T(\rho)$  for a given problem instance  $\mathcal{I}$ . The quantile fraction  $\rho$  can either be given as an input to the algorithm, or left unspecified, as we describe next.

**Contributions.** We show that without any knowledge of the reward distributions of the arms, or of side information such as a distance metric over arms, it is still possible to have strategies to maximise reward over time. We articulate the problem as that of achieving sub-linear  $\rho$ -regret.

1. In Section 3.1, we present our first algorithm, QRM1, which is given a quantile fraction  $\rho$  as input. We show that for a sufficiently large horizon  $T$ , the algorithm incurs  $\mathbb{R}_T(\rho) \in O(\rho^{-1} + \sqrt{(T/\rho) \log(\rho T)})$ . We also provide a lower bound of  $\Omega(\sqrt{T/\rho})$ , establishing tightness up to a logarithmic factor with respect to  $T$ .
2. In Section 3.2, we present our second algorithm, QRM2, which does not take  $\rho$  as an input. Regardless, the algorithm achieves sub-linear  $\rho$ -regret for every  $\rho > 0$ . Specifically, for every  $\rho > 0$  and a sufficiently large horizon  $T$ , QRM2 achieves  $\mathbb{R}_T(\rho) \in o((\frac{1}{\rho} \log \frac{1}{\rho})^{2.89} + T^{0.674})$ .
3. In Section 3.3, we establish a connection between (conventional) regret  $\mathbb{R}_T^*$  and  $\mathbb{R}_T(\rho)$ . Interestingly, we find that when run on instances satisfying a common assumption made in the literature about reward distributions (Herschkorn et al., 1996, Wang et al., 2017), QRM2 also achieves sub-linear *regret*  $\mathbb{R}_T^*$ .

4. In Section 4, we present extensive experimental results comparing QRM2 with three separate categories of algorithms: (1) those assuming that the arms lie in a continuum (Kleinberg et al., 2008, Ray Chowdhury and Gopalan, 2017); (2) those assuming that the mean rewards come from a reservoir distribution (Wang et al., 2008); and (3) algorithms that only retain a constant number of arms in memory (Herschhorn et al., 1996, Berry et al., 1997). In the first two cases we find existing approaches to be sensitive to parameter-tuning, while our parameter-free approach shows robust performance across a variety of problem instances. Except when the arms’ means indeed come from a uniform distribution (as assumed by some constant-memory algorithms), QRM2 also outperforms algorithms from the third category.

We survey the literature on regret minimisation in infinite-armed bandits before presenting our algorithms.

## 2 RELATED WORK

There is a vast body of literature considering regret-minimisation for infinitely many-armed bandits. Based on the assumptions they make, we classify research efforts in the area into three broad categories. We also provide brief mentions of other related work.

**1. Lipschitz-continuity of mean rewards over  $\mathcal{A}$ .** Initiated by Agrawal (1995), the “continuum-armed bandit” models a bandit whose arms come from a segment of the real line, and the rewards are a continuous function of the arms. Generalising this setting, Kleinberg (2005) and Auer et al. (2007) proposed algorithms assuming that the mean reward function  $\mathbb{E}[\mathbb{M}(\cdot)] = \mu(\cdot)$  is Lipschitz-continuous over the set of arms  $\mathcal{A}$ . Their approaches partition  $\mathcal{A}$  into a finite number of intervals, treating each interval (say pulled at its middle arm (Kleinberg, 2005)) as an arm in a finite-armed bandit. The partition is progressively refined at a rate that ensures sub-linear regret. The ZOOMING algorithm proposed by Kleinberg et al. (2008), which assumes that the arms are embedded in a metric space with (known) covering dimension  $d$ , achieves a regret of  $O(T^{\frac{d+1}{d+2}})$ , for a horizon  $T$ . Their algorithm utilises the metric property to focus exploration on intervals potentially containing optimal arms. Understandably, the regret incurred by ZOOMING is sensitive to the definition of metric in the arms’ space, and can degrade with small misspecifications of  $d$ .

A contrasting line of work follows from the work of Srinivas et al. (2010), who introduce a Gaussian Process-based algorithm, GP-UCB, for regret minimisation on infinitely many-armed bandits. Later Valko et al. (2013) proposed KERNELUCB, showing GP-UCB to be a special case. More recently, Ray Chowdhury and Gopalan (2017) have proposed two algorithms: Improved GP-UCB (IGP-UCB) and

GP-Thompson sampling (GP-TS). They assume Gaussian likelihood models for the observed rewards, and Gaussian Process models for the uncertainty over reward functions. Although Ray Chowdhury and Gopalan (2017) show improved regret bounds over previous work, their algorithms are not easy to apply in practice. Foremost, the algorithms themselves give no guidance on the number of arms to explore. Also, the algorithms need several parameters to be tuned, including a confidence parameter  $\delta$ , a free parameter  $\lambda$ , and a schedule  $\gamma_t$  related to information gain.

**2. Particular families of reward distributions.** There is a relatively large body of work that does not assume any embedding of the arms (that is, no side information), but still assumes that the distribution of mean rewards (induced by  $P_{\mathcal{A}}$ ) comes from a particular family. Among the earliest efforts in this direction is that of Berry et al. (1997), who propose several algorithms for infinitely-many armed bandits in which the mean rewards of the arms are uniformly distributed in  $[0, \mu^*]$  for some  $0 < \mu^* \leq 1$ . An additional assumption underlying their work is that for each arm  $a \in \mathcal{A}$ , the reward distribution  $M(a)$  is Bernoulli.

Wang et al. (2008) assume that a randomly-sampled arm’s mean reward  $\mu$  comes from a *reservoir distribution*  $\mathcal{L}$ ; that is,  $\exists \mu^* \in (0, 1]$  and  $\nu > 0$ , for which  $\Pr_{\mu \sim \mathcal{L}}\{\mu > \mu^* - \epsilon\} = \Theta(\epsilon^\nu)$ , for  $\epsilon \rightarrow 0$ . Under this assumption, they present an algorithm that incurs (1)  $R_T^* = \tilde{O}(T^{1/2})$  if  $\mu^* < 1$  and  $\nu \leq 1$ , and (2)  $R_T^* = \tilde{O}(T^{\nu/(1+\nu)})$  otherwise. They have also derived lower bounds that match up to a logarithmic factor. When each arm generates Bernoulli rewards and  $\mu^* = 1$ , Bonald and Proutiere (2013) provide an algorithm that is optimal with the exact constant. In more recent work, Li and Xia (2017) consider a related setting in which the probability of a newly-pulled arm being near-optimal arm depends on the ratio of its expected reward to its expected *cost*, with arms having different random costs.

**3. Constant-memory policies.** A particular novelty of the family of algorithms studied by Berry et al. (1997) is that the algorithms maintain the reward statistics of at most one or two arms at a time. When the arms’ reward distribution is *uniformly* distributed in  $(0, 1)$ , their algorithms are shown to achieve sub-linear regret. No closed-form upper bounds are provided when this condition does not hold. Also in the constant-memory category, Herschhorn et al. (1996) present two approaches for the problem of maximising the almost sure average reward over an infinite horizon. Although they assume that each arm generates i.i.d. Bernoulli rewards, they make no assumption on the distribution of mean rewards. They present two approaches, both of which repeatedly pull an arm until it records a certain number of successive failures. They do not provide an explicit bound on the regret.

**4. Other related work.** Recently, Wang et al. (2017) have proposed CEMAB, a cross-entropy based algorithm for many-armed bandit instances. Like us, they aim to focus

exploration on a small subset of arms. However, they still require the entire set of arms to be finite, which limits their experimentation to instances with a few tens of arms. They do not present any theoretical upper bounds on the regret.

The work we have discussed thus far in this section is all targeted at minimising regret. By contrast, there has also been some effort under the “pure exploration” regime to tackle infinitely-many armed bandits. For example, Carpentier and Valko (2015) aim to minimise *simple regret*, under the same assumption of a mean reservoir distribution assumption as Wang et al. (2008). Our own conception of quantile-regret is motivated by the work of Goschin et al. (2012) and Roy Chaudhuri and Kalyanakrishnan (2017), who study a PAC formulation of identifying arms above a specified reward quantile in infinitely many-armed bandits.

The primary motivation behind our work is to eliminate assumptions regarding structure and side information. Such inductive biases become counterproductive when they are not near-perfect. In Section 4, we show that the algorithms proposed by Kleinberg (2005), Ray Chowdhury and Gopalan (2017), and Wang et al. (2008), all fare poorly when their parameters are misspecified. Constant-memory algorithms, while conceptually elegant, forego the obvious benefit of retaining the statistics of multiple arms (say a few tens or hundreds) in memory. Except in tailormade settings (i.i.d. Bernoulli rewards, uniformly distributed mean rewards), our approach performs significantly better. We proceed to describe our algorithms.

### 3 MINIMISING QUANTILE-REGRET

At the heart of our approach for minimising quantile regret on infinitely many-armed bandit instances is to first sample out suitably-sized finite bandit instances and then to apply conventional regret minimisation algorithms on the latter. For ease of analysis, we choose the MOSS algorithm (Audibert and Bubeck, 2009) for our inner loop, since it incurs optimal (distribution-free) regret (up to a constant factor) on finite bandit instances.

First, we consider the easy case: that is, when  $\rho$  is provided as an input. Then we generalise this setting to one where  $\rho$  is not specified, and the objective is to achieve sub-linear  $\rho$ -regret for all  $\rho > 0$ . Our bounds will hold for sufficiently large (but still finite) horizons  $T$ . On the other hand, for a *fixed* horizon  $T$ , it is impossible to guarantee sub-linear  $\rho$ -regret for all  $\rho > 0$  for all problem instances. For example, consider a problem instance with a fraction  $\rho < 1/T$  of arms all being optimal, and the rest sub-optimal.  $T$  pulls will not suffice even to stumble upon an optimal arm with sufficiently high probability, let alone exploit it. The  $\rho$ -regret on such an instance will have to be linear in  $T$ .

#### 3.1 With quantile fraction specified

In order to minimise  $\rho$ -regret for a given quantile fraction  $\rho \in (0, 1]$ , our primitive operation is to sample a sufficiently large number of arms using  $P_{\mathcal{A}}$ , and to minimise conventional regret on this set of arms by applying MOSS. We implement an “any time” algorithm by repeating this primitive procedure with progressively larger horizons, as shown in Algorithm 1.

---

#### Algorithm 1 QRM1 (with quantile fraction specified)

---

**Require:**  $\mathcal{I}, \rho$

**for**  $r = 1, 2, 3, \dots$  **do**

$$t_r = 2^r, \quad n_r = \left\lceil \frac{1}{\rho} \max\{1, \ln \sqrt{\rho t_r}\} \right\rceil.$$

Form a set  $\mathcal{K}_r$  by selecting additional  $n_r - |\mathcal{K}_{r-1}|$  arms from  $\mathcal{A}$  using  $P_{\mathcal{A}}$ , and adding them to  $\mathcal{K}_{r-1}$ .

Run MOSS( $\mathcal{K}_r, t_r$ ).

**end for**

---

In each phase  $r$ , MOSS is called to run on a finite bandit instance with some  $\mathcal{K}_r$  arms, over a finite horizon  $t_r$ . MOSS is known to incur a regret of at most  $C\sqrt{|\mathcal{K}_r|t_r}$  for some constant  $C$  (Audibert and Bubeck, 2009, Theorem 5). The parameters  $\mathcal{K}_r$  and  $t_r$  are specifically chosen such that with sufficiently high probability, at least one arm from  $\mathcal{TOP}_{\rho}$  is selected in  $\mathcal{K}_r$ , and consequently the overall  $\rho$ -regret remains sub-linear.

Our analysis assumes  $\rho \in (0, 1)$ . The case of  $\rho = 1$  is trivial;  $\mathbb{R}_T(1)$  cannot exceed 0. For  $\rho = 0$ , sub-linear regret can only be achieved under the additional assumption that optimal arms have a positive probability under  $P_{\mathcal{A}}$ . In the analysis that follows, we use  $\log$  to denote the logarithm to the base 2, and  $\ln$  to denote the natural logarithm. We also take the horizon  $T$  to be a power of 2—which only changes our bounds by a constant factor.

**Lemma 3.1.** *For  $\rho \in (0, 1)$  and for sufficiently large  $T$ , QRM1 achieves  $\mathbb{R}_T(\rho) = O\left(\frac{1}{\rho} + \sqrt{\frac{T}{\rho} \log(\rho T)}\right)$ .*

*Proof.* Let us consider the event during phase  $r$  that no arm from  $\mathcal{K}_r$  is in  $\mathcal{TOP}_{\rho}$ . Denote this event  $E_r \stackrel{\text{def}}{=} \{\mathcal{K}_r \cap \mathcal{TOP}_{\rho} = \emptyset\}$ . We upper-bound the  $\rho$ -regret accumulated during phase  $r$ , which we denote  $L_r$ , by conditioning separately on  $E_r$  and  $\neg E_r$ .

In phase  $r$ , the probability of occurrence of  $E_r$  is  $\Pr\{E_r\} = (1 - \rho)^{n_r}$ . Now, letting  $r^* = \log(e^2/\rho)$ , we notice that for  $r \geq r^*$ ,  $t_r \geq e^2/\rho$ , and hence we can upper bound the probability of occurrence of  $E_r$  as  $\Pr\{E_r\} \leq (1 - \rho)^{\rho^{-1} \ln(\sqrt{\rho t_r})} < \sqrt{1/(\rho t_r)}$ . We simply take  $t_r$  as an upper bound on the phase’s contribution to the regret if  $E_r$  has occurred. If  $E_r$  does not occur, then there exists at least one arm from  $\mathcal{TOP}_{\rho}$  in  $\mathcal{K}_r$ . In this case, the regret is upper-bounded by  $C\sqrt{n_r t_r} \leq C\sqrt{t_r \log(\rho t_r)}/\rho$ , for some constant  $C$  (Audibert and Bubeck, 2009). Therefore, for



$r \geq r^*$ , the  $\rho$ -regret from phase  $r$  is upper-bounded as  $L_r \leq t_r \cdot \Pr\{E_r\} + C \cdot \sqrt{n_r t_r} \leq \sqrt{\frac{t_r}{\rho}} + C \cdot \sqrt{\frac{t_r}{\rho} \log(\rho t_r)} \leq C_1 \cdot \sqrt{\frac{t_r}{\rho} \log(\rho t_r)}$  for some constant  $C_1$ .

For phases  $r < r^*$ , the  $\rho$ -regret is trivially upper-bounded by  $t_r$ . Hence summing over all phases, we get  $\mathbb{R}_T(\rho) \leq \sum_{r=1}^{r^*-1} L_r + \sum_{r=r^*}^{\log T} L_r \leq 2^{r^*} + \sum_{r=r^*}^{\log T} C_1 \cdot \sqrt{\frac{t_r}{\rho} \log(\rho t_r)}$ , which is  $\in O\left(\frac{1}{\rho} + \sqrt{\frac{T}{\rho} \log(\rho T)}\right)$ .  $\square$

We show that this upper bound on the  $\rho$ -regret is optimal up to a logarithmic factor in the horizon. Our proof is based on a well-known lower bound for finite bandit instances (Auer et al., 2003, see Theorem 5.1).

**Theorem 3.2.** [Lower bound] *For every algorithm, there exists a problem instance, along with  $\rho \in (0, 1)$  and  $T > 0$ , such that  $\mathbb{R}_T(\rho) \geq \min\left\{\frac{1}{20}\sqrt{\frac{T}{\rho}}, T\right\}$ .*

*Proof.* Let  $ALG$  be any algorithm for sampling infinitely many-armed bandits. Naturally, we can also apply  $ALG$  on finite bandit instances. Given any arbitrary  $K$ -armed bandit instance,  $K < \infty$ , we can create a corresponding problem instance  $((\mathcal{A}, M), P_{\mathcal{A}})$  wherein (1)  $\mathcal{A}$  is the finite set of  $K$  arms, (2)  $M(a)$  is the reward function for  $a \in \mathcal{A}$ , and (3)  $P_{\mathcal{A}}$  samples each arm in  $\mathcal{A}$  with an equal probability of  $1/K$ . Now, if we set  $\rho = 1/K$ , observe that  $\mathcal{TOP}_{\rho}$  can only contain *optimal* arms from  $\mathcal{A}$ , and hence, the  $\rho$ -regret incurred by  $ALG$  on  $((\mathcal{A}, M), P_{\mathcal{A}})$  is the same as the conventional regret on the original finite instance.

Suppose, contrary to the statement of the theorem,  $ALG$  is such that for all input problem instances, for all  $\rho \in (0, 1)$  and for all  $T > 0$ , its  $\rho$ -regret satisfies  $\mathbb{R}_T(\rho) < \min\left\{\frac{1}{20}\sqrt{\frac{T}{\rho}}, T\right\}$ . From the translation described above, it follows that  $ALG$  incurs  $\mathbb{R}_T^* < \min\left\{\frac{1}{20}\sqrt{KT}, T\right\}$  for all finite  $K$ -armed bandit instances,  $K > 0$  and horizons  $T > 0$ . However, Auer et al. (2003, see Theorem 5.1) have shown that no such algorithm exists for finite instances. Our proof is complete.  $\square$

### 3.2 With quantile fraction not specified

We can now drop the requirement that  $\rho$  is given to the algorithm as an input. Rather, we iteratively optimise  $\rho$ -regret for progressively decreasing values of  $\rho$ .

Algorithm 2 follows the same template as Algorithm 1, except that the number of arms to sample in each phase  $r$  is set to be a polynomial function of  $t_r$ , with the power  $\alpha = 0.347$  set to minimise the  $\rho$ -regret's dependence on  $T$ .

Although QRM2, the algorithm specified above, does not require any knowledge of  $\rho$ , we shall analyse its  $\rho$ -regret for some fixed  $\rho > 0$ .

---

#### Algorithm 2 QRM2 (with quantile fraction not specified)

---

**Require:**  $T$

Set  $\alpha = 0.347$  and  $\mathcal{K}_0 = \emptyset$ .

**for**  $r = 1, 2, 3, \dots$  **do**

$t_r = 2^r$ ,  $n_r = \lceil t_r^\alpha \rceil$ .

Form a set  $\mathcal{K}_r$  by selecting additional  $n_r - |\mathcal{K}_{r-1}|$  arms from  $\mathcal{A}$  using  $P_{\mathcal{A}}$ , and adding to  $\mathcal{K}_{r-1}$ .

Run MOSS( $\mathcal{K}_r, t_r$ ).

**end for**

---

**Theorem 3.3.** [Sub-linear quantile-regret of QRM2] *For  $\rho \in (0, 1)$  and for sufficiently large  $T$ , QRM2 incurs  $\mathbb{R}_T(\rho) \in o\left(\left(\frac{1}{\rho} \log \frac{1}{\rho}\right)^{2.89} + T^{0.674}\right)$ .*

*Proof.* Considering some fixed  $\rho \in (0, 1)$ , we upper-bound the  $\rho$ -regret in two parts: (1) when no arms from  $\mathcal{TOP}_{\rho}$  are chosen, and (2) when at least one arm is chosen. To analyse the first part, we show that for  $r^* \stackrel{\text{def}}{=} \lceil (1/\alpha) \log((1/\rho) \log(1/\rho)) \rceil$ , if  $r \geq r^*$ , then  $\mathcal{K}_r$  is sufficiently large to contain an arm from  $\mathcal{TOP}_{\rho}$  with high probability. To show that, like before, we define the event that no arm from  $\mathcal{TOP}_{\rho}$  is in  $\mathcal{K}_r$  as  $E_r(\rho) \stackrel{\text{def}}{=} \{\mathcal{K}_r \cap \mathcal{TOP}_{\rho} = \emptyset\}$ . It follows  $\Pr\{E_r(\rho)\} = (1 - \rho)^{n_r}$ . Now, for  $r \geq r^*$ , using Lemma 5.1 (provided in Appendix A<sup>1</sup>), we get  $\Pr\{E_r(\rho)\} \leq \exp(-[(\alpha(1 + \gamma))^{-1} \cdot \ln t_r^{\log e}]) \leq t_r^{-\alpha \log e / (1 + \gamma)}$ . Hence, if the algorithm runs for  $T$  pulls, then the regret due to occurrence of  $E_r(\rho)$  is upper bounded as

$$\sum_{r=1}^{\log T} t_r \Pr\{E_r(\rho)\} \in O\left(t_{r^*} + T^{1 - \frac{\alpha \log e}{1 + \gamma}}\right). \quad (3)$$

Now we analyse the second part: that is, upper-bounding the regret incurred if at least one from the  $\mathcal{TOP}_{\rho}$  is in  $\mathcal{K}_r$  (the event  $\neg E_r(\rho)$ ). Let us assume that  $C$  is a constant such that the regret incurred by MOSS in phase  $r$  (given  $\neg E_r(\rho)$ ) is at most  $C\sqrt{n_r t_r} = Ct_r^{(1+\alpha)/2}$ . Therefore, assuming total number of pulls as  $T$ , the total regret incurred on  $r^*$ -th phase onward is upper bounded as

$$\sum_{r=r^*}^{\log T} C\sqrt{n_r t_r} \leq C'T^{(1+\alpha)/2} \quad (4)$$

for some constant  $C'$ . The intermediate steps to obtain (3) and (4) are shown in Appendix-A. Combining (3) and (4), and substituting for  $t_{r^*}$ , we get  $\mathbb{R}_T(\rho) = O\left(\left(\frac{1}{\rho} \log \frac{1}{\rho}\right)^{\frac{1}{\alpha}} + T^{1 - \frac{\alpha \log e}{1 + \gamma}} + T^{(1+\alpha)/2}\right)$ . We conclude by noticing that  $\alpha = 0.5 / (0.5 + \log e / (1 + \gamma)) \approx 0.3466$  minimises  $\mathbb{R}_T(\rho)$  with respect to  $T$ .  $\square$

<sup>1</sup>Appearing at the end of the extended version of this paper at [https://www.cse.iitb.ac.in/~shivaram/papers/rk\\_uai\\_2018.pdf](https://www.cse.iitb.ac.in/~shivaram/papers/rk_uai_2018.pdf).

The upper bound in Theorem 3.3 cannot be directly compared with regret bounds in the literature (Kleinberg, 2005, Wang et al., 2008) since our bound is on the  $\rho$ -regret. As yet, we do not know if the dependence of  $\rho$ -regret on the horizon  $T$  can be improved. Even so, the sub-linear upper bound we have shown on  $\mathbb{R}_T(\rho)$  assumes a special significance in the study of infinitely-many armed bandits. Observe that the upper bound holds for every  $\rho > 0$  and for *every* bandit instance. By contrast, conventional regret-minimisation (of  $\mathbb{R}_T^*$ ) cannot assure sub-linear regret unless the bandit instance itself satisfies additional assumptions (of which several have been made in the literature). By taking  $\rho > 0$ , we have chosen to change our objective (albeit slightly), rather than place demands on the input bandit instance.

Interestingly, we find that on bandit instances that do satisfy a standard assumption made to achieve sub-linear  $\mathbb{R}_T^*$ , QRM2, too, achieves sub-linear  $\mathbb{R}_T^*$ , in spite of being designed to minimise  $\mathbb{R}_T(\rho)$  for  $\rho > 0$ . We proceed to discuss this connection between  $\mathbb{R}_T(\rho)$  and  $\mathbb{R}_T^*$ .

### 3.3 Quantile-regret and conventional regret

Given a problem instance  $((\mathcal{A}, M), P_{\mathcal{A}})$ , we first show that minimising  $\mathbb{R}_T^*$  is sufficient to minimise  $\mathbb{R}_T(\rho)$  for all  $\rho > 0$ , but the converse is not true.

**Lemma 3.4.** *For any algorithm and input problem instance, if  $\mathbb{R}_T^* \in o(T)$ , then it must hold that  $\mathbb{R}_T(\rho) \in o(T)$ , for all  $\rho > 0$ . However, the converse is not true.*

*Proof.* For the first part, (1) is written as  $\mathbb{R}_T^* = T \cdot (\mu^* - \mu_\rho) + T \cdot \mu_\rho - \sum_{t=1}^T \mathbb{E}[\mu_t] = T \cdot (\mu^* - \mu_\rho) + \mathbb{R}_T(\rho)$ . Hence,  $\mathbb{R}_T^* \in o(T) \implies \mathbb{R}_T(\rho) \in o(T)$ , for all  $\rho \in [0, 1]$ .

The second part is obtained by considering an infinitely-many armed bandit instance with finitely many optimal arms. Formally, take  $|\mathcal{A}| = \infty$  and  $P_{\mathcal{A}}$  to be the uniform distribution over  $\mathcal{A}$ . Let  $S \subset \mathcal{A}$ , such that  $\forall a \in S, \mu_a = \mu^*$ ,  $|S| < \infty$ , and  $\forall a \in \mathcal{A} \setminus S: \mu_a = \bar{\mu} < \mu^*$ . Now,  $S$  being finite, with probability 1, no arm from  $S$  will be picked by  $P_{\mathcal{A}}$ . Therefore, for  $\rho > 0$ ,  $\mu_\rho = \bar{\mu}$ , and so  $\mathbb{R}_T^* = T \cdot (\mu^* - \bar{\mu}) + \mathbb{R}_T(\rho) \geq T \cdot (\mu^* - \bar{\mu}) \in \Omega(T)$ .  $\square$

Although in general, achieving sub-linear  $\rho$ -regret does not imply achieving sub-linear regret, this turns out asymptotically true for QRM2 on the family of bandit instances considered by Wang et al. (2008), of which the family of instances considered by Berry et al. (1997) is a subset. In these instances, the distribution of the mean reward  $\mu$ , denoted  $D(\mu)$ , has the “reservoir” property, as detailed below.

**Proposition 3.5 (Case Study).** *QRM2 achieves  $\mathbb{R}_T^* \in o(T)$  as  $T \rightarrow \infty$ , under the assumption that  $\Pr_{\mu \sim D(\mu)}\{\mu > \mu^* - \epsilon\} = \Theta(\epsilon^\nu)$ , for  $\epsilon \rightarrow 0$ , where  $\nu$  is a positive constant.*

*Proof.* The assumption amounts to the existence  $\rho_0 \in (0, 1]$  such that for  $0 < \rho < \rho_0$ ,  $c_l(\mu^* - \mu_\rho)^\nu \leq \rho \leq c_u(\mu^* - \mu_\rho)^\nu$ ,

where  $c_u, c_l$  are positive constants. Defining  $h(\rho) \stackrel{\text{def}}{=} \mu^* - \mu_\rho$ , we see that for  $\rho \in (0, \rho_0]$ :  $h(\rho) \leq (\rho/c_l)^{1/\nu}$ .

We know from Theorem 3.3 that for any given  $\rho > 0$ , and for a sufficiently large horizon  $T$ , QRM2 achieves  $\mathbb{R}_T(\rho) \in o(T)$ . Equivalently, for every sufficiently large  $T$ , there exists a  $\rho(T) \in (0, 1]$  such that, for all  $\rho \geq \rho(T)$ , QRM2 achieves  $\mathbb{R}_T(\rho) \in o(T)$ . We also notice that  $\rho(T)$  is a monotonic non-increasing sequence that converges to 0. Hence, there exists a sufficiently large horizon  $T_0$  such that for all  $T \geq T_0$ ,  $\rho(T) \leq \rho_0$ . In other words, for horizon  $T > T_0$ ,  $h(\rho(T)) \leq (\rho(T)/c_l)^{1/\nu}$ . Since  $\lim_{T \rightarrow \infty} (\rho(T)/c_l)^{1/\nu} = 0$  and  $h(\rho(T)) \geq 0$ , we get  $\lim_{T \rightarrow \infty} h(\rho(T)) = 0$ . Since  $\mathbb{R}_T^* = T \cdot h(\rho(T)) + \mathbb{R}_T(\rho)$ , we get  $\lim_{T \rightarrow \infty} \frac{\mathbb{R}_T^*}{T} = \lim_{T \rightarrow \infty} \left( h(\rho(T)) + \frac{\mathbb{R}_T(\rho(T))}{T} \right) = 0$ , which means that  $\mathbb{R}_T^*$  is asymptotically  $o(T)$ .  $\square$

## 4 EXPERIMENTS AND RESULTS

In this section, we compare QRM2 with competing approaches for regret minimisation on infinitely many-armed bandits. We consider representative algorithms from the categories described in Section 2, and investigate the effect of their parameters, in tandem with a comparison with QRM2. Although QRM2 is designed to minimise  $\rho$ -regret for progressively decreasing  $\rho$  values, we use conventional regret as the evaluation metric in all our experiments. This choice essentially amounts to evaluating the total reward accrued over a given horizon, which is perhaps the most relevant measure in practice. In all our experiments, the reward distributions of arms are Bernoulli. Note that both ZOOMING (Kleinberg et al., 2008) and QRM2 proceed through phases, progressively doubling the phase length. To improve sample efficiency, we retain the statistics of pulls from previous phases and correspondingly adjust the “budget” term in the confidence bound.

### 4.1 Comparison with ZOOMING

The ZOOMING algorithm (Kleinberg et al., 2008) works on a bandit instance comprising a set of arms  $\mathcal{A} = [0, 1]$ , with the expected mean reward  $\mu_{(\cdot)}$  being Lipschitz-continuous over  $\mathcal{A}$ . The metric defined on  $\mathcal{A}$  is: for  $x, y \in \mathcal{A}$ ,  $L_d(x, y) = |x - y|^{1/d}$ , where  $d \geq 1$  is a known, user-specified parameter. For a given horizon  $T$ , ZOOMING is shown to incur a regret of  $\tilde{O}(T^{(d+1)/(d+2)})$ . The algorithm proceeds by maintaining confidence bounds on the mean rewards of contiguous regions of  $\mathcal{A}$ ; a new region is created whenever the existing ones fail to cover some portion of  $\mathcal{A}$ . In our implementation, a new region is created by picking an uncovered region uniformly at random. Its “centre” is picked uniformly at random from the points it contains.

We compare QRM2 with ZOOMING on four problem instances, shown in Figure 2 and specified in Appendix-B. On each instance we compare the cumulative regret of QRM2

with that of ZOOMING for  $d \in \{1, 2\}$ . The results at different horizons are presented in Figure 3. Foremost, observe that the performance of ZOOMING is fairly sensitive to  $d$ : on instances I-P and I-S, the variant with  $d = 1$  performs noticeably better; on instances I-N and I-W, the variant with  $d = 2$  is superior. On the instance I-N the better of these two variants performs close to QRM2. On an unknown problem instance, it is unrealistic to expect that the user will be able to guess a good value for  $d$  beforehand.

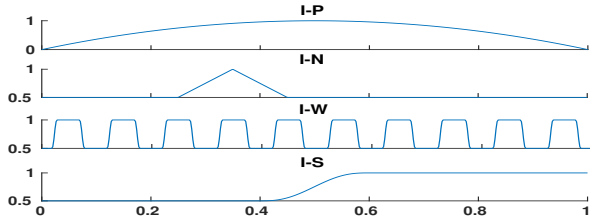


Figure 2: Four problem instances (fully specified in Appendix-B). In all four cases,  $\mathcal{A} = [0, 1]$ , as shown on the x axis. The y axis shows the mean reward  $\mu_a$  for  $a \in \mathcal{A}$ . QRM2 takes  $P_{\mathcal{A}}$  to be the uniform distribution over  $\mathcal{A}$ .

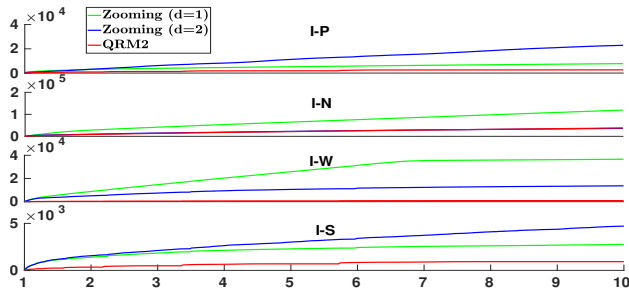


Figure 3: Cumulative regret (y axis) incurred by ZOOMING and QRM2 on the instances in Figure 2. The x axis shows the horizon /  $10^5$ . Each plot is an average of 100 runs.

A second limitation in practice arises from the quality of the features used to generalise across  $\mathcal{A}$ , which effectively determine the Lipschitz constant of the mean reward function. Instances I-W and I-S have exactly the same mean distribution  $D$ —they only differ in the indexing of the arms, which, in practice, would depend on the feature representation. The regret of ZOOMING on these instances (whether with  $d = 1$  or with  $d = 2$ ) varies at least three-fold. By ignoring the arm indices and the metric, QRM2 registers exactly the same regret on both instances.

## 4.2 Comparison with Gaussian Process algorithms

In our next set of experiments, we compare QRM2 with IGP-UCB and GP-TS (Ray Chowdhury and Gopalan, 2017). Our experiments are run on the light sensor data

set<sup>2</sup>, on which the authors have themselves benchmarked their methods. We refer the reader to the original paper for a description of the data set (Ray Chowdhury and Gopalan, 2017, see Section 6), which encodes bandit instances with arms corresponding to sensors. We run IGP-UCB and GP-TS with the same parameters used by the authors.

From Table 1, we find that GP-TS outperforms IGP-UCB, exactly as reported by Ray Chowdhury and Gopalan (2017). However, QRM2 outperforms both GP-TS and IGP-UCB by a large margin. We posit as one reason for the efficiency of QRM2 its use of MOSS as the underlying regret minimisation procedure. On the other hand, using Gaussian Processes to generalise over the space of arms would only work well if nearby arms indeed have similar mean rewards. Without good generalisation, the confidence bounds resulting from Gaussian Processes are likely to be loose, and therefore a poor guide for exploration.

Table 1: Cumulative regret after  $10^6$  pulls, averaged over 192 test instances, with one standard error.

Algorithm	Average cumulative regret
GP-TS	$2.58 \times 10^4 \pm 36.75 \times 10^2$
IGP-UCB	$3.86 \times 10^5 \pm 18.05 \times 10^4$
QRM2	$0.14 \times 10^4 \pm 0.13 \times 10^2$

## 4.3 Comparison with algorithm of Wang et al. (2008)

We compare QRM2 with the algorithm of Wang et al. (2008) for unspecified horizons. Recall that the sub-linear regret bounds shown by Wang et al. (2008) are based on the assumption that the mean distribution is a “reservoir”: that is,  $\Pr_{\mu_a \sim P_{\mathcal{A}}} \{\mu_a > \mu^* - \epsilon\} = \Theta(\epsilon^\nu)$ , for  $\epsilon \rightarrow 0$ . We notice that for  $\mu^* \in (0, 1]$  and  $\nu > 0$ ,  $f(\mu) = \frac{\nu}{\mu^{*\nu}}(\mu^* - \mu)^{\nu-1}$  is a density function of some reservoir distribution. This follows from the fact that CDF of  $f(\mu)$  is given by  $F(\mu) = 1 - \frac{1}{\mu^{*\nu}}(\mu^* - \mu)^\nu$ . Therefore  $\Pr_{\mu \sim f(\mu)} \{\mu > \mu^* - \epsilon\} = 1 - F(\mu) \in \Theta(\epsilon^\nu)$ . It is worth noting that for  $\nu = 1$ ,  $f(\mu)$  is the uniform distribution.

The any-time algorithm given by Wang et al. (2008) requires three parameters: (1) an exploration rate  $\xi_t$ , for the  $t$ -th pull, such that  $2 \ln(10 \ln t) \leq \xi_t \leq \ln t$ , (2) the shape parameter  $\nu$ , and (3) whether  $\mu^* = 1$ . We refer the reader to the original paper (Wang et al., 2008, see Section 2) for a full specification. We test this algorithm along with QRM2 on four problem instances, shown in Table 2. In all cases,  $\mathcal{A} = [0, 1]$ , and we take  $P_{\mathcal{A}}$  as the uniform distribution over  $\mathcal{A}$ . Each problem instance is such that its optimal mean  $\mu^*$  is either 1 or 0.6, and its reward distribution  $D(\mu)$  is either  $\beta(0.5, 2)$  or  $\beta(1, 1)$  (scaled to have support in  $[0, \mu^*]$ ). The algorithm of Wang et al. (2008)’s needs to be supplied  $\nu$  such that the complementary cumulative distribution func-

<sup>2</sup> [www.cs.cmu.edu/~gueztrin/Class/10708-F08/projects/lightsensor.zip](http://www.cs.cmu.edu/~gueztrin/Class/10708-F08/projects/lightsensor.zip)

tion (CCDF) of  $f(\mu)$  will overestimate that of  $D(\mu)$  beyond some  $\mu_0 < \mu^*$ . Also, as the algorithm needs to know whether or not  $\mu^* = 1$ , we supply a representative value  $\mu^\#$  for  $\mu^*$ . Table 2 explicitly shows the parameterisation used for the different instances, and Figure 4 depicts the CCDF of the corresponding  $D(\mu)$ , and that of  $f(\mu)$  for different values of  $\nu$ . In practice, an experimenter might not have a precise estimate of  $(\nu, \mu^*)$ , and hence the values supplied might not meet the above criteria. In Table 2, depending on whether or not the values of  $\nu, \mu^\#$  respect the criteria, the corresponding cells are marked by  $\checkmark$  and  $\times$ , respectively. Also, the values of  $\nu, \mu^\#$  (from our set) for which the CCDF of  $f(\mu)$  coincides with or fits  $D(\mu)$  most closely are marked Exact and Closest, respectively.

Table 2: Summary of problem instances used in Section 4.3, along with different parameterisations of the algorithm of Wang et al. (2008). For explanations see Section 4.3.

Instances	$\mu^*$	$D(\mu) = \beta(a, b)$		$\mu^\#$		$\nu$		
		$a$	$b$	1.0	0.6	0.4	1	2
I-1	1	0.5	2	$\checkmark$	$\times$	$\checkmark$	$\checkmark$	$\checkmark$ Closest
I-2	1	1	1	$\checkmark$	$\times$	$\checkmark$	$\checkmark$ Exact	$\times$
I-3	0.6	0.5	2	$\times$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$ Closest
I-4	0.6	1	1	$\times$	$\checkmark$	$\checkmark$	$\checkmark$ Exact	$\times$

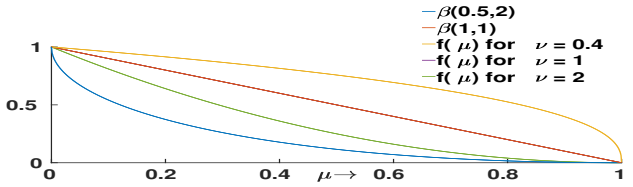


Figure 4: Complementary CDF (CCDF) values (y axis) for various distributions. The CCDF for  $\beta(1, 1)$  coincides with that of  $f(\mu)$  for  $\nu = 1$ .

For a horizon of  $10^6$ , QRM2, which is parameter-free, explores a fixed number of arms (= 94). On the other hand, Wang et al. ’s algorithm explores  $10^4$  arms for  $\nu = 2$ . For  $\nu = 0.4$  it explores 16 and 52 arms for  $\mu^\# = 0.6$  and  $\mu^\# = 1$ , respectively. Figure 5 shows that in spite of providing values of  $(\nu, \mu^\#)$  that closely (or even exactly) track  $D(\mu)$ , QRM2 outperforms Wang et al. ’s algorithm by a significant margin. We note that optimistic values of these parameters helps their result improve, but incorrect parameterisation severely degrades performance. Another non-trivial factor in the performance of their algorithm is the exploration rate  $\xi_t$ . While varying  $\xi_t$  within their prescribed range keeps the regret upper bound unaffected, in practice it is observed to have a significant effect on regret. In Algorithm 2, we set the  $\alpha$  parameter of QRM2 to 0.347 to optimise a *theoretical* bound. In practice, tuning  $\alpha$  for different problem instances further improves QRM2’s performance. In line with our intent to not depend on tuning, we refrain from reporting these optimised results.

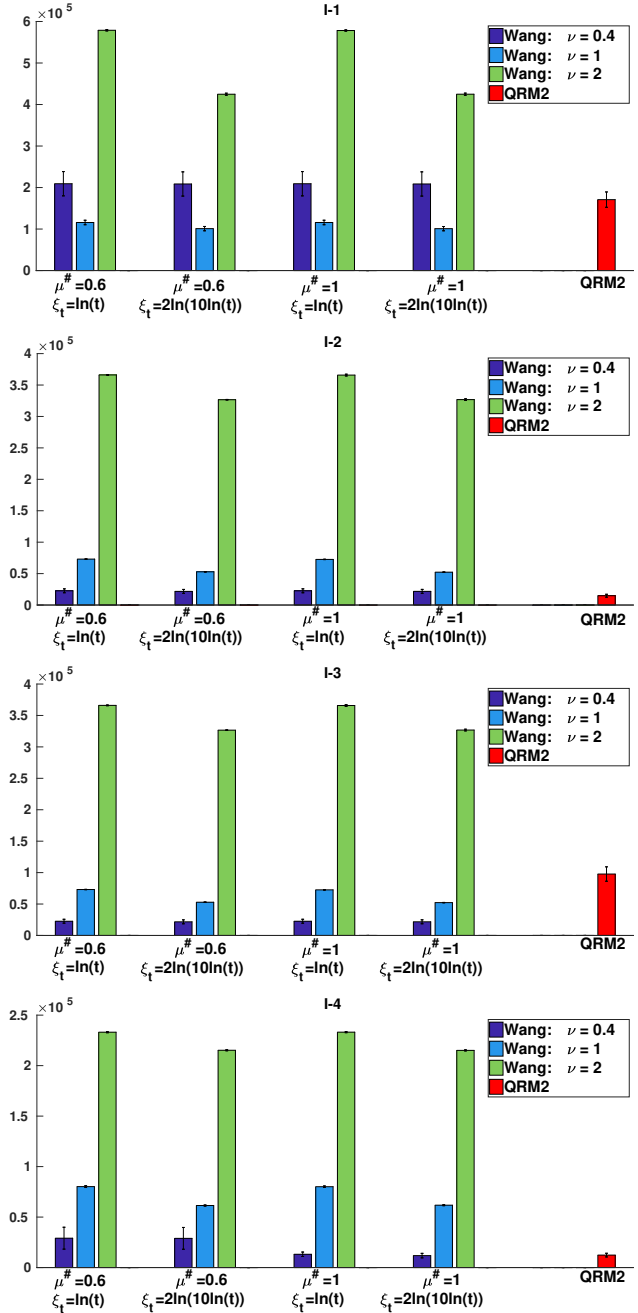


Figure 5: Cumulative regret incurred by QRM2 and the algorithm of Wang et al. (2008) after  $10^6$  pulls on the instances in Table 2. Each bar is an average of 20 runs, and shows one standard error bar. The accompanying parameters are explained in Section 4.3.

#### 4.4 Comparison with constant-memory algorithms

Recall that the algorithms of Herschkorn et al. (1996) and Berry et al. (1997) keep the statistics of only a single arm (or two) in memory. They are specifically designed for bandit instances that yield Bernoulli rewards. The “Non-stationary” algorithm of Herschkorn et al. (1996) repeatedly pulls the  $i$ -th arm, for  $i = 1, 2, \dots$ , until it produces  $i$  consecutive failures—at which point a new arm is pulled. Berry et al. (1997) propose three strategies for problem instances in which the distribution of means is uniform over  $[0, \mu^*]$  for some  $\mu^* \in [0, 1]$ . These strategies assume that the horizon  $T$  is given. For example, the “ $\sqrt{T}$ -run” switches out the current arm upon a single failure, unless the arm produces  $\sqrt{T}$  successes (in which case it is pulled for the remaining horizon). If  $\sqrt{T}$  arms have been pulled and discarded, the arm with the highest observed empirical mean thus far is pulled for the remainder of the run. The “ $\sqrt{T} \ln T$ -learning” strategy and the “Nonrecalling  $\sqrt{T}$ -run” are variants built around a similar theme; we refer the reader to the original paper (Berry et al., 1997) for precise specifications. Table 3 presents a comparison of incurred cumulative regret on the instances I-1, I-2, I-3 and I-4. On I-1, QRM2 outperforms all the other strategies by a significant margin. This result is not surprising, since (1) QRM2 uses additional memory (94 arms for a horizon of  $10^6$ ), and (2) unlike the strategies of Berry et al. (1997), it does not assume that the mean rewards are uniformly distributed. On I-2, which indeed has uniformly-distributed means, the Nonrecalling  $\sqrt{T}$ -run strategy of Berry *et al.* performs marginally better than QRM2. However, this win comes at the expense of incurring very high regret on I-1, in which near-optimal arms are less likely to be encountered. Interestingly, on I-3 and I-4, the Non-stationary policy of Herschkorn et al. (1996) policy outperforms all three from Berry et al. (1997). Yet, all these algorithms are outperformed by QRM2.

Table 3: Cumulative regret ( $/10^5$ ) of QRM2 and strategies proposed by Herschkorn et al. (1996) and Berry et al. (1997) after  $10^6$  pulls, on instances I-1, I-2, I-3 and I-4. Each result is the average of 20 runs, showing one standard error.

Algorithms	I-1	I-2	I-3	I-4
Non-stationary Policy (Herschkorn et al., 1996)	3.58 $\pm$ 0.4	1.11 $\pm$ 0.2	1.64 $\pm$ 0.2	0.79 $\pm$ 0.1
$\sqrt{T}$ -run (Berry et al., 1997)	6.18 $\pm$ 0.5	1.11 $\pm$ 0.4	4.18 $\pm$ 0.3	2.03 $\pm$ 0.3
$\sqrt{T} \ln T$ -learning (Berry et al., 1997)	6.32 $\pm$ 0.4	0.69 $\pm$ 0.3	4.38 $\pm$ 0.2	2.15 $\pm$ 0.3
Nonrecalling $\sqrt{T}$ -run (Berry et al., 1997)	5.35 $\pm$ 0.5	<b>0.03</b> $\pm$ 0.004	4.56 $\pm$ 0.001	2.55 $\pm$ 0.001
QRM2	<b>1.71</b> $\pm$ 0.2	0.15 $\pm$ 0.02	<b>0.98</b> $\pm$ 0.1	<b>0.12</b> $\pm$ 0.01

## 5 CONCLUSION

In this paper, we present an approach to manage the explore-exploit trade-off in bandit instances that contain many more arms than the possible number of experiments. While most

existing approaches in this setting assume special properties of the arms’ reward function or some structure over the set of arms, we make no such assumptions. Rather, we reformulate the problem by introducing the notion of quantile regret (or  $\rho$ -regret), which is defined with respect to the  $(1 - \rho)$ -th quantile of the mean reward distribution—unlike conventional regret, which is defined with respect to the highest mean. We present sub-linear upper bounds on the  $\rho$ -regret when (1)  $\rho$  is specified to the algorithm, and (2)  $\rho$  is not specified to the algorithm. We also prove that our QRM2 algorithm, although it is designed to minimise  $\rho$ -regret for small  $\rho$ , indeed achieves sub-linear regret under the assumption that the instance’s mean rewards come from a reservoir distribution (Wang et al., 2008).

We provide extensive empirical justification for quantile-regret minimisation. Our experiments show that the ZOOMING algorithm (Kleinberg et al., 2008) is sensitive to the given metric and the Lipschitz-continuity of the reward function. With slight perturbations to its parameters, the algorithm incurs a significantly higher regret. The GP-TS, and IGP-UCB algorithms (Ray Chowdhury and Gopalan, 2017) do not explicitly specify the number of arms to explore. Both algorithms perform much worse than QRM2 on the light sensor problem. We find that even when specified the exact distributional parameter, the algorithm proposed by Wang et al. (2008) can incur a higher regret than QRM2. It is infeasible in practice to know the optimal parameter setting for a given problem instance, and it is undesirable to have to find the right parameters using techniques such as cross-validation. The parameter-free approach taken by QRM2 makes it especially appealing to implement as a baseline across different domains and problem instances.

The constant-memory policies proposed by Herschkorn et al. (1996) and Berry et al. (1997) are akin to QRM2 in being simple and parameter-free. On the theoretical side, it seems plausible that their dependence on uniformly-distributed rewards can be removed in lieu of providing a finite time upper bound on the quantile-regret (rather than asymptotic guarantees). Practically, it also seems appealing to generalise these methods to work with larger, even if constant, memory sizes. In future work, we plan to analyse constant-memory policies within the framework of quantile-regret minimisation. We also aim to examine possible improvements to both the upper and lower bounds presented in this paper.

## Acknowledgements

We thank Sayak Ray Chowdhury for sharing code and providing useful guidance. SK was partially supported by SERB grant ECR/2017/002479.

## References

- Rajeev Agrawal. The continuum-armed bandit problem. *SIAM J. Control Optim.*, 33(6):1926–1951, 1995.
- Jean-Yves Audibert and Sébastien Bubeck. Minimax policies for adversarial and stochastic bandits. In *Proc. COLT 2009*, pages 217–226, 2009. URL <https://hal-enpc.archives-ouvertes.fr/hal-00834882/file/COLT09a.pdf>.
- Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *J. Mach. Learn. Res.*, 3:397–422, 2003.
- Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77, 2003.
- Peter Auer, Ronald Ortner, and Csaba Szepesvári. Improved rates for the stochastic continuum-armed bandit problem. In *Proc. COLT 2007*, pages 454–468. Springer, 2007.
- D.A. Berry and B. Fristedt. *Bandit Problems: Sequential Allocation of Experiments*. Chapman & Hall, 1985.
- Donald A. Berry, Robert W. Chen, Alan Zame, David C. Heath, and Larry A. Shepp. Bandit problems with infinitely many arms. *The Annals of Stat.*, 25(5):2103–2116, 1997.
- Thomas Bonald and Alexandre Proutiere. Two-target algorithms for infinite-armed bandits with Bernoulli rewards. In *Adv. NIPS 26*, pages 2184–2192. Curran Associates, Inc., 2013.
- Alexandra Carpentier and Michal Valko. Simple regret for infinitely many armed bandits. In *Proc. ICML 2015*, pages 1133–1141. JMLR, 2015.
- Wei Chu, Lihong Li, Lev Reyzin, and Robert Schapire. Contextual bandits with linear payoff functions. In *Proc. AISTATS 2011*, volume 15, pages 208–214. PMLR, 2011.
- David S. Ebert, F. Kenton Musgrave, Darwyn Peachey, Ken Perlin, and Steven Worley. *Texturing and Modeling: A Procedural Approach*. Morgan Kaufmann publishers Inc., 3rd edition, 2002.
- Avishek Ghosh, Sayak Ray Chowdhury, and Aditya Gopalan. Misspecified linear bandits. In *Proc. AAAI 2017*, pages 3761–3767. AAAI Press, 2017.
- Sergiu Goschin, Ari Weinstein, Michael L. Littman, and Erick Chastain. Planning in reward-rich domains via PAC bandits. In *Proc. EWRL 2012*, volume 24, pages 25–42. JMLR, 2012.
- Stephen J. Herschkorn, Erol Pekz, and Sheldon M. Ross. Policies without memory for the infinite-armed Bernoulli bandit under the average-reward criterion. *Prob. in the Engg. and Info. Sc.*, 10(1):21–28, 1996.
- Yoshiaki Kadono and Naoki Fukuta. Lakube: An improved multi-armed bandit algorithm for strongly budget-constrained conditions on collecting large-scale sensor network data. In *PRICAI 2014: Trends in Artificial Intelligence*, pages 1089–1095. Springer International Publishing, 2014.
- Robert Kleinberg. Nearly tight bounds for the continuum-armed bandit problem. In *Adv. NIPS 17*, pages 697–704. MIT Press, 2005.
- Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. Multi-armed bandits in metric spaces. In *Proc. STOC 2008*, pages 681–690. ACM, 2008.
- Haifang Li and Yingce Xia. Infinitely many-armed bandits with budget constraints. In *Proc. AAAI 2017*, pages 2182–2188. AAAI Press, 2017.
- Sayak Ray Chowdhury and Aditya Gopalan. On kernelized multi-armed bandits. In *Proc. ICML 2017*, volume 70, pages 844–853. PMLR, 2017.
- Arghya Roy Chaudhuri and Shivaram Kalyanakrishnan. PAC identification of a bandit arm relative to a reward quantile. In *Proc. AAAI 2017*, pages 1977–1985. AAAI Press, 2017.
- Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias W. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proc. ICML 2010*, pages 1015–1022. Omnipress, 2010.
- Liang Tang, Romer Rosales, Ajit Singh, and Deepak Agarwal. Automatic ad format selection via contextual bandits. In *Proc. CIKM 2013*, pages 1587–1594. ACM, 2013.
- Long Tran-Thanh, Sebastian Stein, Alex Rogers, and Nicholas R. Jennings. Efficient crowdsourcing of unknown experts using bounded multi-armed bandits. *Artif. Intl.*, 214:89–111, 2014.
- Michal Valko, Nathan Korda, Rémi Munos, Ilias Flaounas, and Nello Cristianini. Finite-time analysis of kernelised contextual bandits. In *Proc. UAI 2013*, pages 654–663. AUAI Press, 2013.
- Erli Wang, Hanna Kurniawati, and Dirk P. Kroese. CEMAB: A cross-entropy-based method for large-scale multi-armed bandits. In *Artif. Life and Computnl. Intl.*, pages 353–365. Springer Intl. Publishing, 2017.
- Yizao Wang, Jean-Yves Audibert, and Rémi Munos. Algorithms for infinitely many-armed bandits. In *Adv. NIPS 21*, pages 1729–1736. Curran Associates Inc., 2008.

---

# Variational Inference for Gaussian Process Models for Survival Analysis

---

**Minyoung Kim\***

Dept. of Electronic Engineering  
Seoul Nat'l Univ. of Science & Technology  
Seoul, Korea

**Vladimir Pavlovic**

Dept. of Computer Science  
Rutgers University  
Piscataway, NJ 08854, USA

## Abstract

Gaussian process survival analysis model (GPSAM) was recently proposed to address key deficiencies of the Cox proportional hazard model, namely the need to account for uncertainty in the hazard function modeling while, at the same time, relaxing the time-covariates factorized assumption of the Cox model. However, the existing MCMC inference algorithms for GPSAM have proven to be slow in practice. In this paper we propose novel and scalable variational inference algorithms for GPSAM that reduce the time complexity of the sampling approaches and improve scalability to large datasets. We accomplish this by employing two effective strategies in scalable GP: i) using pseudo inputs and ii) approximation via random feature expansions. In both setups, we derive the full and partial likelihood formulations, typically considered in survival analysis settings. The proposed approaches are evaluated on two clinical and a divorce-marriage benchmark datasets, where we demonstrate improvements in prediction accuracy over the existing survival analysis methods, while reducing the complexity of inference compared to the recent state-of-the-art MCMC-based algorithms.

## 1 INTRODUCTION

Survival analysis studies statistical dependencies between the time to certain event and the covariates associated with this event. This is an important problem in statistics with applications ranging from medical prognosis in clinical studies (e.g., estimating the time of cancer

recurrence or remission from leukemia based on demographic and individual medical record factors), to other general areas where we seek to predict failure times of a system (e.g., bankruptcy of a firm). The key task in survival analysis is to estimate the conditional density function of the event time  $t$  given the covariates  $\mathbf{x}$ , from which one can immediately derive several important prognostic measures. Two most common instances of such measures are the *survival function*, defined as  $P(T \geq t|\mathbf{x})$ , or the *prognostic index*  $u(\mathbf{x})$  that quantifies the overall (anti) risk (i.e., higher index implies longer survival, and vice versa) of a patient/system with covariates  $\mathbf{x}$ .

A typical data-driven setting for survival analysis assumes availability of time-covariate pairs  $\{(t, \mathbf{x})\}$ , suggesting standard regression problem framing. However, a notable characteristic here is that some of the observed times  $t$  are *censored* in the sense that we only know the actual event time is no earlier than the observed  $t$ . In clinical studies this typically happens when the patient exits the study or the study terminates before the event occurs<sup>1</sup>. Typically, the data provides the information as to whether or not each instance is censored: the event indicator variables  $\delta = 1$  for event, and 0 for censored examples. Therefore, applying standard regression approaches by simply ignoring the censored samples can result in suboptimal use of data.

Several approaches have been proposed to deal with the censored instances. One way is to incorporate a cost-sensitive loss within the regression or ranking framework to learn the prognostic index function  $u(\mathbf{x})$  (Shivaswamy et al., 2007; Khan & Zubek, 2008; Van Belle et al., 2009; Van Belle et al., 2011). The main idea here is to impose

---

<sup>1</sup>This type of censoring is often referred to as the *right-censoring*. *Left-censoring* applies to instances when the event time is never greater than the observed time, while the *interval-censoring* refers to the cases of observed events within an interval. Most cases in practice deal with right-censoring, which we restrict to in this paper.

---

\*Also affiliated with Rutgers University.

an asymmetric loss on the incorrect prediction for censored examples. That is, we penalize the over-estimates (i.e.,  $u(\mathbf{x}) > t$ ) less than the under-estimates ( $u(\mathbf{x}) < t$ ). However, these approaches focus on the prognostic index function directly and are, therefore, inherently unable to provide a measure of uncertainty, namely the distribution of the survival time  $t$  for a given input  $\mathbf{x}$ .

The conditional density  $P(t|\mathbf{x})$  is most commonly modeled using the Cox Proportional Hazard (CoxPH) model (Cox, 1972). The model represents the distribution of the survival time as a first event arrival time in a heterogeneous Poisson process. Unlike the standard Poisson process models, the intensity function (often referred to as the *hazard* function) has a dependency on the input covariates, denoted as  $\lambda(t|\mathbf{x})$ . The CoxPH model further factorizes the hazard function over  $t$  and  $\mathbf{x}$  (see (2) in Sec. 2), allowing simplicity in hazard modeling by separating the input-dependent risk factors from the time-varying effects.

Recent efforts have focused on extending the CoxPH model to address its two drawbacks: i) the proportional and non-crossing hazard rates across instances originating from the factorized form of the hazard function can be too restrictive and oftentimes unlikely, and ii) the lack of proper treatment of uncertainty in the hazard function. (Dempsey et al., 2017) extended the model by introducing latent, continuous-time Markov dynamics to address the former limitation. The latter issue can be resolved by imposing Bayesian priors on the hazard function. Although few approaches along this direction showed initial success (Hjort et al., 2010; Iorio et al., 2009; Martino et al., 2011), they either have practical limitations (e.g., how to incorporate expert knowledge) or fail to overcome the former assumption of the proportional hazard rates. Another related method is the Random Survival Forest (Ishwaran et al., 2008), which can be seen as a generalization of the Kaplan Meier method, the traditional nonparametric hazard function estimator. Recently the Deep Survival Analysis (DSA) method was proposed (Ranganath et al., 2016), which utilizes a deep hierarchical Bayesian model for survival analysis.

To address those limitations, Gaussian process survival analysis model (GPSAM) was proposed in (Fernández et al., 2016). A GP-prioried latent function on the joint input space  $(t, \mathbf{x})$ , coupled with a non-negative link function, introduces stochasticity and removes the factorization assumption of CoxPH. The key advantage is that the Gaussian process circumvents the difficulty of modeling the hazard dependent jointly on  $(t, \mathbf{x})$  through the use of the covariance (kernel) function (Rasmussen & Williams, 2006). In essence, the GPSAM supplements the proportional hazard models with additional flexibil-

ity, while being able to account for uncertainty in the hazard function.

Nevertheless, the inference in the GPSAM model is challenging because the likelihood depends on the latent function values *for an uncountable range* of time inputs  $t \in \mathbb{R}_+$  (Sec. 2.2 for details), and not limited to only those induced by data in standard GP models. Motivated by the sophisticated MCMC inference algorithm for GP-prioried Poisson event models (Adams et al., 2009), the authors in (Fernández et al., 2016) proposed a tractable MCMC dynamics for the GPSAM by exploiting the idea of thinning-based sampling with auxiliary variables. However, the MCMC inference algorithm often exhibits slow convergence. Despite adopting the random feature kernel approximation strategy (Rahimi & Recht, 2008) to circumvent the computationally intensive matrix inversions, the MCMC inference for GPSAM proposed in (Fernández et al., 2016) incurs considerable computational issues when applied to real applications.

In this paper, we propose two novel variational inference algorithms for GPSAM, which address the computational deficiencies of the MCMC approach. To tackle the scalability of the GP nonparametric inference, we incorporate two approximations: the pseudo-input treatment (Titsias, 2009) and the random feature expansion (Rahimi & Recht, 2008). The former approach is similar to (Lloyd et al., 2015) variational inference in the GP modulated Poisson process. However, our approach is different in that we consider the GP latent function in the joint input space within the survival analysis setup. Solutions to variational inference in both approaches admit analytic forms aside from the fast univariate Monte-Carlo estimation of expected log-likelihood. We empirically demonstrate superior performance of our proposed methods over existing survival analysis approaches on several synthetic and real benchmark datasets.

## 2 BACKGROUND

In this section we briefly review the CoxPH model with two popular parameter estimation methods. Then we discuss the recent Gaussian process survival analysis model (GPSAM) (Fernández et al., 2016) that addresses the known drawbacks of the CoxPH model.

### 2.1 COX PROPORTIONAL HAZARD MODEL

The CoxPH model (Cox, 1972; Kleinbaum & Klein, 2005) represents the conditional density

$$P(t|\mathbf{x}) = \lambda(t|\mathbf{x}) \cdot \exp\left(-\int_0^t \lambda(\tau|\mathbf{x}) d\tau\right), \quad (1)$$



where  $t \in \mathbb{R}_+$  is the time of the event (e.g., death or cancer recurrence), and  $\mathbf{x} \in \mathbb{R}^d$  is the  $d$ -dim covariates of the subject (e.g. patient’s medical features). In (1),  $\lambda(t|\mathbf{x})$  is referred to as the *hazard* function, and can be interpreted as the probability of the immediate death at  $t$  given that the survival time is at least  $t$ . The hazard function is the *intensity* function of the (inhomogeneous) Poisson process (Ross, 2006) with (1) being the first event time density, however, in survival analysis this intensity is different from subject to subject, determined by the input covariates  $\mathbf{x}$ .

In the CoxPH model, the hazard function is specifically assumed to follow the factorized parametric form:

$$\lambda(t|\mathbf{x}) = \lambda_0(t) \cdot \exp(\mathbf{b}^\top \mathbf{x}), \quad (2)$$

where the model parameters are comprised of the weight vector  $\mathbf{b} \in \mathbb{R}^d$  and the non-negative function  $\lambda_0(\cdot)$ . The latter is known as the *base hazard* function which is typically modeled by the Weibull or a piecewise constant function. The consequence of the factorized form in (2) is that the hazard ratio between two subjects (with  $\mathbf{x}$  and  $\mathbf{x}'$ ) is constant over time, solely dependent on the covariates (i.e.,  $e^{\mathbf{b}^\top(\mathbf{x}-\mathbf{x}')}$ ). Also, the hazard functions of different subjects are non-crossing with each other.

Given the training data  $\mathcal{D} = \{(\delta_n, t_n, \mathbf{x}_n)\}_{n=1}^N$  where  $\delta_n \in \{0, 1\}$  indicates whether the observation  $n$  is event ( $\delta_n = 1$ ) or right-censored ( $\delta_n = 0$ ), the traditional maximum likelihood learning aims to maximize the data log-likelihood  $\sum_{n=1}^N \log FL(n)$  where

$$FL(n) = P(t_n|\mathbf{x}_n)^{\delta_n} \cdot P(T \geq t_n|\mathbf{x}_n)^{1-\delta_n}. \quad (3)$$

Often we name it the *full-likelihood* to differentiate it from the partial likelihood, discussed next.

Alternatively, also very popular in survival analysis, the parameters can be learned by the *partial likelihood* maximization. The notion of the partial likelihood comes from an alternative view of the data generation process. Namely, at a given time  $t$ , we consider a random process of selecting a subject  $\mathbf{x}$  that will face an event at  $t$ , among all survivors at that moment. The likelihood of this is proportional to the hazard value  $\lambda(t|\mathbf{x})$ . More specifically, for each event instance  $n$  ( $\delta_n = 1$ ), we can regard  $(t_n, \mathbf{x}_n)$  as the selected sample among the survivors  $\{(t_j, \mathbf{x}_j) : t_j \geq t_n\}$ , regardless of  $\delta_j$ ’s. The so-called partial likelihood is then defined as:

$$PL(n) = \frac{\lambda(t_n|\mathbf{x}_n)}{\sum_{j:t_j \geq t_n} \lambda(t_n|\mathbf{x}_j)}, \quad (4)$$

and we maximize  $\sum_{n=1}^N \delta_n \log PL(n)$ .

## 2.2 GAUSSIAN PROCESS SURVIVAL MODEL

Abbreviated as GPSAM, the model aims to address the known drawbacks of the CoxPH model discussed in Sec. 1 by endowing more flexibility and accounting for uncertainty in the hazard function. This is done by imposing Gaussian process prior on the hazard function and having the latent function dependent on both  $t$  and  $\mathbf{x}$ . More specifically,

$$\lambda(t|\mathbf{x}) = \lambda_0(t) \cdot g(f(t, \mathbf{x})), \quad f(\cdot) \sim \mathcal{GP}(0, k(\cdot, \cdot)). \quad (5)$$

Here  $g(\cdot)$  is a non-negative link function to prevent the hazard from being negative. In (Fernández et al., 2016), they used the sigmoid  $g(y) = 1/(1 + e^{-y})$ , and the Weibull for the base hazard,  $\lambda_0(t) = c \cdot t^{r-1}$  for  $c > 0, r \geq 1$ , which subsumes the constant functions ( $r = 1$ ). Note that the kernel function operates on the joint input space  $\mathbb{R}_+ \times \mathbb{R}^d$ . (Fernández et al., 2016) adopted the composite kernel

$$k((t, \mathbf{x}), (t', \mathbf{x}')) = \sum_{j=1}^d x(j) x'(j) k_j(t, t'), \quad (6)$$

where  $x(j)$  indicates the  $j$ -th element of  $\mathbf{x}$ . The kernel on time space,  $k_j(t, t')$  is typically chosen as the squared exponential for  $j = 1, \dots, d$ ,

$$k_j(t, t') = s_j^2 \exp(-0.5(t - t')^2 / l_j^2), \quad (7)$$

with the variance and length-scale parameters ( $s_j^2, l_j^2$ ).

The inference in the GPSAM model is in general difficult mainly due to the form of the likelihood (1), in which the latent function  $f(\cdot)$  is involved with *all*  $\tau \in [0, t]$ . In other words, infinitely many Gaussian latent variables need to be dealt with in principle. In (Fernández et al., 2016) they adopted the thinning-based MCMC sampling strategy motivated from (Adams et al., 2009), where the key idea is to sample<sup>2</sup> from the (inhomogeneous) Poisson process with intensity  $\lambda_0(t)$  while keeping all the thinned samples as auxiliary state variables in the inference of the latent variables<sup>3</sup>. For the censored examples  $n$ , we can do the same thing as if  $t_n$ ’s were exact, but remove all terms related to  $t_n$  from the likelihood function.

However, the MCMC algorithm is generally slow to converge. Furthermore, each MCMC step requires the kernel matrix inversion to sample from the conditional Gaussian given both the data and the thinned samples. As the number of thinned samples can be orders of magnitude larger than the data size, they also had to resort

<sup>2</sup>This sampling must be easy since  $\lambda_0(t) = c \cdot t^{r-1}$  admits a closed-form inverse cumulative function.

<sup>3</sup>Note that this thinning-based sampling is valid since they used the sigmoid link  $g(\cdot)$ , namely  $\lambda_0(t)$  is always an upper bound of  $\lambda(t|\mathbf{x})$  in (5).

to the random feature expansion trick (Rahimi & Recht, 2008) to circumvent the matrix inversion. Nevertheless, the thinned samples can grow arbitrarily large, incurring serious computational overhead. This motivates our work of variational inferences in the following section.

### 3 VARIATIONAL INFERENCE

We begin with the full joint model of the GPSAM with the observed data  $\mathcal{D} = \{(\delta_n, t_n, \mathbf{x}_n)\}_{n=1}^N$ :

$$P_\theta(\mathcal{D}, f) = P_{\theta_0}(\mathcal{D}|f) \cdot P_{\theta_k}(f). \quad (8)$$

Here  $\theta = \{\theta_0, \theta_k\}$  indicates the model parameters, where  $\theta_0 = (c, r)$  is the parameters of the Weibull base hazard  $\lambda_0(t) = c \cdot t^{r-1}$ , and  $\theta_k$  denotes all kernel parameters, specifically  $\{(s_j^2, l_j^2)\}$  in (7). The latter determines the Gaussian process prior  $P(f)$ .

The conditional data likelihood  $P(\mathcal{D}|f)$  in (8) can have either of two different forms. If we follow the full likelihood (3), then the log-likelihood can be written as:

$$\log P(\mathcal{D}|f) = \sum_{n=1}^N \left[ \delta_n \cdot \log \lambda(t_n | \mathbf{x}_n) - \int_0^{t_n} \lambda(\tau | \mathbf{x}_n) d\tau \right], \quad (9)$$

with  $\lambda(t|\mathbf{x})$  from (5). See Appendix A in the supplemental material for the detailed derivations. If we adopt the partial likelihood (4) instead, then  $\log P(\mathcal{D}|f)$  becomes:

$$\sum_{n=1}^N \delta_n \cdot \left[ \log \lambda(t_n | \mathbf{x}_n) - \log \sum_{j:t_j \geq t_n} \lambda(t_n | \mathbf{x}_j) \right]. \quad (10)$$

The posterior distribution of the latent function  $P_\theta(f|\mathcal{D})$  is analytically intractable, and we introduce a tractable density family  $Q_\alpha(f)$  with the parameters  $\alpha$ , and search for  $\alpha$  that makes  $Q_\alpha(f)$  as close as possible to the true posterior. In defining the variational density family  $Q(\cdot)$ , it should be noted that we have to deal with infinitely many latent variables from  $f(\cdot)$ . To this end, we adopt two recent scalable variational inference algorithms: the pseudo-input approximation (Titsias, 2009) and the random feature expansion (Rahimi & Recht, 2008). We frame each of the approaches within GPSAM. They are described in the following sections.

In addition, we use the square non-negative link function, i.e.,  $g(y) = y^2$  instead of the sigmoid, for its merit in analytic derivation of the objective especially in Sec. 3.1, similar in nature as (Lloyd et al., 2015). That is, the hazard function given the latent function is determined as:

$$\lambda(t|\mathbf{x}) = \lambda_0(t) \cdot f(t, \mathbf{x})^2. \quad (11)$$

#### 3.1 APPROXIMATION WITH PSEUDO INPUTS

To address the intractability of dealing with  $f(\cdot)$  at  $(\tau, \mathbf{x})$  for all time epochs  $\tau$  in the domain, stemming from the likelihood (1), we first adopt the scalable pseudo-input approximation techniques recently introduced in (Titsias, 2009; Dezfouli & Bonilla, 2015; Lloyd et al., 2015). We essentially assume that there are  $M$  pseudo inputs denoted by  $\mathcal{Z} = \{z_1, \dots, z_M\} \subset \mathbb{R}_+ \times \mathbb{R}^d$  (denoting  $z_i = (\bar{t}_i, \bar{\mathbf{x}}_i)$ ), where  $M$  is typically chosen to be small so that the inversion of  $(M \times M)$  matrices can be done efficiently. The pseudo inputs can be thought of as the representative points for the joint input space (Quiñonero-Candela & Rasmussen, 2005). We choose the pseudo inputs by clustering the points in the pool that is formed by Cartesian product of uniformly sampled times and randomly sampled covariates from data, although they can also be learned from the data itself.

We define the variational density for the posterior as:

$$Q_\alpha(f) = \int Q_\alpha(\mathbf{f}_{\mathcal{Z}}) P(f|\mathbf{f}_{\mathcal{Z}}) d\mathbf{f}_{\mathcal{Z}}. \quad (12)$$

Here we use the vector notation for the latent function: for a set  $\mathcal{A} = \{(\hat{t}_i, \hat{\mathbf{x}}_i)\}_{i=1}^p \subset \mathbb{R}_+ \times \mathbb{R}^d$ , we denote by  $\mathbf{f}_{\mathcal{A}}$  the  $p$ -dim vector of the function values on the inputs  $(\hat{t}_i, \hat{\mathbf{x}}_i) \in \mathcal{A}$ . The central idea that enables scalability and tractability in (12) is that we only model the low-dimensional density  $Q_\alpha(\mathbf{f}_{\mathcal{Z}})$  while all the other function values can be inferred using  $P(f|\mathbf{f}_{\mathcal{Z}})$ , the conditional density derived from the GP prior. We let  $Q_\alpha(\mathbf{f}_{\mathcal{Z}})$  be a Gaussian with diagonal covariance, namely

$$Q_\alpha(\mathbf{f}_{\mathcal{Z}}) = \mathcal{N}(\mathbf{f}_{\mathcal{Z}}; \boldsymbol{\mu}, \boldsymbol{\Sigma}), \quad (13)$$

where  $\alpha = \{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$  with  $(M \times 1)$  mean vector  $\boldsymbol{\mu}$  and the  $(M \times M)$  diagonal covariance matrix  $\boldsymbol{\Sigma}$ .

The variational parameters  $\alpha$  can be found by minimizing the KL divergence between the true posterior and the variational density (12):

$$\text{KL}(Q_\alpha(f) || P_\theta(f|\mathcal{D})) = \log P_\theta(\mathcal{D}) - \mathcal{L}_{PI}(\theta, \alpha), \quad (14)$$

where  $\mathcal{L}_{PI}(\theta, \alpha)$  is defined as:

$$\mathcal{L}_{PI}(\theta, \alpha) = \mathbb{E}_{Q(f)} [\log P(\mathcal{D}|f)] - \text{KL}(Q(\mathbf{f}_{\mathcal{Z}}) || P(\mathbf{f}_{\mathcal{Z}})). \quad (15)$$

Using the non-negativity of the KL divergence in (14), the  $\mathcal{L}_{PI}$  becomes a lower bound of the log-evidence,

$$\log P_\theta(\mathcal{D}) \geq \mathcal{L}_{PI}(\theta, \alpha). \quad (16)$$

Increasing  $\mathcal{L}_{PI}(\theta, \alpha)$  wrt  $\alpha$  renders the variational density closer to the true posterior, whereas improving it wrt the model parameters  $\theta$  can *potentially*<sup>4</sup> improve the data

<sup>4</sup>Similarly as in other standard variational inferences, this does not guarantee to improve the evidence  $\log P(\mathcal{D})$  since the inequality (16) is not tight.

likelihood of the model. Hence, we maximize  $\mathcal{L}_{PI}(\theta, \alpha)$  wrt all parameters to achieve both variational inference and model selection simultaneously.

Next, we provide detailed derivations necessary to evaluate the objective  $\mathcal{L}_{PI}$ . Since the second term in the right hand side of (15) is the straightforward KL divergence between two Gaussians, we focus on the expected conditional likelihood term.

For the two forms of the likelihood, full in (9) and partial in (10), we write the expectation as:

$$\mathbb{E}_{Q(f)}[\log P(\mathcal{D}|f)] = \sum_{n=1}^N (A_n - B_n), \quad (17)$$

where  $A_n$  is the expectation of the log-hazard at data point  $n$ ,

$$A_n = \delta_n \cdot \left( \log \lambda_0(t_n) + \mathbb{E}_{Q(f_n(t_n))}[\log f_n(t_n)^2] \right), \quad (18)$$

with the abbreviation  $f_n(t) = f(t, \mathbf{x}_n)$ . Whereas  $A_n$  is shared by both likelihoods,  $B_n$  is defined differently.

$$B_n^f = \int_0^{t_n} \lambda_0(\tau) \cdot \mathbb{E}_{Q(f_n(\tau))}[f_n(\tau)^2] d\tau, \quad (19)$$

is for the full likelihood, while the following is for the partial likelihood:

$$B_n^p = \delta_n \cdot \mathbb{E}_{Q(f)} \left[ \log \sum_{j:t_j \geq t_n} \lambda_0(t_n) \cdot f_j(t_n)^2 \right]. \quad (20)$$

**Full likelihood.** Note that in (18) and (19) the distributions over which the expectations are taken are univariate Gaussians, specifically from (12),  $Q(f_n(t)) = \mathcal{N}(\tilde{\mu}_n(t), \tilde{\sigma}_n^2(t))$  where

$$\tilde{\mu}_n(t) = k_{(t, \mathbf{x}_n), \mathcal{Z}} \mathbf{K}^{-1} \mu, \quad (21)$$

$$\tilde{\sigma}_n^2(t) = k_{(t, \mathbf{x}_n), (t, \mathbf{x}_n)} - k_{(t, \mathbf{x}_n), \mathcal{Z}} \mathbf{K}^{-1} k_{\mathcal{Z}, (t, \mathbf{x}_n)} + k_{(t, \mathbf{x}_n), \mathcal{Z}} \mathbf{K}^{-1} \Sigma \mathbf{K}^{-1} k_{\mathcal{Z}, (t, \mathbf{x}_n)}. \quad (22)$$

For two time-covariates input sets  $\mathcal{A}_1$  and  $\mathcal{A}_2$ , we denote by  $k_{\mathcal{A}_1, \mathcal{A}_2}$  the  $(|\mathcal{A}_1| \times |\mathcal{A}_2|)$  kernel matrix obtained by applying  $k(\cdot, \cdot)$  on  $(\mathcal{A}_1 \times \mathcal{A}_2)$ . Also,  $\mathbf{K} = k_{\mathcal{Z}, \mathcal{Z}}$  indicates the  $(M \times M)$  kernel matrix on the pseudo inputs  $\mathcal{Z}$ . The expectation of the squared-log term in (18) can be done by the Monte-Carlo estimation. For it is *univariate* sampling, this does not incur any significant computational overhead. As we also need to compute gradients of  $A_n$  wrt the parameters of the sampling distribution  $Q(f_n(t_n))$ , we adopt the re-parametrized Gaussian sampling technique (Kingma & Welling, 2014) to reduce the variance of the estimate. More specifically, we express the random samples from  $Q(f_n(t_n))$ , denoted by

$f_n^{(s)}(t_n)$  for  $s = 1, \dots, S$ , as:

$$f_n^{(s)}(t_n) = \tilde{\mu}_n(t_n) + (\tilde{\sigma}_n^2(t_n))^{1/2} \epsilon_n^{(s)}, \quad (23)$$

where  $\epsilon_n^{(s)} \sim \mathcal{N}(0, 1)$ . The expectation in  $A_n$  is then estimated as:

$$\frac{1}{S} \sum_{s=1}^S \log \left( \tilde{\mu}_n(t_n) + (\tilde{\sigma}_n^2(t_n))^{1/2} \epsilon_n^{(s)} \right)^2. \quad (24)$$

We sample  $\{\epsilon_n^{(s)}\}$  once, and fix them throughout the optimization, which empirically performed better with a lower variance than re-sampled at each iteration. As we separate randomness ( $\epsilon_n^{(s)}$ ) from the parameters to be optimized, the gradient of (24) can be computed straightforwardly while yielding an unbiased estimate of the gradient of the original (18). Optionally, we can further reduce the variance of the estimate by using the Rao-Blackwellization technique (Casella & Robert, 1996).

For  $B_n^f$  in (19), due to the square link function, the inner expectation equals  $\tilde{\mu}_n(\tau)^2 + \tilde{\sigma}_n^2(\tau)$ , which allows the integral to be derived analytically for the composite squared exponential kernel (6) and (7) (c.f. (Lloyd et al., 2015)). However, the analytic derivation has to resort to certain confluent hyper-geometric function which can be numerically unstable. In the experiments, we rather adopt the numerical integration by having uniformly sampled grid points over the time horizon.

**Partial likelihood.** Looking into  $B_n^p$  in (20), the key difference from the above full likelihood derivation stems from the multiple latent variables (i.e.,  $\{f_j(t_n)\}_{j:t_j \geq t_n}$ ) that are dependent on one another, preventing us from taking advantage of the univariate sampling. Since the number of these variables (i.e.,  $|\{j : t_j \geq t_n\}|$ ) can be as large as the entire data set, naively sampling from  $Q(f)$  jointly can yield an estimate with large variance. Instead, we consider the upper bound of  $B_n^p$  (leading to a lower bound on  $\mathcal{L}_{PI}$ ) using the Jensen's inequality. Specifically, from (20),

$$B_n^p \leq \delta_n \cdot \log \sum_{j:t_j \geq t_n} \lambda_0(t_n) \cdot \mathbb{E}_{Q(f_j(t_n))} [f_j(t_n)^2]. \quad (25)$$

The square link allows an analytical expression of the expectation in  $\tilde{B}_n^p$ ,  $\tilde{\mu}_j(t_n)^2 + \tilde{\sigma}_j^2(t_n)$ , which can be evaluated easily using (21) and (22), likewise its gradients. Note that (regardless of whether we use the upper bound or not) the base hazard term  $\log \lambda_0(t_n)$  cancels out with that in (18) in the final objective (17), preventing one from learning  $\lambda_0(t)$ . This is an inherent problem originating in the hazard form (5), where  $\lambda_0(t)$  is shared across examples. To this end, we simply borrow the estimate of  $\lambda_0(t)$  from the full-likelihood learning.

### 3.2 RANDOM FEATURES APPROXIMATION

To deal with uncountably many random variables and their matrix inversions brought about from the nonparametric Bayesian Poisson process GPSAM, we consider the random features expansion as an alternative approximation strategy. The central idea of the random features (Rahimi & Recht, 2008; Cho & Saul, 2009) is to seek a finite dimensional feature vector representation for input such that the inner product on this feature space equals (in expectation) the kernel value. For the composite kernel function (6), its GP-prior latent function  $f(t, \mathbf{x})$  can be approximated by:

$$\hat{f}(t, \mathbf{x}) = \frac{1}{m} \sum_{j=1}^d x(j) \left( \mathbf{a}_j^\top \cos(\omega_j t) + \mathbf{b}_j^\top \sin(\omega_j t) \right), \quad (26)$$

where  $\mathbf{a}_j$ ,  $\mathbf{b}_j$ , and  $\omega_j$  are all  $m$ -dim iid random variables (samples) with  $\mathbf{a}_j, \mathbf{b}_j \sim \mathcal{N}(0, s_j^2 \mathbf{I}_m)$  and  $\omega_j \sim \mathcal{N}(0, \frac{1}{t_j^2} \mathbf{I}_m)$  for  $j = 1, \dots, d$ . Here  $\mathbf{I}_m$  denotes the  $(m \times m)$  identity matrix. We let  $\mathbf{a} = \{\mathbf{a}_j\}_{j=1}^d$  and the others similarly. It can be shown that  $\text{Cov}(\hat{f}(t, \mathbf{x}), \hat{f}(t', \mathbf{x}')) = k((t, \mathbf{x}), (t', \mathbf{x}'))$ , which implies that the finite dimensional random variables  $\{\mathbf{a}, \mathbf{b}, \omega\}$  are sufficient to represent the latent function  $f \sim \mathcal{GP}(0, k(\cdot, \cdot))$ . The parameter  $m$  is the number of random samples to approximate the kernel, which trades off: large  $m$  reduces the approximation error at the cost of computational overhead.

In this treatment, we aim to infer the posterior distribution  $P(\mathbf{a}, \mathbf{b}, \omega | \mathcal{D})$ , and the variational inference reduces to maximizing:

$$\mathcal{L}_{RF}(\theta, \beta) = \mathbb{E}_{Q(\mathbf{a}, \mathbf{b}, \omega)} [\log P(\mathcal{D} | \hat{f})] - \text{KL}(\theta, \beta), \quad (27)$$

where we use the fully factorized variational density,  $Q(\mathbf{a}, \mathbf{b}, \omega) = Q(\mathbf{a})Q(\mathbf{b})Q(\omega)$ , each modeled as a Gaussian,  $Q(\mathbf{a}) = \prod_{j=1}^d \mathcal{N}(\mathbf{a}_j; \boldsymbol{\mu}_j^a, \boldsymbol{\Sigma}_j^a)$  and similarly for the others. Here  $\boldsymbol{\mu}_j^a$  and  $\boldsymbol{\Sigma}_j^a$  are  $(m \times 1)$  mean vector and  $(m \times m)$  diagonal covariance matrix, respectively, and  $\beta$  includes all these variational parameters. The term  $\text{KL}(\theta, \beta)$  denotes the sum of individual KL terms for  $\mathbf{a}$ ,  $\mathbf{b}$ , and  $\omega$ ; for instance, for  $\mathbf{a}$ , we have:  $\text{KL}(Q(\mathbf{a}) || \mathcal{N}(\mathbf{a}; 0, s_j^2 \mathbf{I}_m))$ . As before, these KL terms all involve Gaussians, having analytic forms, easy to evaluate and take derivatives.

Similarly to Sec. 3.1, the expected log-likelihood term in (27) has two variations, full or partial likelihood, and we decompose it into two parts exactly the same way as (17). For concreteness, with the abbreviation  $\hat{f}_n(t) = \hat{f}(t, \mathbf{x}_n)$ , we approximate the expected log-likelihood term in (27) using the re-parametrized Monte-Carlo estimate. That is, after sampling  $\epsilon_j^{a(s)}$ ,  $\epsilon_j^{b(s)}$  and

$\epsilon_j^{\omega(s)}$  independently from  $\mathcal{N}(0, \mathbf{I}_m)$  for  $j = 1, \dots, d$  and  $s = 1, \dots, S$ , the sampled version of the random weight vector  $\mathbf{a}_j$  is formed as  $(\mathbf{b}_j^{(s)})$  and  $\omega_j^{(s)}$  similarly):

$$\mathbf{a}_j^{(s)} = \boldsymbol{\mu}_j^a + (\boldsymbol{\Sigma}_j^a)^{1/2} \epsilon_j^{a(s)}, \quad (28)$$

Then the posterior samples  $\hat{f}_n^{(s)}(t)$  can be written as:

$$\sum_{j=1}^d \frac{x_n(j)}{m} \left( (\mathbf{a}_j^{(s)})^\top \cos(\omega_j^{(s)} t) + (\mathbf{b}_j^{(s)})^\top \sin(\omega_j^{(s)} t) \right), \quad (29)$$

With these sampled functions, one can basically obtain the estimate of the objective (27) that can be decomposed into forms similar to (18), (19), and (20), which we denote by  $\hat{A}_n$ ,  $\hat{B}_n^f$ , and  $\hat{B}_n^p$ , respectively. Although evaluating  $\hat{A}_n$  and  $\hat{B}_n^p$  (and their gradients) can be done similarly as in Sec. 3.1 with no additional difficulty, working on  $\hat{B}_n^f$  introduces a new computational challenge. Unlike the pseudo-input approximation where we were able to express  $\mathbb{E}_Q[f_n(\tau)^2]$  as an analytic form  $\tilde{\mu}_n(\tau)^2 + \tilde{\sigma}_n^2(\tau)$ , we can only estimate the expectation numerically using the samples (29) from  $Q(\cdot)$ . However, we have an outer integration of this expectation over  $\tau \in [0, t_n]$ , and the (grid-based) numerical integration would incur computational explosion as we need  $(d \cdot m \cdot S \cdot G)$  samples/numbers involved in, where  $G$  is the number of grid points over  $[0, t_n]$  (typically,  $S$  and  $G$  are several thousands, and  $d \approx 10$ ).

To address this difficulty, we propose an alternative estimation strategy for  $\hat{B}_n^f$ . We regard  $\lambda_0(\tau) = c \cdot \tau^{r-1}$  in the integration as an unnormalized density, more specifically,  $\lambda_0(\tau) = \rho_n \cdot p_n(\tau)$  where  $p_n(\tau)$  is the density having the support  $[0, t_n]$  and  $\rho_n = \int_0^{t_n} \lambda_0(\tau) d\tau = \frac{c}{r} t_n^r$  is the normalizer. Thus  $p_n(\tau) = r \frac{\tau^{r-1}}{t_n^r}$  over  $[0, t_n]$ . Then

$$\hat{B}_n^f = \rho_n \cdot \mathbb{E}_{p_n(\tau)} \left[ \mathbb{E}_Q[f_n(\tau)^2] \right]. \quad (30)$$

This allows us to sample  $\tau_n^{(s)} \sim p_n(\tau)$  for  $s = 1, \dots, S$  independently with the random features/weights in (28), and estimate  $\hat{B}_n^f$  as:

$$\rho_n \cdot \frac{1}{S} \sum_{s=1}^S \hat{f}_n^{(s)}(\tau_n^{(s)})^2. \quad (31)$$

Note that sampling from  $p_n(\tau)$  can be done using the inverse transform sampling: for its CDF is  $F_n(\tau) = (\tau/t_n)^r$ , the samples  $\tau_n^{(s)}$  can be expressed as:

$$\tau_n^{(s)} = F_n^{-1}(u^{(s)}) = t_n \cdot (u^{(s)})^{1/r}, \quad (32)$$

where  $u^{(s)}$  are uniform samples from  $[0, 1]$ . We further reduce the variance of the estimate by using the same re-parametrization trick, namely plugging (32) into (31) to separate the randomness from the parameters.

## 4 EMPIRICAL EVALUATIONS

The performance of the proposed variational inference methods is demonstrated on both synthetic and real datasets. Our approaches are denoted as follows:  $\mathbf{VI}_{PI}^f$  and  $\mathbf{VI}_{PI}^p$  are the approximations based on pseudo inputs in Sec. 3.1 with full and partial likelihood, respectively, while  $\mathbf{VI}_{RF}^f$  and  $\mathbf{VI}_{RF}^p$  indicate the variational inference with random feature expansions described in Sec. 3.2. The competing approaches are summarized, with abbreviations, as:

- **MCMC**: The MCMC-based inference method for GPSAM (Fernández et al., 2016), where we used hyperparameters similar to theirs.
- **CoxPH<sup>f</sup>** and **CoxPH<sup>p</sup>**: The full and partial likelihood maximization learning of the CoxPH model.
- **SVCR**: The support vector censored regression approach (Shivaswamy et al., 2007) with no cost imposed for the over-estimation of the censored samples.
- **SVRC**: Another regression-based approach (Khan & Zubek, 2008) that adopts asymmetric costs for over-estimation depending on the violation types.
- **MINLIP**: The ranking-based prognostic function estimation method (Van Belle et al., 2009), which enforces the correct ordering of the prognostic indices for time-comparable pairs of samples. The method further aims to preserve the relative time differences in the prognostic indices.
- **Model2**: The hybrid approach that attempts to combine the ranking constraints and the cost-sensitive loss in estimating the prognostic function (Van Belle et al., 2011).

For the performance measures, we focus on the accuracy of the estimated prognostic index function  $u(\mathbf{x})$ . Whereas the approaches based on regression and/or ranking directly estimate  $u(\mathbf{x})$ , for the CoxPH-based methods we derive it naturally by  $u(\mathbf{x}) = -\mathbf{b}^\top \mathbf{x}$  from the learned CoxPH models. For GPSAM, where the hazard function is not factorized, we estimate the expected event time  $\mathbb{E}[t|\mathbf{x}]$  as the survival index. Two performance measures popular in survival analysis are considered: i) Concordance index – the proportion of the pairs of samples whose predicted survival times are correctly ordered (i.e.,  $(u(\mathbf{x}_i) - u(\mathbf{x}_j))(t_i - t_j) \geq 0$ ), and ii) Log-rank- $\chi^2$  statistics – the statistical test score measuring the difference between two risk groups formed by thresholding the prognostic indices by their median. For both measures, higher scores are better.

For our variational inference methods, we vary the following hyperparameters: the number of pseudo inputs ( $M$ ) for the  $\mathbf{VI}_{PI}$  and the number of random features ( $m$ ) for the  $\mathbf{VI}_{RF}$ . We use the best set obtained by cross validation on the held-out portion of the training data, where the concordance index is used as the selection criterion. The optimal parameters are, consistently across all datasets,  $M = 20$  for the  $\mathbf{VI}_{PI}$  and  $m = 50$  for the  $\mathbf{VI}_{RF}$ . In the latter part of the section and in the supplemental material (Appendix B), we also compare the performances and running times of the other parameter settings. The number of samples for the Monte-Carlo estimation in our variational inference is fixed as 3000. The MCMC approach for GPSAM model (Fernández et al., 2016) used  $m = 50$  random features, and the number of MCMC iterations is chosen as 5000 with the first 1000 samples dropped out. The hyperparameters of the other competing models (e.g., the regularization parameters in the regressions) are also determined by cross validation.

### 4.1 SYNTHETIC DATASETS

To judge the effectiveness and flexibility of the proposed variational inference methods for GPSAM, we consider a synthetic scenario where the data samples are generated from a non-proportional hazard model. In particular, we consider a stratified CoxPH model, which can be seen as a conditional mixture of several CoxPH models. More specifically, the hazard function is defined as  $\lambda(t|\mathbf{x}) = \lambda_{s(\mathbf{x})}(t) \cdot \exp(\mathbf{b}_{s(\mathbf{x})}^\top \mathbf{x})$  where  $s(\mathbf{x}) \in \{1, \dots, K\}$  is a gating function among  $K$  component CoxPH models. The base hazard function  $\lambda_s(t)$  for each component model  $s = 1, \dots, K$ , is defined to be the Weibull function  $c_s \cdot t^{r_s-1}$  with different parameters ( $c_s, r_s$ ) for each  $s$ . The gating function follows a piecewise linear form,  $s(\mathbf{x}) = \arg \max_{1 \leq s \leq K} \mathbf{w}_s^\top \mathbf{x}$ , where we choose  $\mathbf{w}_s$ 's randomly. We set the number of base models as  $K = 3$ . The input covariates  $\mathbf{x}$  are sampled randomly from  $\mathcal{N}(0, \mathbf{I})$ .

To mimic the censoring process, for each generated sample, we randomly turn it into a censored one with probability  $p$ . The observed time  $t$  for the censored sample is then uniformly sampled from  $[0, t_*]$  where  $t_*$  is the original value before censoring. We choose  $p = 0.3$ . After generating 100 samples, we perform 10-fold cross validation where the averaged test scores with standard deviations are depicted in Table 1. For each measure, the best performing method in terms of the average value is boldfaced. To measure the statistical significance, we also conducted the Wilcoxon signed-rank tests, pairwise against the (boldfaced) best performing method. With the null hypothesis that two approaches result in statistically indistinguishable performance, the  $p$ -values

Table 1: (Synthetic dataset) Average test prediction performance. Our variational approximation approaches,  $VI_{PI}$  and  $VI_{RF}$ , adopt  $M = 20$  pseudo inputs and  $m = 50$  random features, respectively. Best average score method is boldfaced. Parentheses indicate the  $p$ -values from the Wilcoxon signed rank test against the best (boldfaced) approaches.

Methods	C-Index (%)	Log-Rank- $\chi^2$
$VI_{PI}^f$	<b>83.04 ± 1.91 (--)</b>	<b>13.88 ± 2.85 (--)</b>
$VI_{PI}^p$	79.68 ± 2.50 (0.002)	10.71 ± 4.73 (0.125)
$VI_{RF}^f$	81.55 ± 2.20 (0.049)	10.55 ± 5.66 (0.250)
$VI_{RF}^p$	81.28 ± 2.54 (0.049)	10.73 ± 5.38 (0.250)
MCMC	77.20 ± 3.66 (0.002)	10.81 ± 6.00 (0.193)
CoxPH <sup>f</sup>	73.43 ± 5.51 (0.002)	8.70 ± 2.88 (0.232)
CoxPH <sup>p</sup>	73.27 ± 4.99 (0.002)	9.14 ± 3.37 (0.160)
SVCR	68.32 ± 6.51 (0.002)	5.55 ± 3.48 (0.027)
SVRC	73.05 ± 4.37 (0.002)	7.81 ± 3.20 (0.084)
MINLIP	65.75 ± 4.34 (0.002)	3.40 ± 2.20 (0.004)
Model2	71.32 ± 3.04 (0.002)	5.63 ± 2.24 (0.027)

are shown in the tables.

The proposed variational inference approaches, both  $VI_{PI}$  and  $VI_{RF}$ , exhibit superior generalization performance compared to all contrasted methods. Specifically, the pseudo-input approximation optimizing the full-likelihood ( $VI_{PI}^f$ ) performs the best in both measures. With regard to the concordance index, it is significantly better than all other models ( $p$ -values  $< 0.05$ ), but leads to marginal improvements in the log-rank- $\chi^2$  measure. The CoxPH-based models (CoxPH<sup>f</sup> and CoxPH<sup>p</sup>) are mostly outperformed by the GPSAM models due to the substantial mismatch with the true data process (simplified modeling assumption of non-crossing hazard functions across instances). Compared to the MCMC approach (Fernández et al., 2016), our proposed VI methods yield higher prediction accuracies, related to improved hazard function estimation. In contrast, the MCMC potentially suffers from computational overhead, preventing convergence to the target distribution.

To investigate the computational benefits of the proposed approaches over the MCMC algorithm, we measure the actual inference times (for our variational inference, we record the entire running time until convergence). Implemented in MATLAB and run on 2.4GHz Intel Xeon CPU, the running times are: 571.7 seconds for  $VI_{PI}^f$ , 1165.1 seconds for  $VI_{RF}^f$ , and 8121.2 seconds for the MCMC, indicating significant computational advantage of our VI approaches. The log-likelihood scores  $\log P(\mathcal{D}_*)$  of GPSAM evaluated on the test data  $\mathcal{D}_*$  are also summarized in Table 3. Although the scores are mostly comparable, the MCMC attains the highest likeli-

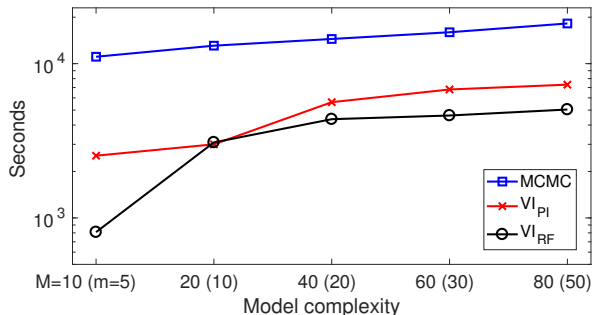


Figure 1: The inference times of three methods: MCMC,  $VI_{PI}^f$ , and  $VI_{RF}^f$  on the MLC dataset.

hood. However, it should be noted that we only compute lower bounds of the log-likelihoods (i.e.,  $\mathcal{L}_{PI}$  in (15) and  $\mathcal{L}_{RF}$  in (27)) for our variational learning.

## 4.2 REAL DATASETS

Next we test the performance of the proposed methods on two clinical and one non-clinical datasets: i) (VLC) Veteran’s lung cancer dataset (Prentice, 1974; Kalbfleisch & Prentice, 2002) contains records of 137 patients with 6 covariates such as age, weight, treatment, cell type, and disease history, ii) (MLC) Mayo lung cancer dataset (Therneau & Grambsch, 2000) having 167 patients with similar 7 covariates, and iii) (Divorce) dataset (Lillard & Panis, 2000) which records the divorce years since marriage for 3771 couples, among which we use a fifth of the samples randomly chosen. The proportions of the censored samples are: 7% for VLC, 28% for MLC, and 70% for Divorce, where the study involved in the latter dataset often fails to track the status of many of the couples, which leads to the large proportion of the censored samples. The Divorce dataset originally provides three categorical features for each sample: the husband education years (categorized into three levels of less than 12 yrs, more than 15, and between two), whether the husband is African American or not, and whether the ethnicity of the couple is different or not. We also introduce additional nonlinear features of pairwise and triple products, yielding 7-dim covariates. Other experimental settings follow the synthetic experiment.

We performed 10-fold cross validation where the test results are shown in Table 2. The best methods, boldfaced with  $p$ -values from the statistical test against other methods, are mostly our variational inference methods (four out of six). Our approaches estimate the posterior of the latent function more accurately than the MCMC while enjoying the benefits of the GP to account for uncertainty in the hazard modeling and relaxing the time-covariates factorized assumption of the CoxPH model.

Table 2: (Real datasets) Average test prediction performance. Our variational approximation approaches,  $VI_{PI}$  and  $VI_{RF}$ , adopt the number of pseudo inputs  $M = 20$  and random features  $m = 50$ , respectively. Best method in terms of the average value is boldfaced. Figures in parentheses indicate the  $p$ -values from the Wilcoxon signed rank test against the best (boldfaced) approaches.

Datasets Methods	VLC		MLC		Divorce	
	C-Index (%)	Log-Rank- $\chi^2$	C-Index (%)	Log-Rank- $\chi^2$	C-Index (%)	Log-Rank- $\chi^2$
$VI_{PI}^f$	71.99 $\pm$ 3.67 (0.0020)	3.89 $\pm$ 2.74 (0.3750)	<b>72.68 <math>\pm</math> 3.00</b> (--)	2.74 $\pm$ 2.07 (0.6250)	63.60 $\pm$ 3.17 (1.0000)	2.27 $\pm$ 1.70 (1.0000)
$VI_{PI}^p$	70.87 $\pm$ 4.75 (0.0039)	3.61 $\pm$ 2.82 (0.2754)	69.44 $\pm$ 6.48 (0.0273)	2.35 $\pm$ 2.27 (0.4316)	62.89 $\pm$ 3.92 (0.1934)	2.25 $\pm$ 1.97 (0.7695)
$VI_{RF}^f$	<b>76.79 <math>\pm</math> 4.08</b> (--)	5.13 $\pm$ 4.17 (1.0000)	68.08 $\pm$ 4.98 (0.0098)	1.64 $\pm$ 1.94 (0.1934)	63.73 $\pm$ 2.67 (0.0703)	2.07 $\pm$ 1.50 (0.7695)
$VI_{RF}^p$	76.49 $\pm$ 4.51 (0.6875)	<b>5.81 <math>\pm</math> 4.27</b> (--)	67.00 $\pm$ 5.35 (0.0098)	1.32 $\pm$ 1.92 (0.1309)	<b>64.56 <math>\pm</math> 3.47</b> (--)	2.24 $\pm$ 1.33 (1.0000)
MCMC	68.48 $\pm$ 2.57 (0.0098)	2.34 $\pm$ 1.54 (0.0840)	66.46 $\pm$ 6.38 (0.0039)	2.15 $\pm$ 2.72 (0.4316)	63.81 $\pm$ 3.15 (1.0000)	<b>2.35 <math>\pm</math> 2.68</b> (--)
CoxPH <sup>f</sup>	69.28 $\pm$ 6.05 (0.0098)	3.12 $\pm$ 3.75 (0.0547)	66.89 $\pm$ 11.05 (0.3223)	3.25 $\pm$ 2.53 (1.0000)	63.33 $\pm$ 2.71 (0.4922)	1.67 $\pm$ 1.38 (0.5566)
CoxPH <sup>p</sup>	69.10 $\pm$ 5.88 (0.0098)	2.64 $\pm$ 3.86 (0.0078)	68.25 $\pm$ 11.14 (0.1934)	<b>3.65 <math>\pm</math> 3.04</b> (--)	63.33 $\pm$ 2.71 (0.4922)	1.67 $\pm$ 1.38 (0.5566)
SVCR	55.41 $\pm$ 9.31 (0.0020)	1.10 $\pm$ 1.68 (0.0020)	56.42 $\pm$ 17.89 (0.0039)	2.52 $\pm$ 2.71 (0.1602)	54.38 $\pm$ 11.21 (0.0098)	0.87 $\pm$ 0.72 (0.3750)
SVRC	55.36 $\pm$ 9.66 (0.0020)	1.10 $\pm$ 1.68 (0.0020)	54.93 $\pm$ 17.65 (0.0039)	2.61 $\pm$ 2.74 (0.2324)	54.43 $\pm$ 10.88 (0.0059)	1.14 $\pm$ 1.19 (0.3223)
MINLIP	65.64 $\pm$ 9.79 (0.0098)	2.59 $\pm$ 3.23 (0.1309)	68.60 $\pm$ 9.28 (0.2754)	3.18 $\pm$ 3.03 (0.8457)	51.75 $\pm$ 5.38 (0.0039)	0.60 $\pm$ 0.96 (0.1602)
Model2	57.40 $\pm$ 10.82 (0.0039)	1.59 $\pm$ 2.07 (0.0020)	68.81 $\pm$ 8.06 (0.1934)	3.19 $\pm$ 3.02 (0.6953)	55.83 $\pm$ 10.45 (0.0195)	0.69 $\pm$ 0.74 (0.2324)

Next we compare running times. To see the effect of the approximation model complexity on the inference time, we vary the parameters in model learning. Specifically,  $M$  is chosen from  $\{10, 20, 40, 60, 80\}$  for  $VI_{PI}$  and  $m$  is from  $\{5, 10, 20, 30, 50\}$  for  $VI_{RF}^f$  and also the MCMC method (Fernández et al., 2016) that employs the random feature approximation. All methods are implemented in MATLAB run on 2.4GHz Intel Xeon CPU. The results on the MLC dataset are visualized in Fig. 1. Results demonstrate that our variational methods are an order of magnitude faster than the MCMC while achieving comparable or often superior prediction performance. For other datasets, refer to Appendix B in the supplemental material. Finally, the test log-likelihood scores of the attained models are depicted in Table 3. Considering we report lower VI bounds of the log-likelihoods, all proposed methods exhibit comparable, or superior, generalization performance to that of the MCMC.

## 5 CONCLUSION

We have proposed a family of novel and highly scalable variational inference methods for the Gaussian process

Table 3: Average test log-likelihood scores attained by  $VI_{PI}$  ( $M = 20$ ),  $VI_{RF}$  ( $m = 50$ ), and the MCMC.

Datasets	Synthetic	VLC	MLC	Divorce
$VI_{PI}^f$	-36.40	-31.50	-40.00	-80.31
$VI_{PI}^p$	-41.13	-35.00	-41.78	-84.10
$VI_{RF}^f$	-35.18	-25.87	-45.48	-86.75
$VI_{RF}^p$	-36.33	-26.05	-45.47	-86.75
MCMC	-34.01	-30.70	-38.61	-105.53

survival analysis model. Our approaches can estimate the posterior of the GP latent function in this flexible non-proportional hazard model more accurately, with running times an order of magnitude faster than the state-of-the-art MCMC algorithm.

## Acknowledgments

MK is supported by National Research Foundation of Korea (NRF-2016R1A1A1A05921948).

## References

- Adams, R. P., Murray, I., & MacKay, D. J. (2009). Tractable nonparametric Bayesian inference in Poisson processes with Gaussian process intensities. International Conference on Machine Learning.
- Casella, G., & Robert, C. P. (1996). Rao-Blackwellisation of sampling schemes. *Biometrika*, 83, 81–94.
- Cho, Y., & Saul, L. K. (2009). Kernel methods for deep learning. In Advances in Neural Information Processing Systems.
- Cox, D. (1972). Regression models and life-tables (with discussion). *Journal of the Royal Statistical Society, Series B*, 34, 187–220.
- Dempsey, W. H., Moreno, A., Scott, C. K., Dennis, M. L., Gustafson, D. H., Murphy, S. A., & Rehg, J. M. (2017). iSurvive: An Interpretable, Event-time Prediction Model for mHealth. International Conference on Machine Learning.
- Dezfouli, A., & Bonilla, E. V. (2015). Scalable inference for Gaussian process models with black-box likelihoods. In Advances in Neural Information Processing Systems.
- Fernández, T., Rivera, N., & Teh, Y. W. (2016). Gaussian processes for survival analysis. In Advances in Neural Information Processing Systems.
- Hjort, N. L., Holmes, C., Miller, P., & Walker, S. G. (2010). *Bayesian nonparametrics*. Cambridge University Press.
- Iorio, M. D., Johnson, W. O., Miller, P., & Rosner, G. L. (2009). Bayesian nonparametric nonproportional hazards survival modeling. *Biometrics*, 65, 762–771.
- Ishwaran, H., Kogalur, U. B., Blackstone, E. H., & Lauer, M. S. (2008). Random survival forests. *The annals of applied statistics*, 2, 841–860.
- Kalbfleisch, J., & Prentice, R. (2002). *The statistical analysis of failure time data*. Wiley Series in Probability and Statistics, New York.
- Khan, F., & Zubek, V. (2008). Support vector regression for censored data (SVRc): A novel tool for survival analysis. In Proceedings of the Eighth IEEE International Conference on Data Mining (ICDM).
- Kingma, D. P., & Welling, M. (2014). Auto-encoding variational Bayes. In Proceedings of the Second International Conference on Learning Representations.
- Kleinbaum, D. G., & Klein, M. (2005). *Survival analysis: A self-learning text (statistics for biology and health)*. Springer.
- Lillard, & Panis (2000). aml multilevel multiprocess statistical software. Release 1.0, EconWare, LA, California.
- Lloyd, C., Gunter, T., Osborne, M. A., & Roberts, S. J. (2015). Variational inference for Gaussian process modulated Poisson processes. International Conference on Machine Learning.
- Martino, S., Akerkar, R., & Rue, H. (2011). Approximate Bayesian inference for survival models. *Scandinavian Journal of Statistics*, 38, 514–528.
- Prentice, R. L. (1974). A log gamma model and its maximum likelihood estimation. *Biometrika*, 61, 539–544.
- Quiñonero-Candela, J., & Rasmussen, C. E. (2005). A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6, 1939–1959.
- Rahimi, A., & Recht, B. (2008). Random features for large-scale kernel machines. In Platt, J. C., Koller, D., Singer, Y., and Roweis, S. T. (eds.), Advances in Neural Information Processing Systems 20.
- Ranganath, R., Perotte, A., Elhadad, N., & Blei, D. (2016). Deep survival analysis. Machine Learning for Health Care.
- Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian processes for machine learning*. The MIT Press.
- Ross, S. M. (2006). *Simulation*. Academic Press.
- Shivaswamy, P., Chu, W., & Jansche, M. (2007). A support vector approach to censored targets. In Proceedings of the Seventh IEEE International Conference on Data Mining (ICDM).
- Therneau, T. M., & Grambsch, P. M. (2000). *Modeling survival data: Extending the Cox model*. Springer-Verlag, New York.
- Titsias, M. K. (2009). Variational learning of inducing variables in sparse Gaussian processes. In Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics.
- Van Belle, V., Pelckmans, K., Suykens, J., & Van Huffel, S. (2009). Learning transformation models for ranking and survival analysis. Tech. Rep., 09-45, ESAT-SISTA, K.U.Leuven (Leuven, Belgium).



Van Belle, V., Pelckmans, K., Van Huffel, S., & Suykens, J. (2011). Support vector methods for survival analysis: A comparison between ranking and regression approaches. *Artificial Intelligence in Medicine*, 53, 107–118.

---

# A Cost-Effective Framework for Preference Elicitation and Aggregation

---

**Zhibing Zhao, Haoming Li,  
Junming Wang**  
RPI  
Troy, NY, USA  
{zhaoz6,lih14,wangj33}@rpi.edu

**Jeffrey O. Kephart,  
Nicholas Mattei, Hui Su**  
IBM Research  
Yorktown, NY, USA  
{kephart,n.mattei,huisuibmres}@us.ibm.com

**Lirong Xia**  
RPI  
Troy, NY, USA  
xial@cs.rpi.edu

## Abstract

We propose a cost-effective framework for preference elicitation and aggregation under the Plackett-Luce model with features. Given a budget, our framework iteratively computes the most cost-effective elicitation questions in order to help the agents make a better group decision.

We illustrate the viability of the framework with experiments on Amazon Mechanical Turk, which we use to estimate the cost of answering different types of elicitation questions. We compare the prediction accuracy of our framework when adopting various information criteria that evaluate the expected information gain from a question. Our experiments show carefully designed information criteria are much more efficient, i.e., they arrive at the correct answer using fewer queries, than randomly asking questions given the budget constraint.

## 1 INTRODUCTION

Consider the hiring decision problem [Bhattacharjya and Kephart, 2014]. With the aid of an intelligent system, a group of people (the *key group*) faces a hiring decision about many candidates who are characterized by attributes, such as experiences, technical skills, communication skills, etc. The goal is to help the key group make a group decision without directly eliciting their full preferences over all candidates, which is often infeasible given the vast number of candidates. Instead, the intelligent system may ask fellow employees (the *regular group*) about their preferences in order to learn about the key group’s preferences. How can the intelligent system decide which member in the regular group to ask and

which questions to ask? Note that we discuss the presence of two groups but our framework is applicable when there is only one group of decision makers as well.

This example illustrates the *preference elicitation* problem, which has been widely studied in the field of recommender systems [Loepp et al., 2014], healthcare [Chajewska et al., 2000, Weernink et al., 2014, Erdem and Campbell, 2017], marketing [Huang and Luo, 2016], stable matching [Drummond and Boutilier, 2014, Rastegari et al., 2016], combinatorial auctions [Sandholm and Boutilier, 2006], etc. Most previous works studied a special case of the aforementioned scenario, in which the regular group is the key group. The objective of preference elicitation is to achieve some goal using as few samples (data) as possible. A common approach is to adaptively ask questions that maximize expected information gain, measured by some information criteria.

Moreover, most previous work focused on specific types of elicitation questions, e.g. pairwise comparisons. In this paper, we consider a more general framework that asks a variety of elicitation questions and can accommodate one or more groups. The diversity of elicitation questions enables us to query cost-effectively. Intuitively, an agent’s preference order over 10 alternatives tells us more about her preference in general than just her top choice among the 10; however, it may take her longer to do so. The key question we want to answer in this paper is:

*How can we compute the most cost-effective questions for preference elicitation under resource constraints?*

### 1.1 OUR CONTRIBUTIONS

We propose a flexible cost-effective preference elicitation and aggregation framework to predict a single agent’s preference or help make a group decision. The main inputs include a budget  $W$ , a set of designs (i.e. questions to ask)  $\mathcal{H}$ , a cost function  $w$ , a randomized

voting rule and an information criterion. We model non-deterministic preferences using the Plackett-Luce model with features.

**Cost-effectiveness.** We propose a flexible, cost-effective preference elicitation framework that accommodates all randomized voting rules, ranking models, and information criteria. This iterative framework leverages the *optimal design* technique. In each iteration, we choose the question that provides the most information per unit cost. The response is then recorded as a data point, leading to an update of the posterior distribution of the parameter, which is treated as the prior for the next iteration. In any iteration, the posterior estimate of the parameter can be used to compute a winner distribution using a randomized voting rule. This procedure is illustrated in Figure 1.

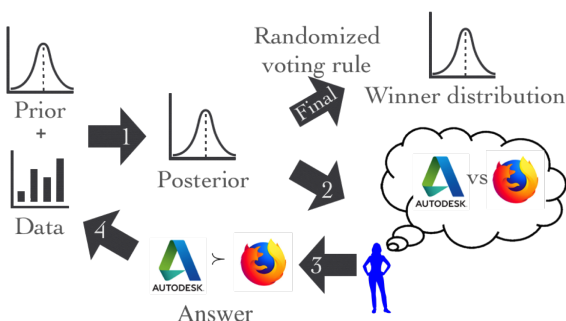


Figure 1: Illustration of the proposed framework.

**Randomized Voting Rules.** We use randomized voting rules to compute the winning alternatives of a group decision, which outputs the distribution of winners (see Section 4 for details). The probability for each alternative to be the winner is proportional to its score based on the voting rule. These probability estimates are more informative than only recommending a winner as it provides a distribution over the candidates as well.

We prove that when people have non-deterministic preferences, the probability of an alternative to be the winner is proportional to the total expected score of this alternative for all agents (Theorem 1). This means the randomized counterpart of any scoring rule can be used in our framework as long as the expected score of each alternative for a single agent is easy to compute. Then we prove that under the Plackett-Luce model, the winner distributions of probabilistic plurality and probabilistic Borda are easy to compute (Corollary 1 and Theorem 2).

**Information Criteria.** An information criterion plays a key role in determining the next elicitation question by measuring the information in the distribution of a parameter. We propose the minimum pairwise certainty

(MPC) criterion, extended from the information criterion by Azari Soufiani et al. [2013], which maximizes the improvement of the least certain pairwise comparison. Other commonly-used information criteria include D-optimality [Wald, 1943, Mood et al., 1946] and E-optimality [Ehrenfeld, 1955], as well as asking a question uniformly at random. All these information criteria are based on the information of the posterior distribution of the model parameter, which is approximated by its asymptotic distribution, a multivariate Gaussian computed based on the composite marginal likelihood method [Pauli et al., 2011].

**Empirical Studies & Experiments.** We carry out Amazon Mechanical Turk experiments to estimate the cost of answering various types of questions for a target domain of ranking hotels. We compare the performances of MPC, D-optimality and E-optimality with simulations and observe that these criteria have similar performance in terms of prediction accuracy, and we observe that all of them significantly outperform random elicitation questions.

## 1.2 RELATED WORK AND DISCUSSIONS

Our work is related to cost-effective experimental designs, which were investigated by Wright et al. [2010], Volkov [2014] in the context of aquatic toxicology and drug development, respectively. Volkov [2014] modeled cost-effectiveness as different types of optimization problems, e.g., minimize cost under information constraints. We take a greedy approach, similar to an algorithm proposed by Wright et al. [2010], and choose the design (elicitation question) that maximizes the expected information gain per unit cost. Our cost varies depending on the type of questions and is estimated empirically, similar to the idea in Volkov [2014]. To our best knowledge, this paper is the first work to apply cost-effective experimental design to preference elicitation.

The greedy approach is also called one-step-lookahead policy, which can be arbitrarily worse than optimal  $t$ -step-lookahead ( $t$ -step myopic active search) policies for  $t \geq 2$  [Garnett et al., 2012]. Arbitrary  $t$ -step myopic active search is hard to compute, as was shown by Jiang et al. [2017], which also proved that nonmyopic active search is computationally hard even to approximate and proposed an efficient searching algorithm. This algorithm is potentially useful in the preference elicitation context and is an interesting future direction.

Most previous works in preference elicitation assumed that people’s preferences are deterministic. For example, Bhattacharjya and Kephart [2014] proposed an even swap algorithm to reveal a single decision maker’s most

preferred alternative; Lu and Boutilier [2011a], Kalech et al. [2011] elicited preferences from a group of people in order to make a group decision under a (deterministic) voting rule. In contrast, we consider non-deterministic preferences of people, which is often the case in real-world. Moreover, we use randomized voting rules, which output the probability of each alternative to be the winner. These probabilities, which can be viewed as normalized scores over all alternatives, provide a quantitative measure of the quality of each alternative. For example, an alternative that wins with probability 0.8 can be seen as being much better than other alternatives.

Non-deterministic preferences were modeled by general random utility models by Azari Soufiani et al. [2013]. They proposed a preference elicitation framework for personalized choice and social choice (aggregated preference). We use the Plackett-Luce model with features, which is a special case of general random utility models but has easy-to-compute probabilities. More importantly, we use randomized voting rules for aggregation, which is very different from parametric modeling of social choices employed by [Azari Soufiani et al., 2013].

Pairwise elicitation questions may be the most widely explored in the literature due to their simplicity [Branke et al., 2017, Eric et al., 2008, Houlby et al., 2012, Lu and Boutilier, 2011a, Pfeiffer et al., 2012]. In contrast, Azari Soufiani et al. [2013] focused on elicitation of full rankings, though their proposed framework also allows for partial orders. Drummond and Boutilier [2014], Lu and Boutilier [2011b] studied a larger set of queries, which includes asking a person to rank her top  $k$  choices over all alternatives. In this paper, we consider an even broader set of queries, asking an agent to rank her top  $k$  choices over a subset of  $l$  alternatives ( $k < l$ ). This enables us to elicit preferences in a more cost-effective manner.

As a key role in preference elicitation, information criteria have been widely investigated for different applications. Standard information criteria include D-optimality (used in [Houlby et al., 2011, 2012, Pfeiffer et al., 2012]) and E-optimality. Drummond and Boutilier [2014] and Lu and Boutilier [2011a] use minimax-regret-based criterion for stable matching and aggregation respectively. Azari Soufiani et al. [2013] proposed yet another criterion, defined on the certainty of the least certain pairwise comparison over the intrinsic utilities (part of the parameter of their general random utility models) of all alternatives. Our MPC criterion extends the criterion by Azari Soufiani et al. [2013]. To predict a single agent's top  $k$  preference, we search over a subset of all pairwise comparisons (see Section 3.3). To help make a group decision, we search over all pairwise comparisons

of all agents in the key group to find the least certain pairwise comparison (Equation (3)).

## 2 PRELIMINARIES

Let  $\mathcal{A} = \{a_1, a_2, \dots, a_m\}$  denote a set of  $m$  alternatives and  $\{1, \dots, n_1, n_1 + 1, \dots, n_1 + n_2\}$  denote  $n_1 + n_2$  agents, where the first  $n_1$  agents belong to the key group, who will be making a group decision. The remaining  $n_2$  agents belong to the regular group. For all  $i = 1, \dots, m$ ,  $a_i$  is characterized by a real-valued column vector of  $K$  attributes  $\vec{z}_i$ . For all  $j = 1, \dots, n_1 + n_2$ , agent  $j$  is characterized by a real-valued column vector of  $L$  attributes  $\vec{x}_j$ . A full ranking  $R$  is often denoted by  $a_{i_1} \succ a_{i_2} \succ \dots \succ a_{i_m}$ , where “ $\succ$ ” means “is preferred over”. We denote the budget by  $W$ , where the money is used to pay the agents for answering elicitation questions.

For  $n_1 = 1$ , we want to predict the single key agent's full or top  $k$  ranking with as much certainty as possible given a budget  $W$ . For  $n_1 \geq 2$ , the goal is to predict the winning alternative of the key group by eliciting preferences from the regular group in the most cost-effective way. More concretely, given  $W$ , we want to output a distribution of winning alternatives, w.r.t. a randomized voting rule, which will be defined in Section 4.

### 2.1 THE PLACKETT-LUCE MODEL WITH FEATURES

Let the parameter  $B = [b_{\kappa l}]_{K \times L}$  be a matrix of real-valued coefficients, transforming features to utilities. Each value  $b_{\kappa l}$  corresponds to the  $\kappa$ -th attribute from an alternative and  $l$ -th attribute from an agent. The parameter space  $\Theta$  is a set of all real-valued  $K \times L$  matrices. Then the utility of an alternative  $a_i$  to an agent  $j$  is

$$u_{ji} = \vec{x}_j^\top B \vec{z}_i. \quad (1)$$

For any agent  $j$  and any full ranking  $R_j = a_{i_1} \succ a_{i_2} \succ \dots \succ a_{i_m}$ , the probability of  $R_j$  is

$$\Pr(R_j) = \frac{\exp(u_{ji_1})}{\sum_{q=1}^m \exp(u_{ji_q})} \times \frac{\exp(u_{ji_2})}{\sum_{q=2}^m \exp(u_{ji_q})} \times \dots \times \frac{\exp(u_{ji_{m-1}})}{\exp(u_{ji_{m-1}}) + \exp(u_{ji_m})}.$$

Given the Plackett-Luce model with features, the probability of alternative  $a_{i_1}$  to be ranked at the top among  $\{a_{i_1}, \dots, a_{i_m}\}$  by agent  $j$  is  $\frac{\exp(u_{ji_1})}{\sum_{q=1}^m \exp(u_{ji_q})}$ . Specifically, for any two alternatives  $a_1$  and  $a_2$ , the probability of  $a_1 \succ a_2$  by agent  $j$  is  $\frac{\exp(u_{j1})}{\exp(u_{j1}) + \exp(u_{j2})}$ .

## 2.2 ONE-STEP BAYESIAN EXPERIMENTAL DESIGN

Given any probabilistic model parameterized by  $B \in \Theta$  and any prior distribution  $\pi(B)$ , a one-step Bayesian experimental design consists of two parts: (i) a set of designs  $\mathcal{H}$ , where each  $h \in \mathcal{H}$  is composed of an agent and a question; (ii) an information measure  $G(\cdot)$ , which maps any distribution of  $B$  over  $\Theta$  to a real-valued scalar: a measure of information in this distribution.

For any design  $h \in \mathcal{H}$ , the distribution of responses can be computed using the ground truth parameter  $B^*$ . We use  $D$  to denote the set of all possible responses. Given a ground truth parameter  $B^*$ , the probability of any data  $d \in D$  can be computed as  $\Pr(d|h)$ . Further, we can compute the posterior distribution of parameter  $\pi(B|d, h)$  over the parameter space  $\Theta$  and the corresponding information criterion  $G(\pi(B|d, h))$ . The expected information is

$$E[G(\pi(B|h))] = \sum_{d \in D} G(\pi(B|d, h)) \Pr(d|h),$$

where the expectation is taken over all possible responses. The goal is to find the design  $h$  that maximize the expected information gain, which is  $E[G(\pi(B|h))] - G(\pi(B))$ , per unit cost. Let  $w(h)$  denote the cost function, which maps the 2-tuple (agent, question) to a positive cost. Given the cost function  $w(h)$ , we can compute the optimal design  $h^*$  that maximizes the expected information gain per unit cost by

$$h^* = \arg \max_h \frac{E[G(\pi(B|h))] - G(\pi(B))}{w(h)}. \quad (2)$$

## 3 COST-EFFECTIVE PREFERENCE ELICITATION

In our proposed framework, we iteratively adapt the one-step experimental design by querying the most cost-effective question in each iteration. At any iteration  $t$ , the prior distribution of  $B$  is the posterior distribution given data  $D^t$ , i.e.  $\pi(B^t|D^t)$ . Given this posterior, we find the most cost-effective design  $h^t$ , which consists of one agent and one question, and query  $h^t$ . The response is combined with  $D^t$  to form  $D^{t+1}$ . Then the budget  $W$  and the set of designs  $\mathcal{H}$  are updated before going to the next iteration. Finally, when  $n_1 = 1$ , we compute the predicted preference of this agent; when  $n_1 \geq 2$ , we compute the distribution of winners based on a randomized voting rule. This framework is formally illustrated in Algorithm 1.

For the rest of this section, we will explain how to approximate the posterior distribution  $\pi(B^t|D^t)$  and how  $G(\pi(B))$  is computed.

---

### Algorithm 1 Cost-Effective Preference Elicitation

---

**Input:** Budget  $W$ , randomize voting rule  $r$ , cost function  $w(h)$ , information criterion  $G(\pi(B))$ , the set of designs  $\mathcal{H}$  where for any  $h \in \mathcal{H}$ ,  $w(h) \leq W$ .

**Output:** A predicted preference when  $n_1 = 1$  or a distribution of winning alternatives for group decision when  $n_1 \geq 2$ .

**Initialization:** Randomly initialize data  $D^1$ .

**while**  $\mathcal{H}$  is not empty **do**

    Compute/approximate  $\pi(B^t|D^t)$ ;

    Compute  $h^t \in \mathcal{H}$  using (2);

    Implement  $h^t$  (query an agent a question). Let  $R^t$  denote her answer. Then  $D^{t+1} \leftarrow D^t \cup \{R^t\}$ ,  $\mathcal{H} \leftarrow \mathcal{H} - h^t$ ,  $W \leftarrow W - w^t$ ;

    Remove all  $h^t$ 's from  $\mathcal{H}$  where  $w(h^t) > W$ .

**end while**

Compute the predicted preference when  $n_1 = 1$  or a distribution of winning alternatives according to the voting rule  $r$  when  $n_1 \geq 2$ .

---

## 3.1 APPROXIMATION OF POSTERIOR DISTRIBUTION

For any prior  $\pi(B)$  and data  $D$ , the posterior distribution is given by  $\pi(B|D) = \frac{\Pr(D|B)\pi(B)}{\int_{\Theta} \Pr(D|B)\pi(B)dB}$  according to Bayes' rule. This posterior is often hard to compute. A commonly-used approach is to approximate it by its asymptotic distribution, which is a multivariate Gaussian distribution characterized by the composite marginal likelihood (CML) method [Pauli et al., 2011].

For convenience we vectorize  $B$  as a column vector, denoted by  $\vec{\beta} = \text{vec}(B)$ . The composite marginal likelihood method [Lindsay, 1988, Zhao and Xia, 2018] computes the estimate of the ground truth parameter from marginal events, e.g., pairwise comparisons. Let  $\{\mathcal{E}_1, \dots, \mathcal{E}_q\}$  denote  $q$  selected marginal events. Then the composite marginal likelihood method computes the estimate  $\vec{\beta}_{\text{CML}}$  by

$$\vec{\beta}_{\text{CML}} = \arg \max_{\vec{\beta} \in \Theta} \text{CLL}(\vec{\beta}) = \arg \max_{\vec{\beta} \in \Theta} \sum_{\lambda=1}^q \ln \Pr(\mathcal{E}_\lambda | \vec{\beta}),$$

where  $\text{CLL}(\vec{\beta})$  denotes the composite log-likelihood function. Under our Plackett-Luce model with features,  $\text{CLL}(\vec{\beta})$  is twice differentiable for all  $\vec{\beta} \in \Theta$ , i.e.  $J(\vec{\beta}) = -\nabla_{\vec{\beta}}^2 \text{CLL}(\vec{\beta})$  exists. From Pauli et al. [2011], asymptotically,  $\pi(\vec{\beta}|D)$  is a multivariate Gaussian distribution, whose mean is  $\vec{\beta}_{\text{CML}}$  and covariance matrix is  $J^{-1}(\vec{\beta})$ .

Computing  $J(\vec{\beta})$  requires computation of second order partial derivatives of  $\ln \Pr(\mathcal{E}_\lambda | \vec{\beta})$  for all  $\lambda$ . We will show the close-form second order partial derivative formula for

any response from an agent.

### 3.2 THE SET OF DESIGNS

Each design  $h \in \mathcal{H}$  is a combination of an agent and a question about her preferences. The agent can be anyone from  $\{1, \dots, n_1 + n_2\}$ . In this paper, for simplicity, we consider the case where only the agents from the regular group  $\{n_1 + 1, \dots, n_1 + n_2\}$  are queried. For any integers  $k < l \leq m$ , we may ask an agent to rank her top  $k$  alternatives over a subset of  $l$  alternatives. When  $k = 1, l = 2$ , the question is a pairwise comparison; when  $k = 1, l > 2$ , the question is to query an agent's top alternative among a subset of alternatives; when  $k = l - 1$ , we are asking a full ranking over a subset of alternatives. The advantage of this type of questions is that the probabilities of responses of these questions are easy to compute, as well as their partial derivatives. W.l.o.g. let  $R_j = a_1 \succ a_2 \succ \dots \succ a_k \succ \text{others}$  be the answer from agent  $j$ . Then we have  $\Pr(R_j|B) = \prod_{p=1}^k \frac{\exp(u_{jp})}{\sum_{i=p}^l \exp(u_{ji})}$ , and  $\ln \Pr(R_j|B) = \sum_{p=1}^k (u_{jp} - \ln \sum_{i=p}^l \exp(u_{ji}))$ .

For any  $1 \leq \kappa \leq K$  and  $1 \leq \iota \leq L$ , let  $b_{\kappa\iota}$  be the  $(\kappa, \iota)$  entry of  $B$ . We have

$$\frac{\partial \ln \Pr(R_j|B)}{\partial b_{\kappa\iota}} = \sum_{p=1}^k \left( \frac{\partial u_{jp}}{\partial b_{\kappa\iota}} - \frac{\sum_{i=p}^l \exp(u_{ji}) \frac{\partial u_{ji}}{\partial b_{\kappa\iota}}}{\sum_{i=p}^l \exp(u_{ji})} \right),$$

where  $\frac{\partial u_{jp}}{\partial b_{\kappa\iota}}$  and  $\frac{\partial u_{ji}}{\partial b_{\kappa\iota}}$  are constants (products of an agent's attribute and an alternative's attribute) by definition. Therefore, for diagonal entries, the second order partial derivatives are given by

$$\frac{\partial^2 \ln \Pr_j(R|B)}{\partial b_{\kappa\iota}^2} = \sum_{p=1}^k \left( \left( \frac{\sum_{i=p}^l \exp(u_{ji}) \frac{\partial u_{ji}}{\partial b_{\kappa\iota}}}{\sum_{i=p}^l \exp(u_{ji})} \right)^2 - \frac{\sum_{i=p}^l \exp(u_{ji}) \left( \frac{\partial u_{jp}}{\partial b_{\kappa\iota}} \right)^2}{\sum_{p=1}^l \exp(u_{jp})} \right),$$

and for non-diagonal entries, we have

$$\begin{aligned} & \frac{\partial^2 \ln \Pr_j(R|B)}{\partial b_{\kappa_1 \iota_1} \partial b_{\kappa_2 \iota_2}} \\ &= \sum_{p=1}^k \left( \frac{(\sum_{i=p}^l \exp(u_{ji}) \frac{\partial u_{ji}}{\partial b_{\kappa_1 \iota_1}}) (\sum_{i=p}^l \exp(u_{ji}) \frac{\partial u_{ji}}{\partial b_{\kappa_2 \iota_2}})}{(\sum_{i=p}^{m'} e^{u_{ji}})^2} \right. \\ & \left. - \frac{\sum_{i=p}^l \exp(u_{ji}) \left( \frac{\partial u_{ji}}{\partial b_{\kappa_1 \iota_1}} \right) \left( \frac{\partial u_{ji}}{\partial b_{\kappa_2 \iota_2}} \right)}{\sum_{i=p}^l \exp(u_{ji})} \right). \end{aligned}$$

### 3.3 INFORMATION CRITERIA

An information criterion maps the distribution of a parameter to a real-valued quality. Standard information

criteria are mostly directly computed from the covariance matrix  $J^{-1}(\vec{\beta})$  or its inverse  $J(\vec{\beta})$ . For example, D-optimality [Wald, 1943, Mood et al., 1946] computes the determinant of  $J(\vec{\beta})$ ; E-optimality [Ehrenfeld, 1955] computes the minimum eigenvalue of  $J(\vec{\beta})$ . We propose the following minimum pairwise certainty (MPC) criterion by extending the criterion from [Azari Soufiani et al., 2013] to our domain.

**MPC for Case  $n_1 = 1$ .** We consider two types of purposes: predicting the agent's (unordered) top  $k$  alternatives and predicting the agent's ranked top  $k$  alternatives. We note that the criterion by Azari Soufiani et al. [2013] only applies to full rankings, which is a special case of our ranked top  $k$ . The intuition of this criterion is to maximize the certainty of the least certain pairwise comparison among a subset of pairwise comparisons. Formally, let  $\mathcal{A}_k$  denote the set of predicted top  $k$  alternatives for this key agent.

- Unordered top- $k$  where  $1 \leq k < m$ :

$$G(\pi(\vec{\beta})) = \min_{i_1 \in \mathcal{A}_k, i_2 \notin \mathcal{A}_k} \frac{|\text{mean}(u_{1i_1} - u_{1i_2})|}{\text{std}(u_{1i_1} - u_{1i_2})}.$$

- Ranked top- $k$  where  $1 < k < m$ :

$$G(\pi(\vec{\beta})) = \min_{i_1 \in \mathcal{A}_k, i_2 \neq i_1} \frac{|\text{mean}(u_{1i_1} - u_{1i_2})|}{\text{std}(u_{1i_1} - u_{1i_2})}.$$

In the above equations,  $\text{mean}(u_{ji_1} - u_{ji_2})$  is computed using  $\vec{\beta}_{\text{CML}}$  and  $\text{std}(u_{ji_1} - u_{ji_2})$  is computed using the approximated covariance matrix  $J^{-1}(\vec{\beta})$  as follows.

Because  $u_{ji_1} - u_{ji_2}$  is linear with  $\vec{\beta}$  (see Equation (1) and recall that  $\vec{\beta}$  is the vectorization of  $B$ ), we write it as  $u_{ji_1} - u_{ji_2} = \sum_{\kappa, \iota} c_{\kappa\iota} b_{\kappa\iota}$ , where  $c_{\kappa\iota}$ 's are constants computed from attributes of  $a_{i_1}, a_{i_2}$  and agent  $j$ . Then we have  $\text{std}(u_{ji_1} - u_{ji_2}) = \sqrt{\sum_{(\kappa_1, \iota_1), (\kappa_2, \iota_2)} c_{\kappa_1 \iota_1} c_{\kappa_2 \iota_2} \text{Cov}(b_{\kappa_1, \iota_1}, b_{\kappa_2, \iota_2})}$ . When  $\kappa_1 = \kappa_2 = \kappa$  and  $\iota_1 = \iota_2 = \iota$ ,  $\text{Cov}(b_{\kappa_1, \iota_1}, b_{\kappa_2, \iota_2})$  reduces to  $\text{Var}(b_{\kappa\iota})$ . Both  $\text{Cov}(b_{\kappa_1, \iota_1}, b_{\kappa_2, \iota_2})$  and  $\text{Var}(b_{\kappa\iota})$  are entries of  $J^{-1}(\vec{\beta})$ .

**MPC for Case  $n_1 \geq 2$ .** Our MPC for this case is different from the criterion by Azari Soufiani et al. [2013] in that we find the least certain pairwise comparison across all agents in the key group. Formally, our MPC for  $n_1 \geq 2$  is

$$G(\pi(\vec{\beta})) = \min_{j \in \{1, \dots, n_1\}, i_2 \neq i_1} \frac{|\text{mean}(u_{ji_1} - u_{ji_2})|}{\text{std}(u_{ji_1} - u_{ji_2})}, \quad (3)$$

where the computation of  $\text{mean}(u_{ji_1} - u_{ji_2})$  and  $\text{std}(u_{ji_1} - u_{ji_2})$  are similar to the  $n_1 = 1$  case.

## 4 RANDOMIZED VOTING RULES

We use randomized voting rules to aggregate the key group's preferences. A randomized voting rule computes the distribution of winners given the preferences of the agents from the key group. Under non-deterministic preferences, this distribution can be computed from the parameter of the model. This section shows that under the Plackett-Luce model with features, probabilistic plurality and probabilistic Borda are easy to compute.

A randomized voting rule assigns a probability for each alternative to be the winner according to the data, usually based on a scoring function. For example, the probabilistic plurality rule, which is equivalent to random dictatorship [Gibbard, 1977], samples a winner from a distribution where the probability of each alternative being the winner is proportional to the plurality score of this alternative. Other randomized voting rules can be defined similarly, including probabilistic Borda [Heckelman, 2003]. The voting rule must have scores associated with it, but this is a very mild restriction because many commonly-studied voting rules including all positional scoring rules, Copeland, range voting, and approval voting, have randomized counterparts.

Recall that  $n_1$  agents are making a group decision among  $m$  alternatives. Let  $P$  denote the preference profile that consists of  $n_1$  full rankings over  $m$  alternatives from the key group. Let  $s_r(a_i, P)$  denote the score of alternative  $a_i$  under voting rule  $r$  and  $\Pr_r(a_i|P)$  be the probability for  $a_i$  to win under the randomized analogy of  $r$  given  $P$ . Then  $\Pr_r(a_i|P)$  is computed by  $\Pr_r(a_i|P) = \frac{s_r(a_i, P)}{\sum_{i=1}^m s_r(a_i, P)}$ .

**Example 1** Suppose the set of alternatives is  $\{a_1, a_2, a_3\}$  and the votes are  $\{a_1 \succ a_2 \succ a_3, a_1 \succ a_3 \succ a_2, a_2 \succ a_1 \succ a_3\}$ . The plurality and Borda scores are shown in Table 1. Under probabilistic plurality rule,  $a_1$  wins with probability  $2/3$  and  $a_2$  wins with probability  $1/3$ . Under probabilistic Borda,  $a_1, a_2, a_3$  win with probabilities  $5/9, 3/9, 1/9$  respectively.

	$a_1$	$a_2$	$a_3$
plurality	2	1	0
Borda	5	3	1

Table 1: Scores under plurality and Borda

We consider non-deterministic preferences from agents, where the preferences from the key agents are independent of each other. Because each agent has  $m!$  possible rankings, there are  $(m!)^{n_1}$  possible preference profiles. Then we have  $\Pr_r(a_i) = \sum_{q=1}^{(m!)^{n_1}} \Pr(P_q) \Pr_r(a_i|P_q)$ ,

where  $P_q$  denotes the  $q$ -th possible preference profile.

Given a voting rule  $r$ , let  $X_{ji}$  be the score of  $a_i$  for agent  $j$ .  $X_{ji}$  is a random variable due to the uncertainty of agent  $j$ 's preference. The following theorem shows that the probability of  $a_i$  being the winner is proportional to the sum of expected score of  $a_i$  for each agent.

**Theorem 1** For any  $1 \leq i \leq m$ ,  $\Pr_r(a_i) \propto \sum_j^{n_1} EX_{ji}$ .

**Proof:** It suffices to prove  $\Pr_r(a_1) \propto \sum_j^{n_1} EX_{j1}$ .

By definition,  $\Pr_r(a_1) = \sum_{q=1}^{(m!)^{n_1}} \Pr(P_q) \Pr_r(a_1|P_q)$ . Let  $S$  denote the score of  $a_1$  under rule  $r$ . Then  $S$  is a random variable defined over the  $(m!)^{n_1}$  cases. Let  $s_q$  denote the value that  $S$  takes for case  $q$ . In any case  $q$ , we have  $\Pr_r(a_1|P_q) \propto s_q$  by the definition of randomized voting rules. We re-write it as  $\Pr_r(a_1|P_q) = \frac{s_q}{M}$ , where  $M$  is the normalization factor. Observe that across all the  $(m!)^{n_1}$  cases,  $M$  does not change because the voting rule  $r$  and the set of agents does not change. So we have

$$\begin{aligned} \Pr_r(a_1) &= \frac{\sum_{q=1}^{(m!)^{n_1}} \Pr(P_q) s_q}{M} \\ &\propto \sum_{q=1}^{(m!)^{n_1}} \Pr(P_q) s_q = ES. \end{aligned} \quad (4)$$

Since  $S = \sum_{j=1}^{n_1} X_{j1}$ , due to linearity of expectation, we have  $ES = E[\sum_{j=1}^{n_1} X_{j1}] = \sum_j^{n_1} EX_{j1}$ . By (4), we have  $\Pr_r(a_1) \propto \sum_j^{n_1} EX_{j1}$ . ■

For probabilistic plurality, the expected score of  $a_i$  for agent  $j$  is exactly the probability of  $a_i$  being ranked at the top by agent  $j$ . So we have the following corollary:

**Corollary 1** Let  $p_j^{a_i}$  be the probability of  $a_i$  being ranked at the top by agent  $j$ . For any  $1 \leq i \leq m$ ,  $\Pr_{\text{plurality}}(a_i) = \frac{1}{n_1} \sum_j^{n_1} p_j^{a_i}$ .

**Theorem 2** Let  $p_j^{a_i \succ a_{i'}}$  denote the probability for agent  $j$  to prefer alternative  $a_i$  over  $a_{i'}$ . Then for any  $1 \leq i \leq m$ ,  $\Pr_{\text{Borda}}(a_i) \propto \sum_j^{n_1} \sum_{i' \neq i} p_j^{a_i \succ a_{i'}}$ .

**Proof:** For any  $i \in \{1, \dots, m\}$ , we have  $\Pr_{\text{Borda}}(a_i) \propto \sum_j^{n_1} EX_{ji}$  by Theorem 1, where  $X_{ji}$  here denotes the score of  $a_i$  for agent  $j$  under Borda. We only need to prove  $EX_{ji} = \sum_{i' \neq i} p_j^{a_i \succ a_{i'}}$ . This is a known result, but we were not able to find a formal proof in literature, except a proof for three alternatives by Chen and Heckelman [2005], which is easy to be extended for arbitrary number of alternatives. For completeness we provide a short proof.

By definition of Borda, we have  $EX_{ji} = \sum_{k=1}^{m-1} (m-k) \sum_{R: a_i \text{ at } k\text{th position of } R} \Pr_j(R)$ , where  $R$  is any full

ranking over the  $m$  alternatives and  $\Pr_j(R)$  is the probability of  $R$  by agent  $j$ . Imagine  $m-1$  bins, each of which is labeled with  $a_i \succ a_{i'}$  for all the remaining  $m-1$   $a_{i'}$ 's. Observe that there are  $m-k$  copies of  $\Pr_j(R)$  for all  $R$  where  $a_i$  beats exactly  $m-k$  other alternatives. We can distribute the  $m-k$  copies to the  $m-k$  bins (one in each) for all  $a_{i'}$ 's that are ranked after  $a_i$ . We do this for all possible rankings and in the end, each bin labeled by  $a_i \succ a_{i'}$  gets the probabilities of all rankings compatible with  $a_i \succ a_{i'}$ . This finishes the proof. ■

We note that for the Plackett-Luce model,  $p_j^{a_i}$ 's and  $p_j^{a_i \succ a_{i'}}$ 's are easy to compute (see Section 2.1).

## 5 EXPERIMENTS

We first introduce an example of empirically estimating the cost of asking different types of questions on MTurk. Then, we show the result of a simulation of cost-effective preference elicitation using synthetic data.

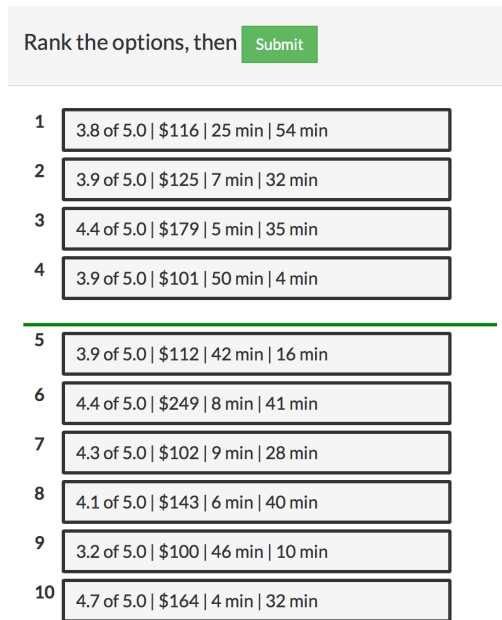


Figure 3: The user interface for a Turker to submit her ranked top 4 over 10 alternatives. The attributes are average ratings, prices per night, time to Times Square, and time to the nearest airport.

### 5.1 ESTIMATING $w(h)$

We recall that a question is defined by a pair of parameters  $(k, l)$ , where  $l$  is the number of alternatives that are presented to an agent and  $k$  is the number of alternatives that the agent is asked to rank at the top  $k$  positions.

In order to map the question types to the time to answer them, we run 2 experiments with multiple tasks on MTurk. Each task required MTurk workers to report their preferences over a set of hotels. We recorded the time they spent on each task, in order to learn such mapping in the following two cases:

- $k, l \in [2, 10], k = l - 1$ : full rankings;
- $k \in [1, 10], l = 10$ : ranked top  $k$  alternatives over 10.

**Experiment Setting.** For the first case, we looked for information on the first 54 Hotels in New York City in alphabetical order. We split the 54 randomly into 9 sets, each containing 2, 3, ..., 10 hotels. We then showed the 9 sets to MTurk workers, randomizing the order of the 9 sets as well as the initial display order of alternatives within each set, and asked them to rearrange by drag-and-dropping the alternatives according to their preferences. The alternatives were anonymous and represented by 4 attributes: average guest rating on a popular travel website, price per night, time to Times Square and time to the nearest airport.

For the second case, a separate experiment is run with another 10 sets of NYC hotels, drawn randomly again from the first 100 hotels in NYC in alphabetical order. In each task, we placed a horizontal green bar underneath the alternative above which are the  $k$  alternatives of interest. We instructed the MTurk workers before the experiment started that only the alternatives above the green bar would count, i.e. only needed to rank-order top- $k$ . All of these were done with goal of minimizing the overhead time for workers to understand the instruction so that the recorded time accurately reflect the time of decision-making. An example of the UI is show in Figure 3, where we asked Turkers to rank-order her top-4 favorite hotels over a set of 10.

The following analysis is made possible by responses from 408 MTurk workers (202 for the first case and 206 for the second).

**Experiment Results.** Although Volkov [2014] considered both linear and quadratic cost function and argued for the superiority of the latter, for simplicity, we perform a linear regression on the dataset to obtain a linear cost function. We regress the time to rearrange the alternatives and submit a full ranking on the number of alternatives in the set. We find that, on average, the time a Turker spent on rank-ordering a full ranking over  $l$  alternatives is  $t_{\text{full-}l} = 5.33l$  (Figure 2 left), and that on rank-ordering her top- $k$  alternatives over 10 alternatives is  $t_{\text{top-}k} = 1.38k + 32.25$  (Figure 2 right). In addition, the 408 workers spent an average of 341.5 seconds on the tasks and were each paid \$0.3. Therefore, the monetary cost of elicitation on average is  $w = 0.00088t$ , which correspond to an hourly wage of \$3.16.



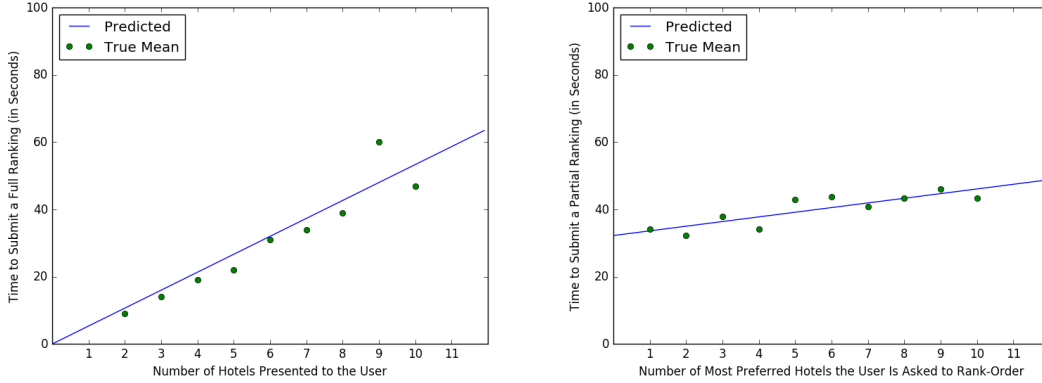


Figure 2: The left subfigure shows the average time a user spent to submit a full ranking over 2, . . . , 10 alternatives; the right subfigure shows the average time a user spent to give her ranked top 1, . . . , 10 alternatives when 10 alternatives were proposed.

Combining these two functions with the hourly wage, we propose the following cost functions, which estimates the cost (in USD) of elicitation about hotel preferences given 4 alternative attributes:  $w_{\text{full-}l} = 0.0047l$  and  $w_{\text{top-}k} = 0.0012k + 0.028$ . We observe that the time a user spent is not very sensitive to  $k$ . This is sensible, as when a MTurk worker ranks her top  $k$  choices, she may follow the following procedure: 1. read the descriptions of all hotels, 2. form their preferences, and 3 choose top  $k$ . Step 1 and 2 do not depend on  $k$  and dominates the time for step 3, as illustrated in the right figure of Figure 2. This suggests that when a fixed number of alternatives is proposed to an agent, it’s likely that the most cost-effective question to ask is a full ranking, as we will see in the next subsection.

## 5.2 COST-EFFECTIVE PREFERENCE ELICITATION

We demonstrate the viability of our cost-effective framework and compare performances of different information criteria on synthetic data.

**Synthetic Data.** We randomly generated 10 alternatives, each of which has 3 attributes, independently normally distributed  $N(0, 1)$ . We then randomly generated 5 agents that forms the key group and 20 the regular group. Each agent also has 3 attributes, independently normally distributed  $N(0, 1)$ .  $B$  was generated from Dirichlet distribution  $\text{Dir}(\vec{1})$ . The result is averaged over 400 trials.

To echo the motivating example from the beginning of this paper, we simulated the process of eliciting key group’s preference by asking agents in the regular group questions. For simplicity, we consider 3 types of ques-

tions, represented in  $(k, l)$ :  $(1, 2)$ ,  $(1, 10)$  and  $(9, 10)$ . We run Algorithm 1 using 3 different information criteria: D-Optimality, E-Optimality, and the proposed MPC. The three elicitation processes utilized the cost function estimated in Section 5.1. They were initialized with the same set of 50 randomly generated pairwise comparisons and was given a \$0.9 budget. Agents’ answers to the elicitation questions are generated from the Plackett-Luce model.

**Metrics.** We use total variation distance to measure the difference between the winner distributions computed from the ground truth parameter and the estimates, denoted by  $\psi^*$  and  $\psi$  respectively. The total variance distance is defined as  $\delta(\psi^*, \psi) = \frac{1}{2} \sum_{i=1}^m |\psi^*(a_i) - \psi(a_i)|$ . To plot the results, at each cost  $w$ , we used the data point that is below  $w$  but closest to  $w$  in each trial. These points were averaged over all trials.

**Observations.** We observe that for both probabilistic plurality and probabilistic Borda, the performances of MPC, D-optimality, and E-optimality are similar, all of which significantly outperforms random elicitation questions (see Figures 4). For example, for probabilistic plurality and probabilistic Borda, at the budget of 0.85 dollars, MPC achieves 15% less total variation distance than that of random elicitation questions. As another example, to achieve the total variation distance of 0.064 under randomized plurality (respectively, randomized Borda), MPC uses 20% (23.5%) less money than that of random elicitation questions.

We also observe that D-optimality almost always choose a full ranking as the most cost-effective question, while MPC tends to choose more full rankings than pairwise comparisons at early stages (see Figure 5). Due to the budget limit, many trials finish after 19 iterations because

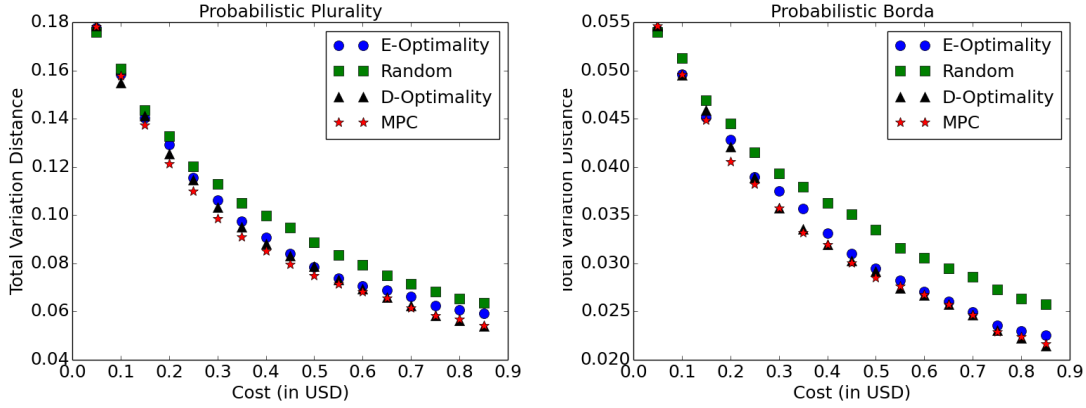


Figure 4: Total variation distance for probabilistic plurality (left) and probabilistic Borda (right).

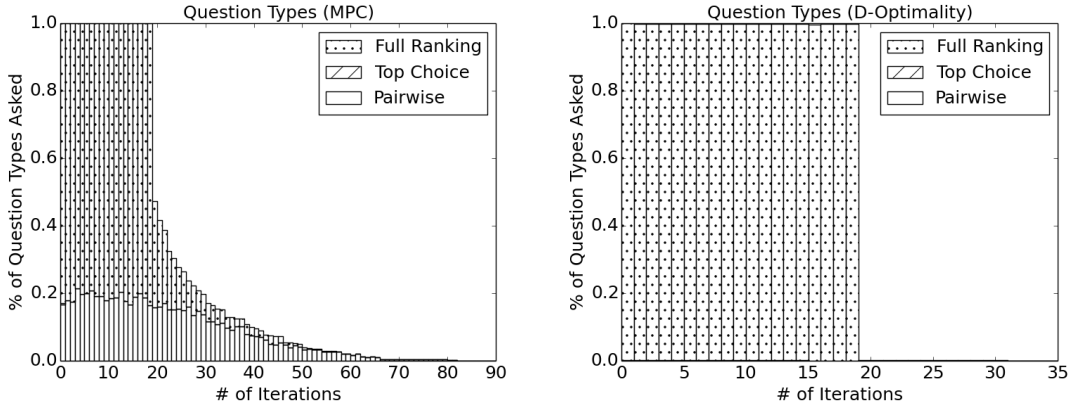


Figure 5: Types of questions chosen by the MPC (left) and D-optimality (right). The legend “Full Ranking”, “Top Choice”, and “Pairwise” correspond to  $(k = 9, l = 10)$ ,  $(k = 1, l = 10)$  and  $(k = 1, l = 2)$  respectively.

they only query full rankings. Others finish at different iterations. The distribution of types of questions for E-optimality is similar to MPC. Under all criteria except random,  $l = 10, k = 1$  questions were rarely asked.

**Discussions.** The meaning of cost-effectiveness in this paper is twofold: (1) in the preference elicitation procedure, we ask elicitation questions that is expected to provide more information per unit cost; and (2) the presence of regular group gives us a belief on the key group’s preferences inexpensively. As we have seen in our experiments, a budget of \$0.9 gives us a reasonably good estimate of the key group’s preferences by querying the regular group.

## 6 CONCLUSIONS AND FUTURE WORK

We proposed a flexible and cost-effective framework for preference elicitation that can be adapted for any ranking

model, any information criterion, and any set of questions. We used randomized voting rules to help make group decisions and proposed MPC for both prediction of one agent’s preference and aggregation of a group of agents’ preferences. Experiments show that MPC and other commonly-used information criteria work better than asking random elicitation questions. For future work we will explore better information criteria for group decisions. We also plan to extend this framework to multiple types of resource constraints.

## Acknowledgements

We thank all anonymous reviewers for helpful comments and suggestions. This work is supported by NSF #1453542 and ONR #N00014-17-1-2621.

## References

- Hossein Azari Soufiani, David C. Parkes, and Lirong Xia. Preference elicitation for general random utility models. In *Proceedings of Uncertainty in Artificial Intelligence (UAI)*, Bellevue, Washington, USA, 2013.
- Debarun Bhattacharjya and Jeffrey Kephart. Bayesian interactive decision support for multi-attribute problems with even swaps. In *Proceedings of the Thirtieth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-14)*, Corvallis, Oregon, 2014.
- Juergen Branke, Salvatore Corrente, Salvatore Greco, and Walter Gutjahr. Efficient pairwise preference elicitation allowing for indifference. *Computers & Operations Research*, 88:175–186, 2017.
- Urszula Chajewska, Daphne Koller, and Ron Parr. Making rational decisions using adaptive utility elicitation. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 363–369, Austin, TX, USA, 2000.
- Frederick H Chen and Jac C Heckelman. Winning probabilities in a pairwise lottery system with three alternatives. *Economic Theory*, 26(3):607–617, 2005.
- Joanna Drummond and Craig Boutilier. Preference elicitation and interview minimization in stable matchings. In *AAAI*, pages 645–653, 2014.
- Sylvain Ehrenfeld. On the efficiency of experimental designs. *The annals of mathematical statistics*, 26(2): 247–255, 1955.
- Seda Erdem and Danny Campbell. Preferences for public involvement in health service decisions: a comparison between best-worst scaling and trio-wise stated preference elicitation techniques. *The European Journal of Health Economics*, 18(9):1107–1123, 2017.
- Brochu Eric, Nando D Freitas, and Abhijeet Ghosh. Active preference learning with discrete choice data. In *Advances in neural information processing systems*, pages 409–416, 2008.
- Roman Garnett, Yamuna Krishnamurthy, Xuehan Xiong, Jeff Schneider, and Richard Mann. Bayesian optimal active search and surveying. In *Proceedings of the 29th International Conference on Machine Learning*, 2012.
- Allan Gibbard. Manipulation of schemes that mix voting with chance. *Econometrica*, 45:665–681, 1977.
- Jac C Heckelman. Probabilistic Borda rule voting. *Social Choice and Welfare*, 21(3):455–468, 2003.
- Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *stat*, 1050:24, 2011.
- Neil Houlsby, Jose Miguel Hernandez-Lobato, Ferenc Huszar, and Zoubin Ghahramani. Collaborative Gaussian processes for preference learning. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, pages 2105–2113, Lake Tahoe, NV, USA, 2012.
- Dongling Huang and Lan Luo. Consumer preference elicitation of complex products using fuzzy support vector machine active learning. *Marketing Science*, 35(3):445–464, 2016.
- Shali Jiang, Gustavo Malkomes, Geoff Converse, Alyssa Shofner, Benjamin Moseley, and Roman Garnett. Efficient nonmyopic active search. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1714–1723, Sydney, Australia, 06–11 Aug 2017. PMLR.
- Meir Kalech, Sarit Kraus, Gal A Kaminka, and Claudia V Goldman. Practical voting rules with partial information. *Autonomous Agents and Multi-Agent Systems*, 22(1):151–182, 2011.
- Bruce G Lindsay. Composite likelihood methods. *Contemporary Mathematics*, 80, 1988.
- Benedikt Loepp, Tim Hussein, and Jüergen Ziegler. Choice-based preference elicitation for collaborative filtering recommender systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3085–3094. ACM, 2014.
- Tyler Lu and Craig Boutilier. Robust approximation and incremental elicitation in voting protocols. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI)*, pages 287–293, Barcelona, Catalonia, Spain, 2011a.
- Tyler Lu and Craig Boutilier. Vote elicitation with probabilistic preference models: Empirical estimation and cost tradeoffs. In *International Conference on Algorithmic Decision Theory*, pages 135–149. Springer, 2011b.
- Alexander M Mood et al. On hotelling’s weighing problem. *The Annals of Mathematical Statistics*, 17(4): 432–446, 1946.
- Francesco Pauli, Walter Racugno, and Laura Ventura. Bayesian composite marginal likelihoods. *Statistica Sinica*, pages 149–164, 2011.
- Thomas Pfeiffer, Xi Alice Gao, Andrew Mao, Yiling Chen, and David G. Rand. Adaptive polling and information aggregation. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 122–128, Toronto, Canada, 2012.
- Baharak Rastegari, Paul Goldberg, and David Manlove. Preference elicitation in matching markets via inter-

- views: A study of offline benchmarks. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 1393–1394, 2016.
- Tuomas Sandholm and Craig Boutilier. Preference elicitation in combinatorial auctions. In Peter Cramton, Yoav Shoham, and Richard Steinberg, editors, *Combinatorial Auctions*, chapter 10, pages 233–263. MIT Press, 2006.
- Oleg Volkov. *Optimal Relaxed Designs of Experiments, with Pharmaceutical Applications*. PhD thesis, Queen Mary University of London, 2014.
- Abraham Wald. On the efficient design of statistical investigations. *The annals of mathematical statistics*, 14(2):134–140, 1943.
- Marieke GM Weernink, Sarah IM Janus, Janine A van Til, Dennis W Raisch, Jeannette G van Manen, and Maarten J IJzerman. A systematic review to identify the use of preference elicitation methods in health-care decision making. *Pharmaceutical medicine*, 28(4):175–185, 2014.
- Stephen E Wright, Belle M Sigal, and A John Bailer. Workweek optimization of experimental designs: exact designs for variable sampling costs. *Journal of Agricultural, Biological and Environmental Statistics*, 15(4):491–509, 2010.
- Zhibing Zhao and Lirong Xia. Composite marginal likelihood methods for random utility models. In *Proceedings of the 35th International Conference on Machine Learning*, Stockholm, Sweden, 2018.

---

# Incremental Learning-to-Learn with Statistical Guarantees

---

**Giulia Denevi**<sup>1,2</sup>   **Carlo Ciliberto**<sup>3</sup>   **Dimitris Stamos**<sup>3</sup>   **Massimiliano Pontil**<sup>1,3</sup>  
giulia.denevi@iit.it   c.ciliberto@ucl.ac.uk   d.stamos.12@ucl.ac.uk   massimiliano.pontil@iit.it

<sup>1</sup> Computational Statistics and Machine Learning, Istituto Italiano di Tecnologia, 16163 Genova, Italy

<sup>2</sup> Department of Mathematics, University of Genova, 16146 Genova, Italy

<sup>3</sup> Department of Computer Science, University College of London, WC1E 6BT, London, United Kingdom

## Abstract

In learning-to-learn the goal is to infer a learning algorithm that works well on a class of tasks sampled from an unknown meta-distribution. In contrast to previous work on batch learning-to-learn, we consider a scenario where tasks are presented sequentially and the algorithm needs to adapt incrementally to improve its performance on future tasks. Key to this setting is for the algorithm to rapidly incorporate new observations into the model as they arrive, without keeping them in memory. We focus on the case where the underlying algorithm is Ridge Regression parametrised by a symmetric positive semidefinite matrix. We propose to learn this matrix by applying a stochastic strategy to minimize the empirical error incurred by Ridge Regression on future tasks sampled from the meta-distribution. We study the statistical properties of the proposed algorithm and prove non-asymptotic bounds on its excess transfer risk, that is, the generalization performance on new tasks from the same meta-distribution. We compare our online learning-to-learn approach with a state-of-the-art batch method, both theoretically and empirically.

## 1 INTRODUCTION

Learning-to-learn (LTL) or meta-learning aims at finding an algorithm that is best suited to address a class of learning problems (tasks). These tasks are sampled from an unknown meta-distribution and are only partially observed via a finite collection of training examples, see (Baxter, 2000; Maurer, 2005; Thrun & Pratt, 1998) and references therein. This problem plays a large role in artificial intelligence in that it can improve the efficiency

of learning from human supervision. In particular, substantial improvement over “learning in isolation” (also known as independent task learning, ITL) is to be expected when the sample size per task is small, a setting which naturally arises in many applications, see e.g. (Camoriano et al., 2017; Rebuffi et al., 2017; Rohrbach et al., 2013).

LTL is particularly appealing when considered from an online or incremental perspective. In this setting, which is sometimes referred to as lifelong learning, see e.g. (Ruvolo & Eaton, 2013), the tasks are observed sequentially – via corresponding sets of training examples – from a common environment and we aim to improve the learning ability of the underlying algorithm on future yet-to-be-seen tasks from the same environment. Practical scenarios of lifelong learning are wide ranging, including computer vision (Rebuffi et al., 2017), robotics (Camoriano et al., 2017), user modelling and many more.

Although LTL is naturally suited for the incremental setting, surprisingly, theoretical investigations are lacking. Previous studies, starting from the seminal paper (Baxter, 2000) and (Maurer, 2009; Maurer et al., 2013; 2016; Pentina & Lampert, 2014), have almost exclusively considered the setting in which the tasks are given in one batch, that is, the meta-algorithm processes multiple datasets from the environment jointly and only once as opposed to sequentially and indefinitely.

The papers (Balcan et al., 2015; Herbster et al., 2016) present results in an online framework which applies to a finite number of tasks using different performance measures. Perhaps most related to our work is (Alquier et al., 2017), where the authors consider a general PAC-Bayesian approach to lifelong learning based on the exponentially weighted aggregation procedure. Unfortunately, this approach is not efficient for large scale applications as it entails storing the entire sequence of datasets during the meta-learning process.

LTL also bears strong similarity to multi-task learning

(MTL), see e.g. (Caruana, 1997), and much work has been done on the theoretical study of both batch (Ando & Zhang, 2005; Maurer et al., 2013) and online (Cavallanti et al., 2010) multi-task learning algorithms. However multi-task learning aims to solve the different problem of learning well on a prescribed set of tasks (the learned model is tested on the same tasks used during training) whereas LTL aims to extrapolate to new tasks.

The principal contribution of this paper is to propose an incremental approach to learning-to-learn and to analyse its statistical guarantees. This incremental approach is appealing in that it efficiently processes one dataset at the time, without the need to store previously encountered datasets. We study in detail the case of linear representation learning, in which an underlying learning algorithm receives in input a sequence of datasets and incrementally updates the data representation so as to better learn future tasks. Following previous work on LTL, e.g. (Baxter, 2000; Maurer, 2009), we measure the performance of the incremental meta-algorithm by the *transfer risk*, namely the average error obtained by running the underlying algorithm with the learned representation, over tasks sampled from the meta-distribution.

Specifically, in this work we choose the underlying algorithm to be Ridge Regression parametrised by a symmetric positive semidefinite matrix. The incremental LTL approach we propose aims at optimizing the *future empirical error* (Maurer, 2009; Maurer et al., 2016) incurred by Ridge Regression over a class of linear representations. For this purpose, we propose to apply Projected Stochastic Subgradient Algorithm (PSSA). We show that the objective function of the resulting meta-algorithm is convex and we give a non-asymptotic convergence rate for the algorithm in high probability. A remarkable feature of our learning bound is that it is comparable to previous bounds for batch LTL. Our proof technique leverages previous work on learning-to-learn (Maurer, 2009) with tools from online convex optimization, see (Cesa-Bianchi et al., 2004; Hazan, 2016) and references therein.

The paper is organized as follows. In Sec. 2, we review the LTL problem and describe in detail the case of linear feature learning with Ridge Regression. In Sec. 3, we present our incremental meta-algorithm for linear feature learning. Sec. 4 contains our bound on the excess transfer risk for the proposed algorithm and in Sec. 5 we compare the bound to a previous bound for the batch setting. In Sec. 6, we report preliminary numerical experiments for the proposed algorithm and, finally, Sec. 7 summarizes the paper and highlight directions of future research. The detailed proofs of the statements in the paper are reported in the appendix.

## 2 PROBLEM FORMULATION

In the standard independent task learning setting the goal is to learn a functional relation between an input space  $\mathcal{X}$  and an output space  $\mathcal{Y}$  from a finite number of training examples. More precisely, given a loss function  $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$  measuring prediction errors and given a distribution  $\mu$  on the joint data space  $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ , the goal is to find a function  $f : \mathcal{X} \rightarrow \mathcal{Y}$  minimizing the *expected risk*

$$\mathcal{R}_\mu(f) = \mathbb{E}_{z \sim \mu} \ell(f, z) \quad (1)$$

where, with some abuse of notation, for any  $z = (x, y) \in \mathcal{Z}$  we denoted  $\ell(f, z) = \ell(f(x), y)$ . In most practical situations the underlying distribution is *unknown* and the learner is only provided with a finite set  $Z = (z_i)_{i=1}^n \in \mathcal{Z}^n$  of observations independently sampled from  $\mu$ . The goal of a learning algorithm is therefore, given such a *training* dataset  $Z$  to return a “good” estimator  $A(Z) = f_Z$  whose expected risk is small and tends to the minimum of Eq. (1) as  $n$  increases.

A well-established approach to tackle the learning problem is offered by *regularized empirical risk minimization*. This corresponds to the family of algorithms  $A_\phi$  such that, for any  $Z \in \mathcal{Z}^n$ ,

$$A_\phi(Z) = \operatorname{argmin}_{f \in \mathcal{F}_\phi} \mathcal{R}_Z(f) + \lambda \|f\|_{\mathcal{F}_\phi}^2 \quad (2)$$

where  $\phi : \mathcal{X} \rightarrow \mathcal{F}_\phi$  is a feature map,  $\mathcal{F}_\phi$  is the Hilbert space of functions  $f : \mathcal{X} \rightarrow \mathcal{Y}$  such that  $f(x) = \langle f, \phi(x) \rangle_{\mathcal{F}_\phi}$  for any  $x \in \mathcal{X}$  and

$$\mathcal{R}_Z(f) = \frac{1}{n} \sum_{i=1}^n \ell(f, z_i)$$

denotes the *empirical risk* of function  $f$  on the set  $Z$ .

### 2.1 LINEAR FEATURE LEARNING

In this work we will focus on the case that  $\mathcal{Y} \subseteq \mathbb{R}$ ,  $\mathcal{X} \subseteq \mathbb{R}^d$ ,  $\ell$  is the square loss and  $\phi : \mathbb{R}^d \rightarrow \mathbb{R}^m$  is a *linear* feature map (also known as a representation), corresponding to the action  $\phi(x) = \Phi x$  of a matrix  $\Phi \in \mathbb{R}^{m \times d}$  on the input space. It is well known, see e.g. (Argyriou et al., 2008), that, setting  $D = \frac{1}{\lambda} \Phi^\top \Phi \in \mathbb{R}^{d \times d}$ , any problem of the form in Eq. (2) can be equivalently formulated as

$$A_D(Z) = \operatorname{argmin}_{w \in \operatorname{Ran}(D)} \mathcal{R}_Z(w) + w^\top D^\dagger w \quad (3)$$

where, with some abuse of notation, we denoted with  $\mathcal{R}_Z(w)$  the empirical risk of the linear function  $x \mapsto w^\top x$ , for any  $x \in \mathcal{X}$ . Here,  $D^\dagger$  denotes the pseudoinverse of  $D$ , which is symmetric positive semidefinite (PSD) but not necessarily invertible; when it is not

invertible the constraint requiring  $w$  to be in the range  $\text{Ran}(D) \subseteq \mathbb{R}^d$  of  $D$  is needed to grant the equivalence with Eq. (2). Since for any linear feature map  $\phi$  there exists a symmetric PSD matrix  $D$  such that Eq. (2) and Eq. (3) are equivalent, in the following we will refer to  $D$  as the *representation* used by algorithm  $A_D$ .

## 2.2 LEARNING TO LEARN $D$

A natural question is how to choose a good representation  $D$  for a given family of related learning problems. In this work we consider the approach of *learning* it from data. In particular, following the seminal work of (Baxter, 2000), we consider a setting where we are provided with an increasing number of tasks and our goal is to find a joint representation  $D$  such that the corresponding algorithm  $A_D$  is suited to address all such learning problems. The underlying assumption is that all the tasks that we observe *share a common structure* that algorithm  $A_D$  can leverage in order to achieve better prediction performance.

More formally, we assume that the tasks we observe are independently sampled from a meta-distribution  $\rho$  on the set of probability measures on  $\mathcal{Z}$ . According to the literature on the topic, see e.g. (Baxter, 2000; Maurer, 2005), we refer to the meta-distribution  $\rho$  as the *environment* and we identify each task sampled from  $\rho$  by its corresponding distribution  $\mu$ , from which we are provided with a *training* dataset  $Z \sim \mu^n$  of  $n$  points sampled independently from  $\mu$ . While it is possible to consider a more general setting, for simplicity in this work we study the case where for each task we sample the same fixed number  $n$  of training points. In line with the independent task learning setting, the goal of a “learning-to-learn” algorithm is therefore to find the best parameter  $D$  minimizing the so-called *transfer risk*

$$\mathcal{E}(D) = \mathbb{E}_{\mu \sim \rho} \mathbb{E}_{Z \sim \mu^n} \mathcal{R}_\mu(A_D(Z)) \quad (4)$$

over a set  $\mathfrak{D}$  of candidate representations. The term  $\mathcal{E}(D)$  is the expected risk that the corresponding algorithm  $A_D$ , when trained on the dataset  $Z$ , would incur *on average with respect to the distribution of tasks  $\mu$  induced by  $\rho$* . That is, to compute the transfer risk, we first draw a task  $\mu \sim \rho$  and a corresponding  $n$ -sample  $Z \in \mathcal{Z}^n$  from  $\mu^n$ , we then apply the learning algorithm to obtain an estimator  $A_D(Z)$  and finally we measure the risk of this estimator on the distribution  $\mu$ .

The problem of minimizing the transfer risk in Eq. (4) given a finite number  $T$  of training datasets  $Z_1, \dots, Z_T$  sampled from the corresponding tasks  $\mu_1, \dots, \mu_T$ , has been subject of thorough analysis in literature, see e.g. (Baxter, 2000; Maurer, 2005; Maurer et al., 2016). Most work has been focused on the so-called “batch” setting,

where all such training datasets are provided at once. However, by its nature, LTL is an ongoing (possibly never ending) process, with training datasets observed a few at the time. In such a scenario the meta-algorithm should allow for an evolving representation  $D$ , which improves over time as new datasets are observed. In the following we propose a meta-algorithm to learn  $D$  *on-line* with respect to the tasks, allowing us to transfer past experience about the environment in an efficient manner, *without requiring the memorization of training data*, which could be prohibitive in large scale applications. We will study the statistical guarantees of the proposed algorithm and compare it to its batch counterpart in terms of both theoretical and empirical performance.

## 2.3 CONNECTION WITH MULTI-TASK LEARNING

LTL is strongly related to *multi-task learning* (MTL) and in fact, as we will see later for the algorithm in Eq. (3), approaches developed for MTL can be used as inspiration to design algorithms for LTL. In multi-task learning a fixed number of tasks  $\mu_1, \dots, \mu_T$  is provided up front and, given  $T$  datasets  $Z_1, \dots, Z_T$ , each sampled from its corresponding distribution, the goal is to find a joint representation  $D$  incurring a small *average expected risk*  $\frac{1}{T} \sum_{t=1}^T \mathcal{R}_{\mu_t}(A_D(Z_t))$ . In this sense, the main difference between LTL and MTL is that the former aims to guarantee good prediction performance on *future tasks*, while the latter aims to guarantee good prediction performance on the same tasks used to train  $D$ .

A well-established approach to MTL is *multi-task feature learning* (Argyriou et al., 2008). This method consists in solving the optimization problem

$$\min_{D \in \mathfrak{D}_\lambda} \frac{1}{T} \sum_{t=1}^T \min_{w \in \text{Ran}(D)} \mathcal{R}_{Z_t}(w_t) + w_t^\top D^\dagger w_t$$

over the set

$$\mathfrak{D}_\lambda = \{D \in \mathbb{S}_+^d \mid \text{tr}(D) \leq 1/\lambda\} \quad (5)$$

where  $\mathbb{S}_+^d$  denotes the set of  $d \times d$  symmetric PSD matrices,  $\text{tr}(D)$  is the trace of  $D$  and  $\lambda$  is a positive parameter which controls the degree of regularization. In the subsequent analysis the parameter  $\lambda$  must be intended as a fixed hyper-parameter, which will be chosen by cross-validation in the experiments. This choice for  $\mathfrak{D}_\lambda$  is motivated by the following variational form, see e.g. (Argyriou et al., 2008, Prop. 4.2), of the square trace norm of  $W = [w_1, \dots, w_T] \in \mathbb{R}^{d \times T}$

$$\|W\|_1^2 = \frac{1}{\lambda} \inf_{D \in \text{Int}(\mathfrak{D}_\lambda)} \sum_{t=1}^T w_t^\top D^{-1} w_t$$

where  $\text{Int}(\mathcal{D}_\lambda)$  is the interior of  $\mathcal{D}_\lambda$ , namely the set of the symmetric PSD invertible matrices with trace strictly smaller than  $1/\lambda$ . This leads to the equivalent problem

$$\min_{W \in \mathbb{R}^{d \times T}} \frac{1}{T} \sum_{t=1}^T \mathcal{R}_{Z_t}(w_t) + \gamma \|W\|_1^2 \quad (6)$$

with  $\gamma = \lambda/T$ . The trace norm of a matrix is defined as the sum ( $\ell_1$ -norm) of its singular values, and it is known to induce low-rank solutions for Problem (6). Intuitively, this means that tasks are encouraged to *share a common set of features (or representation)*. In this paper, we adopt this perspective to design our online LTL approach for linear feature learning.

### 3 ONLINE LEARNING-TO-LEARN

Motivated by the above connection with multi-task learning, we propose an online LTL approach to approximate the solution of the learning problem

$$\min_{D \in \mathcal{D}_\lambda} \mathcal{E}(D)$$

over the set  $\mathcal{D}_\lambda$  introduced in Eq. (5). We consider the setting in which we are provided with a stream of independent datasets  $Z_1, \dots, Z_T, \dots$ , each sampled from an individual task distribution  $\mu_1, \dots, \mu_T, \dots$  coming from the environment  $\rho$  and our goal is to find an estimator in  $\mathcal{D}_\lambda$  that improves *incrementally* as the number of observed tasks  $T$  increases.

#### 3.1 MINIMIZING THE EMPIRICAL TRANSFER RISK

A key observation motivating the online procedure proposed in this work, is that in the *independent task learning* setting, standard results from learning theory, see e.g. (Shalev-Shwartz & Ben-David, 2014), allow one to control the statistical performance of regularized empirical risk minimization, providing bounds on the *generalization error* of  $A_D$  as

$$\mathbb{E}_{Z \sim \mu^n} |\mathcal{R}_\mu(A_D(Z)) - \mathcal{R}_Z(A_D(Z))| \leq G(D, n) \quad (7)$$

where  $G(\cdot, n)$  is a decreasing function converging to 0 as  $n \rightarrow +\infty$ , while  $G(D, \cdot)$  is a measure of complexity of  $D$ , which is large for more “expressive” representations and smaller otherwise.

Eq. (7) suggests us to use the empirical risk  $\mathcal{R}_Z$  as a proxy for the expected risk  $\mathcal{R}_\mu$ . Therefore, we introduce the so-called *future empirical risk* (Maurer, 2009; Maurer et al., 2016),

$$\hat{\mathcal{E}}(D) = \mathbb{E}_{\mu \sim \rho} \mathbb{E}_{Z \sim \mu^n} \mathcal{R}_Z(A_D(Z))$$

---

#### Algorithm 1 PSSA applied to $\hat{\mathcal{E}}$

---

**Input:**  $T$  number of tasks,  $\lambda > 0$  hyper-parameter,  $\{\gamma_t\}_{t \in \mathbb{N}}$  step sizes.

**Initialization:**  $D^{(1)} \in \mathcal{D}_\lambda$

**For**  $t = 1$  to  $T$ :

Sample  $\mu_t \sim \rho, Z_t \sim \mu_t^n$ .

Choose  $U_t \in \partial \mathcal{L}_{Z_t}(D^{(t)})$

Update  $D^{(t+1)} = \text{proj}_{\mathcal{D}_\lambda}(D^{(t)} - \gamma_t U_t)$

**Return**  $\bar{D}_T = \frac{1}{T} \sum_{t=1}^T D^{(t)}$

---

and consider the related problem

$$\min_{D \in \mathcal{D}_\lambda} \hat{\mathcal{E}}(D), \quad (8)$$

which in the sequel, introducing the shorthand notation  $\mathcal{L}_Z(D) = \mathcal{R}_Z(A_D(Z))$  for any  $D \in \mathbb{S}_+^d$ , will be rewritten as

$$\min_{D \in \mathcal{D}_\lambda} \mathbb{E}_{\mu \sim \rho} \mathbb{E}_{Z \sim \mu^n} \mathcal{L}_Z(D) \quad (9)$$

to highlight the dependency on  $Z$ .

Problem (9) can be approached with stochastic optimization strategies. Such methods proceed by sequentially sampling a point (dataset in this case)  $Z$  and performing an update step. In recent years, stochastic optimization, finding its origin in the Stochastic Approximation method by (Robbins & Monro, 1951), has been effectively used to deal with large scale applications. We refer to (Nemirovski et al., 2009) for a more comprehensive discussion about this topic. We therefore propose to apply Projected Stochastic Subgradient Algorithm (PSSA) (Shamir & Zhang, 2013), to solve the optimization problem in Eq. (9). The candidate representation coincides in this case with the mean after  $T$  iterations  $\bar{D}_T$  and it is known as Polyak-Ruppert averaging scheme (Nemirovskii & Yudin, 1985; Polyak & Juditsky, 1992) in the optimization literature. Alg. 1 reports the application of PSSA to  $\hat{\mathcal{E}}$  when  $\mathcal{L}_Z$  is convex on the set  $\mathbb{S}_+^d$ . It requires iteratively: *i*) sampling a dataset  $Z$ , *ii*) performing a step in the direction of a subgradient of  $\mathcal{L}_Z$  at the current point, and *iii*) projecting onto the set  $\mathcal{D}_\lambda$  (which can be done in a finite number of iterations, see Lemma 16 in App. E). Note that in this case, since the function  $\mathcal{L}_Z$  is convex, there is no ambiguity in the definition of the subdifferential  $\partial \mathcal{L}_Z$ , see e.g. (Bertsekas et al., 2003), and we can rely on the convergence of Alg. 1 to a global minimum of  $\hat{\mathcal{E}}$  over  $\mathcal{D}_\lambda$  for a suitable choice of step-sizes, as discussed in Sec. 4.



### 3.2 LTL WITH RIDGE REGRESSION

In this work, we focus on the case that the loss function  $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$  corresponds to the square loss, namely  $\ell(y, y') = (y - y')^2$  for any  $y, y' \in \mathcal{Y} \subseteq \mathbb{R}$ . In this setting, given a dataset  $Z \in \mathcal{Z}^n$ , algorithm  $A_D$  is equivalent to perform the following variant to *Ridge Regression*

$$\min_{w \in \text{Ran}(D)} \frac{1}{n} \|\mathbf{y} - Xw\|^2 + w^\top D^\dagger w \quad (10)$$

where  $X \in \mathbb{R}^{n \times d}$  is the matrix with rows corresponding to the input points  $x_i \in \mathbb{R}^d$  in the dataset  $Z$  and  $\mathbf{y} \in \mathbb{R}^n$  the vector with entries equal to the corresponding output points  $y_i \in \mathbb{R}$ . The solution to Eq. (10) can be obtained in closed form, in particular, see e.g. (Argyriou et al., 2008; Maurer, 2009),

$$A_D(Z) = DX^\top (XDX^\top + nI)^{-1} \mathbf{y}. \quad (11)$$

Plugging this solution in the definition of  $\mathcal{L}_Z(D)$ , a direct computation yields that

$$\mathcal{L}_Z(D) = n \|(XDX^\top + nI)^{-1} \mathbf{y}\|^2. \quad (12)$$

The following result characterizes some key properties of the function  $\mathcal{L}_Z$  in Eq. (12), which will be useful in our subsequent analysis. We denote by  $\mathcal{B}_r \subseteq \mathbb{R}^d$  the ball of radius  $r > 0$  centered at 0.

**Proposition 1** (Properties of  $\mathcal{L}_Z$  for the Square Loss). *Let  $\mathcal{X} \subseteq \mathcal{B}_1$ ,  $\mathcal{Y} \subseteq [0, 1]$  and  $\ell$  be the square loss. Then, for any dataset  $Z \in \mathcal{Z}^n$  the following properties hold:*

1.  $\mathcal{L}_Z$  is convex on the set  $\mathbb{S}_+^d$ .
2.  $\mathcal{L}_Z$  is  $\mathcal{C}^\infty$  and, for every  $D \in \mathbb{S}_+^d$ ,

$$\nabla \mathcal{L}_Z(D) = -nX^\top M(D)^{-1} S(D) M(D)^{-1} X$$

where

$$\begin{aligned} M(D) &= XDX^\top + nI \\ S(D) &= \mathbf{y}\mathbf{y}^\top M(D)^{-1} + M(D)^{-1} \mathbf{y}\mathbf{y}^\top. \end{aligned}$$

3.  $\mathcal{L}_Z$  is 2-Lipschitz w.r.t. the Frobenius norm.
4.  $\nabla \mathcal{L}_Z$  is 6-Lipschitz w.r.t. the Frobenius norm.
5.  $\mathcal{L}_Z(D) \in [0, 1]$ , for any  $D \in \mathbb{S}_+^d$ .

The proposition above establishes the convexity of Problem (8) for the case of the square loss. This fact is important in that it guarantees no ambiguity in applying Alg. 1 to our setting and moreover, since  $\mathcal{L}_Z$  is differentiable, Alg. 1 becomes a *Projected Stochastic Gradient Algorithm*.

## 4 THEORETICAL ANALYSIS

In this section, we study the statistical properties of Alg. 1 for the case of the square loss. Below we report the main result of this work, which characterizes the non-asymptotic behavior of the estimator  $\bar{D}_T$  produced by Alg. 1 with respect to a minimizer  $D_* \in \text{argmin}_{D \in \mathcal{D}_\lambda} \mathcal{E}(D)$ . To present our results we introduce the  $d \times d$  matrix  $C_\rho = \mathbb{E}_{\mu \sim \rho} \mathbb{E}_{(x, y) \sim \mu} [xx^\top]$  denoting the covariance of the input data, obtained by averaging over all input marginals sampled from  $\rho$ . We also denote with  $\|C_\rho\|_\infty$  the operator norm of  $C_\rho$ , which corresponds to the largest eigen-value.

**Theorem 2** (Online LTL Bound). *Let  $\mathcal{X} \subseteq \mathcal{B}_1$ ,  $\mathcal{Y} \subseteq [0, 1]$  and  $\ell$  be the square loss. Let  $\mu_1, \dots, \mu_T$  be independently sampled from  $\rho$  and  $Z_t$  sampled from  $\mu_t^n$  for  $t \in \{1, \dots, T\}$ . Let  $\bar{D}_T$  be the output of Alg. 1 with step sizes  $\gamma_t = (\lambda\sqrt{2t})^{-1}$ . Then, for any  $\delta \in (0, 1]$*

$$\begin{aligned} \mathcal{E}(\bar{D}_T) - \mathcal{E}(D_*) &\leq \frac{4\sqrt{2\pi}\|C_\rho\|_\infty^{1/2}}{\sqrt{n}} \frac{1 + \sqrt{\lambda}}{\lambda} \\ &\quad + \frac{4\sqrt{2}}{\lambda\sqrt{T}} + \sqrt{\frac{8 \log(2/\delta)}{T}} \end{aligned}$$

with probability at least  $1 - \delta$  with respect to the independent sampling of the tasks  $\mu_t \sim \rho$  and training sets  $Z_t \sim \mu_t^n$  for any  $t \in \{1, \dots, T\}$ .

In Sec. 5, we will compare Thm. 2 with the statistical bound available for a state-of-the-art LTL batch procedure. We will see that the statistical behaviour of these two approaches is essentially equivalent, with the online LTL approach being more appealing given the lower requirements in terms of both number of computations and memory. In the rest of this section we give a sketch of the proof for Thm. 2. Proofs of intermediate results are reported in the appendix.

### 4.1 ERROR DECOMPOSITION

The statistical analysis of Alg. 1 hinges upon the following decomposition for the excess transfer risk of the estimator  $\bar{D}_T$ :

$$\begin{aligned} \mathcal{E}(\bar{D}_T) - \mathcal{E}(D_*) &= \mathcal{E}(\bar{D}_T) \pm \hat{\mathcal{E}}(\bar{D}_T) \pm \hat{\mathcal{E}}(D_*) - \mathcal{E}(D_*) \\ &\leq 2 \sup_{D \in \mathcal{D}_\lambda} |\mathcal{E}(D) - \hat{\mathcal{E}}(D)| + \hat{\mathcal{E}}(\bar{D}_T) - \hat{\mathcal{E}}(D_*) \\ &\leq 2 \underbrace{\sup_{D \in \mathcal{D}_\lambda} |\mathcal{E}(D) - \hat{\mathcal{E}}(D)|}_{\text{Uniform generalization error}} + \underbrace{\hat{\mathcal{E}}(\bar{D}_T) - \hat{\mathcal{E}}(\hat{D}_*)}_{\text{Excess future empirical risk}} \end{aligned} \quad (13)$$

where the matrix  $\hat{D}_*$  denotes a minimizer of the future transfer risk over  $\mathcal{D}_\lambda$ , that is,  $\hat{D}_* \in \text{argmin}_{D \in \mathcal{D}_\lambda} \hat{\mathcal{E}}(D)$ .

Eq. (13) decomposes  $\mathcal{E}(\bar{D}_T) - \mathcal{E}(D_*)$  in a *uniform generalization error*, implicitly encoding the complexity of the class of algorithms parametrised by  $D$  and an *excess future empirical risk*, measuring the discrepancy between the estimator  $\bar{D}_T$  and the minimizer  $\hat{D}_*$  of  $\hat{\mathcal{E}}$ . In the following we describe how to bound these two terms.

## 4.2 BOUNDING THE UNIFORM GENERALIZATION ERROR

Results providing generalization bounds for the class of regularized empirical risk minimization algorithms  $A_D$  considered in this work are well known. The following result, which is taken from (Maurer, 2009), leverages an explicit estimate of the generalization bound  $G(D, n)$  introduced in Sec. 3.1 for independent task learning, see Eq. (7), to obtain a uniform bound over the class of algorithms parametrized by  $\mathcal{D}_\lambda$ .

**Proposition 3** (Uniform Generalization Error Bound). *Let  $\mathcal{X} \subseteq \mathcal{B}_1$ ,  $\mathcal{Y} \subseteq [0, 1]$  and let  $\ell$  be the square loss, then*

$$\sup_{D \in \mathcal{D}_\lambda} |\mathcal{E}(D) - \hat{\mathcal{E}}(D)| \leq \frac{2\sqrt{2\pi}\|C_\rho\|_\infty^{1/2}}{\sqrt{n}} \frac{1 + \sqrt{\lambda}}{\lambda}.$$

For completeness, we report the proof of this proposition in App. B.3.

## 4.3 BOUNDING THE EXCESS FUTURE EMPIRICAL RISK

Providing bounds for the excess future empirical risk introduced in Eq. (13) consists in studying the convergence rates of Alg. 1 to the minimum of  $\hat{\mathcal{E}}$  over  $\mathcal{D}_\lambda$  in high probability with respect to the sample of  $T$  tasks  $\mu_t$  from  $\rho$  and datasets  $Z_t$  from  $\mu_t^n$  for any  $t \in \{1, \dots, T\}$ .

To this end, we leverage classical results from the online learning literature (Hazan, 2016). In online learning, the performance of an online algorithm returning a sequence  $\{D^{(t)}\}_{t=1}^T$  over  $T$  trials is measured in terms of its *regret*, which in the context of this work corresponds to

$$R_T = \frac{1}{T} \sum_{t=1}^T \mathcal{L}_{Z_t}(D^{(t)}) - \min_{D \in \mathcal{D}_\lambda} \frac{1}{T} \sum_{t=1}^T \mathcal{L}_{Z_t}(D).$$

Differently from the statistical setting considered in this work, in the online setting no assumption is made about the data generation process of  $Z_1, \dots, Z_T$ , which could be even adversely generated. Therefore, an algorithm that is able to solve the online problem (i.e. if its regret vanishes as  $T \rightarrow \infty$ ) can be also expected to solve the corresponding problem in the statistical setting. This is indeed the case for Alg. 1, for which the following lemma provides a non-asymptotic regret bound.

**Lemma 4** (Regret Bound for Alg. 1). *Let  $\mathcal{X} \subseteq \mathcal{B}_1$ ,  $\mathcal{Y} \subseteq [0, 1]$  and  $\ell$  be the square loss. Then the regret of Alg. 1 with step-sizes  $\gamma_t = (\lambda\sqrt{2t})^{-1}$  is such that*

$$R_T \leq \frac{4\sqrt{2}}{\lambda\sqrt{T}}.$$

The above lemma is a corollary of Prop. 1 combined with classical results on regret bounds for Projected Online Subgradient Algorithm (Hazan, 2016). We refer the reader to App. D.1 for a more in-depth discussion and for a detailed proof.

In our setting, the datasets  $Z_1, \dots, Z_T$  are assumed to be independently sampled from the underlying environment. Combining this assumption with the regret bound in Lemma 4, we can control the excess future empirical risk by means of so-called *online-to-batch conversion* results (Cesa-Bianchi et al., 2004; Hazan, 2016), leading to the following proposition.

**Proposition 5** (Excess Future Empirical Risk Bound for Alg. 1). *Let  $\mathcal{X} \subseteq \mathcal{B}_1$ ,  $\mathcal{Y} \subseteq [0, 1]$  and let  $\ell$  be the square loss. Let  $\mu_1, \dots, \mu_T$  be independently sampled from  $\rho$  and  $Z_t$  sampled from  $\mu_t^n$  for  $t \in \{1, \dots, T\}$ . Let  $\bar{D}_T$  be the output of Alg. 1 with step sizes  $\gamma_t = (\lambda\sqrt{2t})^{-1}$ . Then, for any  $\delta \in (0, 1]$*

$$\hat{\mathcal{E}}(\bar{D}_T) - \hat{\mathcal{E}}(\hat{D}_*) \leq \frac{4\sqrt{2}}{\lambda\sqrt{T}} + \sqrt{\frac{8 \log(2/\delta)}{T}}$$

with probability at least  $1 - \delta$  with respect to the independent sampling of the tasks  $\mu_t \sim \rho$  and training sets  $Z_t \sim \mu_t^n$  for any  $t \in \{1, \dots, T\}$ .

The result above follows by combining Prop. 1 with online-to-batch results, see e.g. (Hazan, 2016, Thm. 9.3) and (Cesa-Bianchi et al., 2004). In App. D.2 we provide the complete proof of this statement together with a more detailed discussion about this topic. At this point we are ready to give the proof of Thm. 2.

**Proof of Thm. 2.** The claim follows by combining Prop. 3 and Prop. 5 in the decomposition of the error  $\mathcal{E}(\bar{D}_T) - \mathcal{E}(D_*)$  given in Eq. (13). ■

## 5 ONLINE LTL VERSUS BATCH LTL

In this section, we compare the statistical guarantees obtained for our online meta-algorithm with a state-of-the-art batch LTL method for linear feature learning. We also comment on the computational cost of both procedures.

### 5.1 STATISTICAL COMPARISON

Given a finite collection  $\mathbf{Z} = \{Z_1, \dots, Z_T\}$  of datasets, a standard approach to approximate a minimizer of the

future empirical risk  $\hat{\mathcal{E}}$  is to take a representation  $\hat{D}_T$  minimizing the multi-task empirical risk

$$\hat{\mathcal{E}}_{\mathbf{Z}}(D) = \frac{1}{T} \sum_{t=1}^T \mathcal{R}_{Z_t}(A_D(Z_t)) \quad (14)$$

over the set  $\mathcal{D}_\lambda$ . Such a choice has been extensively studied in the LTL literature (Baxter, 2000; Maurer, 2009; Maurer et al., 2013; 2016). Here we report a result analogous to [Thm. 2](#), characterizing the discrepancy between the transfer risks of  $\hat{D}_T$  and  $D_*$ .

**Theorem 6** (Batch LTL Bound). *Let  $\mathcal{X} \subseteq \mathcal{B}_1$ ,  $\mathcal{Y} \subseteq [0, 1]$  and let  $\ell$  be the square loss. Let tasks  $\mu_1, \dots, \mu_T$  be independently sampled from  $\rho$  and  $Z_t$  sampled from  $\mu_t^n$  for  $t \in \{1, \dots, T\}$ . Let  $\hat{D}_T$  be a minimizer of the multi-task empirical risk in Eq. (14) over the set  $\mathcal{D}_\lambda$ . Then, for any  $\delta \in (0, 1]$*

$$\begin{aligned} \mathcal{E}(\hat{D}_T) - \mathcal{E}(D_*) &\leq \frac{4\sqrt{2\pi}\|C_\rho\|_\infty^{1/2}}{\sqrt{n}} \frac{1 + \sqrt{\lambda}}{\lambda} \\ &\quad + \frac{2\sqrt{2\pi}}{\lambda\sqrt{T}} + \sqrt{\frac{2\log(2/\delta)}{T}} \end{aligned}$$

with probability at least  $1 - \delta$  with respect to the independent sampling of the tasks  $\mu_t \sim \rho$  and training sets  $Z_t \sim \mu_t^n$  for any  $t \in \{1, \dots, T\}$ .

The result above is obtained by further decomposing the error  $\mathcal{E}(\hat{D}_T) - \mathcal{E}(D_*)$  as done in Eq. (13). In particular, since the multi-task empirical error provides an estimate for the future empirical risk, it is possible to control the overall error by further bounding the term  $|\hat{\mathcal{E}}(D) - \hat{\mathcal{E}}_{\mathbf{Z}}(D)|$  uniformly with respect to  $D \in \mathcal{D}_\lambda$ . This last result was originally presented in (Maurer, 2009); in [App. C](#) we report the complete analysis of such decomposition, leading to the bound in [Thm. 6](#).

## 5.2 STATISTICAL CONSIDERATIONS

For a fixed value of  $\lambda$ , we can now compare the bounds on the excess transfer risk for the representations resulting from the application of the online procedure (see [Thm. 2](#)) and the batch one (see [Thm. 6](#)). Since the approximation error due to the choice of  $\lambda$  will be the same for both approaches, this comparison provides a first indication of their statistical behavior. However, it should be kept in mind that we are comparing upper bounds, hence our considerations are not conclusive and further analysis by means of lower bounds for both algorithms would be valuable.

[Thm. 2](#) and [Thm. 6](#) are both composed of three terms. The first term is exactly the same for both procedures and this is obvious looking at the decompositions used

to deduce both results. This term can be interpreted as a within-task-estimation error, that depends on the number of points  $n$  used to train the underlying learning algorithm (in our case Ridge Regression with a linear feature map). This term, similarly to the MTL setting, highlights the advantage of exploiting the relatedness of the tasks in the learning process in comparison to independent task learning (ITL). Indeed, if the inputs are distributed on a high dimensional manifold, then  $\|C_\rho\|_\infty \ll 1$ , while upper bounds for ITL have a leading constant of 1. In particular,  $\|C_\rho\|_\infty = 1/d$  if the marginal distributions of the tasks are uniform on the  $d - 1$  dimensional unit sphere; see (Maurer, 2009; Maurer et al., 2016) for a more detailed discussion about this point. The last term in the bounds expresses the dependency on the confidence parameter  $\delta$  and it is again approximately the same for the batch and the online case. It follows that the main role in the comparison between the online and batch bounds is driven by the middle term, which expresses the dependency of the bound on the number of tasks  $T$ . This term originates in different ways: in the batch approach it is derived from the application of uniform bounds and it can be interpreted as an inter-task estimation error, while in the online approach, it plays the role of an optimization error. Despite the different derivations, we can ascertain from the explicit formula of the bounds that this term is approximately the same for both procedures. This is remarkable since it implies that the representation resulting from our online procedure enjoys the same statistical guarantees than the batch one, despite its more parsimonious memory and computational requirements.

## 5.3 COMPUTATIONAL CONSIDERATIONS

After discussing the theoretical comparison between the online and the batch LTL approach, in this section we point out some key aspects regarding the computational costs of both procedures.

**Memory.** The batch LTL estimator corresponds to a minimizer of the multi-task empirical risk in Eq. (14) over *all tasks observed so far*. The corresponding approach therefore requires storing in memory all training datasets as they arrive in order to perform the optimization. This is clearly not sustainable in the incremental setting, since tasks are observed sequentially and, possibly indefinitely, inevitably leading to a memory overflow. On the contrary, in line with stochastic methods, online LTL has a small memory footprint, since it requires to store only one dataset at the time, allowing to “forget” it as soon as one gradient step is performed.

**Time.** Online LTL is also advantageous in terms of the number of iterations performed whenever a new task is observed. Indeed, for every new task, online LTL per-

forms *only one step* of gradient descent for a total of  $T$  steps after  $T$  tasks. On the contrary, batch LTL requires finding a minimizer for Eq. (14), which cannot be obtained in closed form but requires adopting an iterative method such as Projected Gradient Descent, see e.g. (Combettes & Wajs, 2005). These methods typically require  $k$  iterations to achieve an error of the order of  $O(1/k)$  from the optimum (better rates are possible adopting accelerated schemes). However, since for any new task batch LTL needs to find a minimizer for the multi-task empirical error from scratch, this leads to a total of  $Tk$  iterations after  $T$  tasks. Noting that every such iteration requires to compute  $T$  gradients of  $\mathcal{L}_Z$  in contrast to the single one of PSSA, this shows that online LTL requires much less operations. In the batch case, a “warm-restart” strategy can be adopted to initialize the Projected Gradient Descent with the representation learned during the previous step, however, as we empirically observed in Sec. 6, online LTL is still significantly faster than batch.

## 6 EXPERIMENTS

In this section, we report preliminary empirical evaluations of the online LTL strategy proposed in this work; the Python implementation of our algorithm is available at <https://github.com/dstamos>. In particular we compare our method with its batch (or offline) counterpart and independent task learning (ITL), i.e. standard Ridge Regression, which does not leverage any shared structure among the tasks.

In all experiments, we obtain the online and batch estimators  $\hat{D}_{\lambda, T_{tr}}$  and  $\hat{D}_{\lambda, T_{tr}}$  by learning them on a dataset  $\mathbf{Z}_{tr}$  of  $T_{tr}$  training tasks, each comprising  $n$  input-output pairs  $(x, y) \in \mathcal{X} \times \mathcal{Y}$ . Below to simplify our notation we omit the subscript  $T_{tr}$  in these estimators. We perform this training for different values of  $\lambda \in \{\lambda_1, \dots, \lambda_p\}$  and select the best estimator based on the prediction error measured on a separate set  $\mathbf{Z}_{va}$  of  $T_{va}$  validation tasks. Once such optimal  $\lambda$  value has been selected, we report the generalization performance of the corresponding estimator on a set  $\mathbf{Z}_{te}$  of  $T_{te}$  test tasks. Note that the tasks in the test and validation sets  $\mathbf{Z}_{te}$  and  $\mathbf{Z}_{va}$  are all provided with both a training and test datasets  $Z, Z' \in \mathcal{Z}^n$ . Indeed, in order to evaluate the performance of a representation  $D$ , we need to first train the corresponding algorithm  $A_D$  on  $Z$ , and then test its performance on  $Z'$  (sampled from the same distribution), by computing the empirical risk  $\mathcal{R}_{Z'}(A_D(Z))$ . For all methods considered in this setting, we perform parameter selection over  $p = 30$  candidate values of  $\lambda$  over the range  $[10^{-6}, 10^3]$  with logarithmic spacing. In the online setting the training datasets arrive one at the time, therefore model se-

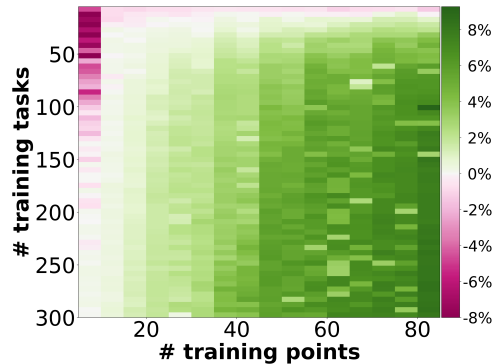


Figure 1: Relative improvement (in %) of our online LTL algorithm over the ITL baseline for a varying range of training tasks  $T_{tr}$  and number of samples  $n$  per task, during 30 trials.

lection is performed *online*: the system keeps track of all candidate representation matrices  $\bar{D}_{\lambda_1}, \dots, \bar{D}_{\lambda_m}$  and whenever a new training task is presented, these matrices are all updated by incorporating the corresponding new observations. The best representation is then returned at each iteration, based on its performance on the validation set  $\mathbf{Z}_{va}$ . Finally, in the subsequent experiments, we set the step sizes of the online LTL method in Alg. 1 equal to  $\gamma_t = c/\sqrt{t}$ , for some constant  $c > 0$  chosen by model selection. Moreover, we computed the batch LTL estimator by classical Projected Gradient Descent method up to convergence, within  $10^{-6}$  relative descent of the objective function.

**Synthetic Data.** We considered a regression problem on  $\mathcal{X} \subseteq \mathbb{R}^d$  with  $d = 50$  and a variable number of training tasks  $T_{tr}$  and training points  $n$ . We also generated  $T_{te} = 300$  test tasks and we sampled a number  $T_{va}$  of validation tasks equal to 50% of  $T_{tr}$ . For each task, the corresponding dataset  $(x_i, y_i)_{i=1}^n$  was generated according to the linear regression equation  $y = w^\top x + \epsilon$ , with  $x$  sampled uniformly on the unit sphere in  $\mathbb{R}^d$  and  $\epsilon$  sampled from a Normal distribution,  $\epsilon \sim \mathcal{N}(0, 0.2)$ . The tasks predictors  $w$  were generated as  $P\tilde{w}$  with the components of  $\tilde{w} \in \mathbb{R}^{d/2}$  sampled from  $\mathcal{N}(0, 1)$  and then  $\tilde{w}$  normalized to have unit norm, with  $P \in \mathbb{R}^{d \times d/2}$  a matrix with orthonormal rows. In this way, the tasks reflect the assumption of sharing a low dimensional representation, which needs to be inferred by the LTL algorithm.

Fig. 1 reports the comparison between the baseline ITL and the proposed online LTL approach in terms of the relative difference of the prediction error on test tasks for the two methods. More precisely, given the mean squared errors (MSE)  $R_{oLTL}$  of online LTL and  $R_{ITL}$  of ITL averaged across the test tasks, we report the ratio  $(R_{ITL} - R_{oLTL})/R_{ITL}$  as a percentage improvement. Results are reported across a range of  $T_{tr}$  and  $n$ . We note that the regime considered for these experiments is par-

Table 1: Time (in seconds) for computing online and batch LTL for  $T_{tr}$  training tasks and  $n$  of samples per task.

	$T_{tr}$ 50		$T_{tr}$ 100		$T_{tr}$ 150	
	$n$ 20	$n$ 50	$n$ 20	$n$ 50	$n$ 20	$n$ 50
<b>Batch</b>	85	227	246	617	428	2003
<b>Online</b>	36	86	108	273	227	776

ticularly favorable to LTL, almost always outperforming ITL. However, when the number of training points per task is small, the LTL algorithm, as expected, is unable to capture the underlying representation, unless several tasks are used in training.

To provide further evidence of the performance of online LTL, Fig. 2 (Top) compares the prediction error of online LTL, batch LTL, and ITL as the number of training tasks  $T_{tr}$  increases one at the time and the different methods update their corresponding representation accordingly. In this case, the number of samples per task is fixed to  $n = 40$ . We also added to the comparison the multi-task algorithm (MTL) described in Sec. 2.3, performing trace norm regularization *on the test set*. As expected, the performance of both ITL and MTL does not depend on the number of training tasks. Consistently to what observed before, ITL is outperformed by both LTL methods, which tend to converge to the MTL method as more training tasks are provided. In general, when, as in this case, the number of test tasks is large enough, the MTL method is expected to outperform LTL, since MTL optimizes the representation directly on the test tasks. Concerning the LTL methods, consistently with the theory presented in Sec. 4, the performance of the online method is equivalent to that of its batch counterpart, which is, as already stressed in Sec. 5.3, less appealing from the computational point of view. To confirm this aspect, we report in Tab. 1 the computational times required on average by online LTL and batch LTL as  $T_{tr}$  and  $n$  vary. Online LTL is faster than batch LTL.

**Schools Dataset.** We evaluated online LTL on the Schools dataset, consisting of examination records from 139 schools, see (Argyriou et al., 2008). Each school is associated to a regression task, individual students correspond to the input and their exam score to the output. In this case, the sample size  $n$  varies across the tasks and the features belong to an input space  $\mathcal{X} \subseteq \mathbb{R}^d$ , with  $d = 26$ . We randomly sampled 25% and 50% of the 139 tasks for LTL training and validation respectively and the remaining tasks were used as test set. Fig. 2 (Bottom) reports the performance of online LTL, batch LTL, ITL and MTL. Performance is reported in terms of the Explained Variance on the tasks (Argyriou et al., 2008), higher values correspond to better performance. Results are consistent with synthetic experiments; in particular, online

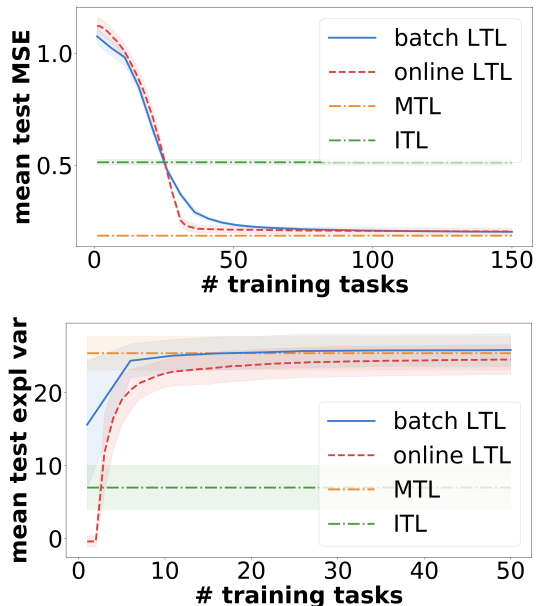


Figure 2: Performance of online LTL, batch LTL, ITL and MTL (on the test set) during 30 trials on the synthetic dataset (Top) and the Schools dataset (Bottom) as the number of training tasks increases incrementally.

and batch LTL are comparable.

## 7 CONCLUSION AND FUTURE WORK

We proposed an on-line (incremental) approach to LTL for linear data representation learning. Compared with its batch counterpart, this approach is computationally more efficient both in terms of memory and number of operations, while enjoying the same generalization properties. Preliminary experiments have highlighted the favorable learning capability of the proposed LTL strategy. Our analysis opens several future research directions. First, it would be valuable to investigate whether the same statistical guarantees hold for a projection-free meta-algorithm which does not require the computation of the entire SVD (e.g. certain variants of Frank Wolfe algorithm (Hazan & Kale, 2012), which do not require memorizing the sequence of datasets). Second, from a modeling perspective, we could take inspiration from the vast MTL literature to design new LTL methods in order to deal with tasks that are not necessarily spanning a low-rank subspace but are for instance organized into clusters (Jacob et al., 2009) or share a sparse set of relations (Ciliberto et al., 2015a;b). Finally, extending our analysis to non-convex settings would allow one to tackle more general families of learning algorithms as well as recent empirical meta-learning approaches (e.g. Franceschi et al., 2018) which implicitly attempt to directly minimize the transfer risk.

## References

- Alquier, Pierre, Mai, The Tien, and Pontil, Massimiliano. Regret bounds for lifelong learning. In *International Conference on Artificial Intelligence and Statistics*, 2017.
- Ando, Rie Kubota and Zhang, Tong. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 2005.
- Argyriou, Andreas, Evgeniou, Theodoros, and Pontil, Massimiliano. Convex multi-task feature learning. *Machine Learning*, 2008.
- Balcan, Maria-Florina, Blum, Avrim, and Vempala, Santosh. Efficient representations for lifelong learning and autoencoding. In *Conference on Learning Theory*, 2015.
- Bartlett, Peter L and Mendelson, Shahar. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.
- Bauschke, Heinz H, Combettes, Patrick L, et al. *Convex analysis and monotone operator theory in Hilbert spaces*. Springer.
- Baxter, Jonathan. A model of inductive bias learning. *J. Artif. Intell. Res.*, 12(149–198):3, 2000.
- Bertsekas, Dimitri P, Nedi, Angelia, and Ozdaglar, Asuman. *Convex analysis and optimization*. Athena Scientific, 2003.
- Bhatia, Rajendra. Matrix analysis, volume 169 of graduate texts in mathematics, 1997.
- Boucheron, Stéphane, Lugosi, Gábor, and Bousquet, Olivier. Concentration inequalities. In *Advanced Lectures on Machine Learning*, pp. 208–240. Springer, 2004.
- Camoriano, Raffaello, Pasquale, Giulia, Ciliberto, Carlo, Natale, Lorenzo, Rosasco, Lorenzo, and Metta, Giorgio. Incremental robot learning of new objects with fixed update time. In *ICRA*, 2017.
- Caruana, Rich. Multitask learning. *Machine Learning*, 1997.
- Cavallanti, Giovanni, Cesa-Bianchi, Nicolo, and Gentile, Claudio. Linear algorithms for online multitask classification. *Journal of Machine Learning Research*, 2010.
- Cesa-Bianchi, Nicolo, Conconi, Alex, and Gentile, Claudio. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 2004.
- Ciliberto, Carlo, Mroueh, Youssef, Poggio, Tomaso, and Rosasco, Lorenzo. Convex learning of multiple tasks and their structure. In *ICML*, 2015a.
- Ciliberto, Carlo, Rosasco, Lorenzo, and Villa, Silvia. Learning multiple visual tasks while discovering their structure. In *CVPR*, 2015b.
- Combettes, Patrick L and Wajs, Valérie R. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.
- Franceschi, Luca, Frasconi, Paolo, Grazi, Riccardo, Salzo, Saverio, and Pontil, Massi. Bilevel programming for hyperparameter optimization and meta-learning. In *ICML*, 2018.
- Grimmett, Geoffrey and Stirzaker, David. *Probability and random processes*. Oxford university press, 2001.
- Hazan, Elad. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2016.
- Hazan, Elad and Kale, Satyen. Projection-free online learning. *arXiv preprint arXiv:1206.4657*, 2012.
- Herbster, Mark, Pasteris, Stephen, and Pontil, Massimiliano. Mistake bounds for binary matrix completion. In *Advances in Neural Information Processing Systems*, 2016.
- Jacob, Laurent, Vert, Jean-philippe, and Bach, Francis R. Clustered multi-task learning: A convex formulation. In *Advances in neural information processing systems*, 2009.
- Kollo, Tonu and von Rosen, Dietrich. *Advanced Multivariate Statistics with Matrices*, volume 579. Springer Science & Business Media, 2006.
- Maurer, Andreas. Algorithmic stability and meta-learning. *Journal of Machine Learning Research*, 2005.
- Maurer, Andreas. Transfer bounds for linear feature learning. *Machine learning*, 75(3):327–350, 2009.
- Maurer, Andreas, Pontil, Massi, and Romera-Paredes, Bernardino. Sparse coding for multitask and transfer learning. In *ICML*, 2013.
- Maurer, Andreas, Pontil, Massimiliano, and Romera-Paredes, Bernardino. The benefit of multitask representation learning. *The Journal of Machine Learning Research*, 17(1): 2853–2884, 2016.
- McDonald, Andrew M, Pontil, Massimiliano, and Stamos, Dimitris. New perspectives on k-support and cluster norms. *Journal of Machine Learning Research*, 2016.
- Nemirovski, Arkadi, Juditsky, Anatoli, Lan, Guanghui, and Shapiro, Alexander. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 2009.
- Nemirovskii, A. and Yudin, D. B. Problem complexity and method efficiency in optimization. *SIAM Review*, 1985.
- Pentina, Anastasia and Lampert, Christoph. A PAC-Bayesian bound for lifelong learning. In *International Conference on Machine Learning*, pp. 991–999, 2014.
- Petersen, Kaare Brandt and Pedersen, Michael Syskind. The matrix cookbook. *Technical University of Denmark*, 2008.
- Polyak, Boris T and Juditsky, Anatoli B. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.
- Rebuffi, Sylvestre-Alvise, Kolesnikov, Alexander, and Lampert, Christoph H. iCaRL: Incremental classifier and representation learning. In *Proc. CVPR*, 2017.
- Robbins, Herbert and Monro, Sutton. A stochastic approximation method. *The Annals of Mathematical Statistics*, 1951.
- Rohrbach, Marcus, Ebert, Sandra, and Schiele, Bernt. Transfer learning in a transductive setting. In *Advances in Neural Information Processing Systems*, pp. 46–54, 2013.
- Ruvolo, Paul and Eaton, Eric. Ella: An efficient lifelong learning algorithm. In *ICML*, 2013.
- Shalev-Shwartz, Shai and Ben-David, Shai. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.
- Shamir, Ohad and Zhang, Tong. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *ICML*, 2013.
- Thrun, Sebastian and Pratt, Lorien. *Learning to Learn*. Springer, 1998.

---

# Bandits with Side Observations: Bounded vs. Logarithmic Regret

---

**Rémy Degenne**

LPSM, CNRS, Sorbonne Université,  
Université Paris Diderot, 75013 Paris, France;  
CMLA, ENS Cachan, CNRS,  
Université Paris-Saclay, 94235 Cachan, France

**Evrard Garcelon**

CMLA, ENS Cachan,  
94235 Cachan, France

**Vianney Perchet**

CMLA, ENS Cachan, CNRS,  
Université Paris-Saclay, 94235 Cachan, France  
Criteo AI Lab  
75009 Paris

## Abstract

We consider the classical stochastic multi-armed bandit but where, from time to time and roughly with frequency  $\epsilon$ , an extra observation is gathered by the agent for free. We prove that, no matter how small  $\epsilon$  is the agent can ensure a regret uniformly bounded in time.

More precisely, we construct an algorithm with a regret smaller than  $\sum_i \frac{\log(1/\epsilon)}{\Delta_i}$ , up to multiplicative constant and  $\log \log$  terms. We also prove a matching lower-bound, stating that no reasonable algorithm can outperform this quantity.

## 1 INTRODUCTION

We consider the celebrated multi-armed bandit framework (sometimes also called online learning), a repeated decision problem where an agent (or an algorithm, a machine, a player, etc.) takes sequentially decisions from a finite set. Each decision gives a stochastic reward to the agent of fixed expectation. The main objective is to derive an algorithm maximizing the cumulative reward or minimizing its normalized version, the so-called “regret”. The latter is simply the difference between the cumulative expected reward of an agent knowing in hindsight the optimal decision, and the cumulative reward of the algorithm.

Online learning can be traced back to the 30’s, when Thompson analysed random clinical trial using an analogy with finding the best slot-machine in a casino by pulling sequentially their arms in order to minimize the total loss. During the 20th century, many improvements have been made, at least on the asymptotic version of the problem. The quantity of theoretical studies and practical applications of bandits have exploded since the early

2000. There are several reasons for that. First of all, a simple yet almost optimal algorithm called UCB has been developed. Its simple structure allows to adapt it to many different settings. As a consequence, many possible applications of online learning have been developed. Amongst them, we can mention the routing problem: given a network with congested edges, one must find the quickest way from some origin to a destination (this setting incorporates a combinatorial structure); this can be used to send packets in a network, as well as finding the quickest itinerary from a point A to a point B. Online advertising is another application: given a possible set of ads, one must find the ad with the highest probability of click. The last application we mention is concerned with wireless network and/or cognitive radio, where either a radio can change from an available channel to other channels to improve its reception or emission quality, or alternatively a wireless source, in a relay selection problem where multiple relays are available, can explore those nodes to achieve better transmissions rates. One of the typical and crucial assumption of all these models is that the agent only observes the outcome of his decisions, but not what the other decisions would have given him. For instance, using a slot machine only gives you a feedback on the performance of that very machine, displaying an ad only gives information of the probability of clicks on that specific ad, etc. This assumption is actually called “bandit feedback”. At the other end of the spectrum, the dual assumption (mostly used in non-stationary environment that we are not concerned with in that paper) is the “full information feedback”, where all the outcomes of all decisions are observed at all stages. However, none of our motivating examples satisfies this strong assumption.

However, we argue that the bandit feedback is also too strong and that in many cases more informations are available to the agent. Typically, the agent will always observe the outcome of his own decision, but with some small probability he might also get one (or several, but

that is irrelevant to our setting) extra “free” information. For instance, consider the original multi-armed bandit problem. A gambler is in a casino and wants to find out which slot machine is the best one. From time to time, he might observe other gamblers playing nearby machines. Even if this does not cost him anything, he gets feedback on the other machines. This effect also appears in other settings. In wireless network, a source with an allocated transmission capacity (because of a power-saving allocation protocol for instance) sends data through a relay and may have the opportunity to send another custom packet (so that the energy needed to send this packet is less than the available energy) through another relay in order to estimate transmissions rates. In online advertisement (and actually many other industrial markets), companies are willing to spend a small fraction of their data, say with probability  $\varepsilon$  as in the celebrated  $\varepsilon$ -greedy algorithm, just to acquire new information. An algorithm is only evaluated on the remaining (of proportion  $1 - \varepsilon$ ) fraction of the data treated. In a multi-armed bandit setting, this means that with probability  $\varepsilon$ , the next decision is “free”. Finally, we can also think that in the congested network problem, an algorithm can from time to time send “fake”, but free, packets to test the congestion; conversely, an app trying to minimize the congestion time of its users might be able to use free information if it notices that a bucket of users (for instance, those that are registered) might explore new road willingly, i.e., without uninstalling the app.

We therefore focus on the classical multi-armed bandits but where some extra and free information is available from time to time. Clearly, if the probability  $\varepsilon$  that it happens is arbitrarily close to 0, the improvement will be negligible. But we aim at constructing “optimal” algorithm, i.e., whose regret is small and in a multiplicative constant of the best regret achievable regret by “meaningful” algorithms. All these concepts are explained in details in the remaining of the paper that is organized as follows.

The model is introduced in Section 2, where we provide a very naïve algorithm achieving bounded regret (uniformly in time). We exhibit in Section 3 non-trivial lower bounds (we emphasize here that traditional bandit lower bounds are void in our setting). Algorithms are described and analysed in Section 4. Finally, Section 5 is dedicated to experiments illustrating the different guarantees and dependencies in the parameters of the models.

## 1.1 RELATED WORKS

This paper is not the first one to consider additional, free informations, available to the agents while optimizing. There are many different ways of modelling this idea,

but our paper is the first one (to our knowledge) that also focus on strategical aspects of obtaining these free informations to reduce regret, especially in the stochastic case.

There exists models where when a specific decision is taken, automatically (resp. with some probability), the performance of some other decision are observed [Alon et al., 2015, Chen et al., 2016, Caron et al., 2012]. Those models assume that there exists a directed (resp. weighted) graph whose set of nodes is the set of decisions. When the agent takes a decision, he also observes the outcome of any node linked (resp. with a probability proportional to the weight of the edge) to the current decision node. Our passive model could be recast as a specific case of that setting, but our results are much finer than the ones available for the general case.

In [Yu and Mannor, 2009] the rewards are stochastic but their means change at unknown time points. Free additional informations are queried by the algorithm in order to detect these change points. They however are not used to decrease the regret of the base bandit algorithm.

Another trend of literature of additional free information in multi-armed bandit studies the “adversarial” case, where no stationary assumption is made on the sequence of rewards (namely, there are not i.i.d.) [Audibert and Bubeck, 2010, Cesa-Bianchi et al., 2006, Mannor and Shamir, 2011]. However the rate of convergence in the two extreme cases (bandit and full information) have the same dependency in  $T$ , the total number of stages. To be precise, the regret is either of the order of  $\sqrt{KT}$  (in the bandit case) or  $\sqrt{\log(K)T}$  (in the full information case), where  $K$  is the number of decisions. Intermediate settings (where  $1 + M$  observations are available at each stage) interpolate between those two cases.

In the stochastic case though, regret is uniformly bounded with full information and grows logarithmically in the bandit case. As a consequence, even the rate of convergence will depend on the size of free informations.

## 2 MULTI-ARMED BANDITS, REGRET MINIMIZATION AND FEEDBACKS

In that section, we describe precisely the stochastic multi-armed bandit problem and its objective, the minimization of regret.

### 2.1 STOCHASTIC MULTI-ARMED BANDITS

#### 2.1.1 Bandit vs Full-Information

At each successive stage  $t \in \mathbb{N}^*$ , an agent takes a decision (or *pulls an arm* using the multi-armed bandit lingo)



$i_t$  in the finite set  $[K] := \{1, \dots, K\}$ . After pulling this arm, the agent receives the reward  $X_t^{(i_t)} \in \mathbb{R}$ , which is sampled from a real reward distribution  $\nu^{(i_t)}$  of expectation  $\mu^{(i_t)}$ . As a consequence, the stochastic bandit problem is parametrised by the vector of distribution,  $(\nu^{(1)}, \dots, \nu^{(K)})$ , or alternatively in the non-parametric case, by the vector of expected rewards  $(\mu^{(1)}, \dots, \mu^{(K)})$ . Throughout the paper, the results are stated using the arbitrary ordering  $\mu^{(1)} > \mu^{(2)} \geq \dots \geq \mu^{(K)}$ . Obviously, those vectors are unknown to the agent, who is aiming at optimizing her cumulative expected reward  $\sum_{t=1}^T \mu^{(i_t)}$ . Actually, instead of this cumulative reward, the objective is normalized into *cumulative regret* minimization.

The cumulative regret (or simply regret) of an algorithm at stage  $T$  is defined as

$$R_T = T \max_{i \in [K]} \mu^{(i)} - \sum_{t=1}^T \mu^{(i_t)},$$

i.e., it is the difference between the maximal possible cumulative reward up to stage  $T$  and the expectation of the reward gained by the successive choices of arms  $i_1, \dots, i_T$ . Following the classical notations, we define  $\mu^* = \max_{i \in [K]} \mu^{(i)}$  and the gaps  $\Delta_i = \mu^* - \mu^{(i)}$ . In the non-parametric case, these gaps are the relevant quantities characterising the complexity of a bandit problem.

There are different standard assumption on the feedbacks available to the agent before taking a new decision. In the *bandit* setting, she observes only her reward  $X_t^{(i_t)}$  (and, specifically, not the other  $X_t^{(k)}$ ) at the end of stage  $t \in \mathbb{N}^*$ . In the *full information* setting, she observes the full vector of rewards  $(X_t^{(1)}, \dots, X_t^{(K)}) \in \mathbb{R}^K$ . With full information, the *Follow The Leader* (FTL) algorithm that selects the arg max of the empirical average  $\bar{X}_t^{(i)} := \frac{1}{t} \sum_{s=1}^t X_s^{(i)}$  attains a uniformly bounded regret (with respect to  $T$ ). In the bandit setting, FTL gets a linear regret, yet the logarithmic optimal dependency in  $T$  is achieved by many algorithms. One of the most popular, called *Upper Confidence Bound* (UCB), selects the argmax of the empirical average augmented of an error term  $\hat{\mu}_t^{(i)} + \sqrt{6 \frac{\log(t)}{N_i(t)}}$  where  $N_i(t)$  is the number of pulls of arm  $i$  up to stage  $t$ , while  $\hat{\mu}_t^{(i)} := \frac{1}{N_i(t)} \sum_{s:i_s=i} X_s^{(i_s)}$ .

Many other algorithms are variants of UCB, by modifying the error term, changing some parameters, specifying it for a given class of parametric distributions, etc.

### 2.1.2 Additional Informations

As specified and motivated in the Introduction, we aim at analysing intermediate settings between bandit and full information, in which a subset of the reward vector might

be observed. More precisely, at some stages, the agent not only observes an arm by pulling it but might also observe a second arm for free, i.e., without getting a reward (and without incurring any regret). We consider several ways in which these free observations can be obtained: they can be deterministically available periodically (for instance every  $1/\varepsilon$  rounds) or arrive randomly (at each stage with probability  $\varepsilon$ ); the agent can also be a *passive observer* if she can not choose from which arm she gets an extra information (the environment chooses it for her, in a manner to be specified latter on), or she can be an *active observer* if she can choose the arm to observe freely.

We end this section with some notations. In the random time arrival of free information, we assume that at each stage  $t \in \mathbb{N}^*$  a Bernoulli random variable  $Z_t$  with expectation  $\varepsilon_t$  (whose law is denoted by  $\text{Ber}(\varepsilon_t)$ ) is sampled and a free observation is available if  $Z_t = 1$ . The particular setting in which  $\varepsilon_t$  is constant will be called static random. We will denote by  $i_t$  the arm pulled and by  $f_t$  the arm chosen to be observed using the free information (if available). The total number of pulls of arm  $i$  up to stage  $t$  is  $N_i(t)$ , the number of free observations  $F_i(t)$  and the total number of observation of arm  $i$  is  $O_i(t) = N_i(t) + F_i(t)$ .

## 2.2 A FINITE REGRET SETTING

It is not really difficult to devise a naïve algorithm with a (uniformly) bounded regret at least in the deterministic case, when a free observation is obtained every  $1/\varepsilon$  round. We consider for simplicity the case of  $K = 2$  arms in this section as it gives all the intuitions. Consider the following (heavily sub-optimal) strategy, which we denote by FTL-robin: pull the *leading arm* (the one with the highest empirical average  $\hat{\mu}_t^{(i)}$ ) and when a free sample is available, observe arms in a round-robin fashion.

After a period of  $1/\varepsilon$  stages, both arms have their observation counters increased by at least one. As a consequence, this simple algorithm FTL-robin can be seen as a full-information algorithm which would take  $1/\varepsilon$  stages to get the observations. To simplify intuitions

**Lemma 1.** *The regret of the FTL-robin algorithm on the deterministic setting with  $K = 2$  satisfies*

$$\mathbb{E} R_T \leq \frac{\bar{c}}{\varepsilon} \frac{1}{\Delta}, \text{ where } \Delta = |\mu^{(1)} - \mu^{(2)}|,$$

and there exist distributions  $(\nu^{(1)}, \nu^{(2)})$  such that

$$\frac{c}{\varepsilon} \frac{1}{\Delta} \leq \mathbb{E} R_T,$$

where  $c, \bar{c} > 0$  are universal constants that do not involve any parameter of the problem.

This lemma shows that even the simplest algorithm gets a finite regret in this setting. The proof is almost trivial and omitted. To provide some insights, just assume that  $\nu^{(1)} = \mathcal{N}(\Delta, 1)$  and  $\nu^{(2)} = \delta_0$ . Then the regret of FTL-robin is equal to the  $\Delta/\epsilon$  times the number of times that  $\overline{X}_t^{(1)}$  is smaller than 0. Basic computations show that this number is of order  $\frac{1}{\Delta^2}$ .

The relevant question is then not the asymptotic regime, but what is the precise optimal dependency on  $\epsilon$ . Indeed, when  $\epsilon < \frac{1}{\log T}$ , this bound gets larger than the  $O(\log T)$  regret of another naive approach, which is to use an algorithm for bandits and discard the additional information.

This free information problem is characterized by a transition from "small"  $\epsilon$ , where the amount of additional information is not enough to improve the performance of bandit algorithm, to "big"  $\epsilon$ , where the regret is finite and the setting is closer to full-information.

We answer the question of what "small" and "big" mean in this context and where the transition occurs and we display algorithms enjoying both logarithmic regret when  $\epsilon$  is small and finite regret when it is big.

### 3 LOWER BOUNDS

We first consider the definition of *optimality* of an algorithm, that is, what is the minimal regret achievable by any "reasonable" algorithm, in a sense we will make precise. Our lower bounds will highlight a transition from logarithmic (with respect to the horizon  $T$ ) to finite regimes when  $\epsilon$  gets big enough.

There are now quite standard techniques to devise lower bounds for stochastic bandits problems, but surprisingly these techniques are inadequate in our case, due to the finiteness of the optimal regret. As a finite regret is possible, a traditional, asymptotic lower bound for  $\frac{\mathbb{E} R_T}{\log T}$  [Lai and Robbins, 1985] could only be 0 and hence would not be informative. We can obtain a finite time version of this type of bound as in [Garivier et al., 2016] by imposing that our algorithm should perform better than a reference algorithm.

**Definition 1.** An algorithm is said to be sub-logarithmic with constants  $C, C_0$  if on all bandit problems it verifies for all stages  $T \in \mathbb{N}^*$ ,

$$\mathbb{E} R_T \leq C \sum_{i=1}^K \frac{\log T}{\Delta_i} + C_0 \sum_{i=2}^K \Delta_i.$$

There exists sub-logarithmic algorithms (UCB for example, with constants  $C = 8, C_0 = (1 + \pi^2/3)$  [Auer et al., 2002]). A sub-logarithmic algorithm is performing at least as good as the UCB baseline. This finite time

constraint on the performance of the algorithm translates into a lower bound: to perform relatively well on all bandit problems, an algorithm cannot outperform the lower bound guarantee on any of them.

#### 3.1 PASSIVE OBSERVER

When the observer is passive (i.e., she does not choose the arm  $f_t$  to observe freely), we assume that  $f_t$  is equal to  $i \in [K]$  with probability  $p_t^{(i)}$  chosen by the environment. Consider the static setting in which for all  $t$ ,  $Z_t \sim \text{Ber}(\epsilon)$  and the probabilities  $p_t^{(i)}$  do not depend on the stage  $t$  (we will thereafter omit the subscript  $t$ ).

Standard lower bound techniques proceed as follows: at stage  $T$ , the expected number of pulls of an arm is linked to the Kullback-Leibler divergence between the bandit problem studied and a related alternative, in which this arm would be the best one (roughly speaking, in order to be able to "test" that the problem is not the alternative one, a minimum number of samples of that arm must be gathered in the original problem).

A bound on this divergence gives a constraint of the form  $\mathbb{E} O_i(T) \geq h_i(t)/\Delta_i^2$  for some function  $h_i(T) = O(\log T)$ . Then a lower bound for the regret is the minimal value of  $\sum_{i=2}^K \Delta_i \mathbb{E} N_i(t)$  respecting all these constraints, that can be computed through some linear program. With this proof technique, we obtain lemma 2.

**Lemma 2.** *The regret of a sub-logarithmic algorithm with constants  $C, C_0$  must verify*

$$\mathbb{E}_1 R_T \geq \sum_{i=2}^K \max \left\{ 0, \frac{h_i(T)}{2\Delta_i} - \epsilon p^{(i)} T \Delta_i \right\}.$$

where  $h_i(T) = O(\log T)$  (see appendix for a detailed definition).

As mentioned above, this lower bound is void as it reaches 0 as soon as  $T$  is big enough, bigger than  $\frac{1}{\epsilon} \max_{j \geq 2} \frac{h_j(T)}{2p^{(j)} \Delta_j^2}$ .

We want to explain why this lower bound fails to provide relevant informations as our algorithm (see Section 4) are somehow inspired by this. Recall that the lower bound only states that any reasonable algorithm must have gathered, for each sub-optimal arm, a given number of observations, namely  $\frac{h_i(T)}{2\Delta_i^2}$ . However,  $h_i(T)$  grows sub-linearly, while the number of free observations grows linearly. So if  $T$  is large enough, there will be in total enough free observations to allocate  $\frac{h_i(T)}{2\Delta_i^2}$  of them to arm  $i$  and an optimal algorithm should somehow have used only free information to explore.

However, this is only possible if the  $\epsilon T$  free observa-

tions were gathered *at the beginning* of the problem and not scarcely with time! Indeed, in the traditional lower bounds techniques, the fact that arm  $i$  is observed at the beginning or at the end of time is irrelevant (since the cost of one pull is constant throughout time). They totally discard the fact that the quantities  $\mathbb{E} N_i(t)$  and  $\mathbb{E} R_t$  must be non-decreasing. Tighter, relevant lower bounds can be recovered using this monotonicity.

**Theorem 1.** *The regret of a sub-logarithmic algorithm with constants  $C, C_0$  must verify*

$$\mathbb{E} R_T \geq \sum_{i=2}^K \frac{1}{2\Delta_i} r_T^{(i)}$$

where

$$r_T^{(i)} = \log\left(\frac{T\Delta_i^2}{2C \log T \sum_{j \neq i} \frac{\Delta_i}{\Delta_i + \Delta_j}}\right) + \eta_i(T) - 2\epsilon p^{(i)} \Delta_i^2 T$$

if  $T \leq 1/(2\epsilon p^{(i)} \Delta_i^2)$  and otherwise

$$r_T^{(i)} = \left[ \log\left(\frac{1}{\epsilon 4C p^{(i)} \sum_{j \neq i} \frac{\Delta_i}{\Delta_i + \Delta_j}}\right) - \log \log\left(\frac{1}{2\epsilon p^{(i)} \Delta_i^2}\right) + \eta_i\left(\frac{1}{2\epsilon p^{(i)} \Delta_i^2}\right) - 1 \right].$$

The function  $\eta_i(T)$  goes to zero in  $O(1/\log T)$ . See appendix for details.

Theorem 1 correctly reports a lower bound increasing with the horizon. It shows a transition from a  $O(\log T)$  optimal regret for  $T \ll 1/(2\epsilon p^{(i)} \Delta_i^2)$  to a finite regret function of  $\epsilon$  when  $T$  gets bigger. According to Theorem 1, the correct dependency in  $\epsilon$  in the regret should be in  $O(\log(1/\epsilon))$ , not  $O(1/\epsilon)$  as seen for the naive FTL-robin algorithm.

We can also wonder what is the most favorable passive setting. Simple computations show that free observations should be drawn according to the probability vector  $(p_\star^{(1)}, \dots, p_\star^{(K)})$  where  $p_\star^{(i)}$  is proportional to  $\frac{1}{\Delta_i}$  (here, we actually ignore the log log and  $\eta$  terms of Theorem 1), leading to a lowest lower bound

$$\begin{aligned} \mathbb{E}_1 R_T &\geq \sum_{i=2}^K \frac{1}{2\Delta_i} \log\left(\frac{1}{\epsilon 4C \sum_{j \neq i} \frac{1}{\Delta_j}}\right) + \alpha \\ &\geq \sum_{i=2}^K \frac{1}{2\Delta_i} \log\left(\frac{1}{4C\epsilon}\right) + \alpha, \end{aligned}$$

where  $\alpha$  regroups the log log and  $\eta$  terms in theorem 1. This lower bound shows in particular that when all sub-optimal arms have the same gap, the optimal sample distribution is uniform and the lower bound is of order  $\frac{K}{\Delta} \log(\frac{1}{\epsilon})$ .

### 3.2 ACTIVE OBSERVER

An active observer has the possibility to chose the weights  $p_t^{(i)}$  at each stage  $t \leq T$ , potentially achieving a much better distribution of the free observations up to stage  $T$  than any static distribution. As before, standard techniques give the following lower bound.

**Lemma 3.** *The regret of a sub-logarithmic algorithm with constants  $C, C_0$  verifies*

$$\mathbb{E} R_T \geq \sum_{i=2}^k \frac{h_i(T)}{2\Delta_i} - \Delta_k(\epsilon T - \sum_{j>k} \frac{h_j(T)}{2\Delta_j^2}),$$

where  $k = \min\{i \in \{2, \dots, K\} : \sum_{j>i} \frac{h_j(T)}{2\Delta_j^2} \leq \epsilon T\}$ .

The structure of the solution to the optimization problem in this case is again educational: an optimal algorithm presented with a given amount of free observations would spend them at the beginning, before costly pulls, and will spend them on the worst arms. This intuition drove the construction of algorithms for active observer in section 4:

First gather free observations, ideally accordingly to the proportion  $(p_\star^{(1)}, \dots, p_\star^{(K)})$  then discards arms for which enough information were gathered, and use a standard optimal bandit algorithm on the remaining ones.

As in the passive observer case, although this lower bound can be meaningful for small horizon  $T$ , it becomes void for larger horizons. A better lower bound using the monotony of the number of pulls and of the regret is provided in the next theorem.

**Theorem 2.** *For  $k \in \{2, K-1\}$  let  $t_k = \max\{t \geq 1 : \sum_{j=k+1}^K \frac{h_j(t)}{2\Delta_j^2} > \epsilon t\}$ . The regret of any active sub-logarithmic algorithm with constants  $C, C_0$  verifies*

$$\begin{aligned} \mathbb{E} R_T &\geq \max_{k:t_k \leq T} \sum_{i=2}^k \frac{1}{\Delta_i} \left[ \log\left(\frac{1}{\epsilon 4C \sum_{j \neq i} \frac{\Delta_i^2}{\Delta_j}}\right) \right. \\ &\quad \left. - \log \log\left(\frac{1}{\epsilon \sum_{j=k+1}^K \frac{1}{2\Delta_j^2}}\right) + \eta\left(\frac{1}{\epsilon \sum_{j=k+1}^K \frac{1}{2\Delta_j^2}}\right) \right]. \end{aligned}$$

When all gaps are equal to the same value  $\Delta > 0$ , the leading term of this lower bound is of the form

$$\max_{k:t_k \leq T} \frac{k-1}{\Delta} \log\left(\frac{1}{\epsilon} \frac{K-k}{K}\right).$$

In particular, this result states that as  $T$  goes to infinity, the regret is asymptotically lower bounded by  $\frac{K-1}{\Delta} \left[ \log(\frac{1}{\epsilon}) - \log \log(\frac{\epsilon}{\epsilon}) \right]$ .

## 4 ALGORITHMS AND UPPER-BOUNDS

In this section, we exhibit algorithms matching the lower bounds derived in the previous section, up to  $\log \log(\cdot)$  terms, showing that they indeed represent accurately the problem complexity.

### 4.1 PASSIVE OBSERVER

A passive observer does not get to choose the arms on which free information is gained. As in the classical stochastic multi-armed bandit, the only decision is therefore which arm to pull. It is then natural to extend known algorithms by taking into account all observations from both provenances.

As UCB pulls the arm with maximal index  $\bar{X}_t^{(i)} + \sqrt{\frac{6 \log t}{N_i(t)}}$ , we extend it by using all available observations both in the empirical mean and exploration term. Algorithm 1 pulls  $i_t = \arg \max_i \bar{X}_t^{(i)} + \sqrt{\frac{6 \log t}{O_i(t)}}$ .

---

**Algorithm 1** UCB with passive observations.

---

Pull each arm once.

**loop:** at stage  $t$ ,

$$i_t = \arg \max_i \bar{X}_t^{(i)} + \sqrt{\frac{6 \log t}{O_i(t)}}$$

Pull arm  $i_t$ , observe  $X_t^{(i)}$ .

If  $Z_t = 1$ , sample  $f_t$  and observe  $X_t^{(f_t)}$ .

Update  $\bar{X}_t, N_i(t), F_i(t), O_i(t) = N_i(t) + F_i(t)$ .

**end loop**

---

**Theorem 3.** *Consider the static passive observer case, where  $f_t$  follows the categorical distribution with parameters  $(p^{(1)}, \dots, p^{(K)})$  and the probability of getting a free observation is  $\epsilon \in (0, 1]$  for all stages  $t \geq 1$ .*

Then the regret of ucb verifies both

$$\mathbb{E} R_T \leq \sum_{i=2}^K \frac{24}{\Delta_i} \log T,$$

and

$$\begin{aligned} \mathbb{E} R_T \leq & \sum_{i=2}^K \frac{24}{\Delta_i} \log \frac{50}{\epsilon p^{(i)}} \\ & + \sum_{i=2}^K \frac{24}{\Delta_i} \max \left\{ \log \frac{1}{e \Delta_i^2}, \log \log \frac{20}{\epsilon p^{(i)}} \right\}. \end{aligned}$$

Hence UCB with passive observations recovers the  $\log(\frac{1}{\epsilon})$  dependency in  $\epsilon$ , up to a doubly logarithmic term when  $\epsilon p^{(i)}$  is small compared to the squared gaps. When the dominant term in this maximum is  $\log \frac{1}{e \Delta_i^2}$ , the regret

due to arm  $i$  has the form  $\frac{1}{\Delta_i} \log \frac{1}{\epsilon p^{(i)} \Delta_i^2}$ , which is sub-optimal with respect to  $\Delta_i$  (see Theorem 1). This is due to the sub-optimality of UCB itself: while the regret of UCB on a bandit problem is  $O(\sum_{i=2}^K \frac{\log T}{\Delta_i})$ , other algorithms of the same family like UCB2 [Auer et al., 2002], Improved-UCB, [Auer and Ortner, 2010] or MOSS [Audibert and Bubeck, 2009, Degenne and Perchet, 2016] get an improved regret of order  $O(\sum_{i=2}^K \frac{\log(T \Delta_i^2)}{\Delta_i})$ .

The dependency in  $\log(\frac{1}{\epsilon})$  means that  $\epsilon$  as small as  $\frac{1}{T}$  gives useful information to a learner. Obviously there is no gain to be had if  $\epsilon < \frac{1}{T}$ , as there is in average less than one additional observation before  $T$ , but few more free observations are enough to improve the regret.

### 4.2 ACTIVE OBSERVER

While a uniform allocation of the free observations over the arms gets the right  $\log(\frac{1}{\epsilon})$  dependency in  $\epsilon$ , having the choice of the arm which will be observed allows an algorithm to get the right dependency in the parameters of the bandit problem. In the active setting, the algorithm can choose freely which of the  $[K]$  arms will get an additional observation, when such an observation is available.

To devise an algorithm taking advantage of this possibility, we try to mimic the lower bound for fixed stage, as in Lemma 3. A good algorithm should use the available free observations first to discard the worse arms, before using costly pulls only on the remaining arms.

We introduce an algorithm combining two subroutines: an Explore-Then-Commit (ETC) [Even-Dar et al., 2006, Perchet and Rigollet, 2013] algorithm on the free observations is used to narrow the set of arms which need to be pulled and an algorithm of the UCB family is used on this set. As we seek for optimality with respect to the problem parameters we use OCUCB-n [Lattimore, 2016], which is the UCB-type algorithm closest to it. ETC is described in Algorithm 3. OCUCB-n with parameters  $\eta > 1$  and  $\rho \in [1/2, 1]$  pulls at stage  $t \in \mathbb{N}^*$  the arm with maximal index

$$\bar{X}_t^{(i)} + \sqrt{\frac{2\eta \log B_{t-1}^{(i)}}{N_i(t)}}$$

where

$$B_{t-1}^{(i)} = \max \left\{ e, \log(t), \frac{t \log t}{\sum_{i=1}^K \min\{N_i, N_j^\rho N_i^{1-\rho}\}} \right\}$$

where  $N_i$  is a shorthand notation for  $N_i(t)$ .

The main algorithm use a succession of epochs. In epoch number  $m \in \mathbb{N}$ , the ETC subroutine collects (free) information on all the arms in  $[K]$ , while OCUCB-n pulls

arms in an available subset of the arms  $S_m$ . At the end of epoch  $m$ , the free observations gathered are used to discard arms from  $[K]$  which are not optimal with high enough confidence, forming  $S_{m+1}$ . There is a finite  $m_i \in \mathbb{N}$  depending on  $\epsilon$  and the gaps such that with high probability,  $i \notin S_m$  for  $m > m_i$ , hence arm  $i$  contributes to the regret only up to epoch  $m_i$  and the regret is finite.

---

**Algorithm 2** Active Algorithm.

---

**Require:** parameters  $\rho \in [1/2, 1]$ ,  $\alpha \geq 1$ ,  $\eta > 1$ .

Initialize  $S_0 = [K]$ .

**loop:** at epoch  $m$ , with duration  $d_m = 2^{2^m}$ ,

Pull arms according to OCUCB-n with parameters  $\eta$  and  $\rho$  on  $S_m$ ,

Use free observations according to ETC with parameter  $\alpha$  and horizon  $T = d_{m+1}^{3/2} \log d_{m+1}$ .

Set  $S_{m+1}$  to the set returned by ETC.

**end loop**

---



---

**Algorithm 3** Explore-Then-Commit

---

**Require:** parameter  $\alpha \geq 1$ , horizon  $T \in \mathbb{N}^*$ .

Initialize  $s = 0$ ,  $S = [K]$ .

**loop**

Observe all arms in  $S$ .

Discard from  $S$  any arm  $i$  such that

$$\hat{\mu}_s^{(i)} + \sqrt{\frac{2\alpha}{s} \log\left(\frac{T}{s}\right)} < \max_{j \in S} \hat{\mu}_s^{(j)} - \sqrt{\frac{2\alpha}{s} \log\left(\frac{T}{s}\right)}.$$

$s \leftarrow s + 1$ .

**end loop**

**return**  $S$ .

---

In order to write a regret upper bound for our active algorithm, we introduce quantities  $H_{i,\rho}$  for  $i \in \{2, \dots, K\}$  and  $\rho \in [1/2, 1]$ ,

$$H_{i,\rho} = \frac{i}{\Delta_i^2} + \sum_{j=i+1}^K \frac{1}{\Delta_i^{2(1-\rho)} \Delta_j^{2\rho}}.$$

These constants transcribe the difficulty of the problem. A number of observations of order  $\frac{1}{\epsilon} H_{i,1}$  will be necessary for ETC to eliminate arm  $i$  with high confidence.

**Theorem 4.** *The regret of the active algorithm 2 with parameters  $\rho \in [1/2, 1]$  and  $\alpha = 1$  on problems with rewards in  $[0, 1]$  is*

$$\mathbb{E} R_T \leq C_\eta \sum_{i=2}^K \frac{4}{\Delta_i} \max \left\{ \log\left(\frac{1}{\epsilon}\right), \log \sqrt{H_{i,\rho}} \right\} + 51K + O \left( \sum_{i=2}^K \frac{1}{\Delta_i} (\log \log \frac{H_{i,1}}{\epsilon})^2 \right)$$

with  $C_\eta$  a constant that depends only on  $\eta$  (see [Lattimore, 2016] for details on  $C_\eta$ ).

Our analysis of Explore-Then-Commit relies on a new maximal concentration inequality which can be of independent interest.

**Lemma 4.** *Let  $Z_t$  be a  $\sigma^2$ -sub-Gaussian martingale difference sequence then, for every  $\delta \in (0, 0.2]$  and every integers  $T \in \mathbb{N}^*$ ,*

$$\mathbb{P} \left\{ \exists t \leq T, \bar{Z}_t \geq \sqrt{\frac{2\sigma^2}{t} \log\left(\frac{T}{\delta t}\right)} \right\} \leq 6\delta \sqrt{\log\left(\frac{1}{\delta}\right)}.$$

Asymptotically, we obtain

$$\limsup_{\delta \rightarrow 0} \frac{\mathbb{P} \left\{ \exists t \leq T, \bar{Z}_t \geq \sqrt{\frac{2\sigma^2}{t} \log\left(\frac{T}{\delta t}\right)} \right\}}{\delta \sqrt{\log\left(\frac{1}{\delta}\right)}} \leq \sqrt{e/8}.$$

This value is  $\sqrt{e/8} \approx 0.6$ .

#### 4.2.1 Heuristics and Influence of $\epsilon$

Besides the algorithm already discussed, we also experimented on the following heuristic: choose a bandit algorithm of the UCB family, which pulls the arm with a maximal index; use it to pull the arm with maximal index and if an observation is available, observe the second maximal arm. We provide no regret analysis for this heuristic but study its performance in the experimental section.

Concerning the dependency in  $\epsilon$ , we can make the following interesting remark. To simplify notations, we will assume that all arms have the same gap  $\Delta$  and we remove constants for this analysis. With these simplifications, we proved that regret at stage  $T$  is of the order of  $R_T \simeq \frac{K}{\Delta} \log\left(\frac{1}{\epsilon}\right)$ . Obviously, if  $\epsilon$  is almost equal to 0, this upper-bound is void and the algorithm should not depend on the free observations. One might ask what is the threshold at which free informations become relevant at stage  $T$ .

Notice that standard information theory arguments yield that if  $\epsilon T \leq \frac{K}{2\Delta^2}$ , and even if the free observations were gathered at the beginning of the problem, only  $\frac{K}{2}$  arms could be removed (with high probability) from the set of possible optimal arms. Hence these free information are not useful for at least  $K/2$  arms and regret will have to scale as  $\frac{K}{2\Delta} \log\left(\frac{2T\Delta^2}{K}\right)$ , the optimal rate for the bandit problem with  $K/2$  arms with equal gaps  $\Delta$ .

On the other hand, if  $\epsilon T \geq \frac{K}{2\Delta^2}$ , then (up to multiplicative constant),  $\frac{K}{\Delta} \log\left(\frac{1}{\epsilon}\right)$  dominates  $\frac{K}{\Delta} \log\left(\frac{T\Delta^2}{K}\right)$ . As a consequence, the relevant threshold for the probability of free observations after  $T$  stages is

$$\epsilon^* = \frac{1}{T} \frac{K}{\Delta^2}.$$

## 5 EXPERIMENTS

All experiments are performed with Gaussian rewards with unit variance.

**Influence of  $\epsilon$ .** The goal of this first experiment is to confirm the scaling of the regret with  $\epsilon$ . That is to say, the regret scales with  $\sum_{i:\Delta_i>0} \frac{1}{\Delta_i} \log(\frac{1}{\epsilon\Delta_i^2})$ . The experiment is performed with a passive observer with either a uniform distribution or the optimal one, as defined in Section 3.1. To do so, the experiment is performed in the passive setting associated with a uniform distribution and the optimal one, as defined in Section 4.1. Also, when free observations are scarce,  $\epsilon \sim \frac{1}{T}$ , the average number of those is approximately 1 during the experience. Therefore, the regret is similar to the one suffered by an UCB algorithm in a classic multi-armed bandit setting, a behaviour captured by the function  $f$ . On Figure 1 and 2, experiments are run on four Gaussian arms with expectations 2, 1.8, 0.5, 0.2, the error bars are quantile at 10% and 90%.

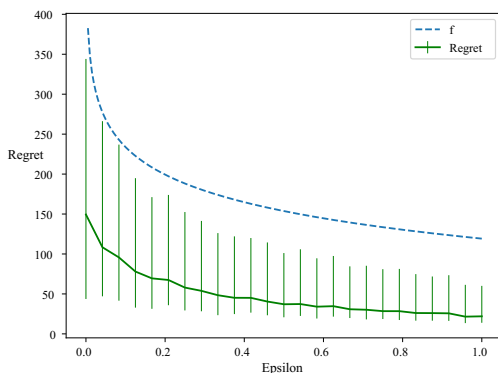


Figure 1: Dependence on  $\epsilon$  of the regret of UCB as passive observer, with a uniform distribution of the free observations, averaged over 300 runs.

### Passive Observer: optimal sampling distribution.

This second experiment illustrates the induced regret in the passive setting with a probability distribution  $p^{(i)} = \frac{1}{\Delta_i}$ . This distribution is considered to be optimal because, as mentioned in Section 3.1, it achieves the lowest lower bound. It also suggests a paradigm for algorithms in the active setting i.e sampling freely as much as possible the arm with the lowest  $\Delta_i$ . A way to do so is to run an UCB type algorithm to choose which arm to pull, and use another UCB type algorithm on other arms to determine which will be observed if a free observation is available. The results of this type of policy is presented in the next paragraph.

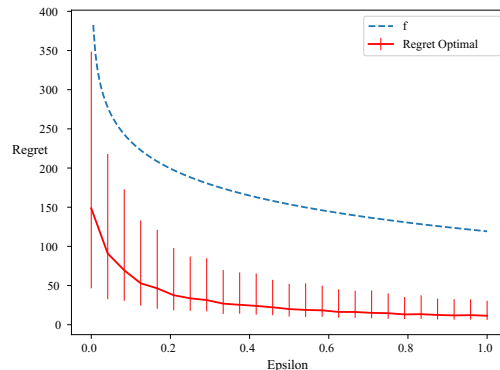


Figure 2: Dependence on  $\epsilon$  of the regret of UCB as passive observer, with the optimal distribution of the free observations, averaged over 300 runs.

The experiment is run on the same set of arms as previously with a uniform distribution, the optimal distribution and a suboptimal one such that  $p^{(i)} = \frac{1}{\Delta_i^2}$ , referred as SubOptimal in Figure 3. Color filled regions are 25% and 75% quantiles.

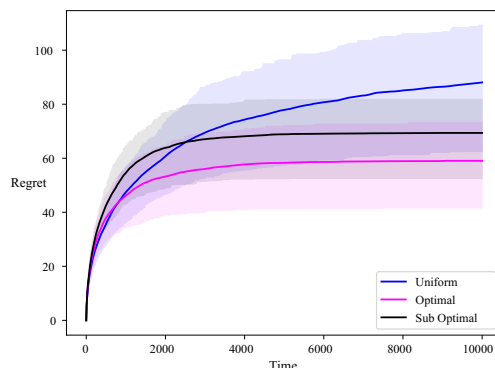


Figure 3: Regret averaged over 300 runs

**Active Observer: comparison of algorithms.** This subsection is dedicated to the comparison of algorithms introduced earlier : UCB1-Double, ETC-OCUCB and ETC-OCUCB-2.

UCB1-Double uses a UCB algorithm and select the free observation as the second index maximising arm. The optimal allocation in the passive setting samples better arms more often, therefore we use the free observation to sample the arm next to optimal (according to its UCB index). The second algorithm, referred to as ETC-OCUCB, is the algorithm studied in the above section. In particular, its ETC subroutine checks for potentially

removable arms every  $C|S|$  pulls, with  $C$  a fixed parameter and  $S$  the set of currently active arm. Finally, the algorithm referred to as ETC-OCUCB-2 is a variant of ETC-OCUCB where elimination checks are made every  $2^k$  stages, thus behaving less aggressively than ETC-OCUCB. In addition, we introduced in this experiment a parameter  $p$  so that the epoch length is  $d_m = p^{2^m}$  in ETC-OCUCB. This enables us to adapt the growth of epochs to the horizon, here  $T = 10^4$ . Other parameters are :  $\alpha = 1$ ,  $\rho = \frac{1}{2}$ ,  $\eta = 2$  and  $C = 10$ .

The experiments is run on five Gaussian arms with expectations 2, 1.8, 1.5, 1 and 0.5. Color filled regions are 25% and 75% quantiles.

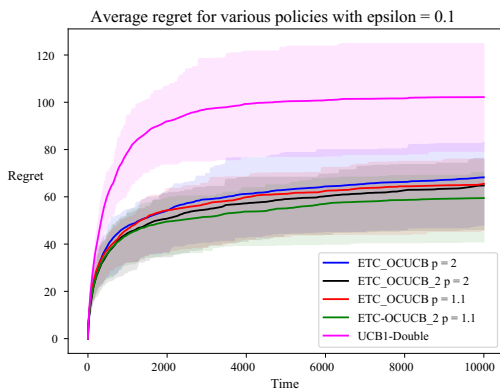


Figure 4: Regret for  $\epsilon = 0.1$  averaged over 100 runs

Figure 4 illustrates that:

- UCBI-Double reaches rapidly its final regret value after a logarithmic exploration phase where informations are gathered so that the policy doesn't pull an other suboptimal arm after this phase.
- ETC-OCUCB and ETC-OCUCB-2 algorithms have similar performances and the parameter  $p$  offers a control how often the set of active arms is updated which offers a slight performance increase for lower  $p$ .

ETC-OCUCB and ETC-OCUCB-2 maintain two distinct tracks of rewards, one for rewards obtained after pulling an arm and the other for rewards after sampling freely an arm. Therefore, it may be possible to increase their performance by using both sources of information in both subroutines. In the Figure below, these variants are referred as ETC-OCUCB-all-info and ETC-OCUCB-all-info-2.

This simple modification provides a clear improvement whether for the final regret or the speed at which this value is reached.

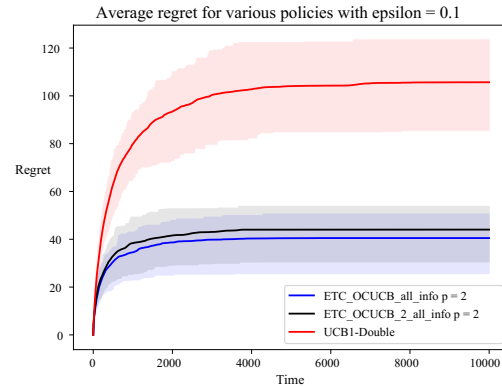


Figure 5: Regret for  $\epsilon = 0.1$  averaged over 300 runs for  $p = 2$

## 6 CONCLUSION

We analysed the multi-armed bandit problem with just a few extra free information. Interestingly, as the regret is uniformly bounded in time, standard lower bounds are void. However, a careful analysis allowed us to exhibit non-trivial guarantee that no reasonable algorithm can out-perform and we finally provided an optimal algorithm, whose regret matches the lower bound up to doubly logarithmic terms.

We would like to finally emphasize that our algorithm can be used even if the  $\epsilon T$  observations are not free. Since we used ETC on these observations, we get that our algorithm has a regret smaller (discarding multiplicative constants and log log terms) than

$$\sum_{i=2}^K \frac{\log(\epsilon T \Delta_i^2)}{\Delta_i} + \sum_{i=2}^K \frac{\log(1/\epsilon)}{\Delta_i}$$

where the first term is the guarantee of ETC on  $\epsilon T$  samples, and the second one is the guarantee of our algorithm with “free” observations. As a consequence, no matter the value of  $\epsilon$  (as long as the log log terms do not become dominant), its dependency vanishes, and we recover the expected performance of ETC.

## Acknowledgements

V. Perchet has benefited from the support of the ANR (grant n.ANR-13- JS01-0004-01), of the FMJH Program Gaspard Monge in optimization and operations research (supported in part by EDF), from the Labex LMH and from the CNRS, PEPS project Lacreme.

## References

- N. Alon, N. Cesa-Bianchi, O. Dekel, and T. Koren. Online learning with feedback graphs: Beyond bandits. In *Conference on Learning Theory*, pages 23–35, 2015.
- J.-Y. Audibert and S. Bubeck. Minimax policies for adversarial and stochastic bandits. In *COLT*, pages 217–226, 2009.
- J.-Y. Audibert and S. Bubeck. Regret bounds and minimax policies under partial monitoring. *Journal of Machine Learning Research*, 11(Oct):2785–2836, 2010.
- P. Auer and R. Ortner. Ucb revisited: Improved regret bounds for the stochastic multi-armed bandit problem. *Periodica Mathematica Hungarica*, 61(1-2):55–65, 2010.
- P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.
- S. Caron, B. Kveton, M. Lelarge, and S. Bhagat. Leveraging side observations in stochastic bandits. *arXiv preprint arXiv:1210.4839*, 2012.
- N. Cesa-Bianchi, G. Lugosi, and G. Stoltz. Regret minimization under partial monitoring. *Mathematics of Operations Research*, 31(3):562–580, 2006.
- I. Chatzigeorgiou. Bounds on the lambert function and their application to the outage analysis of user cooperation. *IEEE Communications Letters*, 17(8):1505–1508, 2013.
- W. Chen, Y. Wang, Y. Yuan, and Q. Wang. Combinatorial multi-armed bandit and its extension to probabilistically triggered arms. *The Journal of Machine Learning Research*, 17(1):1746–1778, 2016.
- R. Degenne and V. Perchet. Anytime optimal algorithms in stochastic multi-armed bandits. In *International Conference on Machine Learning*, pages 1587–1595, 2016.
- E. Even-Dar, S. Mannor, and Y. Mansour. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of machine learning research*, 7(Jun):1079–1105, 2006.
- A. Garivier, P. Ménard, and G. Stoltz. Explore first, exploit next: The true shape of regret in bandit problems. *arXiv preprint arXiv:1602.07182*, 2016.
- T. L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22, 1985.
- T. Lattimore. Regret analysis of the anytime optimally confident ucb algorithm. *arXiv preprint arXiv:1603.08661*, 2016.
- S. Mannor and O. Shamir. From bandits to experts: On the value of side-observations. In *Advances in Neural Information Processing Systems*, pages 684–692, 2011.
- M. Okamoto. Some inequalities relating to the partial sum of binomial probabilities. *Annals of the institute of Statistical Mathematics*, 10(1):29–35, 1959.
- V. Perchet and P. Rigollet. The multi-armed bandit problem with covariates. *The Annals of Statistics*, pages 693–721, 2013.
- J. Y. Yu and S. Mannor. Piecewise-stationary bandit problems with side observations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1177–1184. ACM, 2009.



---

# Sampling and Inference for Beta Neutral-to-the-Left Models of Sparse Networks

---

**Benjamin Bloem-Reddy**  
Department of Statistics  
University of Oxford

**Adam Foster**  
Department of Statistics  
University of Oxford

**Emile Mathieu**  
Department of Statistics  
University of Oxford

**Yee Whye Teh**  
Department of Statistics  
University of Oxford

## Abstract

Empirical evidence suggests that heavy-tailed degree distributions occurring in many real networks are well-approximated by power laws with exponents  $\eta$  that may take values either less than and greater than two. Models based on various forms of exchangeability are able to capture power laws with  $\eta < 2$ , and admit tractable inference algorithms; we draw on previous results to show that  $\eta > 2$  cannot be generated by the forms of exchangeability used in existing random graph models. Preferential attachment models generate power law exponents greater than two, but have been of limited use as statistical models due to the inherent difficulty of performing inference in non-exchangeable models. Motivated by this gap, we design and implement inference algorithms for a recently proposed class of models that generates  $\eta$  of all possible values. We show that although they are not exchangeable, these models have probabilistic structure amenable to inference. Our methods make a large class of previously intractable models useful for statistical inference.

## 1 INTRODUCTION

Sparsity and heavy-tailed degree distributions are believed to occur in many real networks (Newman, 2005; Clauset, Shalizi, and Newman, 2009). Sparsity has been well-studied and is an intuitive concept: The typical social network user interacts with only a vanishing fraction of all users as the network grows. Heavy-tailed degree distributions and the mechanisms that generate them are not as well understood. However, empirical evidence indicates that heavy-tailed distributions are expressed in a wide range of settings, including network degrees (Clauset,

Shalizi, and Newman, 2009). Power law degree distributions, in which the proportion of vertices with degree  $d$  is  $\propto d^{-\eta}$ , are often used as models for real degree distributions, and serve as a useful analytic tool for characterizing the asymptotic properties of random network models.

Many statistical network models make the assumption of exchangeability over vertices, appealing to the Aldous–Hoover theorem (Hoover, 1979; Aldous, 1981) for theoretical justification. Noting that networks sampled from these models cannot be sparse, Orbanz and Roy (2015) posed a question paraphrased as, “Can a probabilistic model for random graphs produce sparse networks and have some useful notion of probabilistic symmetry?” A generation of models answered in the affirmative by incorporating other notions of exchangeability: In an exchangeable point process representation of a network (Caron and Fox, 2017; Veitch and Roy, 2015; Borgs et al., 2016), or as an exchangeable sequence of edges (Crane and Dempsey, 2017; Cai, Campbell, and Broderick, 2016; Williamson, 2016). Under certain parameterizations, these models generate sparse networks. They are able to generate asymptotic power law degree distributions, providing a better fit to real network data than their vertex exchangeable counterparts. However, the power law exponent of the degree distribution in both model classes is constrained to the interval  $\eta \in (1, 2)$ . That interval is not an artifact of particular model specifications. Rather, it is a basic property resulting from the fact that the average vertex degree is asymptotically unbounded; vertex degrees grow, on average, linearly in the number of edges. For some data, this property may be undesirable; such properties ideally would be inferred from a model able to capture a larger range of power law behavior.

In a largely disjoint literature, so-called preferential attachment (PA) models have been studied primarily for their ability to generate power law degree distributions from a simple size-biased reinforcement mechanism, and for their analytical tractability (e.g., Barabási and Albert, 1999; Berger et al., 2014; Peköz, Röllin, and Ross, 2017).

PA models have power law exponents  $\eta > 2$ . As we explain in Section 2, the exponent range is tied to PA models' non-exchangeability—a property that has made them, until now, of limited use as statistical models. In particular, if the history of the network is unobserved, the order of the edges must be inferred or marginalized; even for networks of modest size, such inference over permutations is generally intractable.

Recently, Bloem-Reddy and Orbanz (2017) introduced a class of models that can generate random graphs with power law degree distributions of any exponent  $\eta \in (1, \infty)$ . For reasons discussed below, we propose naming them Beta Neutral-to-the-Left (BNTL) models. BNTL models generalize many known models that have a size-biased reinforcement mechanism, including a sub-class of edge exchangeable models based on the Pitman–Yor process, and variations of the PA model. The cost of the additional flexibility is exchangeability; BNTL models depend on the times at which new vertices arrive and are not exchangeable in any known sense. However, as we show in Section 3, BNTL models have probabilistic structure—namely, left-neutrality—that may be exploited for efficient computation, making a large class of previously intractable models useful for statistical inference.

Bloem-Reddy and Orbanz (2017) established the asymptotic properties of BNTL models; statistical modeling and inference were left unstudied. Our contributions are:

- We identify left-neutrality as the key property that yields tractable inference schemes.
- We categorize and give solutions to the BNTL inference problem based on what data are available: We design schemes for maximum likelihood estimation when vertex arrival times are observed, and for Bayesian inference when an unlabeled network is observed.
- We implement these schemes on real networks of various sizes, from modest ( $\sim 10^2$  vertices) to massive ( $\sim 10^6$  vertices).

## 2 POWER LAWS IN RANDOM GRAPH MODELS

This section provides some context, and collects and interprets various results from random graph models with asymptotic degree distributions exhibiting power law tails. Although none of the results here are new, to our knowledge they have not been coherently synthesized in the literature. Technical details are omitted; they may be found in the references given throughout the section. We focus our attention on edge exchangeable and PA models because they are most similar to the BNTL framework.

A graph  $G$  is a set of vertices,  $\mathcal{V}(G)$ , and of edges,<sup>1</sup>  $\mathcal{E}(G)$ , between them. A multigraph allows for multiple edges between vertices; we consider each edge to be distinct, rather than as one integer-valued edge. We consider only multigraphs and henceforth refer to them as graphs. A sequence of growing graphs  $G_1, G_2, \dots$  is a stochastic process  $\mathbf{G}$ , indexed by the number of edges,  $n$ . Hence,  $G_n$  may be interpreted as  $G_{n-1}$  with an additional edge, either between two vertices in  $\mathcal{V}(G_{n-1})$ , between two new vertices, or between one old and one new vertex. We assume that the edges are labeled according to the order in which they appear, though this assumption is not necessary for edge exchangeable models (discussed below). As such,  $G_n$  may be viewed simply as a sequence of edges  $\mathbf{E}_n := (E_1, \dots, E_n)$  or, even more simply, as a sequence of *ends of edges*, denoted  $\mathbf{Z}_{2n} = (Z_1, \dots, Z_{2n})$ . We denote by  $G(\mathbf{Z}_{2n})$  the labeled graph with  $n$  edges constructed from  $\mathbf{Z}_{2n}$ . (For convenience, we will use the subscript  $n$  for all sequences when there is no risk of confusion.)

For a graph  $G(\mathbf{Z}_n)$ ,  $K_n := |\mathcal{V}(G(\mathbf{Z}_n))|$  is the number of vertices (i.e., the number of unique values in  $\mathbf{Z}_n$ ); the degree of vertex  $j$ ,  $d_{j,n} := \sum_{i=1}^n \mathbb{1}\{Z_i = j\}$ , is equal to the number of ends of edges connected to it. Let  $m_n(d)$  denote the number of vertices with degree  $d$ . The asymptotic degree distribution of  $G_1, G_2, \dots$  is said to have power law tail with exponent  $\eta > 1$  if

$$p_n(d) = \frac{m_n(d)}{K_n} \xrightarrow[n \rightarrow \infty]{p} p_d \stackrel{a^{\uparrow} \infty}{\sim} L(d)d^{-\eta} \quad \text{for large } d,$$

such that  $\sum_{d \geq 1} p_d = 1$ , for some slowly varying function  $L(d)$ :  $\lim_{x \rightarrow \infty} L(rx)/L(x) = 1$  for all  $r > 0$  (Bingham, Goldie, and Teugels, 1989). For power law tails, we state the following fact (see Appendix A).

*Fact.* As  $n \rightarrow \infty$ , if the expected average degree is unbounded, then  $\eta \in (1, 2)$ ; if it is bounded,  $\eta \in (2, \infty)$ .

**Edge exchangeable models (Crane and Dempsey, 2017; Cai, Campbell, and Broderick, 2016).** Let  $G_n$  be specified by its sequence of edges  $\mathbf{E}_n$  (not necessarily ends of edges), which is assumed to be exchangeable: Its distribution is invariant under all permutations of the order of the edges for all  $n$ , i.e., the labels carry no information about their distribution. As a consequence of the law of large numbers for exchangeable sequences, the counts of all non-zero multi-edges grow linearly in  $n$  and thus so do the vertex degrees. That is,  $d_{j,n} = \Theta(n)$ . Furthermore,  $K_n = o(n)$ . The average degree is unbounded, implying that if the degree distribution tail follows a power law, then  $\eta \in (1, 2)$ .

<sup>1</sup>We treat all graphs as undirected; extension to directed graphs is straightforward.

As an example, consider sampling  $\mathbf{Z}$  from the the Pitman–Yor process ( $\mathcal{PYP}$ ) (Ishwaran and James, 2001) with parameters  $\tau \in (0, 1)$ ,  $\theta > -\tau$ ,

$$\mathbb{P}[Z_{n+1} \in \cdot | \mathbf{Z}_n] = \frac{\theta + K_n \tau}{n + \theta} \delta_{K_{n+1}}(\cdot) + \frac{n - K_n \tau}{\theta + n} \sum_{j=1}^{K_n} \frac{d_{j,n} - \tau}{n - K_n \tau} \delta_j(\cdot). \quad (1)$$

It can be shown that the asymptotic degree distribution has power law tail (Pitman, 2006),

$$n^{-\tau} m_d(n) \xrightarrow[n \rightarrow \infty]{\text{a.s.}} p_d \stackrel{d \uparrow \infty}{\sim} d^{-(1+\tau)},$$

which implies that  $\eta_\tau = 1 + \tau \in (1, 2)$ .

The predictive rule (1) demonstrates why the expected average degree is unbounded. The probability that  $Z_{n+1}$  corresponds to a new vertex is  $\frac{\theta + K_n \tau}{n + \theta}$ , which is arbitrarily close to zero as  $n \rightarrow \infty$ . For large  $n$ , the expected interarrival time between new vertices becomes arbitrarily large, and edges pile up on the existing vertices. Intuitively, vertex  $j$  takes part in a constant fraction of all interactions as  $n$  grows. This property is shared by all edge exchangeable models; an analogous property holds for exchangeable point process models (see Appendix B).

**Preferential attachment models.** Although the  $\mathcal{PYP}$  has the same size-biased reinforcement mechanism common to all PA models, typically it is not considered to be part of the same class as the PA models in the probability literature, of which Barabási and Albert (1999) provide the prototypical example. However, the difference between them amounts to how frequently new vertices appear (Bloem-Reddy and Orbanz, 2017). For our purposes, this is best illustrated with a simple PA model, the Yule–Simon (YS) model (Simon, 1955). For  $\beta \in (0, 1)$ ,  $\mathbf{Z}$  is generated via the predictive rule

$$\mathbb{P}[Z_{n+1} \in \cdot | \mathbf{Z}_n] = \beta \delta_{K_{n+1}}(\cdot) + (1 - \beta) \sum_{j=1}^{K_n} \frac{d_{j,n}}{n} \delta_j(\cdot). \quad (2)$$

The YS model is known to generate power law degree distributions with  $\eta_\beta = 1 + \frac{1}{1-\beta} \in (2, \infty)$  (Simon, 1955). Different versions of PA exhibit a range of possible  $\eta$ 's, but it is generally the case that  $\eta_{\text{PA}} > 2$ , and this is tied to their lack of exchangeability. The average rate at which new vertices arrive is constant in  $n$ ; hence,  $K_n = \Omega(n)$ , implying bounded expected average degree. The “edge pileup” phenomenon of exchangeable models does not occur:  $d_{j,n} = o(n)$ . In the YS model,  $d_{j,n} = \Theta(n^{1-\beta})$ .

### 3 BETA NTL MODELS

BNTL models were introduced under the name  $(\alpha, T)$ -models by Bloem-Reddy and Orbanz (2017), who studied their distributional and asymptotic properties. We briefly review the definition of BNTL models and describe the properties that make them amenable to inference.

In the predictive distributions (1)-(2), the probability that  $Z_{n+1}$  is a new vertex is independent of the degrees  $d_{j,n}$ , which allows the sampling of  $\mathbf{Z}$  to be separated into two parts: A sequence  $T_1 < T_2 < \dots$  of *arrival times* of new vertices, and size-biased reinforcement at all steps not associated with an arrival time. As such, a BNTL model is parameterized by a scalar “discount parameter”  $\alpha \in (-\infty, 1)$  and a probability distribution  $\Lambda$  on strictly increasing integer-valued sequences, which specifies the law of the arrival times  $T_1, T_2, \dots$ . A sequence  $\mathbf{Z}$  is said to have law  $\text{BNTL}(\alpha, \Lambda)$  if, for a random arrival time sequence  $\mathbf{T} = (T_1, T_2, \dots) \sim \Lambda$ ,  $\mathbf{Z}$  is sampled as

$$\mathbb{P}[Z_{n+1} \in \cdot | \mathbf{Z}_n, \mathbf{T}] = \mathbb{1}\{n+1 = T_{K_{n+1}}\} \delta_{K_{n+1}}(\cdot) + \mathbb{1}\{n+1 < T_{K_{n+1}}\} \sum_{j=1}^{K_n} \frac{d_{j,n} - \alpha}{n - K_n \alpha} \delta_j(\cdot). \quad (3)$$

In practice, it may be simpler to specify the distribution of *interarrival times*  $\Delta_j = T_j - T_{j-1}$ , and use their partial sums to construct  $\mathbf{T}$ ; we discuss this in more detail in Section 4. The similarity of (3) to (1)-(2) is not coincidental. The  $\mathcal{PYP}$  and the YS model each correspond to particular parameterizations of the BNTL model: The YS model corresponds to i.i.d.  $\Delta_j \sim \text{Geom}(\beta)$ ; the arrival time distribution induced by the  $\mathcal{PYP}$  in (1) also has known form (see (11)).

For a given  $\mathbf{T}$ , the probability of any  $G(\mathbf{Z}_n)$  is

$$\mathbb{P}[G(\mathbf{Z}_n) | \mathbf{T}] = \mathbb{P}[G(\mathbf{Z}_n) | \mathbf{T}_{K_{n+1}}, K_n] = \frac{\Gamma(d_{1,n} - \alpha)}{\Gamma(n - K_n \alpha)} \prod_{j=2}^{K_n} \frac{\Gamma(T_j - j\alpha) \Gamma(d_{j,n} - \alpha)}{\Gamma(T_j - 1 - (j-1)\alpha) \Gamma(1 - \alpha)}. \quad (4)$$

A crucial property that makes BNTL models amenable to inference is that conditioned on  $\mathbf{T}$ , the joint probability (4) factorizes over the vertices; each term is expressed in terms of its arrival time,  $T_j$ , and its degree,  $d_{j,n}$ . Note that given  $\mathbf{T}$ , the degree sequence  $\mathbf{d}_{K_n} := (d_{1,n}, \dots, d_{K_n,n})$  is a sufficient statistic for  $\alpha$ . Furthermore, the distribution of the arrival times (and therefore  $K_n$ ) is independent of the degrees. The factorization becomes explicitly useful in the Gibbs sampling updates and in the maximum likelihood estimating equations in Section 4.

**Sampling representation.** Like their exchangeable counterpart the  $\mathcal{PYP}$ , BNTL models have a sampling representation in terms of products of independent beta random

variables: (3) is an urn sequence corresponding to the following (Bloem-Reddy and Orbanz, 2017):

- $\mathbf{T} \sim \Lambda$ .
- $\Psi_j | T_j \sim \text{Beta}(1 - \alpha, T_j - 1 - (j - 1)\alpha)$  for  $j \geq 1$ .
- $P_{j, K_n} = \Psi_j \prod_{\ell=j+1}^{K_n} (1 - \Psi_\ell)$
- $Z_n \sim \begin{cases} \delta_{K_n}(\cdot) & \text{for } n = T_{K_n} \\ \text{Categorical}(P_{j, K_n}) & \text{o.w.} \end{cases}$

(By convention,  $\text{Beta}(a, 0)$  is a point mass on 1, so  $\Psi_1 = 1$ .) The last two items specify that when there are  $k$  vertices in the graph,  $Z_n$  is sampled from a categorical distribution over those vertices, each with probability  $P_{j, k}$ . After the subsequent arrival time,  $T_{k+1}$ , when there are  $k + 1$  vertices, the probability that  $Z_n = j$  is

$$P_{j, k+1} = \begin{cases} P_{j, k}(1 - \Psi_{k+1}), & j \in \{1, \dots, k\} \\ \Psi_{k+1}, & j = k + 1 \end{cases}.$$

That is, the vector of probabilities  $\mathbf{P}_k = (P_{1, k}, \dots, P_{k, k})$  grows in length as each new vertex arrives, and each of the previous entries is scaled by  $(1 - \Psi_{\text{new}})$ .

**Neutrality.** The recursive scaling of  $P_{j, k}$  is the essence of a neutral-to-the-left (NTL) sequence. A random vector  $\mathbf{X} = (X_1, \dots, X_k) \in \mathbb{R}_+^k$ , is NTL if the increments,

$$R_j := \frac{X_j}{\sum_{i=1}^j X_i}, \quad (5)$$

form a sequence of mutually independent random variables; a non-decreasing stochastic process  $Z$  defined on  $\mathbb{R}$  is NTL if the vector of increments  $\frac{Z(t_j) - Z(t_{j-1})}{Z(t_j)}$  is NTL for any finite partition  $-\infty \leq t_1 < \dots < t_k \leq \infty$  of  $\mathbb{R}$  (Doksum, 1974). A bit of algebra shows that  $\mathbf{P}_k$  is NTL: The corresponding sequence of increments is  $R_j = \Psi_j$ , for all  $k$ . Intuitively, this must be the case due to the recursive scaling construction. Together with the beta random variables in the sampling representation, left-neutrality characterizes these models; hence the name.

Neutral-to-the-right (NTR) processes are better known than NTL processes, and appear throughout the Bayesian statistics literature, both explicitly (Walker and Muliere, 1997; James, 2006) and implicitly in the form of the stick-breaking constructions of the Dirichlet Process and the  $\mathcal{PYP}$  (e.g., Ishwaran and James, 2001). The properties of right- and left-neutrality are, as their names suggest, symmetric opposites: A NTR vector in reverse order is NTL, and vice versa.

The independence properties that make NTR stick-breaking constructions useful for modeling and inference purposes transfer in large part to NTL models. The  $\Psi_j$ 's are conditionally independent given the  $T_j$ 's; along with

the parameters of the beta distribution, this independence induces the factorized form in (4). In the exchangeable random partitions literature, a model with joint probability that factorizes over the blocks and the probability of having  $K_n$  blocks is known as *Gibbs-type* (Gnedin and Pitman, 2006).

**Sparsity and power law tails in BNTL models.** The asymptotic behavior of BNTL models is controlled primarily by the arrival times,  $\mathbf{T}$ . In order to obtain sparse graphs,  $K_n$  must be  $\omega(n^{1/2})$ . If  $\mathbf{T}$  are the arrival times from an exchangeable sequence  $\mathbf{Z}$ , then  $K_n$  is at most  $\Theta(n^\delta)$ , for some  $\delta \in (0, 1)$ , in which case  $\eta = 1 + \delta$  (Pitman, 2006); thus, sparse graphs generated this way have  $\eta \in (3/2, 2)$ . For the  $\mathcal{PYP}$ ,  $\delta = \tau$ . Alternatively, for  $\mathbf{T}$  sampled such that the mean interarrival time,

$$\bar{\Delta}_{K_n} := \frac{1}{K_n - 1} \sum_{j=2}^{K_n} \Delta_j, \quad (6)$$

converges to some finite  $\mu$ , then  $K_n = \Theta(n)$  and  $\eta = 1 + \frac{\mu - \alpha}{\mu - 1} > 2$ . Furthermore, vertex degrees grow as  $d_{j, n} = \Theta(n^{\frac{\mu-1}{\mu-\alpha}})$  (Bloem-Reddy and Orbanz, 2017). Thus, depending on the specification of the arrival time distribution, BNTL models can achieve any  $\eta \in (1, \infty)$ .

**Microclustering in BNTL partitions.** The sequence  $\mathbf{Z}_n$  can be transformed into an arrival-ordered partition  $\Pi(\mathbf{Z}_n) := \{B_{1, n}, \dots, B_{K_n, n}\}$  of  $[n] := \{1, \dots, n\}$  by grouping  $\mathbf{Z}_n$  into blocks  $B_{j, n} := \{i \in [n] : Z_i = j\}$ . There is a bijective mapping between  $\Pi(\mathbf{Z}_n)$  and  $G(\mathbf{Z}_n)$  for all  $n$  (Bloem-Reddy and Orbanz, 2017), which puts blocks of the partition in correspondence with vertices of the graph. Hence, properties of  $G(\mathbf{Z}_n)$  translate into properties of  $\Pi(\mathbf{Z}_n)$ . In particular, the growth rate of vertex degrees translates to the growth rate of blocks sizes. Recent work (Betancourt et al., 2016; Di Benedetto, Caron, and Teh, 2017) has explored the so-called microclustering property, which is defined as block sizes that grow sub-linearly in  $n$ . The  $\eta > 2$  range of BNTL models corresponds precisely with this property. Although we do not make explicit statements about partition-valued data, statements about graphs are easily translated into statements about partitions via the correspondence between blocks and vertices. In particular, the inference algorithms in Section 4 are valid for partition-valued data.

## 4 INFERENCE

Although PA models exhibit a range of power laws not captured by exchangeable models, they face a significant barrier to use as statistical models due to their inherent lack of exchangeability. At a high level, applying a non-exchangeable model to data for which the order is unknown requires inference over permutations of the data.

This is, in general, a prohibitively difficult problem even for modest  $n$ . However, using the probabilistic structure of BNTL models, we design a Gibbs sampling algorithm that overcomes this difficulty for networks with thousands of vertices (Section 4.1). If the ordered edge sequence is observed, maximum likelihood estimation scales to networks with millions of vertices (Section 4.2).

Given the hierarchical nature of BNTL models, inference may be performed at a number of levels. In the simplest case, suppose the data are a sequence of edge-ends,  $\mathbf{Z}_n$ . From this sequence the arrival times and the arrival-ordered graph can be perfectly reconstructed, and inferring the parameters  $\phi$  of the arrival distribution  $\Lambda^\phi$  and the parameters  $\Psi_{K_n} := (\Psi_j)_{j=1}^{K_n}$  is straightforward: Simple maximum likelihood estimators exist for  $\Psi_{K_n}$  (see Appendix C), and for the parameters of many arrival time distributions of interest, or equally simple MCMC samplers may be constructed for Bayesian inference.

More challenging are the situations in which some aspect of the data is not perfectly observed. For graph-valued observations, the following table summarizes the range of possibilities, in order of increasing difficulty of inference:

Observation	Unobserved variables
End of edge sequence $\mathbf{Z}_n$	$\alpha, \phi, \Psi_{K_n}$
Vertex arrival-ordered graph	$\alpha, \phi, \Psi_{K_n}, \mathbf{T}_{K_n}$
Unlabeled graph	$\alpha, \phi, \Psi_{K_n}, \mathbf{T}_{K_n}, \sigma[K_n]$

The last row presents a significant challenge. In particular, the unobserved variables include a permutation  $\sigma$  mapping the arrival-ordered sequence to some arbitrary ordering of the vertices (by which the vertices are uniquely identified). For a graph with  $K_n$  vertices, there are  $K_n!$  possible permutations. Conditioned on a sequence of arrival times, some permutations have zero posterior probability, making the problem space both high-dimensional and constrained. Despite these difficulties, the inference problem is much simpler than that of a generic non-exchangeable model for a sequence of  $n$  data points: Even in sparse graphs, typically  $K_n \ll n$  and thus the dimension of the problem is exponentially smaller. Furthermore, the form of (4) yields simple conditional distributions for Gibbs sampling.

#### 4.1 GIBBS SAMPLING

In this section, we build from the simplest inference problem to the hardest, progressing through the table in the previous section. The full sampler infers the posterior distributions of the parameters  $\Psi_{K_n}$  and  $\alpha$ , of the arrival times  $\mathbf{T}_{K_n}$ , of the parameters of the arrival time distribution, and of the permutation of the vertices. In order to maintain the structure of the factorization over vertices

in (4), we assume that the arrival time distribution has a Markov factorization (with a slight abuse of notation):

$$\Lambda^\phi(\mathbf{T}_k) = \delta_{T_1}(1) \prod_{j=2}^k \Lambda_j^\phi(\Delta_j | T_{j-1}), \quad (7)$$

with  $\phi$  representing any parameters. Examples are i.i.d. interarrivals such that  $\Lambda_j^\phi(\Delta_j | T_{j-1}) = p_\phi(\Delta_j)$ ; and interarrivals that depend on the previous interarrivals through their sum and the number of previous arrivals, such as the interarrival sequence generated by exchangeable Gibbs-type sequences (De Blasi et al., 2015).

Suppose we observe a sequence of edge-ends  $\mathbf{Z}_n$ . Denote the partial sums of the ordered degree sequence as  $\bar{d}_{j,n} = \sum_{i=1}^j d_{i,n}$ . For any fixed  $\alpha$  and  $\phi$ ,

$$p_{\alpha,\phi}(\mathbf{Z}_n, \Psi_{K_n}) = \prod_{j=2}^{K_n} \frac{\Psi_j^{d_{j,n} - \alpha - 1} (1 - \Psi_j)^{\bar{d}_{j-1,n} - (j-1)\alpha - 1}}{B(1 - \alpha, T_j - 1 - (j-1)\alpha)} \Lambda_\phi(T_j | T_{j-1}) \times \Lambda_\phi(T_{K_n+1} > n | T_{K_n}), \quad (8)$$

where  $B(a, b)$  is the beta function, and  $\Lambda_\phi(T_{K_n+1} > n | T_{K_n})$  is the censored probability of vertex  $K_n + 1$ 's unobserved arrival time. Note that marginalizing  $\Psi_{K_n}$  recovers (4).

**Updates for  $\Psi_{K_n}$ .** From (8) it is clear that

$$\Psi_j | \mathbf{Z}_n, \Psi_{\setminus j} \sim \text{Beta}(d_{j,n} - \alpha, \bar{d}_{j-1,n} - (j-1)\alpha), \quad (9)$$

where  $\Psi_{\setminus j}$  is shorthand for the sequence  $\Psi_{K_n}$  with  $\Psi_j$  excluded. That is, given the arrival-ordered block sizes, the  $\Psi_{K_n}$  are independent of each other and of the arrival times, and the beta distribution is the conjugate prior for the BNTL sampling process. To understand this, consider a second scenario in which a graph is observed with vertices labeled in order of arrival (though not their time of arrival). The data consist of an ordered sequence of degrees,  $\mathbf{d}_{K_n} = (d_1, \dots, d_{K_n})$ , which corresponds to more than one possible edge-end sequence  $\mathbf{Z}_n$ . The model places equal probability on each sequence that gives rise to the same arrival-ordered degree sequence  $\mathbf{d}_{K_n}$  and the same arrival times; summing over these sequences yields

$$p_{\alpha,\phi}(\mathbf{d}_{K_n}, \Psi_{K_n} | \mathbf{T}_{K_n}, K_n) = \prod_{j=2}^{K_n} \frac{\Psi_j^{d_j - \alpha - 1} (1 - \Psi_j)^{\bar{d}_{j-1} - (j-1)\alpha - 1}}{B(1 - \alpha, T_j - 1 - (j-1)\alpha)} \binom{\bar{d}_j - T_j}{d_j - 1}. \quad (10)$$

The binomial coefficients count the number of sequences  $\mathbf{Z}_n$  that yield  $\mathbf{d}_{K_n}$ , given  $\mathbf{T}_{K_n}$  (Griffiths and Spanò, 2007). (10) is a product of binomial likelihoods with beta priors. Hence, the conjugacy derived in (9).

**Updates for  $\alpha$ .** In both observation scenarios, generic MCMC methods such as slice sampling (Neal, 2003) can be used to sample from the full conditional distribution of  $\alpha$ . We use slice sampling in the experiments in Section 5.

**Updates for  $\phi$ .** Many models of i.i.d. interarrival times will yield conjugate updates for  $\phi$ . For other models, generic MCMC methods can be used. In the experiments in Section 5, we consider three interarrival models: i.i.d.  $\text{Geom}(\beta)$  and i.i.d.  $\text{Pois}_+(\lambda)$ , which is the Poisson distribution shifted to the positive integers; and the interarrival distribution induced by the  $\mathcal{PYP}$ , which is (Griffiths and Spanò, 2007)

$$\begin{aligned} \Lambda_{j+1}^{\theta, \tau}(\Delta_{j+1} = s \mid T_j) & \quad (11) \\ & = (\theta + j\tau) \frac{\Gamma(\theta + T_j)\Gamma(T_j + s - 1 - j\tau)}{\Gamma(\theta + T_j + s)\Gamma(T_j - j\tau)}. \end{aligned}$$

In the former two cases, conjugate updates are performed (conditioned on  $\mathbf{T}_{K_n}$ ); in the latter case, we perform univariate slice sampling for each of  $\theta$  and  $\tau$ .

**Updates for  $\mathbf{T}_{K_n}$ .** The assumed Markov structure of the arrival times induces a simple conditional distribution for  $T_j$  that is supported on the set  $S_j = \{T_{j-1} + 1, \dots, T_{j-1} + M_j\}$ , where  $M_j = \min\{T_{j+1} - T_{j-1} - 1, \bar{d}_{j-1} - T_{j-1} + 1\}$ . The support set enforces the constraints that  $\bar{d}_{j-1} \geq T_j - 1$ , and that  $T_{j-1} < T_j < T_{j+1}$ . Conditioning on  $T_{j-1}$  and  $T_{j+1}$ , updating  $T_j$  is equivalent to updating  $\Delta_j$  and  $\Delta_{j+1}$ ; for  $j = 2, \dots, K_n - 1$ ,

$$\begin{aligned} p_{\alpha, \phi}(\Delta_j = s, \Delta_{j+1} = T_{j+1} - T_{j-1} - s \mid \mathbf{T}_{\setminus j}, \mathbf{d}_n) & \\ \propto \frac{\Lambda_{j+1}^{\phi}(T_{j+1} - T_{j-1} - s \mid T_{j-1} + s)\Lambda_j^{\phi}(s \mid T_{j-1})}{B(1 - \alpha, T_{j-1} + s - 1 - (j - 1)\alpha)} & \\ \times \binom{\bar{d}_j - T_{j-1} - s}{d_j - 1}. & \end{aligned}$$

For  $j = K_n$ ,

$$\begin{aligned} p_{\alpha, \phi}(\Delta_{K_n} = s \mid \mathbf{T}_{\setminus K_n}, \mathbf{d}) & \\ \propto \Lambda_{K_n}^{\phi}(s \mid T_{K_n-1}) \binom{n - T_{K_n-1} - s}{d_{K_n} - 1} & \\ \times \frac{\Lambda_{K_n+1}^{\phi}(\Delta_{K_n+1} > n - T_{K_n-1} - s \mid T_{K_n-1} - s)}{B(1 - \alpha, T_{K_n-1} + s - 1 - (K_n - 1)\alpha)}, & \end{aligned}$$

and  $M_{K_n} = \min\{n - T_{K_n-1} - 1, \bar{d}_{j-1} - T_{j-1} + 1\}$ .

For i.i.d. interarrivals with distribution  $p_{\phi}$ , the updates are particularly easy to compute because

$$\begin{aligned} \Lambda_{j+1}^{\phi}(T_{j+1} - T_{j-1} - s \mid T_{j-1} - s)\Lambda_j^{\phi}(s \mid T_{j-1}) & \\ = p_{\phi}(T_{j+1} - T_{j-1} - s)p_{\phi}(s). & \quad (12) \end{aligned}$$

$p_{\phi}(s)$  can be computed for each  $s \in \{1, \dots, M_j\}$ ; each term multiplied by the corresponding term in  $s \in$

$\{M_j, \dots, 1\}$  yields (12). In the case of  $\text{Geom}(\beta)$  interarrivals, the distribution is uniform on  $s \in \{1, \dots, M_j\}$ :

$$\begin{aligned} \Lambda_{j+1}^{\phi}(T_{j+1} - T_{j-1} - s \mid T_{j-1} - s)\Lambda_j^{\phi}(s \mid T_{j-1}) & \quad (13) \\ = \beta(1 - \beta)^{T_{j+1} - T_{j-1} - s - 1}\beta(1 - \beta)^{s-1} \propto 1. & \end{aligned}$$

**Updates for  $\sigma[K_n]$ .** Given a sample of  $\mathbf{T}_{K_n}$ , the order of the vertices can be updated via a series of adjacent swap proposals. Let  $\sigma_j$  be the identity of the  $j$ -th vertex in the current sampling iteration. A sampling update of  $\sigma$  proposes swapping  $\sigma_j \leftrightarrow \sigma_{j+1}$  with probability proportional to the value of (10), with  $\Psi_{K_n}$  marginalized and with  $d_j$  and  $d_{j+1}$  swapped. Due to the factorization over vertices, all but the  $j$ -th and  $j + 1$ -st terms are the same; as a result, swap proposals are inexpensive to compute (for compactness, ‘-’ indicates all other variables):

$$\begin{aligned} p_{\alpha, \phi}(\sigma_j \leftrightarrow \sigma_{j+1} \mid -) & \propto \frac{\Gamma(\bar{d}_{j-1} + d_{j+1} - T_j + 1)}{\Gamma(\bar{d}_{j+1} - d_j - T_{j+1} + 2)} \\ p_{\alpha, \phi}(\sigma_j \not\leftrightarrow \sigma_{j+1} \mid -) & \propto \frac{\Gamma(\bar{d}_{j-1} + d_j - T_j + 1)}{\Gamma(\bar{d}_j - T_{j+1} + 2)}. \end{aligned}$$

The simplicity of swap proposals enables many swaps to be sampled in a short amount of computational time, helping to overcome the high dimensionality of the sample space. We note that in general, local proposals of all possible permutations of  $m > 1$  consecutive vertices are possible and  $m > 2$  would likely enhance exploration of the state space; here we consider only  $m = 2$ .

**Computational complexity.** The slice sampling updates for  $\alpha$ , which require evaluation of (4), are of complexity  $\mathcal{O}(K_n)$ , as are the permutation swap proposals. Updates for the arrival parameter(s)  $\phi$  depend on the model, but as they depend only on the  $K_n$  arrival times, they are at most  $\mathcal{O}(K_n)$ . The most expensive update is that of  $\mathbf{T}_{K_n}$ , which is  $\mathcal{O}(n)$ , though the constant hidden in  $\mathcal{O}$  may vary greatly across arrival models.

## 4.2 MAXIMUM LIKELIHOOD FOR PARAMETERS IN EDGE SEQUENCES

Suppose the edge-end sequence  $\mathbf{Z}_n$  is observed. For arrival time distribution  $\Lambda^{\phi}$ ,  $\phi$  and  $\alpha$  can be estimated by maximum likelihood (ML). The likelihood admits the factorization

$$p_{\alpha, \phi}(\mathbf{Z}_n) = p_{\alpha}(\mathbf{Z}_n \mid \mathbf{T}_{K_n})\Lambda_{\phi}(\mathbf{T}_{K_n}), \quad (14)$$

with the practical implication that the estimating equations for  $\alpha$  and  $\phi$  can be solved separately. In particular,

$$\hat{\alpha} = \arg \max_{\alpha \in (-\infty, 1)} \log p_{\alpha}(\mathbf{Z}_n \mid \mathbf{T}_{K_n}), \quad (15)$$

where  $p_{\alpha}(\mathbf{Z}_n \mid \mathbf{T}_{K_n})$  is as in (4).

Table 1: Results of Gibbs sampling experiments on synthetic data ( $\alpha^* = 0.75$ ). The top four rows show results from each of four different BNTL models fit to a synthetic graph with 500 edges generated by the coupled  $\mathcal{PYP}$  BNTL model; the bottom four rows show the same BNTL models fit to a synthetic graph with  $\text{Geom}(0.25)$ -distributed interarrivals.

Gen. arrival distn.	$K_n$	Inference model	$ \hat{\alpha} - \alpha^* $	$ \hat{\mathbf{S}} - \mathbf{S}^* $	Pred. log-lik.	Runtime (sec.)	ESS
$\mathcal{PYP}(1.0, 0.75)$	260	$(\tau, \mathcal{PYP}(\theta, \tau))$	<b>0.046 ± 0.002</b>	<b>28.5 ± 0.7</b>	<b>-2637.0 ± 0.1</b>	297.6 ± 0.2	0.80 ± 0.09
$\mathcal{PYP}(1.0, 0.75)$	260	$(\alpha, \mathcal{PYP}(\theta, \tau))$	<b>0.045 ± 0.003</b>	33.4 ± 1.0	-2638.4 ± 0.2	313.6 ± 0.4	0.77 ± 0.07
$\mathcal{PYP}(1.0, 0.75)$	260	$(\alpha, \text{Geom}(\beta))$	0.049 ± 0.004	66.8 ± 1.2	-2660.5 ± 0.7	90.5 ± 0.1	0.78 ± 0.09
$\mathcal{PYP}(1.0, 0.75)$	260	$(\alpha, \text{Pois}_+(\lambda))$	0.054 ± 0.004	68.0 ± 0.7	-2902.5 ± 1.4	112.5 ± 0.1	0.79 ± 0.07
$\text{Geom}(0.25)$	251	$(\tau, \mathcal{PYP}(\theta, \tau))$	0.086 ± 0.002	56.6 ± 1.3	-2386.8 ± 0.1	295.4 ± 0.6	0.83 ± 0.06
$\text{Geom}(0.25)$	251	$(\alpha, \mathcal{PYP}(\theta, \tau))$	0.078 ± 0.003	54.2 ± 2.0	-2387.5 ± 0.5	312.7 ± 0.3	0.66 ± 0.09
$\text{Geom}(0.25)$	251	$(\alpha, \text{Geom}(\beta))$	<b>0.043 ± 0.003</b>	24.8 ± 0.8	<b>-2382.6 ± 0.2</b>	87.2 ± 0.1	0.92 ± 0.04
$\text{Geom}(0.25)$	251	$(\alpha, \text{Pois}_+(\lambda))$	<b>0.041 ± 0.003</b>	<b>21.0 ± 0.5</b>	-2562.2 ± 0.2	109.5 ± 0.1	0.91 ± 0.05

Closed-form MLEs are known for many i.i.d. interarrival distributions. For the  $\text{Geom}(\beta)$  and  $\text{Pois}_+(\lambda)$  distributions used in Section 5,  $\hat{\beta} = \frac{K_n - 1}{n - K_n}$  and  $\hat{\lambda} = \frac{n - K_n}{K_n - 1}$ . MLEs for  $\theta$  and  $\tau$  in  $\mathcal{PYP}$ -induced interarrivals can be found by numerically optimizing the product over arrival times of (11). See Appendix D for details. Maximum a posteriori (MAP) estimates are straightforward to compute by placing priors on the model parameters and including the relevant prior probabilities in (14)-(15).

### 4.3 RELATED WORK

There is relatively little previous work on statistical inference for non-exchangeable models of network data. Bloem-Reddy and Orbanz (2018) develop sequential Monte Carlo methods for non-exchangeable models; those methods are feasible only for networks with hundreds of vertices. See references therein for related ideas based on importance sampling. Where BNTL models overlap with edge exchangeable models, there exist inference algorithms that do not account for arrival times. Namely, if  $\mathbf{Z}_n$  is assumed to be an exchangeable sequence of edge-ends, then the sampling and estimation algorithms for Gibbs-type partitions can be used. For example, Gibbs sampling methods for the  $\mathcal{PYP}$  are derived in Ishwaran and James (2001). Crane and Dempsey (2017) give maximum likelihood estimating equations. However, neither method infers arrival times, and the inference techniques do not extend to the wider class of non-exchangeable BNTL models.

Wan et al. (2017) propose MLEs for the parameters of a class of PA models when the edge sequence is observed. A MLE of the parameter  $\alpha$  in a slightly different PA model was proposed by Gao and van der Vaart (2017) for observed edge sequences. The PA model considered there has random initial degrees, rather than random arrival times, but the initial degrees play a similar role to the arrival times. Those authors find that conditioned on

the initial degrees, the degree sequence at step  $n$ ,  $\mathbf{d}_{K_n}$ , is sufficient for  $\alpha$ , and that the MLE is asymptotically normal. Based on the similarities of the models and the corresponding log-likelihoods, it is plausible that similar properties hold for BNTL models.

## 5 EXPERIMENTS

We apply the inference methods developed in Section 4 to data. The first set of experiments is in the unlabeled network setting, in which the posterior distribution over vertex ordering must be inferred along with the model parameters. In a second set of experiments, we consider graphs with edges labeled in order of appearance, and demonstrate that maximum likelihood and MAP estimation scale to networks with millions of nodes.<sup>2</sup>

### 5.1 BAYESIAN INFERENCE

We first apply the Gibbs sampler from Section 4.1 to synthetic data, which allows us to study the effects of model misspecification on parameter estimation, and to demonstrate the feasibility of inference over the vertex order and the arrival times. We generated two synthetic graphs, each with 1,000 edges: One from a  $\mathcal{PYP}(\theta, \tau)$  sequence (1) in which  $\tau$  is forced to be equal to the BNTL parameter  $\alpha$ , which corresponds to the edge exchangeable Hollywood model of Crane and Dempsey (2017); and one from a BNTL model with i.i.d.  $\text{Geom}(\beta)$ -distributed interarrival times. We set  $\theta = 1.0$ ,  $\beta = 0.25$ , and in both cases,  $\alpha = 0.75$ .

For each of the graphs, we held out the final 500 edges for prediction, and we fit four different BNTL models to the first 500 edges whose order we treated as unknown: One with  $\mathcal{PYP}(\theta, \tau)$ -induced arrivals and  $\alpha = \tau$  (the ‘‘coupled  $\mathcal{PYP}$ ’’ model), which is the same as the generative

<sup>2</sup>Julia code is available at <https://github.com/emilemathieu/NTL.jl>.

Table 2: Scaling performance of the Gibbs sampler.

	100 edges	1,000 edges	10,000 edges
$ \hat{\alpha} - \alpha^* $	0.12 $\pm$ 0.01	0.03 $\pm$ 0.00	0.01 $\pm$ 0.00
$ \hat{\beta} - \beta^* $	0.02 $\pm$ 0.00	0.01 $\pm$ 0.00	0.00 $\pm$ 0.00
$ \hat{\mathbf{S}} - \mathbf{S}^* $	10.3 $\pm$ 0.4	33.9 $\pm$ 0.9	343 $\pm$ 1.6
ESS	0.90 $\pm$ 0.04	0.85 $\pm$ 0.05	0.75 $\pm$ 0.08
Runtime (s)	21 $\pm$ 0.0	213 $\pm$ 0.4	2267 $\pm$ 2

model of the first synthetic dataset; one with  $\mathcal{PYP}(\theta, \tau)$ -induced arrivals and  $\alpha$  allowed to vary separately from  $\tau$  (the “uncoupled  $\mathcal{PYP}$ ” model); and two i.i.d. interarrival models, with  $\text{Geom}(\beta)$ - and  $\text{Pois}_+(\lambda)$ -distributed interarrivals. We ran 125,000 Gibbs sampling iterations, including a burn-in of 25,000, and collected one in every 1,000 iterations for a total of 1,000 samples. To assess performance, we calculated the average absolute error (relative to the true value) of MCMC samples of  $\alpha$ , and of  $\mathbf{S} := \frac{1}{K_n - 1} \sum_{j>1} (\bar{d}_{j-1} - T_j)$ . The latter statistic captures how well the sampler recovers the vertex permutation and the arrival times. We also calculated the predictive log-likelihood of a further 500 edges. Average runtimes<sup>3</sup> and effective sample size (ESS) factors, based on the log of the normalized  $L_1$  distance between the sampled degree sequence and the true degree sequence, are also shown.

Table 1 summarizes the results, averaged over 10 repetitions. The top four rows show the results of fitting four BNTL inference models to the coupled  $\mathcal{PYP}$  dataset. Unsurprisingly, the inference models with arrivals induced by the  $\mathcal{PYP}$  achieve the lowest errors in  $\alpha$  and  $\mathbf{S}$ , and highest predictive log-likelihood. The bottom four rows show the same four inference models fit to the  $\text{Geom}(0.25)$  BNTL dataset; the i.i.d. interarrival models achieve lower errors, and the  $\text{Geom}(\beta)$  inference model attains the highest predictive log-likelihood. Although the  $\text{Pois}_+(\lambda)$  inference model attains low errors in  $\alpha$  and  $\mathbf{S}$ , the low variance of the Poisson distribution compared to the Geometric distribution means that it attains low predictive probability due to the relatively frequent occurrence of large interarrivals.

As discussed in Section 4.1, the most expensive Gibbs update is that of the arrival time sequence. As such, the  $\text{Geom}(\beta)$  interarrival inference model benefits greatly from (13), which implies that computation of  $\Lambda_j^\phi$  is not required. The i.i.d. interarrival models each have conjugate updates for their parameters, whereas the  $\mathcal{PYP}$  interarrival models require slice sampling for  $\phi = (\theta, \tau)$ . These differences are reflected in the runtimes shown in Table 1. Finally, all four inference models exhibit good ESS factors, indicating that the sampler is exploring per-

<sup>3</sup>All Gibbs sampling experiments were run on a quad-core (3.1 GHz) Dell desktop running Linux.

mutation space beyond simply swapping vertices of the same degree.

**Scaling in  $n$ .** In order to study how sampling and computational efficiency scale with the size of the network, we generated a single BNTL network of 10,000 edges with i.i.d.  $\text{Geom}(0.25)$ -distributed arrival times, and performed Gibbs sampling using the subgraphs formed by the first  $n$  edges, with  $n \in \{100, 1,000, 10,000\}$ . Table 2 shows the results of 10 repetitions, each of 150,000 Gibbs iterations; samples were collected once every 1,000 iterations after a burn-in period of 75,000 iterations. Parameter estimation is increasingly accurate for increasing  $n$  without major decrease in ESS, indicating that the sampler is taking advantage of the increased statistical signal in the bigger network. Runtimes increase at a rate linear in  $n$ .

## 5.2 MAXIMUM LIKELIHOOD ESTIMATION ON EDGE SEQUENCES

For observed edge sequences, maximum likelihood estimation scales to networks with millions of vertices and tens of millions of edges. To demonstrate, we compute MLEs on a collection of temporal network datasets available from the Stanford Network Analysis Project (SNAP) (Leskovec and Krevl, 2014).

For each of the datasets listed in Table 3, we fit MLEs of  $\alpha$  and of the parameters of three different interarrival models: coupled  $\mathcal{PYP}(\theta, \alpha)$ ; uncoupled  $\mathcal{PYP}(\theta, \tau)$ ; and  $\text{Geom}(\beta)$ . Table 4 displays the MLEs of the model parameters and the plug-in estimates of the asymptotic power law degree exponent,  $\eta$ . (The asymptotic degree distribution of the uncoupled  $\mathcal{PYP}$  model is unknown.) Note that due to the factorization of the likelihood in (14),  $\hat{\alpha}$  is the same for any model in which  $\alpha$  is not coupled to the arrival distribution. In order to assess model fitness, we fit MLEs for the same BNTL models to the first 80% of the edges in each network and calculated the predictive log-likelihood based on the MLEs of the remaining 20%; this is also shown in Table 4. For context, the arrival time sequence of each dataset is plotted in Figure 1. Unsurprisingly, whether or not the arrival times are approximately linear in  $n$  largely determines which BNTL model fits

Table 3: SNAP temporal network datasets.

Dataset	# of vertices	# of edges
Ask Ubuntu	159,316	964,437
UCI social network	1,899	20,296
EU email	986	332,334
Math Overflow	24,818	506,550
Stack Overflow	2,601,977	63,497,050
Super User	194,085	1,443,339
Wikipedia talk pages	1,140,149	7,833,140



Table 4: MLEs on full datasets, and predictive log-likelihood for final 20% of edges based on MLEs fit to the first 80%, for three different BNTL models. Note that the uncoupled  $\mathcal{PYP}(\theta, \tau)$  and  $\text{Geom}(\beta)$  interarrival models have the same  $\hat{\alpha}$  due to the factorization in (14).

Dataset	Coupled $\mathcal{PYP}(\theta, \alpha)$			$\hat{\alpha}$	Uncoupled $\mathcal{PYP}(\theta, \tau)$		Geom( $\beta$ )		
	$(\hat{\theta}, \hat{\alpha})$	$\hat{\eta}$	Pred. l-l.		$(\hat{\theta}, \hat{\tau})$	Pred. l-l.	$\hat{\beta}$	$\hat{\eta}$	Pred. l-l.
Ask Ubuntu	(18080, 0.25)	1.25	-3.707e6	-2.54	(-0.99, 0.99)	-3.678e6	0.083	2.32	<b>-3.678e6</b>
UCI social network	(320.4, 4.4e-11)	-	-1.600e5	-4.98	(5.50, 0.52)	<b>-1.595e6</b>	0.016	2.10	-1.596e5
EU email	(113.6, 2.5e-14)	-	<b>-8.06e5</b>	-1.86	(113.6, 9.2e-10)	<b>-8.06e5</b>	0.001	2.00	-8.07e5
Math Overflow	(2575, 0.15)	1.15	-1.685e6	-6.62	(-0.97, 0.997)	-1.670e6	0.025	2.19	<b>-1.670e6</b>
Stack Overflow	(297600, 0.11)	1.11	-3.358e8	-8.94	(-1.0, 1.0)	-3.333e8	0.020	2.21	<b>-3.333e8</b>
Super User	(20640, 0.24)	1.24	-5.855e6	-4.19	(-0.996, 1.0)	<b>-5.775e6</b>	0.067	2.37	-5.775e6
Wikipedia talk pages	(14870, 0.54)	1.54	-3.074e7	-0.25	(-1.0, 1.0)	<b>-3.066e7</b>	0.073	2.10	-3.066e7

best. The two densest networks, the EU email and UCI social networks, exhibit arrival times that are sub-linear in  $n$ ; as such, the  $\mathcal{PYP}$  models fit best. Note that the coupled  $\mathcal{PYP}$  model estimates  $\hat{\alpha} \approx 0$ , indicating the lack of a power law tail in the degree distribution. In the rest of the networks, the arrival times appear approximately linear in  $n$ . The  $\text{Geom}(\beta)$  and uncoupled  $\mathcal{PYP}$  model fit best. However, we note that in these cases the MLEs for the uncoupled  $\mathcal{PYP}$  model are at the boundaries of the parameter range ( $\hat{\theta} \approx -1, \hat{\tau} \approx 1$ ). This illustrates that although the uncoupled  $\mathcal{PYP}$  model is more flexible than the coupled version, the underlying arrival time model cannot capture linear arrival time sequences without driving the parameters to the boundaries.

## 6 DISCUSSION

BNTL models are a useful tool to reason about asymptotic properties of a network. For example, the exponent of the asymptotic power law tail is a function of model parameters, which can be estimated from finite-size networks without dealing with the large fluctuations of heavy-tailed degree distributions in finite samples. Furthermore, the ability to capture the full range of power law exponents and sparsity levels within the same model class allows for model fitness comparisons using the same set of techniques, as in Section 5. We have designed a set of inference algorithms for these models; in doing so, we have made a large class of previously intractable models useful for statistical inference.

**Future research directions.** The full Gibbs sampler scales reasonably well to networks with thousands of vertices; in order to scale to larger networks, further work is needed. One possible approach is via Metropolis–Hastings with cheap joint proposals of the arrival times and the permutation, which may be able to take larger steps in sample space. A different direction is variational inference, though permutations pose a significant challenge in that context; recent work (Linderman et al., 2018) is a step in that direction.

## Acknowledgments

BBR, EM, YWT’s research leading to these results received funding from the European Research Council under the European Union’s Seventh Framework Programme (FP7/2007-2013) ERC grant agreement no. 617071. EM, YWT acknowledge Microsoft Research and EPSRC for partially funding EM’s studentship. AF acknowledges funding from EPSRC grant no. EP/N509711/1.

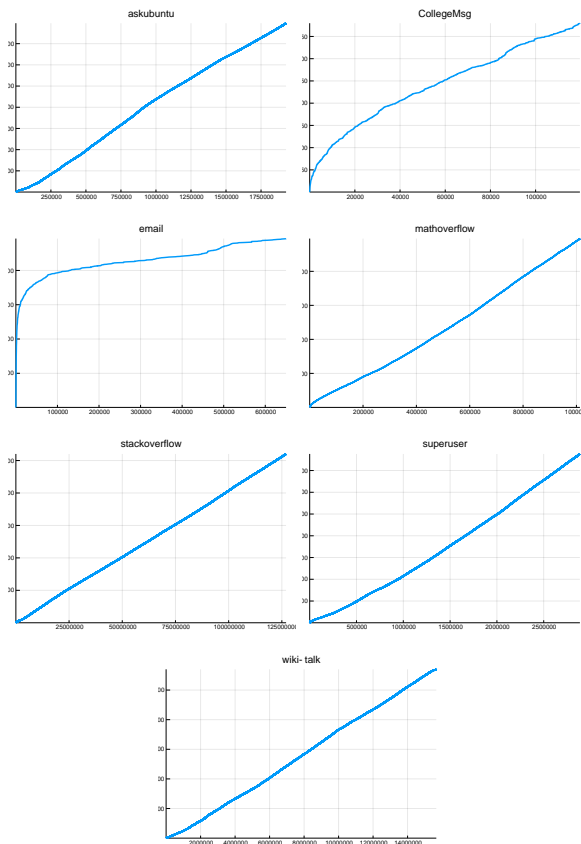


Figure 1: Arrival time sequences for SNAP data. Vertical axis is number of vertices,  $K_n$ ; horizontal axis is  $n$ .

## References

- Aldous, D. J. (1981). “Representations for partially exchangeable arrays of random variables”. In: *Journal of Multivariate Analysis* 11.4, pp. 581–598.
- Barabási, A.-L. and R. Albert (1999). “Emergence of scaling in random networks”. In: *Science* 186.5439, pp. 509–512.
- Berger, N. et al. (2014). “Asymptotic behavior and distributional limits of preferential attachment graphs”. In: *Ann. Probab.* 42.1, pp. 1–40.
- Betancourt, B. et al. (2016). “Flexible Models for Microclustering with Application to Entity Resolution”. In: *NIPS* 29, pp. 1417–1425.
- Bingham, N. H., C. M. Goldie, and J. L. Teugels (1989). *Regular Variation*. Vol. 27. Encyclopedia of Mathematics and its Applications. Cambridge University Press.
- Bloem-Reddy, B. and P. Orbanz (2017). “Preferential Attachment and Vertex Arrival Times”. In: arXiv: 1710.02159 [math.PR].
- (2018). “Random Walk Models of Network Formation and Sequential Monte Carlo Methods for Graphs”. In: *Journal of the Royal Statistical Society: Series B*. To appear.
- Borgs, C. et al. (2016). “Sparse exchangeable graphs and their limits via graphon processes”. In: arXiv: 1601.07134 [math.PR].
- Cai, D., T. Campbell, and T. Broderick (2016). “Edge-exchangeable graphs and sparsity”. In: *NIPS* 29, pp. 4242–4250.
- Caron, F. and E. B. Fox (2017). “Sparse graphs using exchangeable random measures”. In: *Journal of the Royal Statistical Society: Series B* 79.5, pp. 1–44.
- Clauset, A., C. R. Shalizi, and M. E. J. Newman (2009). “Power-Law Distributions in Empirical Data”. In: *SIAM Review* 51.4, pp. 661–703.
- Crane, H. and W. Dempsey (2017). “Edge exchangeable models for interaction networks”. In: *Journal of the American Statistical Association*.
- De Blasi, P. et al. (2015). “Are Gibbs-Type Priors the Most Natural Generalization of the Dirichlet Process?” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.2, pp. 212–229.
- Di Benedetto, G., F. Caron, and Y. W. Teh (2017). “Non-exchangeable random partition models for microclustering”. In: arXiv: 1711.07287 [stat.ME].
- Doksum, K. (1974). “Tailfree and Neutral Random Probabilities and Their Posterior Distributions”. In: *Ann. Probab.* 2.2, pp. 183–201.
- Gao, F. and A. van der Vaart (2017). “On the asymptotic normality of estimating the affine preferential attachment network models with random initial degrees”. In: *Stochastic Processes and their Applications*.
- Gnedin, A. and J. Pitman (2006). “Exchangeable Gibbs partitions and Stirling triangles”. In: *Journal of Mathematical Sciences* 138.3, pp. 5674–5685. ISSN: 1573-8795.
- Griffiths, R. C. and D. Spanò (2007). “Record Indices and Age-Ordered Frequencies in Exchangeable Gibbs Partitions”. In: *Electron. J. Probab.* 12, pp. 1101–1130.
- Hoover, D. N. (1979). *Relations on probability spaces and arrays of random variables*. Tech. rep. Institute of Advanced Study, Princeton.
- Ishwaran, H. and L. F. James (2001). “Gibbs Sampling Methods for Stick-Breaking Priors”. In: *Journal of the American Statistical Association* 96.453, pp. 161–173.
- James, L. F. (2006). “Poisson calculus for spatial neutral to the right processes”. In: *Ann. Statist.* 34.1, pp. 416–440.
- Leskovec, J. and A. Krevl (2014). *SNAP Datasets: Stanford Large Network Dataset Collection*. <http://snap.stanford.edu/data>.
- Linderman, S. W. et al. (2018). “Reparameterizing the Birkhoff Polytope for Variational Permutation Inference”. In: *AISTATS* 21.
- Neal, R. M. (2003). “Slice Sampling”. In: *Ann. Statist.* 31.3, pp. 705–767.
- Newman, M. E. J. (2005). “Power laws, Pareto distributions and Zipf’s law”. In: *Contemporary physics* 46.5, pp. 323–351.
- Orbanz, P. and D. M. Roy (2015). “Bayesian Models of Graphs, Arrays and Other Exchangeable Random Structures”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.2, pp. 437–461.
- Peköz, E. A., A. Röllin, and N. Ross (2017). “Joint degree distributions of preferential attachment random graphs”. In: *Advances in Applied Probability* 49.2, pp. 368–387.
- Pitman, J. (2006). *Combinatorial Stochastic Processes*. Vol. 1875. Ecole d’Été de Probabilités de Saint-Flour XXXII. Springer-Verlag Berlin Heidelberg.
- Simon, H. A. (1955). “On a class of skew distribution functions”. In: *Biometrika* 42.3–4, pp. 425–440.
- Veitch, V. and D. M. Roy (2015). “The Class of Random Graphs Arising from Exchangeable Random Measures”. In: arXiv: 1512.03099 [math.ST].
- Walker, S. G. and P. Muliere (1997). “Beta–Stacy processes and a generalization of the Pólya-urn scheme”. In: *Ann. Statist.* 25.4, pp. 1762–1780.
- Wan, P. et al. (2017). “Fitting the linear preferential attachment model”. In: *Electron. J. Statist.* 11.2, pp. 3738–3780.
- Williamson, S. A. (2016). “Nonparametric Network Models for Link Prediction”. In: *Journal of Machine Learning Research* 17.202, pp. 1–21.

---

# Clustered Fused Graphical Lasso

---

**Yizhi Zhu**  
yzhu44@illinois.edu

**Oluwasanmi Koyejo**  
sanmi@illinois.edu

## Abstract

Estimating the dynamic connectivity structure among a system of entities has garnered much attention in recent years. While usual methods are designed to take advantage of temporal consistency to overcome noise, they conflict with the detectability of anomalies. We propose *Clustered Fused Graphical Lasso* (CFGL), a method using pre-computed clustering information to improve the signal detectability as compared to typical Fused Graphical Lasso methods. We evaluate our method in both simulated and real-world datasets and conclude that, in many cases, CFGL can significantly improve the sensitivity to signals without a significant negative effect on the temporal consistency.

## 1 INTRODUCTION

In recent years, undirected graph models have become a popular topic in machine learning. In an undirected graphical model, vertices represent entities in the system, and edges represent bi-directional effects between entities. The inverse covariance matrix is the preferred estimator for such structures since it indicates partial correlations, i.e., an off-diagonal entry is zero if and only if the entities of the corresponding column and row are conditionally independent given all the other entities. Therefore, two adjacent vertices in an estimated network correspond to a non-zero off-diagonal entry and a direct dependency. One of the most popular methods for estimating the sparse precision matrix is *Graphical Lasso* (Glasso) [Friedman et al., 2008], which assumes the connectivity structure is static. However, this assumption is not satisfied in many fields like functional MRI [Monti et al., 2014], financial markets [Namaki et al., 2011], or

social network analysis [Ahmed and Xing, 2009]. In such cases, data comes from a time series of collections, and the underlying structures are usually assumed to be non-static across time. Consequently, estimating dynamic networks at each time point becomes necessary in order to better understand the complex systems.

Compared to the static case, fewer observations at each time point are available in dynamic estimation. The lack of samples leads to higher level of noise, and thus introduces additional difficulty in estimation. Temporal consistency is a natural assumption with time-varying networks, based on the idea that in most cases, only few changes should occur between consecutive networks. Given this assumption, one may like to place an additional penalty on the difference of neighboring networks. *Fused Graphical Lasso* (Fused Glasso) achieves this using an element-wise  $l_1$  penalty, and it has become the default choice for many studies in the structure estimation field [Monti et al., 2014, Hallac et al., 2017, Danaher et al., 2014, Ahmed and Xing, 2009].

Several Fused Glasso based algorithms have been proposed in the literature on time varying network estimation, and they all have some issues with change detection. *SINGLE* [Monti et al., 2014] avoids accurate change detection by assuming temporal homogeneity (i.e., small and slow changes) on functional MRI data. It uses Fused Glasso on sample covariance estimates, which are smoothed using a Gaussian kernel, so that all abrupt changes are transferred into trends. In another study [Gibberd and Nelson, 2017], the ability to recover change points is specifically targeted. *Grouped Fused Graphical Lasso* (GFGL) uses a group  $l_{2,1}$  smoothing. The drawback is that compared to Fused Glasso results, GFGL has performance loss on static periods on a similar scale as the performance gain at change points. *Time-Varying Graphical Lasso* (TVGL) [Hallac et al., 2017] proposes a general framework, allowing various penalty functions to be applied to fit different situations. But again, it is difficult for a single penalty function to satisfy

both temporal consistency and temporal diversity. More importantly, estimating dynamic functional structures is an unsupervised task. Thus, it is usually impossible to know the correct situation beforehand or to objectively compare methods with different penalties.

In this work, we propose the *Clustered Fused Graphical Lasso* (CFGL) method, obtaining good detectability of changing events and taking care of temporal consistency. Motivated by the property that the thresholded hierarchical clustering is closely related to the connected components of Glasso estimated graphs [Mazumder and Hastie, 2012], we make the key observation that this clustering information indicates evidence of structure-change events and can be a reasonable heuristic. We propose a clustering framework to enhance the evidence of changes. CFGL incorporates the precomputed information into the smooth penalty so that local structures are free to detect change points.

Section 2 formulates the problem and briefly reviews the related background on Graphical Lasso and Fused Graphical Lasso estimation. Section 3 proposes the CFGL method and clarifies the algorithm details. Section 4 compares CFGL to existing methods in three different simulations. Section 5 evaluates CFGL in real cases. Section 6 concludes the paper and talks about future extensions.

## 2 BACKGROUND

### 2.1 PROBLEM DEFINITION

Say there is a sequence of multivariate observations at time points  $t_1 \leq \dots \leq t_T$ . At each time  $t_i$ ,  $n_i \geq 1$  observation vectors form  $X_i = \{x_i^1, \dots, x_i^{n_i}\} \in \mathcal{R}^{n_i \times p}$  with  $x_i \sim \mathcal{N}(0, \Sigma_i)$ . We would like to infer the underlying connectivity structures across time. Based on the aforementioned precision matrix properties, an equivalent problem is to estimate the corresponding precision matrices  $\{\Theta_i\} = \{\Theta_1, \dots, \Theta_T\}$ , one at each time point.

### 2.2 GRAPHICAL LASSO

We start from the static case,  $T = 1$ . An assumption on the number of dependencies (i.e., sparsity) is usually applied, so that only a subset with most dependencies is chosen. This would require placing a prior on the parameters to induce additional zeros on the off-diagonal entries of  $\{\Theta_i\}$ .

$$\arg \min_{\Theta} -l(\Theta) + \lambda_1 \|\Theta\|_1 \quad (1)$$

The equation (1) is *Graphical lasso* [Friedman et al., 2008], which is known to be one of the most effective

methods for this problem [Wang et al., 2012]. The empirical covariance  $S$  is defined to be  $\frac{1}{n} \sum_{i=1}^n x_i x_i^T$ , and then the log likelihood  $l(\Theta)$  is

$$l(\Theta_i) = \log \det(\Theta_i) - \text{trace}(S_i \Theta_i). \quad (2)$$

Minimizing  $-l(\Theta)$  would encourage  $\Theta$  to be close to the inverse covariance  $S^{-1}$  [Yuan and Lin, 2007]. The  $\lambda_1$  in Equation (1) is a non-negative tuning parameter to trade off the sparsity and likelihood.  $\|\Theta\|_1$  is defined to be the element-wise  $l_1$  norm of  $\Theta$ .

### 2.3 FUSED GRAPHICAL LASSO

In the case of  $T > 1$ , to estimate time series of graphs, we seek to take advantage of neighborhood information indicating temporal consistency, i.e., assume structures are similar to their neighbors. Previous methods [Danaher et al., 2014, Monti et al., 2014, Yang et al., 2015, Hallac et al., 2017] implement this assumption by adding an additional penalty to encourage smoothness. In particular, SINGLE [Monti et al., 2014], and  $l_1$ -penalized TVGL [Hallac et al., 2017] define this penalty to be an element-wise  $l_1$  norm of the difference between consecutive estimations:

$$\arg \min_{\{\Theta_i\}} \sum_{i=1}^T -l(\Theta_i) + \lambda_1 \sum_{i=1}^T \|\Theta_i\|_1 + \lambda_2 \sum_{i=2}^T \|\Theta_i - \Theta_{i-1}\|_1 \quad (3)$$

Equation (3) is usually called *Fused Graphical Lasso* (Fused Glasso), due to its relationship to its vector regularization analogue – *Fused Lasso* [Tibshirani et al., 2005], for estimating a sparse time-varying vector.

The variational norm in Fused Lasso is effective for introducing smoothness, but Qian and Jia [2012] prove that Fused lasso can recover exact patterns only if there are no consecutive change points on the timeline and all changes keep switching directions. Many real-world signal patterns obviously do not satisfy these conditions. In addition, even with the aforementioned conditions satisfied, simulated experiments on Fused Glasso show large  $F_1$  score performance drop in the vicinity of the only change point [Gibberd and Nelson, 2017]. In response, we propose the CFGL to improve signal recovery.

## 3 CLUSTERED FUSED GRAPHICAL LASSO

### 3.1 PROPOSED METHOD

Instead of assigning a uniform smoothing weight on all the entries of all precision matrices, we explore methods to distinguish between stable and non-stable connections

in the network and only apply smooth penalties on stable connections.

Mazumder and Hastie [2012] proves the close relation between connected components of the thresholded sample covariance graph and connected components of the Glasso-estimated graph. Tan et al. [2015] extends the conclusion and proves the following theorem, which states the relation between Glasso and clustering results.

Denote  $|S|$  as the matrix of element-wise absolute values of  $S$ , i.e.,  $|S|^{k,l} = |S^{k,l}|$ , where  $S$  is the normalized  $p \times p$  covariance matrix.

**Theorem 3.1.** Let  $C^1, \dots, C^K$  denote the clusters that result from performing *Single Linkage Hierarchical Clustering* (SLC) using similarity matrix  $|S|$  and cutting the resulting dendrogram at a height of  $0 \leq \lambda_1 \leq 1$ . Let  $D^1, \dots, D^R$  denote the connected components of the graphical lasso solution with tuning parameter  $\lambda_1$ . Then,  $K = R$ , and there exists a permutation  $\pi$  such that  $C^k = D^{\pi(k)}$  for  $k = 1, \dots, K$ .

Following Theorem 3.1, it is clear that the regularization parameter  $\lambda_1$  is the threshold to define connected components in the Glasso solution. We note that other clustering algorithms like *average linkage clustering* (ALC) are also used as alternative methods in Tan et al. [2015]. In the following sections, we will use *clusters* and *connected components* interchangeably. We also assume that  $\lambda_1$  is a good threshold resulting in clustering with reasonable accuracy.

Define  $V = \{v_1, \dots, v_p\}$  as the vertex set representing the entities, and  $C_i = \{C_i^1, \dots, C_i^K\}$  as the clustering result on  $|S_i|$  with parameter  $\lambda_1$ , where  $C_i(v_k)$  is the cluster label of  $v_k$ . We construct an undirected graph  $G_i = (V_i, E_i)$ , where  $E_i$  is represented in a similar form to adjacency matrix.  $E_i^{k,l} = 1$  if and only if there is a path between  $v_k$  and  $v_l$ , i.e.,

$$E_i^{k,l} = \mathbb{1}_{\{C_i(v_k)=C_i(v_l)\}}, \quad (4)$$

where  $\mathbb{1}$  is the indicator function.

If  $E_i^{k,l} = 0$ ,  $v_k$  and  $v_l$  have zero partial correlation at time  $t_i$  and thus  $\Theta_i^{k,l} = 0$ . If  $E_i^{k,l} \neq 0$ ,  $v_k$  and  $v_l$  have non-zero partial correlation on intuition and thus  $\Theta_i^{k,l} \neq 0$ .

Given consecutive clustering results  $C_{i-1}$  and  $C_i$ , we want to use the evidence of partial correlation change on between  $\Theta_{i-1}$  and  $\Theta_i$ , and define the weight matrix  $W_i \in \{0, 1\}^{p \times p}$  to help decide whether to apply the  $l_1$  smooth penalty on each entry.

- If  $E_{i-1}^{k,l} = E_i^{k,l}$ ,  $v_k$  and  $v_l$  both have either zero or non-zero partial correlation in time  $t_{i-1}$  and  $t_i$ , and we set  $W_i^{k,l} = 1$ .

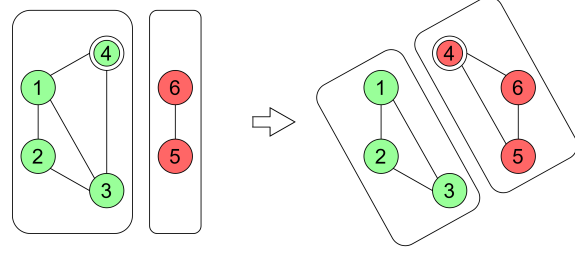


Figure 1: Vertex 4 detaches from the green cluster and merges to the red cluster.

- If  $E_{i-1}^{k,l} \neq E_i^{k,l}$ , we want the partial correlation between  $v_k$  and  $v_l$  to change freely. Thus  $W_i^{k,l} = 0$ .

A simple example is illustrated in Figure 1. At time  $t_{i-1}$  (left),  $E_{i-1}^{4,l} = 1$  for  $l \in \{1, 2, 3\}$  and  $E_{i-1}^{4,l} = 0$  for  $l \in \{5, 6\}$ . At time  $t_i$ ,  $E_i^{4,l} = 0$  for  $l \in \{1, 2, 3\}$  and  $E_i^{4,l} = 1$  for  $l \in \{5, 6\}$ . Thus we set  $W_i^{4,l} = 0$  for  $l \in \{1, 2, 3, 5, 6\}$  to allow vertex 4 to freely change clusters.

Assuming the clustering threshold  $\lambda_1$  is known, we propose the following steps:

1. Perform the chosen clustering method on each empirical covariance matrix  $S_i$  to obtain a sequence of cluster sets  $C_1, C_2, \dots, C_T$ .
2. Let  $\oplus$  denote the **XOR** logical operations, and define the weight matrix set  $\{W_i\}$  as

$$W_i^{k,l} = 1 - \mathbb{1}_{\{E_{i-1}^{k,l} \oplus E_i^{k,l}\}}. \quad (5)$$

3. Apply these predefined weights and solve the following CFGL optimization problem:

$$\arg \min_{\{\Theta_i\}} \sum_{i=1}^T -l(\Theta_i) + \lambda_1 \sum_{i=1}^T \|\Theta_i\|_1 + \lambda_2 \sum_{i=2}^T \|(\Theta_i - \Theta_{i-1}) \circ W_i\|_1, \quad (6)$$

where the  $\circ$  denotes the element-wise Hadamard product.

### 3.2 MORE STABLE CLUSTER CHANGES ACROSS TIME

The previously defined procedure in section 3.1 is sometimes sensitive to the choice of parameter  $\lambda_1$ . If  $\lambda_1$  is small, clustering is sensitive to noise and very large clusters are always formed; if  $\lambda_1$  is large, sparse clustering is achieved, but vertex pairs with true partial correlation are

likely to be overlooked. Even if  $\lambda_1$  is within an acceptable range, the variations of values around the threshold may lead to grouped and detached clusters, introducing redundant switching. Therefore, no matter what value  $\lambda_1$  is picked, the precomputed clustering change information tends to be noisy.

Since SLC is more unstable with noise and usually has undesirable chain structures [Hastie et al., 2009], Tan et al. [2015] use ALC instead of SLC for clustering on a single network. However, when merging individual and small clusters, ALC is similar to SLC and still suffers severely from noise. The frequent switching problem occurs on boundary values as well. We propose a framework to increase the stability of clustering changing across time (Algorithm 1) and make it applicable to most clustering algorithms.

The idea can be easily explained in the simplest case of hierarchical clustering. Consider there are two vertices,  $v_k$  and  $v_l$ , we propose to have two thresholds  $\lambda_1$  and  $\lambda_1^*$ , with  $\lambda_1^*$  smaller than  $\lambda_1$ , i.e.,  $\lambda_1 = \lambda_1^* + \gamma$  and  $\gamma > 0$ .

- $\lambda_1$  is used to judge whether  $v_k$  and  $v_l$  should be grouped at time  $t_i$  if not grouped at time  $t_{i-1}$ .
- $\lambda_1^*$  is used to judge whether they should be grouped again if they are grouped at time  $t_{i-1}$ .

In other words, we define the condition of  $v_k$  and  $v_l$  being clustered together as

$$\text{Cond}_{C_i(v_k)=C_i(v_l)} = \begin{cases} |S_i^{k,l}| \geq \lambda_1, & \text{if } i = 0 \text{ or } E_{i-1}^{k,l} = 0 \\ |S_i^{k,l}| \geq \lambda_1^*, & \text{if } E_{i-1}^{k,l} = 1 \end{cases} \quad (7)$$

To generalize the idea to complex cases (e.g., merging two clusters), Algorithm 1 is proposed. The input  $\{S_i\}$  is the sequence of similarity matrices (normalized empirical covariance),  $\gamma$  can be understood as the gap or stabilizing parameter, and  $f$  is the clustering algorithm (i.e. represented as a function).

---

**Algorithm 1** Stable Clustering Framework across Time

---

**Input:**  $\{S_i\}, \gamma, f$   
**Output:**  $\{C_i\}, \{E_i\}$   
1:  $C_1 = f(|S_1|)$   
2: Construct  $E_1$  on  $C_1$  using Equation (4)  
3: **for**  $i = 2$  to  $T$  **do**  
4:  $\hat{S}_i = |S_{i-1}| + \gamma E_{i-1}$   
5:  $C_i = f(\hat{S}_i)$   
6: Construct  $E_i$  on  $C_i$  using Equation (4)  
7: **end for**

---

The choice of  $\gamma$  depends on the clustering algorithm  $f$ , and also on the level of noise  $e$  we define (to maintain

sparsity). Here we propose a heuristic choice for the hierarchical clustering case. Under the assumption that the previous clustering is accurate, there are two kinds of errors related to  $\gamma$ :

1.  $|S_i|^{k,l} \geq \lambda_1 - \gamma$  with true  $E_{i-1}^{k,l} = 1 \wedge E_i^{k,l} = 0$ .
2.  $|S_i|^{k,l} \leq \lambda_1 - \gamma$  with true  $E_{i-1}^{k,l} = 1 \wedge E_i^{k,l} = 1$ .

Let  $n$  be the sample size of each estimated  $S_i$ , the standard error of correlation coefficient  $r = S_i^{k,l}$  is  $se(r, n) = \sqrt{\frac{1-r^2}{n-2}}$ . If we assume normally distributed error, the sampled correlation is  $\bar{r} \sim \mathcal{N}(r, se(r, n)^2)$ .

Let us assume the exact true correlation coefficient of all the  $k, l$  pairs is  $\lambda_1$  if  $E^{k,l} = 1$ , and  $e$  if  $E^{k,l} = 0$ . Intuitively, we want  $\lambda_1 - \gamma$  to be as far as possible from both  $\lambda_1$  and  $e$  on their standard error normalized distances. Thus, we can heuristically estimate  $\gamma$  as:

$$\gamma = \frac{se(\lambda_1, n)(\lambda_1 - e)}{se(\lambda_1, n) + se(e, n)} \quad (8)$$

### 3.3 PARAMETER TUNING

All that remains is to tune the parameters  $\lambda_1$  and  $\lambda_2$ . Following Hallac et al. [2017] and Monti et al. [2014], we use Akaike Information Criteria (AIC) to tune these hyperparameters. For a given pair  $(\lambda_1, \lambda_2)$ , we define the AIC as:

$$AIC(\lambda_1, \lambda_2) = 2 \sum_{i=1}^T -l(\Theta_i) + 2K. \quad (9)$$

where the estimated degree of freedom  $K$  is slightly different from the definition in Tibshirani et al. [2005], and is given by:

$$K = \sum_{k,l} \sum_{i=2}^T \mathbb{1}_{\{(\Theta_i^{k,l} \neq 0 \oplus \Theta_{i-1}^{k,l} \neq 0) \wedge (\Theta_i^{k,l} \neq 0 \wedge W_i^{k,l} \neq 0)\}} \quad (10)$$

In equation (10), we do not penalize the changes unrelated to 0 since graphical lasso focuses more on the occurrence of edges. Term  $W_i^{k,l}$  is added to avoid penalizing intentionally allowed changes.

Observe that a small  $\lambda_1$  may result in huge clusters and  $W_i^{k,l} \neq 0$  everywhere, which makes CFGL equivalent to Fused Glasso. In addition to a typical grid search with AIC score, CFGL requires  $\lambda_1$  to be in a smaller pre-decided range. Therefore, we need to first tune a series of static graphical lasso (Static Glasso) with AIC to achieve an appropriate range of  $\lambda_1$ . Other methods can be used

to predefine the range, as long as they distinguish CFGL with typical Fused Glasso.

For a fixed sparse penalty  $\lambda_1$ , the clustering threshold can be either set to be  $\lambda_1$  or slightly higher in order to compensate for the use of  $\gamma$ . The CFGL algorithm is described in Algorithm 2.

---

**Algorithm 2** CFGL and Parameter Tuning

---

**Input:**  $\{S_i\}, f$

**Output:**  $\{\Theta_i\}$

- 1: Tune  $\lambda_1^*$  on static graphical lasso using AIC score
  - 2: Obtain a range of  $\lambda_1$  near the best  $\lambda_1^*$  from 1
  - 3: **for**  $\lambda_1$  among choices **do**
  - 4:   **for**  $\lambda_2$  among choices **do**
  - 5:     Compute  $\{E_i\}$  using  $\{S_i\}, f$  and  $\lambda_1$  via Algorithm 1
  - 6:     Compute  $\{W_i\}$  using  $\{E_i\}$  and equation (5)
  - 7:     Obtain  $\{\Theta_i\}_{\lambda_1, \lambda_2}$  that minimizes equation (6)
  - 8:     Compute AIC score of  $\{\Theta_i\}$
  - 9:   **end for**
  - 10: **end for**
  - 11: **return**  $\{\Theta_i\}_{\lambda_1, \lambda_2}$  which minimizes the AIC score
- 

### 3.4 OPTIMIZATION ALGORITHM

We use Alternating Directions Method of Multipliers (ADMM) [Boyd et al., 2011] algorithm to solve the optimization problem. Firstly, define the problem as:

$$\begin{aligned} \underset{\{\Theta_i\}, \{Z_i\}}{\text{minimize}} \quad & \sum_{i=1}^T -l(\Theta_i) + \lambda_1 \sum_{i=1}^T \|Z_i\|_1 \\ & + \lambda_2 \sum_{i=2}^T \|(Z_i - Z_{i-1}) \circ W_i\|_1 \end{aligned} \quad (11)$$

subject to:  $\Theta_i = Z_i$ , for  $i = 1, 2, 3, \dots, T$ .

The augmented Lagrangian corresponding to equation (11) is defined as:

$$\begin{aligned} \mathcal{L}_\rho(\{\Theta_i\}, \{Z_i\}, \{U_i\}) = & \sum_{i=1}^T -l(\Theta_i) \\ & + \lambda_1 \sum_{i=1}^T \|Z_i\|_1 + \lambda_2 \sum_{i=2}^T \|(Z_i - Z_{i-1}) \circ W_i\|_1 \\ & + \frac{\rho}{2} \sum_{i=1}^T (\|\Theta_i - Z_i + U_i\|_2^2 - \|U_i\|_2^2), \end{aligned} \quad (12)$$

where  $\{U_i\}$  are scaled dual variables, and  $\rho$  is a constant penalty parameter in ADMM which is usually set to one. Consequently, we get the update rule for  $U_i, \Theta_i$  for  $i =$

$1, \dots, T$ , and  $\{Z_i\}$  in  $j + 1$ th iteration:

$$\Theta_i^{j+1} = \arg \min_{\Theta_i^{j+1}} \frac{\rho}{2} \|\Theta_i - Z_i^j + U_i^j\|_2^2 - l(\Theta_i) \quad (13)$$

$$\{Z_i^{j+1}\} = \arg \min_{\{Z_i^{j+1}\}} \mathcal{L}_\rho(\{\Theta_i^{j+1}\}, \{Z_i\}, \{U_i^j\}) \quad (14)$$

$$U_i^{j+1} = U_i^j + \Theta_i^{j+1} - Z_i^{j+1} \quad (15)$$

Thus, the update step (13) is same in a typical fused graphical lasso, and the solution is discussed in detail by Monti et al. [2014] and Danaher et al. [2014]. For the  $Z$  update step (14), since all of these are element-wise operations, we can solve each  $\{Z_i^{j+1}\}^{k,l}$  separately.

$$\begin{aligned} & \arg \min_{\{Z_i^{j+1}\}^{k,l}} \frac{\rho}{2} \sum_{i=1}^T \|\{\Theta_i^{j+1} - Z_i + U_i^j\}^{k,l}\|_2^2 \\ & + \lambda_1 \sum_{i=1}^T \|Z_i^{k,l}\|_1 + \lambda_2 \sum_{i=2}^T \|(Z_i^{k,l} - Z_{i-1}^{k,l}) \circ W_i^{k,l}\|_1 \end{aligned}$$

Set  $y_i = (\Theta_i^{j+1} + U_i^j)^{k,l}$  and  $\beta_i = (Z_i)^{k,l}$ . We get the following equation:

$$\begin{aligned} \mathbf{f}_{1,T}^*(\beta) = & \sum_{i=1}^T \frac{\rho}{2} \|y - \beta_i\|_2^2 + \lambda_1 \sum_{i=1}^T \|\beta_i\|_1 \\ & + \sum_{i=2}^T \lambda_{i,i-1} \|\beta_i - \beta_{i-1}\|_1, \end{aligned}$$

where the  $\lambda_{i,i-1}$  is defined to be  $\lambda_2 W_i^{k,l}$ .

Note that the nonzero value of  $\lambda_{i,i-1}$  enforces  $\beta_i$  and  $\beta_{i-1}$  to be close, and a zero value in  $\lambda_{i,i-1}$  splits the above equation into several smaller pieces. If  $W_i^{k,l} \neq 0$  for all  $i$ , the equation is equivalent to a 1-dimensional FLSA as shown below, whose solver has been well discussed by Friedman et al. [2007], Hoefling [2010]:

$$\begin{aligned} \mathbf{f}_{1,T}(\beta) = & \sum_{i=1}^T \frac{\rho}{2} \|y - \beta\|_2^2 + \lambda_1 \sum_{i=1}^T \|\beta_i\|_1 \\ & + \lambda_2 \sum_{i=2}^T \|\beta_i - \beta_{i-1}\|_1. \end{aligned}$$

If  $W_i^{k,l} = 0$  for  $i \in \{m_1, \dots, m_D\}$ ,  $m_0 = 1, m_1 \geq 2, m_D \leq T$  and  $m_{D+1} = T$ , solving the above problem is equivalent to separately solving  $D + 1$  independent 1-dimensional FLSA. In other words,  $\beta$  becomes the concatenation of  $\{\beta^1, \dots, \beta^{D+1}\}$ , and

$$\beta^d = \arg \min_{\beta} f_{m_{d-1}, m_d}(\beta).$$

It is shown in the supplement that  $\{W_i\}$  does not introduce any additional complexity. Also, the update steps (13) and (14) can be easily parallelized, making the optimization algorithm very scalable.

## 4 SIMULATED EXPERIMENTS

To perform simulations, we first construct three groups of signal patterns, outlined in Figure 2, ranging from sudden stimuli to long-term switches. Group 1 has two short-term changes lasting for three time points. Group 2 has three short stimuli happening in only one time point. Signals in Group 3 have equal length and are sequentially distributed across time.

### 4.1 EXPERIMENT SETUP

We generate the simulated data from an Erdős–Rényi random graph  $G = \{V, E\}$  under the controlled sparsity  $|E| = 0.5 |V|$ . To form the precision matrix  $\hat{\Theta}_i$  and sampled observation data, we use the following method, similar to Gibberd and Nelson [2017]

1. Set an empty  $|V| \times |V|$  matrix and insert off-diagonal terms based on edges in  $G$  with values chosen from  $Unif(0.6, 0.9)$ .
2. Equally shift all the diagonal entries by a positive value so that the smallest eigenvalue is 0.1 to ensure positive semi-definiteness.
3. Normalize the matrix to have value 1 on diagonal.
4. Generate observation data  $X_i = \{x_1^1, \dots, x_1^{n_i}\}$  and  $x_i^j \sim \mathcal{N}(0, \hat{\Theta}_i^{-1})$ .

For all the three simulations datasets, we set  $|V| = 25$ . The generated precision matrices  $\{\hat{\Theta}_i\}$  have off-diagonal terms with values around 0.5. There are total 30 time points, each with 25 observations, i.e.,  $T = 30$  and  $n_i = 25$  for all  $i$ .

We compare to four state-of-the-art baseline methods, the static graphical lasso [Friedman et al., 2008], the fused graphical lasso from SINGLE [Monti et al., 2014], the  $l_1$  penalized TVGL [Hallac et al., 2017], and the group fused graphical lasso (GFGL) [Gibberd and Nelson, 2017]. For Static Glasso, the graph at each time  $t_i$  is estimated independently. So we solve a total of  $T$  Glasso (1) problems to get  $T$  separate graphs. The degree of freedom  $K$  in Glasso is defined as the number of non-zero entries [Tibshirani et al., 2005]. Although both SINGLE and  $l_1$ -TVGL have exactly the same fused graphical lasso objective in optimization, they use different optimization solvers which often result in different performance. We put both methods here as baselines and denote them as FGL-SINGLE and FGL-TVGL accordingly. Also, it is worth mentioning that CFGL shares a similar optimization solver as FGL-SINGLE.

We use four different versions of CFGL to compare to three baseline methods, listed in table 1. The  $\Gamma$  repre-

Table 1: Notation of CFGL Related Methods

Notation	Clustering Method	Threshold
CFG <sub>L</sub> -alc	ALC	$\lambda_1$
CFG <sub>L</sub> -slc	SLC	$\lambda_1$
CFG <sub>L</sub> -alc2	ALC	$\lambda_1 + \Gamma/2$
CFG <sub>L</sub> -slc2	SLC	$\lambda_1 + \Gamma/2$

sents a rough estimation by applying equation (8) on AIC tuned Static Glasso  $\lambda_1$ . CFGL related methods are tuned using Algorithm 2. The four baseline methods are tuned by a typical grid search to minimize their correspondingly defined AIC score. Gibberd and Nelson [2017] tentatively propose a BIC score to tune GFGL in unsupervised tasks, but we find AIC generates slightly better results in our simulations. To eliminate the potential performance difference caused by tuning range, baseline methods are also tuned by Algorithm 2, and the result with better  $F_1$ -score is chosen. We repeat the whole process 10 times to reduce randomness, each time generating a new set of Erdős–Rényi graphs and observations. The averaged performance results are shown in Table 2.

### 4.2 PERFORMANCE METRICS

We measure the performance of each method using three metrics:  $F_1$ -score,  $F_1$ -ratio, and edge deviation ratio.

#### 4.2.1 $F_1$ Score

This measures how close the captured structures are to the true graphs. The  $F_1$  Score shown in Table 2 is the averaged  $F_1$  score across all the time points. Thus it provides an overview of performance for both static points and signals.

#### 4.2.2 $F_1$ Ratio

We define  $F_1$  ratio to be the average ratio between  $F_1$  scores at the starting points of changing signals and the overall averaged  $F_1$  score.  $F_1$  ratio indicates the performance of accurate edge detection at change points.

#### 4.2.3 Edge Deviation (ED) Ratio

We define ED Ratio to be the average ratio between edge deviations (number of changing edges) at starting points of changing signals and the overall averaged edge deviations. Unlike  $F_1$  ratio which pays more attention to the correctness, ED ratio focuses more on changing detecting and provides an unsupervised measure.



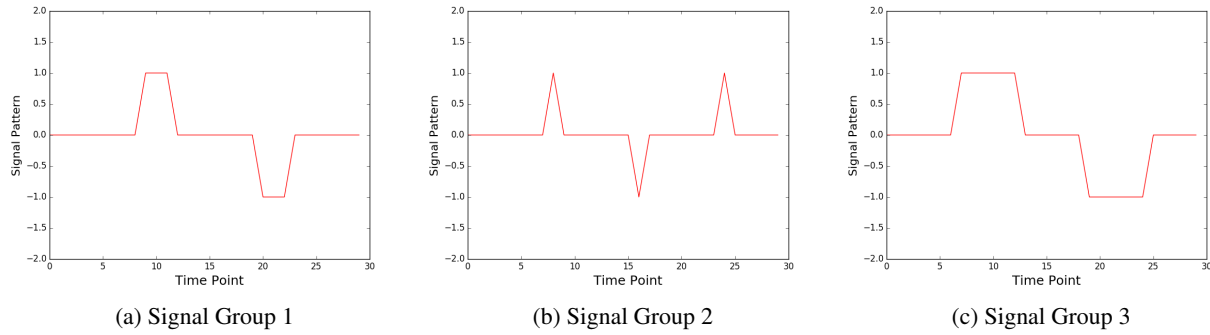


Figure 2: Signal patterns with same numerical values have exactly same graph structure and ground truth precision matrix. Signal patterns with same absolute values but opposite signs have the same graph structure but opposite signs in the off diagonal values in precision matrix.

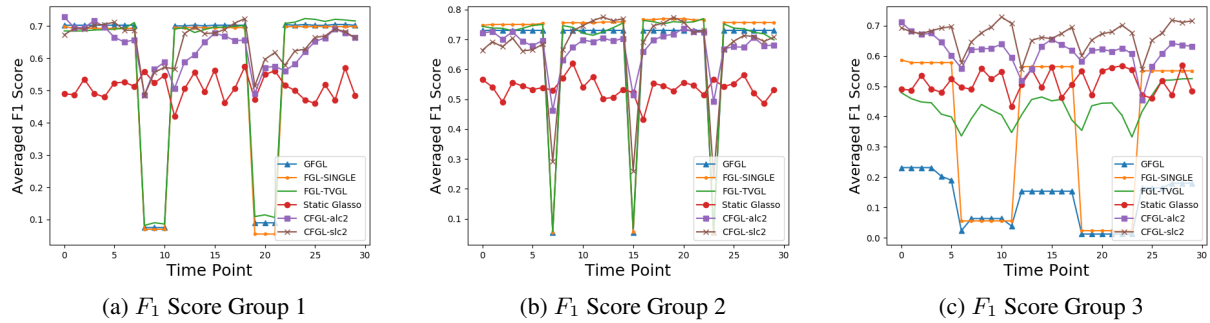


Figure 3: Each point on the curve represents the averaged  $F_1$  score at that time point among 10 repetitions.

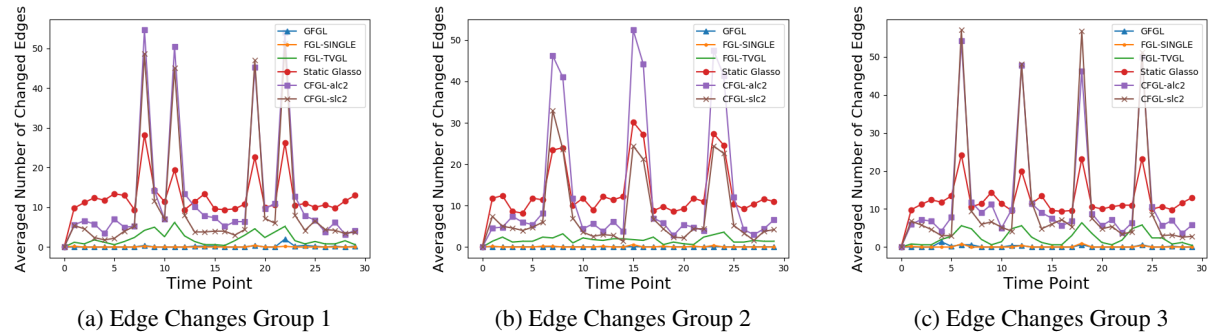


Figure 4: Each point on the curve represents the averaged number of changing edges between consecutive time points among 10 repetitions.

Table 2: Performance Comparison

Dataset	Signal Group1			Signal Group2			Signal Group3		
	$F_1$ Score	$F_1$ Ratio	ED Ratio	$F_1$ Score	$F_1$ Ratio	ED Ratio	$F_1$ Score	$F_1$ Ratio	ED Ratio
GFGL	0.580	0.143	2.6	0.662	0.081	0	0.129	0.147	3.6
FGL-SINGLE	0.570	0.110	<b>9</b>	<b>0.687</b>	0.076	<b>5.45</b>	0.400	0.112	<b>9.6</b>
FGL-TVGL	0.593	0.164	2.15	0.672	0.085	1.37	0.438	0.797	2.53
Static Glasso	0.526	<b>1.01</b>	2.04	0.545	<b>1.00</b>	2.02	0.523	<b>1.02</b>	1.93
<b>CFGL-alc</b>	0.631	0.848	3.43	0.630	0.800	2.81	0.618	0.934	3.40
<b>CFGL-slc</b>	0.555	0.115	2.14	<b>0.687</b>	0.078	2.8	0.626	0.932	5.18
<b>CFGL-alc2</b>	0.633	0.775	3.87	0.676	0.724	3.61	0.618	0.927	3.94
<b>CFGL-slc2</b>	<b>0.653</b>	0.768	4.55	0.670	0.392	3.35	<b>0.668</b>	0.882	4.90

### 4.3 SIMULATION RESULTS

CFGL-alc2 and CFGL-alc have stably good performance in all three metrics. The comparison between CFGL-slc2 and CFGL-slc indicates that SLC based clustering is more sensitive to the choice of thresholds.

Although Static Glasso always achieves good values on  $F_1$  ratio and ED ratio, its low  $F_1$  score implies not using neighborhood information. The performance of FGL-SINGLE shows that it tends to choose parameters which strongly highlight temporal consistency, resulting in overlooking signals. FGL-TVGL performs slightly better on signal detection, but both the metrics in Table 2 and the curves in Figures 3 and 4 show that it fails to take care of temporal consistency and diversity at the same time. For GFGL, although our simulation perfectly satisfies their assumption that change time points tend to group together and the other time points remain static, GFGL estimated structures do not reflect the changing periods properly. The static periods are also highly disturbed by long changing periods in signal group 3. This may be due to the difficulty finding good parameters for GFGL in unsupervised scenarios, which was mentioned by Gibberd and Nelson [2017] as well.

## 5 CASE STUDIES

In this section, we apply CFGL to more complicated real-world datasets to show how the CFGL method can be used to provide insights into real-world multivariate time series datasets.

### 5.1 EEG EYE STATE

Electroencephalography (EEG) is a medical imaging technique that reads scalp electrical activities generated by brain structures, and EEG measurements are commonly used in medical and research areas to infer brain

activities [Teplan et al., 2002]. The dataset we use is the EEG eye state dataset [Roesler, 2013] from the UC Irvine repository. All data is from one continuous EEG measurement with the Emotiv EEG Neuroheadset. The eye state was detected via a camera during the measurement and added later manually after analyzing the video frames.

Table 3: Cross Validated Results on Estimated Structures

Notation	Cross Validated Accuracy
Static Glasso	0.6625
FGL-TVGL	0.800
FGL-SINGLE	0.6875
CFGL-alc	0.8375
CFGL-alc2	<b>0.8875</b>

In this experiment, we pick the first 2000 observations, and we group every 25 observations to form a time point, which is roughly 0.2 seconds. All the methods use the aforementioned tuning procedure. Considering there is no ground truth graph structure in EEG data, we treat the video captured eye state as a ground truth signal. We use the estimated structures as transformed features (i.e., edges present or not) and train a linear SVM model to predict the corresponding eye state. We observe that the prediction performance is related to the feature sparsity, and thus we tune the parameters so that all methods have similar sparsity (about 25% edges present). The result of 10 fold cross-validation is shown in Table 3. In addition, we provide an aligned comparison between the eye state and edge changes. Considering the uncontrollable brain activities and the latency between brains and eye states, we smooth the edge-change signals by the Gaussian kernel. As shown in Figure 5, CFGL performs very well for capturing eye state switching events.

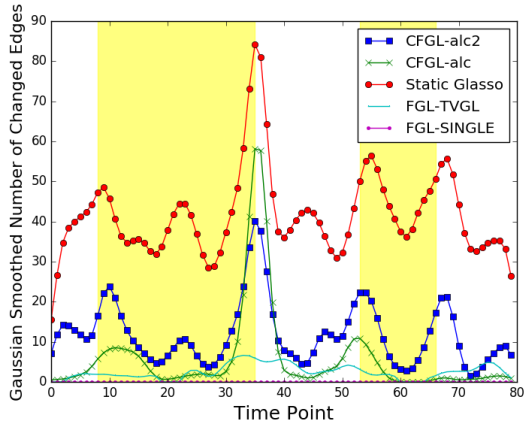


Figure 5: Edge changes are smoothed by Gaussian kernel. The yellow regions represent the period when subject closed his/her eyes.

## 5.2 STOCK MARKET

In this experiment, we apply CFGL to the financial data in the stock market to explore the economic structures of stocks. These structures can be used to provide high-level understanding of company relations. In particular, the stock prices are natural multivariate time-series datasets, and they are also good indicators of company conditions. We assume each company’s stock price is dependent on companies in the same or related fields, and is more likely to be independent to companies in unrelated fields. We pick 20 big companies, roughly 2 from each category of OS, Internet Service, PC, Auto, Restaurant, Finance, Energy, and Sales. Then we infer their structure changes during the global financial crisis of 2008<sup>1</sup>. We pick the dates starting from June 1st 2006 to August 4th 2009, total 800 days in the stock market, and use 20 days to form a time point so that each roughly represents a month.

TED spread is defined as the difference between the interest rates on interbank loans and on short-term U.S. government debt. It is usually treated as the indicator of perceived credit risk in the general economy [Boudt et al., 2017]. Thus we use the TED spread’s changes as the reference of financial events. Figure 6 shows the comparison between the structural changes in estimated stock networks, and the TED spread changes between consecutive periods. It can be observed that the two curves follow similar patterns. There is a shift between the largest change of TED spread and stock market structure. We further investigated and found that the stock market started to drop in early September (around

<sup>1</sup>Data freely available online from <https://quantquote.com/historical-stock-data>.

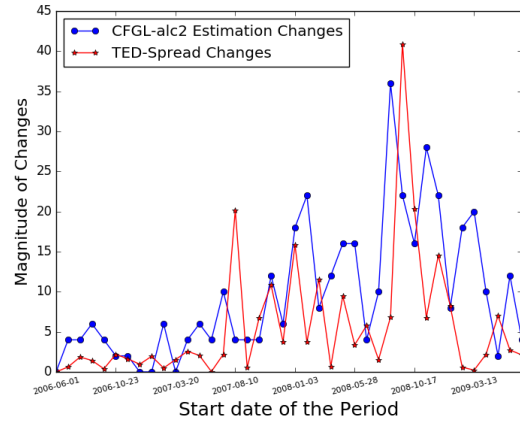


Figure 6: Edge Changes and TED Spread Changes

September 8th), which is exactly at the time of the blue peak but one time point earlier than the red peak. This result may indicate that TED spread has some latency for detecting stock market changes.

## 6 CONCLUSIONS AND FUTURE WORK

We propose *Clustered Fused Graphical Lasso* (CFGL) to improve the signal sensitivity of Fused Graphical Lasso (FGL). CFGL applies clustering based heuristic information on the smooth penalty so that temporal consistency and temporal diversity are simultaneously considered. Our experimental results show that the clustering information often makes CFGL more sensible for capturing signal changes, and CFGL outperforms FGL methods on datasets with time-varying underlying structures. The incorporated clustering information is independent of the smooth penalties. Therefore, there are many possible extensions either applying this information to other penalties, or setting up better methods for clustering.

### Acknowledgements

We would like to thank to Microsoft Azure for computing resources.

### References

- Amr Ahmed and Eric P Xing. Recovering time-varying networks of dependencies in social and biological studies. *Proceedings of the National Academy of Sciences*, 106(29):11878–11883, 2009.
- Kris Boudt, Ellen CS Paulus, and Dale WR Rosenthal. Funding liquidity, market liquidity and TED spread:

- A two-regime model. *Journal of Empirical Finance*, 43:143–158, 2017.
- Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.
- Patrick Danaher, Pei Wang, and Daniela M Witten. The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(2): 373–397, 2014.
- Jerome Friedman, Trevor Hastie, Holger Höfling, Robert Tibshirani, et al. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 1(2):302–332, 2007.
- Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.
- Alexander J Gibberd and James DB Nelson. Regularized estimation of piecewise constant Gaussian graphical models: The group-fused graphical lasso. *Journal of Computational and Graphical Statistics*, 26(3): 623–634, 2017.
- David Hallac, Youngsuk Park, Stephen Boyd, and Jure Leskovec. Network inference via the time-varying graphical lasso. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 205–213. ACM, 2017.
- T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning; Data Mining, Inference and Prediction*. Springer Verlag, New York, 2009.
- Holger Hoefling. A path algorithm for the fused lasso signal approximator. *Journal of Computational and Graphical Statistics*, 19(4):984–1006, 2010.
- Rahul Mazumder and Trevor Hastie. Exact covariance thresholding into connected components for large-scale graphical lasso. *Journal of Machine Learning Research*, 13(Mar):781–794, 2012.
- Ricardo Pio Monti, Peter Hellyer, David Sharp, Robert Leech, Christoforos Anagnostopoulos, and Giovanni Montana. Estimating time-varying brain connectivity networks from functional MRI time series. *NeuroImage*, 103:427–443, 2014.
- A Namaki, AH Shirazi, R Raei, and GR Jafari. Network analysis of a financial market based on genuine correlation and threshold method. *Physica A: Statistical Mechanics and its Applications*, 390(21-22):3835–3841, 2011.
- Junyang Qian and Jinzhu Jia. On pattern recovery of the fused lasso. *arXiv preprint arXiv:1211.5194*, 2012.
- Oliver Roesler. UCI machine learning repository, 2013. URL <https://archive.ics.uci.edu/ml/datasets/EEG+Eye+State>.
- Kean Ming Tan, Daniela Witten, and Ali Shojaie. The cluster graphical lasso for improved estimation of Gaussian graphical models. *Computational statistics & data analysis*, 85:23–36, 2015.
- Michal Teplan et al. Fundamentals of EEG measurement. *Measurement science review*, 2(2):1–11, 2002.
- Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1):91–108, 2005.
- Hao Wang et al. Bayesian graphical lasso models and efficient posterior computation. *Bayesian Analysis*, 7(4):867–886, 2012.
- Sen Yang, Zhaosong Lu, Xiaotong Shen, Peter Wonka, and Jieping Ye. Fused multiple graphical lasso. *SIAM Journal on Optimization*, 25(2):916–943, 2015.
- Ming Yuan and Yi Lin. Model selection and estimation in the Gaussian graphical model. *Biometrika*, 94(1): 19–35, 2007.

---

# Unsupervised Learning of Latent Physical Properties Using Perception-Prediction Networks

---

David Zheng<sup>1</sup>, Vinson Luo<sup>2</sup>, Jiajun Wu<sup>1</sup>, and Joshua B. Tenenbaum<sup>1</sup>

<sup>1</sup>Computer Science and Artificial Intelligence Laboratory, MIT

<sup>2</sup>Department of Computer Science, Stanford University

## Abstract

We propose a framework for the completely unsupervised learning of latent object properties from their interactions: the *perception-prediction network* (PPN). Consisting of a perception module that extracts representations of latent object properties and a prediction module that uses those extracted properties to simulate system dynamics, the PPN can be trained in an end-to-end fashion purely from samples of object dynamics. The representations of latent object properties learned by PPNs not only are sufficient to accurately simulate the dynamics of systems comprised of previously unseen objects, but also can be translated directly into human-interpretable properties (*e.g.* mass, coefficient of restitution) in an entirely unsupervised manner. Crucially, PPNs also generalize to novel scenarios: their gradient-based training can be applied to many dynamical systems and their graph-based structure functions over systems comprised of different numbers of objects. Our results demonstrate the efficacy of graph-based neural architectures in object-centric inference and prediction tasks, and our model has the potential to discover relevant object properties in systems that are not yet well understood.

## 1 INTRODUCTION

The physical properties of objects, combined with the laws of physics, govern the way in which objects move and interact in our world. Assigning properties to objects we observe helps us summarize our understanding of those objects and make better predictions of their future behavior. Often, the discovery of such properties can be performed with little supervision. For instance, by watching an archer shoot several arrows, we may conclude

that properties such as the tension of the bowstring, the strength and direction of the wind, and the mass and drag coefficient of the arrow affect the arrow's ultimate trajectory. Even when given observations from entirely novel microworlds, humans are still able to learn the relevant physical properties that characterize a system [1].

Our work utilizes recent advances in neural relation networks in order to learn latent physical properties of a system in an unsupervised manner. In particular, the neural relation architectures [2, 3] have proven capable of accurately simulating complex physical interactions involving objects with known physical properties. Relation networks have several characteristics that make them particularly suitable for our task: they are fully differentiable, allowing them to be applied to a variety of different situations without the need for any architectural change; they have a modular graph-based structure that generalizes over differing numbers of objects; and their basic architecture can be easily applied to both dynamics prediction and the learning of latent properties.

We use relation networks to construct the perception-prediction network (PPN), a novel system that uses a representation learning [4] paradigm to extract an encoding of the properties of a physical system purely through observation. Unlike previous neural relation architectures, which only use relation networks to predict object states with known property values, we use relation networks to create both a *perception network*, which derives property values from observations, and a *prediction network*, which predicts object positions given property values. The PPN is able to derive unsupervised representations of the latent properties relevant to physical simulations purely by observing the dynamics of systems comprised of objects with different property values. These learned representations can be translated directly into human-interpretable properties such as mass and coefficient of restitution.

One crucial aspect of our system is generalization, which humans excel at when inferring latent properties of novel

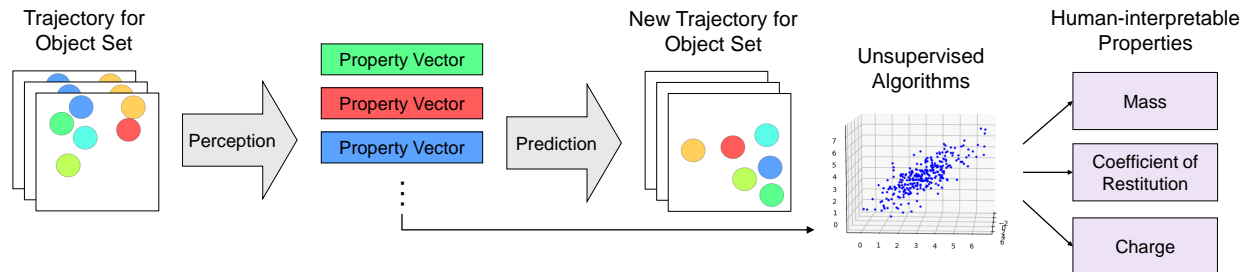


Figure 1: **Model overview.** The unsupervised object property discovery paradigm that the PPN follows extracts property vectors from samples of object dynamics to accurately predict new trajectories of those same objects. Applying unsupervised learning methods to the learned vectors allows for the extraction of human-interpretable object properties.

systems. Our proposed system is robust under several forms of generalization, and we present experiments demonstrating the ability of our unsupervised approach to discern interpretable properties even when faced with different numbers of objects during training and testing as well as property values in previously unseen ranges.

We evaluate the PPN for two major functionalities: the accuracy of dynamics prediction for unseen objects and the interpretability of properties learned by the model. We show that our model is capable of accurately simulating the dynamics of complex multi-interaction systems with unknown property values after only a short observational period to infer those property values. Furthermore, we demonstrate that the representations learned by our model can be easily translated into relevant human-interpretable properties using entirely unsupervised methods. Additionally, we use several experiments to show that both the accuracy of dynamics prediction and interpretability of properties generalize well to new scenarios with different numbers and configurations of objects. Ultimately, the PPN serves as a powerful and general framework for discovering underlying properties of a physical system and simulating its dynamics.

## 2 RELATED WORK

Previous methods of modeling intuitive physics have largely fallen under two broad categories: top-down approaches, which infer physical parameters for an existing symbolic physics engine [1, 5, 6, 7, 8, 9], and bottom-up approaches, which directly predict physical quantities or future motion given observations [10, 11, 12, 13, 14, 15, 16]. While top-down approaches are able to generalize well to any situation supported by their underlying physics engines (*e.g.* different numbers of objects, previously unseen property values, etc.), they are difficult to adapt to situations not supported by their underlying description languages, requiring manual modifications to support new types of interactions. On the other hand, bottom-up approaches are often capable of learning the dynamics of formerly unseen situations without any fur-

ther modification, though they often lack the ability to generalize in the same manner as top-down approaches.

Recently, a hybrid approach has used neural relation networks, a specific instance of the more general class of graph-based neural networks [17, 18], to attain the generalization benefits of top-down approaches without requiring an underlying physics engine. Relation networks rely on the use of a commutative and associative operation (usually vector addition) to combine pairwise interactions between object state vectors in order to predict future object states [19]. These networks have demonstrated success in simulating multiple object dynamics under interactions including Coulomb charge, object collision (with and without perfect elasticity), and spring tension [2, 3, 20, 21]. Much like a top-down approach, relation networks are able to generalize their predictions of object position and velocity to different numbers of objects (training on 6 objects and testing on 9, for instance) without any modification to the network weights; furthermore, they are fully differentiable architectures that can be trained via gradient descent on a variety of interactions. Our paper leverages the interaction network in a novel way, demonstrating for the first time its efficacy as a perception module and as a building block for unsupervised representation learning.

Additional research has looked at the supervised and unsupervised learning of latent object properties, attempting to mirror the inference of object properties that humans are able to perform in physical environments [1]. Wu *et al.* [9] leverages a deep model alongside set physical laws to estimate properties such as mass, volume, and material from raw video input. Fraccaro *et al.* [22] uses a variational autoencoder to derive the latent state of a single bouncing ball domain, which they then simulate using Kalman filtering. Chang *et al.* [3] demonstrate that their relation network based physics simulator is also capable of performing maximum-likelihood inference over a discrete set of possible property values by comparing simulation output for each possibility to reality. Our paper goes one step further by showing that physical proper-

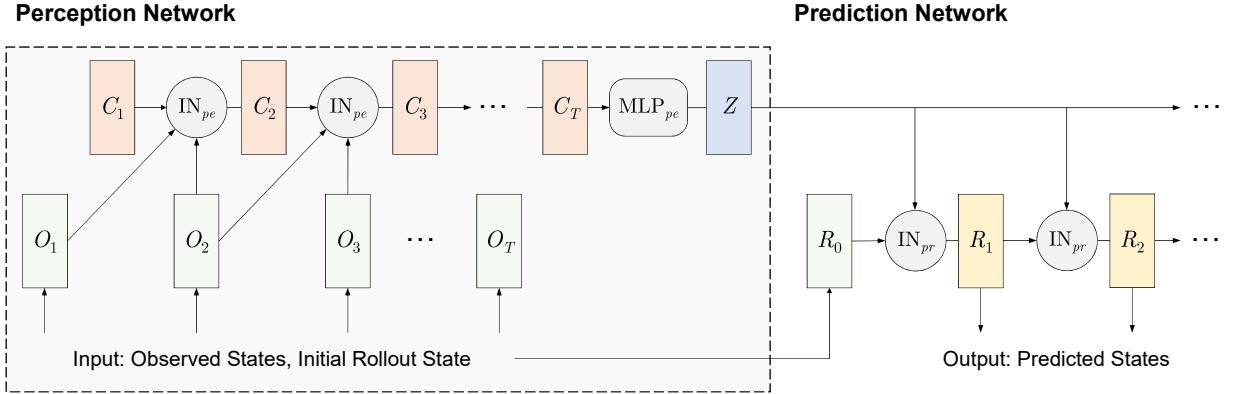


Figure 2: **Model architecture.** The PPN takes as input a sequence of observed states  $O_1, \dots, O_T$  as well an initial state  $R_0$  to begin a new rollout. Code vectors  $C_1, \dots, C_T$  are derived from the observed states using interaction networks and a final property vector  $Z$  is produced by the perception network. The property vector is then utilized by the prediction network to recursively predict future object states  $R_1, R_2, \dots$  for a new rollout given initial state  $R_0$ . We train the PPN to minimize the L2 distance between the predicted rollout states and the ground truth states for those timesteps.

ties can be learned from no more than raw motion data of multiple objects. Recently, Kipf *et al.* [23] has also utilized relation networks to infer the identity of categorical interactions between objects; in contrast, our paper is concerned with the learning of object properties.

### 3 MODEL

#### 3.1 PERCEPTION-PREDICTION NETWORK

The PPN observes the physical dynamics of objects with unknown latent properties (*e.g.* mass, coefficient of restitution) and learns to generate meaningful representations of these object properties that can be used for later simulations. An overview of the full network is shown in Figure 1. The PPN consists of the following two components:

- The **perception network** takes as input a sequence of frames on the movements of objects over a short observation window. It outputs a *property vector* for each object in the scene that encodes relevant latent physical properties for that object. Each input frame is a set of *state vectors*, consisting of each object’s position and instantaneous velocity. During training, no direct supervision target is given for the property vectors.
- The **prediction network** uses the property vectors generated by the perception network to simulate the objects from a different starting configuration. The network takes as input the property vectors generated by the perception network and new initial state vectors for all objects. Its output is a rollout of the objects’ future states from their new starting state. The training target for the prediction network is the ground truth states of the rollout sequence.

We implement both the perception and prediction networks using interaction networks [2], a specific type of neural relation network that is fully differentiable and generalizes to arbitrary numbers of objects. This enables us to train both networks end-to-end using gradient descent with just the supervision signal of the prediction network’s rollout target, as the property vectors output by the perception network feed directly into the prediction network.

#### 3.2 INTERACTION NETWORK

An interaction network (IN) is a relation network that serves as the building block for both the perception and prediction networks. At a high level, interaction networks use multilayer perceptrons (MLPs) to implement two modular functions, the relational model  $f_{\text{rel}}$  and the object model  $f_{\text{obj}}$ , which are used to transform a set of object-specific input features  $\{x^{(1)}, \dots, x^{(N)}\}$  into a set of object-specific output features  $\{y^{(1)}, \dots, y^{(N)}\}$ , where  $N$  is the number of objects in a system. Given input features for two objects  $i$  and  $j$ ,  $f_{\text{rel}}$  calculates the “effect” vector of object  $j$  on object  $i$  as  $e^{(i,j)} = f_{\text{rel}}(x^{(i)}, x^{(j)})$ . The net effect on object  $i$ ,  $e^{(i)}$ , is the vector sum of all pairwise effects  $\sum_{j \neq i} e^{(i,j)}$  on object  $i$ . Finally, the output for object  $i$  is given by  $y^{(i)} = f_{\text{obj}}(x^{(i)}, e^{(i)})$ . Importantly,  $f_{\text{obj}}$  and  $f_{\text{rel}}$  are shared functions that are applied over all objects and object-object interactions, allowing the network to generalize across variable numbers of objects.

Interaction networks are capable of learning state-to-state transition functions for systems with complex physical dynamics. More generally, however, interaction networks can be used to model functions where input and output features are specific to particular objects and the relationship between input and output is the same for each object.

While our prediction network uses an interaction network to simulate state transitions, our perception network uses an interaction network to make incremental updates on the values of object latent properties from observed evidence.

### 3.3 PERCEPTION NETWORK

The perception network produces object-specific property vectors,  $Z$ , from a sequence of observed states  $O$ . As shown in Figure 2, our perception network is a recurrent neural network that uses an interaction network as its core recurrent unit. The perception network begins with object-specific code vectors,  $C_1$ , initialized to zero vectors, with some fixed size  $L_C$  for each object. At each step  $t$ , the IN takes in the previous code vectors,  $C_{t-1}$ , as well as the last two observed states,  $O_{t-1}$  and  $O_t$ , to produce updated code vectors,  $C_t$ , also of size  $L_C$ . After processing all  $T_O$  observation frames, the perception network feeds the final code vectors  $C_{T_O}$  into a single code-to-property MLP that converts each object’s code vector into an “uncentered” property vector of size  $L_Z$  per object. We denote the final collection of uncentered property vectors as  $Z_u$ .

In many physical systems, it may be impossible or undesirable to measure the latent properties of objects on an absolute scale. For example, in a system where two balls collide elastically, a collision can only inform us on the mass of each object relative to the other object, not their absolute mass values. In order to allow for the inference of absolute property values, we let the first object of every system serve as a *reference object* and take on the same property values in each system. In doing so, we can infer the absolute property values of all other objects by observing their value relative to the reference object. To enforce inference relative to the reference object, we “center” the property vectors by subtracting the reference object’s uncentered property vector from each object’s uncentered property vector, producing the final property vectors  $Z$ . Note that this ensures that the reference object’s property vector is always a zero vector, agreeing with the fact that its properties are known to be constant. We can summarize the perception network with the following formulas:

$$C_1 = \mathbf{0} \quad (1)$$

$$C_t = \text{IN}_{pe}(C_{t-1} \| O_{t-1} \| O_t), \text{ for } t = 2, \dots, T_O \quad (2)$$

$$Z_u^{(i)} = \text{MLP}_{pe} \left( C_{T_O}^{(i)} \right), \text{ for } i = 1, \dots, N \quad (3)$$

$$Z^{(i)} = Z_u^{(i)} - Z_u^{(1)}, \text{ for } i = 1, \dots, N \quad (4)$$

where  $\|$  is the object-wise concatenation operator,  $\text{IN}_{pe}$  is the perception interaction network,  $\text{MLP}_{pe}$  is the code-to-property MLP, and  $Z_u^{(1)}$  is the reference object’s uncentered property vector.

### 3.4 PREDICTION NETWORK

The prediction network performs state-to-state rollouts of the system from a new initial state,  $R_0$ , using the property vectors produced by the perception network. Like the perception network, the prediction network is a recurrent neural network with an Interaction Network core. At step  $t$ , the IN takes in the previous state vectors,  $R_{t-1}$ , and the property vectors,  $Z$ , and outputs a prediction of the next state vectors,  $R_t$ . In other words,

$$R_t = \text{IN}_{pr}(R_{t-1} \| Z), \text{ for } t = 1, \dots, T_R \quad (5)$$

where  $\text{IN}_{pr}$  is the prediction interaction network and  $T_R$  is the number of rollout frames.

The prediction loss for the model is the total MSE between the predicted and true values of  $\{R_t\}_{t=1 \dots T_R}$ .

## 4 EXPERIMENTS

### 4.1 PHYSICAL SYSTEMS

For our experiments, we focus on 2-D domains where both the latent property inference task and the subsequent dynamics prediction task are challenging. In all systems, the first object serves as the reference object and has fixed properties. All other objects’ properties can be inferred relative to the reference object’s properties. We evaluate the PPN on the following domains (see Fig. 5):

- **Springs** Balls of equal mass have a fictitious property called “spring charge” and interact as if all pairs of objects were connected by springs governed by Hooke’s law\*. The reference object has a spring charge of 1, while all other objects have spring charges selected independently at random from the log-uniform† distribution over  $[0.25, 4]$ . The spring constant of the spring connecting any given pair of objects is the product of the spring charges of the two objects, and the equilibrium distance for all springs is a fixed constant.
- **Perfectly Elastic Bouncing Balls** Balls of fixed radius bounce off each other elastically in a closed box. The reference object has a mass of 1. Each other ball has a mass selected independently at random from the log-uniform distribution over  $[0.25, 4]$ . The four walls surrounding the balls have infinite mass and **do not move**.

\*Two objects connected by a spring governed by Hooke’s law are subject to a force  $F = -k(x - x_0)$ , where  $k$  is the spring constant of the spring,  $x$  is the distance between the two objects, and  $x_0$  is the spring’s equilibrium distance. The force is directed along the line connecting the two objects but varies in sign: it is attractive if  $x > x_0$  and repulsive if  $x < x_0$ .

†We use the phrase log-uniform distribution over  $[A, B]$  to indicate the distribution of  $\exp(x)$ , where  $x$  is drawn uniformly at random over the interval  $[\log A, \log B]$ .



- **Inelastic Bouncing Balls** Building off the previous domain, we introduce additional complexity by adding coefficient of restitution (COR) as another varying latent property of each object. The COR of a collision is the ratio of the final to initial relative velocity between the two colliding objects along the axis perpendicular to the contact plane. In a perfectly elastic domain, for example, all collisions would have a COR of 1. In our new domain, each object has a random COR selected uniformly from  $[0.5, 1]$ . The reference object has a COR of 0.75. The COR used to compute the dynamics in a collision between two balls is defined as the maximum of the two colliding objects’ CORs. When a ball collides with a wall, the ball’s COR is used for the collision.

For each domain, we train the PPN on a 6-object dataset with  $10^6$  samples and validate on a 6-object dataset with  $10^5$  samples. Each sample consists of 50 observation frames used as input into the perception network and 24 rollout frames used as targets by the prediction network. We evaluated our model on 3-object, 6-object, and 9-object test sets, each with  $10^5$  samples.

In addition, we also wish to demonstrate the PPN’s ability to generalize to new objects whose latent properties are outside of the range of values seen during training. For this experiment, we test our model on a new 2-object perfectly elastic balls dataset with  $10^5$  samples. The mass of the first ball remains fixed at 1, while the mass of the second ball is selected from 11 values ranging from  $32^{-1}$  to 32, spaced evenly on a log scale. We perform a similar experiment on the springs domain, using the same 11 values as the spring charge of the second object.

We use matter-js<sup>‡</sup>, a general-purpose rigid-body physics engine, to generate ground truth data. In all simulations, balls are contained in a  $512 \text{ px} \times 512 \text{ px}$  closed box. Each ball has a 50 px radius and randomly initialized positions such that no ball overlaps. In the springs domain, initial x- and y-velocity components are selected uniformly at random from the range  $[-15, 15]$  px/sec, the equilibrium displacement for each spring is 150, and the mass of all balls is  $10^4$ . In the perfectly elastic balls domain, initial velocity components are selected from the range  $[-9, 9]$  px/sec. In the inelastic balls domain, they are selected from the range  $[-13, 13]$  px/sec. Each dataset’s frames are sampled at 120 fps.

In the creation of our bouncing ball datasets, we use rejection sampling to filter out simulations in which some object latent properties cannot be inferred from the observation frames. In both bouncing ball domains, we must

be able to infer the mass of every object. In order to guarantee this, each object must collide directly with the reference object or be linked indirectly to it through a sequence of collisions. For the inelastic domain, we must ensure that each object’s COR can be inferred as well. In a ball-ball collision, only the higher object COR is used in determining collision dynamics, and so only the higher object COR can be inferred from the collision. For this reason, every ball must either collide with a ball of lower COR or a wall.

## 4.2 MODEL ARCHITECTURE

We use a single model architecture for all of our experiments. We set  $L_C$ , the size of each code vector, to 25 and  $L_Z$ , the size of each property vector, to 15. All MLPs in the model, including those in the interaction networks, use linear hidden layers with ReLU activation and a linear output layer.

Following the overall structure of Battaglia *et al.* [2], the perception network’s IN core consists of a 4-layer relation-centric MLP with sizes  $[75, 75, 75, 50]$  and a 3-layer object-centric MLP with sizes  $[50, 50, 25]$ . The final code vectors output by the IN feed into another object-centric MLP of size  $[15, 15, 15]$  to produce the final latent property vectors of size 15. The prediction network’s IN core consists of a 5-layer relation-centric MLP with sizes  $[100, 100, 100, 100, 50]$  and a 3-layer object-centric MLP with sizes  $[50, 50, 4]$  used to predict each object’s next position and velocity.

The perception network and prediction network are trained end-to-end using a single training loss, which we call the prediction loss. The prediction loss is the unweighted sum of the MSE of the predicted vs actual state vectors of all objects during the 24 rollout timesteps. In addition, we apply L2 regularization on the “effects” layer of both the perception and prediction networks. This regularization encourages minimal information exchange during interactions and proves to be a crucial component to generalization to different numbers of objects. We selected the penalty factor for each regularization term via grid search. We also experimented with the use of  $\beta$ -VAE regularization [24, 25] on property vectors to encourage the learning of interpretable and factorized properties.

In order to improve stability when simulating long rollouts, we added a small amount of Gaussian noise to each state vector during rollout, forcing the model to self-correct for errors. Empirically, we found that setting the noise std. dev. equal to  $0.001 \times$  the std. dev. of each state vector element’s values across the dataset stabilized rollout positions without affecting loss.

We trained the model for 150 epochs and optimized the parameters using Adam [26] with mini-batch size 256.

<sup>‡</sup><http://brm.io/matter-js/>

Component #	Springs		Perfectly Elastic Balls		Inelastic Balls		
	EVR	$R^2$ w/ log charge	EVR	$R^2$ w/ log mass	EVR	$R^2$ w/ log mass	$R^2$ w/ COR
1	0.94	0.95	0.99	0.94	0.73	0.90	0.02
2	0.06	0.02	0.006	0	0.27	0.02	0.81
3	0	0	0	0	0.006	0	0
4	0	0	0	0	0	0	0

Table 1: **Principal component analysis.** Applying PCA on the property vectors yields principal components that are highly correlated with human-interpretable latent properties such as COR and the log of mass. We compute statistics on the first four principal components of the property vectors for each training set. Explained variance ratio or EVR is the explained variance of the principal component as a fraction of overall variance, and  $R^2$  is the squared in-sample correlation between the principal component and a particular ground truth property. Values less than  $10^{-3}$  round to 0.

# Training Data	# Test Objects	Springs	Perfectly Elastic Balls	Inelastic Balls	
		$R^2$ w/ log charge	$R^2$ w/ log mass	$R^2$ w/ log mass	$R^2$ w/ COR
$10^5$	6	0.60	0.91	0.55	0.03
$2 \times 10^5$	6	0.95	0.96	0.95	0.65
$5 \times 10^5$	6	0.94	0.94	0.91	0.77
$10^6$	6	0.95	0.94	0.90	0.80
$10^6$	3	0.90	0.97	0.92	0.86
	9	0.87	0.92	0.90	0.68

Table 2: **Data-efficiency and number of objects generalization.** The PPN learns to capture physical properties with  $10^5$  training data points and converges when given  $2 \times 10^5$  instances. Its predictions generalize well to out-of-sample test sets with varying numbers of objects. We train the PPN on a 6-object dataset and test it on entirely new datasets comprised of 6, 3, and 9 objects. Above, we report the  $R^2$  when using the property vector’s first principal component to predict log mass and the second principal component to predict COR (for the inelastic balls case). Note that even in the 3 and 9 object cases the PPN is able to extract mass and coefficient of restitution with high  $R^2$ .

We used a waterfall schedule that began with a learning rate of  $5 \times 10^{-4}$  and downscaled by 0.8 each time the validation error, estimated over a window of 10 epochs, stopped decreasing.

## 5 RESULTS

### 5.1 EXTRACTING LATENT PROPERTIES

Our results show that the physical properties of objects are successfully encoded in the property vectors output by the perception network. In fact, we can extract the human-interpretable notions of spring charge, mass, and COR by applying principal component analysis (PCA) to the property vectors generated by the perception network during training. We find that the first principal component of each property vector is highly correlated with the log of spring charge in the spring domain and the log of object mass in both bouncing ball domains. In the inelastic balls domain, we also find that the second principal component of the property vector is highly correlated with COR. Table 1 shows the explained variance ratio (EVR) of each of the first 4 principal components of the learned property vectors in all three domains, along with the  $R^2$  when each component is used to predict ground truth object prop-

erties<sup>§</sup>. Since PCA is an unsupervised technique, these scalar quantities can be discovered without prior notions of mass and COR, and we can use the order-of-magnitude difference between certain principal components’ EVR to identify which components represent meaningful properties and which merely capture noise.

We also find that each learned property vector only contains information about its associated object and not any other objects. We test this hypothesis by using linear least squares to calculate the in-sample  $R^2$  between the ground truth latent properties of each object and the concatenation of the property vectors of all other objects. This  $R^2$  is less than 5% for each of the three domains and their relevant latent properties.

In order to test the generalization properties of our perception network, we calculate the out-of-sample  $R^2$  when using the perception network (trained on 6 object dynamics) and PCA to predict property values for test sets with varying number of objects, as shown in Table 2. The table

<sup>§</sup>By default, the property values produced by PCA will not be in the same scale as our ground truth values. For the purposes of correlation analysis, we linearly scale predictions to match the mean and std. dev. of the ground truth latent values.

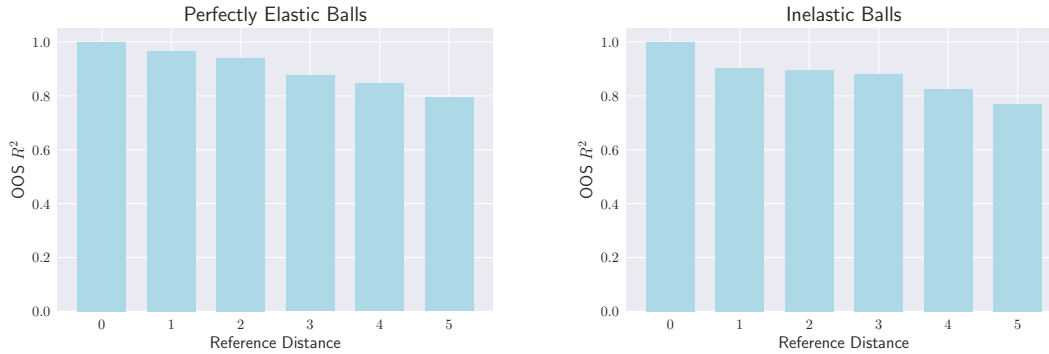


Figure 3: **Mass prediction vs. reference distance.** Out-of-sample  $R^2$  on the two 6-object bouncing balls datasets for predicting log mass at different reference distances. The PPN must combine a sequence of intermediate mass inferences to accurately infer the mass of an object with large reference distance.

also shows how PPN performs when given a different number of training instances. In all bouncing balls test sets, for our model trained on  $10^6$  data points, the OOS  $R^2$  for log mass is above 90%, the OOS  $R^2$  for COR is above 68%, and the OOS  $R^2$  for log spring charge in the springs domain is above 87%.

We also compare the PPN against a **LSTM-PPN** baseline. The LSTM-PPN replaces each of the perception and prediction networks in the PPN with stacked LSTMs. Unlike an interaction network, an LSTM does not factorize input and output by object. Instead, state vectors for each object are concatenated and processed together, and a single property vector is learned for all objects. Table 3 shows that the LSTM-PPN does not learn meaningful latent properties. In each scenario, the linear least squares in-sample  $R^2$  between true object properties and property vectors is less than 2%. We also experiment with different values of  $\beta$  in the regularization term of the property vectors  $Z$  as in  $\beta$ -VAE [25]. The value of  $\beta$  does not impact the PPN’s performance on learning object properties.

For the two bouncing balls domains, the relative masses of objects are inferred through collisions, but not all objects collide directly with the reference object. We define the *reference distance* of an object to be the minimum number of collisions needed during observation to relate the object’s mass to that of the reference object. Inference on an object with reference distance of 3, for example, depends on the inference of the mass of two intermediate objects. Figure 3 shows the relation between the PPN’s prediction  $R^2$  and reference distance for each of the 6-object test sets. While there is a decay in  $R^2$  as reference distance increases due to compounding errors during inference, the PPN clearly demonstrates the ability to use transitivity to infer the mass of objects with large reference distance.

## 5.2 ROLLOUT PREDICTIONS

Although the PPN’s primary objective is the unsupervised learning of latent physical properties, the network can

Methods	Springs	Elastic Balls	Inelastic Balls	
	log charge	log mass	log mass	COR
LSTM	0.02	0.03	0.02	0.03
PPN ( $\beta = 0$ )	<b>0.95</b>	<b>0.94</b>	0.90	<b>0.80</b>
PPN ( $\beta = 0.01$ )	<b>0.95</b>	0.93	<b>0.93</b>	0.79
PPN ( $\beta = 1$ )	0.92	<b>0.94</b>	<b>0.93</b>	0.65

Table 3: **Comparing with baseline methods.** Varying the value of  $\beta$  in the regularization term as in  $\beta$ -VAE does not change the PPN’s performance significantly. The PPN consistently outperforms the baseline LSTM.

also be used to simulate object dynamics. To evaluate the PPN’s prediction performance, we use the mean Euclidean prediction error, or the mean Euclidean norm between the ground truth and predicted rollout positions, averaged over all samples and objects. We compare the PPN’s performance against two benchmarks. The **Mean Properties Perfect Rollout (MPPR)** baseline outputs a perfect rollout from the starting state, but incorrectly assumes that all object masses and spring charges are 1. For the inelastic balls domain, it also assumes that all object CORs are 0.75. The **Ground Truth Properties Interaction Network (GPIN)** benchmark is an IN with the same architecture as the PPN’s prediction network. Unlike the PPN, it has direct access to ground truth latent values as input, though it is still only trained on 6-object datasets. Figure 4 lists the three models’ mean Euclidean prediction errors for various scenes and shows how the prediction errors vary for different rollout steps. The PPN’s mean Euclidean prediction error is significantly better than the MPPR baseline and comes reasonably close to the GPIN model, especially for the springs and perfectly elastic balls datasets.

Finally, Figure 5 shows visualizations of the PPN’s rollout trajectories. Randomly selected simulations can be found at <http://ppn.csail.mit.edu>. Like the original IN, the PPN’s rollouts are sensitive to small prediction errors in early timesteps, but remain visually convincing.

Model	Springs			Perfectly Elastic Balls			Inelastic Balls		
	6 balls	3 balls	9 balls	6 balls	3 balls	9 balls	6 balls	3 balls	9 balls
PPN	0.020	0.078	0.057	0.025	0.017	0.032	0.048	0.041	0.054
MPPR	0.124	0.082	0.139	0.038	0.027	0.046	0.062	0.045	0.073
GPIN	0.005	0.068	0.043	0.019	0.015	0.027	0.029	0.021	0.039

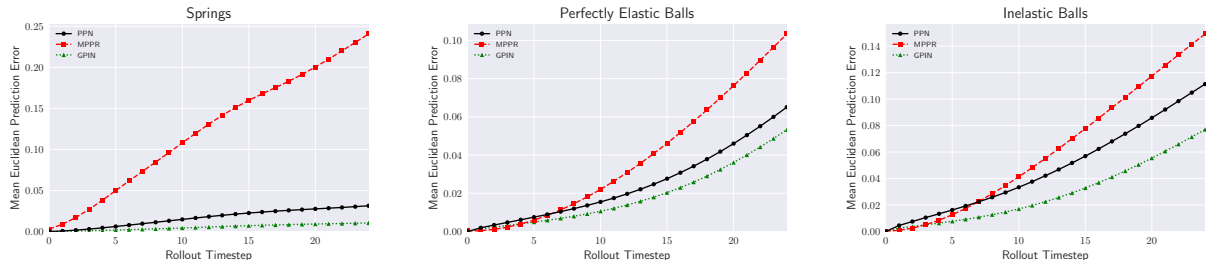


Figure 4: **Mean Euclidean prediction error.** *Top:* Mean Euclidean prediction error over all timesteps and samples for each test set measured as fraction of framewidth. For each domain, the PPN and GPIN are trained on 6-object systems and tested on new systems with 6, 3, and 9 objects. *Bottom:* Mean Euclidean prediction error at different rollout timesteps for each of the 6-object scenarios. Plots for the 3-object and 9-object scenarios exhibit similar behavior.

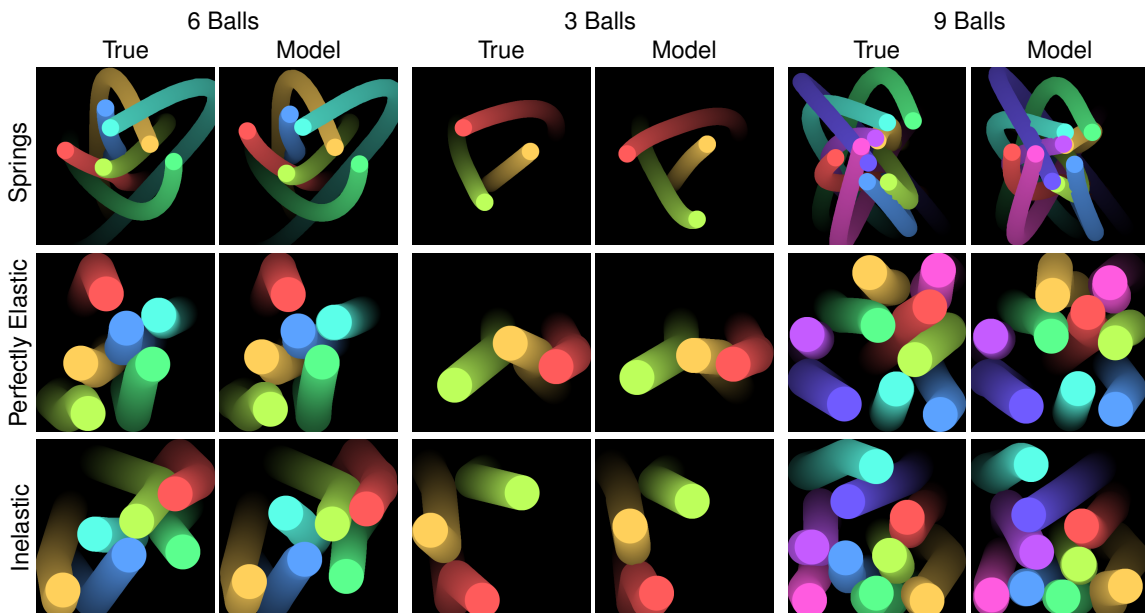


Figure 5: **Rollout trajectories.** Sample rollout trajectories (over 24 timesteps) from each of the six test sets. Each domain’s model was trained on 6-object samples and tested on 6-, 3-, and 9-object samples.

### 5.3 GENERALIZING TO NEW OBJECTS

Our experiments also explore generalizations to objects whose property values are outside the range found in the training set. We test the PPN framework on a 2-object perfectly elastic test set where the second ball’s mass varies from  $32^{-1}$  to 32. Mass values in the range  $[0.25, 4]$  are found within the training set, while mass values outside this range require the PPN to extrapolate its understanding of mass to values it has not previously been exposed to. We perform a similar experiment on the springs domain, in which the second object’s spring charge varies from  $32^{-1}$  to 32. Figure 6 plots the relationship between true

and predicted property values for the second ball in the two domains, using the same PCA technique described in Section 5.1 to make predictions.

In the perfectly elastic balls domain, the PPN continues to offer accurate predictions of mass even when the true value lies far outside training range, despite an overall tendency to underestimate large mass values and overestimate small mass values. In the springs domain, the PPN is able to predict objects with large spring charge relatively well but performs poorly on objects with low spring charge. This is likely due to the fact that objects with low spring charge tend to feel very little spring force overall,

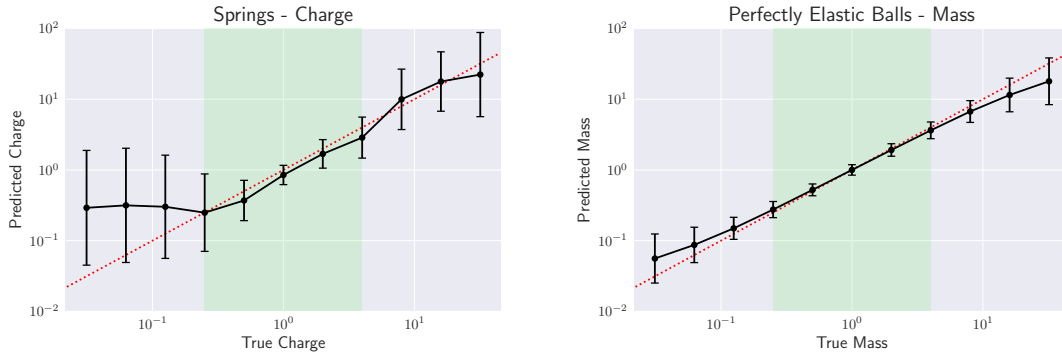


Figure 6: **Property value generalization.** Predicted property values vs. true property values of the second object in the 2-object test sets for both the springs and perfectly elastic balls domains. The true property values range from  $32^{-1}$  to 32, and the green region,  $4^{-1}$  to 4, indicates property values which appear to the PPN during training. Error bars show 95% confidence intervals. On the whole, the PPN continues to make reasonable predictions on mass and spring charge values well outside the training set, though the prediction of objects with lower spring charge than previously encountered is noticeably worse.

making the difference between charges of  $32^{-1}$  and  $16^{-1}$  much less noticeable than the difference between charges of 16 and 32.

## 6 DISCUSSION

We have presented the PPN, a model that is capable of discovering latent object properties in an entirely unsupervised manner from samples of object dynamics. Through our experiments, we showed not only that the representations of object properties learned by the PPN are sufficient to accurately simulate the dynamics of new systems under the same laws; but also that these learned representations can be readily transformed into relevant, human-interpretable properties such as mass and coefficient of restitution via principal component analysis.

The PPN demonstrates robustness by generalizing to novel scenarios with little loss in the accuracy of dynamical predictions or latent property inference. By using interaction networks as the basic building block of both our perception and prediction modules, we enabled our model to scale to arbitrary numbers of objects and interactions without architectural change. Our perception network architecture, in particular, is a simple but effective combination of relation and recurrent networks that may be useful in other time series inference tasks involving interacting objects. We also established the PPN’s ability to infer latent properties outside the range of values seen during training, further boosting its potential in discovering the relevant latent properties of new systems.

Several extensions would further improve the applicability of our model to the general discovery of latent object properties. In particular, there are a few general classes of problems that which interaction network-based architectures haven’t been able to solve: collision detec-

tion between rigid bodies of an arbitrary shape, dense fluid simulation, etc. Extending interaction networks to particle-based object representations is a promising future research direction [27].

While the interaction network framework is generally extensible to arbitrary numbers of objects, the computational time required to process all objects scales quadratically with the number of objects due to the presence of interaction terms between all pairs of objects, making it impractical for very large systems. One way to improve the computational efficiency of both the perception and prediction modules is to only consider interactions from objects in the neighborhood of target objects (with the interpretation that most interactions are only strong on shorter length scales), similar to Chang et al. [3]. A smaller, global interaction net could still be used to model longer range interactions.

The PPN provides a promising method for deriving the underlying properties governing the dynamics of systems, in addition to being a more general learnable physics engine capable of reasoning about potentially unknown object properties. The entirely unsupervised manner of its operation and its many generalization characteristics make the PPN suitable for application to a variety of systems, and it may even be able to discover relevant latent properties in domains that are yet to be well understood.

## 7 ACKNOWLEDGMENTS

We thank Michael Chang for his important insights and the anonymous reviewers for their useful suggestions. This work was supported by ONR MURI N00014-16-1-2007, the Center for Brain, Minds and Machines (NSF #1231216), Facebook, and the Toyota Research Institute.

## References

- [1] Tomer Ullman, Andreas Stuhlmüller, Noah Goodman, and Joshua B Tenenbaum. Learning physics from dynamical scenes. In *Annual Conference of the Cognitive Science Society*, 2014.
- [2] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. In *Advances in Neural Information Processing Systems*, 2016.
- [3] Michael B Chang, Tomer Ullman, Antonio Torralba, and Joshua B Tenenbaum. A compositional object-based approach to learning physical dynamics. In *International Conference on Learning Representations*, 2017.
- [4] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.
- [5] Peter W Battaglia, Jessica B Hamrick, and Joshua B Tenenbaum. Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences*, 110(45):18327–18332, 2013.
- [6] Christopher Bates, Peter Battaglia, Ilker Yildirim, and Joshua B Tenenbaum. Humans predict liquid dynamics using probabilistic simulation. In *Annual Conference of the Cognitive Science Society*, 2015.
- [7] Jessica Hamrick, Peter Battaglia, and Joshua B Tenenbaum. Internal physics models guide probabilistic judgments about object dynamics. In *Annual Conference of the Cognitive Science Society*, 2011.
- [8] Jiajun Wu, Ilker Yildirim, Joseph J Lim, Bill Freeman, and Josh Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *Advances in Neural Information Processing Systems*, 2015.
- [9] Jiajun Wu, Joseph J Lim, Hongyi Zhang, Joshua B Tenenbaum, and William T Freeman. Physics 101: Learning physical object properties from unlabeled videos. In *British Machine Vision Conference*, 2016.
- [10] Pulkit Agrawal, Ashvin V Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning to poke by poking: Experiential learning of intuitive physics. In *Advances in Neural Information Processing Systems*, 2016.
- [11] Sebastien Ehrhardt, Aron Monszpart, Niloy J Mitra, and Andrea Vedaldi. Learning a physical long-term predictor. *arXiv preprint arXiv:1703.00247*, 2017.
- [12] Katerina Fragkiadaki, Pulkit Agrawal, Sergey Levine, and Jitendra Malik. Learning visual predictive models of physics for playing billiards. In *International Conference on Learning Representations*, 2016.
- [13] Adam Lerer, Sam Gross, and Rob Fergus. Learning physical intuition of block towers by example. In *International Conference on Machine Learning*, 2016.
- [14] Roozbeh Mottaghi, Hessam Bagherinezhad, Mohammad Rastegari, and Ali Farhadi. Newtonian scene understanding: Unfolding the dynamics of objects in static images. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.
- [15] Roozbeh Mottaghi, Mohammad Rastegari, Abhinav Gupta, and Ali Farhadi. what happens if... learning to predict the effect of forces in images. In *European Conference on Computer Vision*, 2016.
- [16] Ilya Sutskever, Geoffrey E Hinton, and Graham W Taylor. The recurrent temporal restricted boltzmann machine. In *Advances in Neural Information Processing Systems*, 2009.
- [17] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.
- [18] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. In *International Conference on Learning Representations*, 2016.
- [19] David Raposo, Adam Santoro, David Barrett, Razvan Pascanu, Timothy Lillicrap, and Peter Battaglia. Discovering objects and their relations from entangled scene representations. In *ICLR Workshop*, 2017.
- [20] Nicholas Watters, Andrea Tacchetti, Theophane Weber, Razvan Pascanu, Peter Battaglia, and Daniel Zoran. Visual interaction networks. In *Advances in Neural Information Processing Systems*, 2017.
- [21] Jiajun Wu, Erika Lu, Pushmeet Kohli, Bill Freeman, and Josh Tenenbaum. Learning to see physics via visual de-animation. In *Advances in Neural Information Processing Systems*, 2017.
- [22] Marco Fraccaro, Simon Kamronn, Ulrich Paquet, and Ole Winther. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In *Advances in Neural Information Processing Systems*, 2017.

- [23] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In *International Conference on Machine Learning*, 2018.
- [24] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.
- [25] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017.
- [26] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.
- [27] Damian Mrowca, Chengxu Zhuang, Elias Wang, Nick Haber, Li Fei-Fei, Joshua B Tenenbaum, and Daniel LK Yamins. Flexible neural representation for physics prediction. *arXiv preprint arXiv:1806.08047*, 2018.

---

# Subsampled Stochastic Variance-Reduced Gradient Langevin Dynamics

---

Difan Zou\*

Department of Computer Science  
University of California  
Los Angeles, CA 90095, USA

Pan Xu\*

Department of Computer Science  
University of California  
Los Angeles, CA 90095, USA

Quanquan Gu

Department of Computer Science  
University of California  
Los Angeles, CA 90095, USA

## Abstract

Stochastic variance-reduced gradient Langevin dynamics (SVRG-LD) was recently proposed to improve the performance of stochastic gradient Langevin dynamics (SGLD) by reducing the variance of the stochastic gradient. In this paper, we propose a variant of SVRG-LD, namely SVRG-LD<sup>+</sup>, which replaces the full gradient in each epoch with a subsampled one. We provide a nonasymptotic analysis of the convergence of SVRG-LD<sup>+</sup> in 2-Wasserstein distance, and show that SVRG-LD<sup>+</sup> enjoys a lower gradient complexity<sup>1</sup> than SVRG-LD, when the sample size is large or the target accuracy requirement is moderate. Our analysis directly implies a sharper convergence rate for SVRG-LD, which improves the existing convergence rate by a factor of  $\kappa^{1/6}n^{1/6}$ , where  $\kappa$  is the condition number of the log-density function and  $n$  is the sample size. Experiments on both synthetic and real-world datasets validate our theoretical results.

## 1 INTRODUCTION

Markov chain Monte Carlo (MCMC) methods used for posterior sampling have achieved great successes in Bayesian machine learning and Bayesian statistics. Recently, a family of gradient-based MCMC algorithms derived from Langevin dynamics (Parisi, 1981) has become a research hotspot in both Bayesian sampling (Welling & Teh, 2011; Ahn et al., 2012; Wang et al., 2013; Dalalyan, 2014) and optimization (Raginsky et al., 2017; Zhang et al., 2017; Xu et al., 2017). The Langevin dynamics

is defined by the following stochastic differential equation (SDE)

$$d\mathbf{X}_t = -\nabla f(\mathbf{X}_t)dt + \sqrt{2}d\mathbf{B}_t, \quad (1.1)$$

where  $\mathbf{X}_t \in \mathbb{R}^d$  is a  $d$ -dimensional stochastic process,  $\mathbf{B}_t \in \mathbb{R}^d$  represents the standard  $d$ -dimensional Brownian motion and  $-\nabla f(\mathbf{x})$  is called the drift coefficient. It can be shown that the Langevin dynamics converges to an invariant stationary distribution  $\pi \propto \exp(-f)$  (Chiang et al., 1987). Based on this observation, various Langevin dynamics based numerical algorithms (Roberts & Tweedie, 1996; Mattingly et al., 2002) have been designed to sample from the target distribution  $\pi$ . Directly applying Euler-Maruyama discretization (Kloeden & Platen, 1992) to SDE (1.1) gives rise to

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \nabla f(\mathbf{x}_k)\eta + \sqrt{2\eta}\epsilon_k, \quad (1.2)$$

where  $\eta$  denotes the step size, and  $\epsilon_k \sim N(0, \mathbf{I}_{d \times d})$  is a  $d$ -dimensional standard Gaussian random vector. The sampling algorithm using (1.2) as its update formula is typically known as the Langevin Monte Carlo (LMC) algorithm, which has been extensively studied when the target distribution is both log-smooth and strongly log-concave, or even log-Hessian-Lipschitz (Dalalyan, 2014; Durmus & Moulines, 2016; Dalalyan, 2017; Dalalyan & Karagulyan, 2017).

On the other hand, modern machine learning problems often involve an extremely large amount of data. Suppose the dataset consists of  $n$  observations, it is often assumed that the function  $f$  in the drift term of (1.1) can be written as an average of  $n$  finite component functions, i.e.,

$$f(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^n f_i(\mathbf{x}), \quad (1.3)$$

where each  $f_i$  is smooth and  $f$  is strongly convex. When  $n$  is very large, the LMC algorithm can be inefficient

---

\*Equal contribution

<sup>1</sup>Gradient complexity is defined as the required number of stochastic gradient evaluations to reach a target accuracy.



since the gradient evaluation is computationally very expensive. Following the same idea in stochastic optimization, Welling & Teh (2011) proposed the stochastic gradient Langevin dynamics (SGLD) algorithm by replacing the full gradient in (1.2) with a stochastic gradient computed only on a minibatch of data. The SGLD algorithm has been successfully applied to Bayesian learning (Welling & Teh, 2011; Ahn et al., 2012) and training deep neural networks (Chaudhari et al., 2016; Ye et al., 2017), because it can dramatically decrease the number of stochastic gradient evaluations and save a lot computation in practice. Nevertheless, the convergence rate of SGLD is much slower than LMC, which may lead to a worse runtime complexity in certain regime. Regarding the true computational cost of SGLD, Nagapetyan et al. (2017) argued that SGLD is at most better by a constant factor relative to an Euler discretization with full gradients, and raised questions about the good performance of SGLD under the big-data setting. In order to fairly evaluate the performances of stochastic algorithms, one often uses gradient complexity to indicate the efficiency of a sampling algorithm in large scale machine learning problems. When  $f$  is smooth, strongly convex and Hessian Lipschitz, Dalalyan & Karagulyan (2017) proved that the gradient complexity of LMC to converge to the stationary distribution  $\pi$  in 2-Wasserstein distance is  $\tilde{O}(n\kappa^2 d^{1/2}/\epsilon)$ , where  $\epsilon$  represents the target accuracy and  $\kappa$  is the condition number of  $f$ . In comparison, the gradient complexity of SGLD is  $\tilde{O}(\kappa^2 d\sigma^2/\epsilon^2)$  (Dalalyan, 2017; Dalalyan & Karagulyan, 2017), which is slower than LMC when  $n \lesssim d^{1/2}\sigma^2/\epsilon$ , where  $d\sigma^2$  is an upper bound on the variance of the stochastic gradient.

In order to achieve the best of both worlds, i.e., save the gradient computation of LMC as well as boost the convergence rate of SGLD, Dubey et al. (2016) proposed stochastic variance-reduced gradient Langevin dynamics (SVRG-LD) and stochastic average gradient Langevin dynamics (SAGA-LD), which adapts the idea of variance reduction in stochastic optimization such as SVRG (Johnson & Zhang, 2013; Allen-Zhu & Hazan, 2016; Reddi et al., 2016) and SAGA (Defazio et al., 2014) to gradient-based Monte Carlo methods. However, Dubey et al. (2016) only investigated the performance of both algorithms in terms of mean square error (MSE) of the averaged sample path. Baker et al. (2017) applied zero variance control variates to stochastic MCMC method, and showed that such technique is able to reduce the computational cost of stochastic gradient Langevin dynamics to  $O(1)$ . Recently, Chatterji et al. (2018) analyzed the convergence rates of SVRG-LD and SAGA-LD to the stationary distribution in 2-Wasserstein distance, and showed that SAGA-LD has a lower gradient complexity compared with SVRG-LD. However, they also observed

that when considering low target accuracy regime or the samples size is very large, both of these variance reduction based LMC algorithms perform worse than SGLD, which can converge even within a single data pass. However, their theoretical results suggest that SAGA-LD attains a faster convergence rate than SVRG-LD, which is not consistent with the convergence analyses of SAGA and SVRG for optimization, where both methods have been proved to have the same gradient complexity (Johnson & Zhang, 2013; Defazio et al., 2014). Therefore, Chatterji et al. (2018) raised a question that whether SVRG is less suited than SAGA to work with sampling methods.

In this paper, in order to overcome the shortcomings of SVRG-LD and SAGA-LD, we propose a variant of SVRG-LD, namely SVRG-LD<sup>+</sup>, by replacing the full gradient computation in the outer loop of SVRG-LD with a subsampled one. The idea of using subsampled gradient instead of full gradient in variance reduction algorithms is originated from the recent work on variance reduction for stochastic optimization (Harikandeh et al., 2015; Lei & Jordan, 2016; Lei et al., 2017), and has also been adopted to Langevin based algorithm by Chen et al. (2017). It is worthy noting that the algorithm proposed in Chen et al. (2017), namely practical vrSG-MCMC, is similar to our algorithm. Nevertheless, the practical SVRG-LD algorithm needs to output all the iterates because its theoretical guarantee is on the sample path. In contrast, our algorithm only needs to output the last iterate, because our theory holds for the last iterate.

## 1.1 OUR CONTRIBUTIONS

We highlight the major contributions of our work as follows.

- We propose the SVRG-LD<sup>+</sup> algorithm and analyze its convergence rate to the target distribution in Wasserstein distance. Specifically, we prove that the SVRG-LD<sup>+</sup> algorithm requires  $\tilde{O}((n + \kappa^{3/2}n^{1/2}d^{1/2}/\epsilon) \wedge \kappa^2 d\sigma^2/\epsilon^2)$  stochastic gradient evaluations to converge to the target distribution in 2-Wasserstein distance within  $\epsilon$ -accuracy. Our result suggests that when the sample size  $n$  is large or the target accuracy  $\epsilon$  is moderate, the gradient complexity of SVRG-LD<sup>+</sup> is better than that of SVRG-LD and SAGA-LD (Chatterji et al., 2018). In addition, the gradient complexity of SVRG-LD<sup>+</sup> is never worse than that of SGLD.
- Since SVRG-LD is a special case of SVRG-LD<sup>+</sup> when the subsampled gradient is chosen to be the full gradient, our analysis of SVRG-LD<sup>+</sup> directly implies a sharp convergence rate of SVRG-LD,

which improves the recent result in Chatterji et al. (2018) by a factor of  $\kappa^{1/6}n^{1/6}$ , and matches the convergence rate of SAGA-LD (Chatterji et al., 2018). This suggests that both SVRG and SAGA are equally suited to work with sampling methods, and therefore answers the question raised in (Chatterji et al., 2018). Our experiments on both synthetic and real data also show that SVRG-LD and SAGA-LD have comparable performance, which verifies our theory.

We summarize the gradient complexities of existing LMC methods in Table 1, from which we can see that SVRG-LD<sup>+</sup> achieves the lowest gradient complexity among all methods. Detailed discussions will be provided in the main theory section.

## 1.2 ADDITIONAL RELATED WORK

Another line of research that is related to LMC is Hamiltonian Monte Carlo (HMC) method (Neal, 2011), which is based on Hamiltonian dynamics by introducing fictitious momentum variables. Recently, the HMC method has been widely studied and developed both experimentally and theoretically. Specifically, Chen et al. (2014) proposed a stochastic gradient HMC (SG-HMC) algorithm and demonstrated its better performance than SGLD in learning Bayesian neural networks. Chen et al. (2015) conducted a comprehensive analysis for a family of SG-MCMC algorithms including SG-HMC in terms of MSE, and showed that SG-HMC attains a better performance than SGLD if adopting an appropriate discretization method. Ma et al. (2015) proposed a general framework to design samplers from the target distribution, and generated a new state-adaptive sampler on the Riemannian manifold. The nonasymptotic convergence analysis of HMC and SG-HMC was provided in Cheng et al. (2017), where the authors analyzed an underdamped Langevin MCMC algorithm and proved the convergence guarantees in 2-Wasserstein distance. Zou et al. (2018) proposed a stochastic variance-reduced HMC algorithm and proved its convergence rate in 2-Wasserstein distance. Li et al. (2018) analyzed the mean square error of the HMC based algorithm for different discretization schemes. Our work is focused on LMC and is complementary to this line of research.

<sup>2</sup>The convergence of SGLD does not require the Hessian Lipschitz condition. However, Dalalyan & Karagulyan (2017) proved that the convergence rate of SGLD remains the same even with additional Hessian Lipschitz condition.

## 1.3 NOTATION

We use  $[n]$  to denote the index set  $\{1, \dots, n\}$ . For a random vector  $\mathbf{x}_k \in \mathbb{R}^d$ , we denote its probability distribution function by  $P(\mathbf{x}_k)$ . The 2-Wasserstein distance between two probability measures  $u$  and  $v$  is defined as follows,

$$\mathcal{W}_2(u, v) = \left( \inf_{\zeta \in \Gamma(u, v)} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|\mathbf{X}_u - \mathbf{X}_v\|_2^2 d\zeta(\mathbf{X}_u, \mathbf{X}_v) \right)^{1/2},$$

where the infimum is over all joint distributions  $\zeta$ . We use  $a_n = O(b_n)$  to denote that  $a_n \leq Cb_n$  for some constant  $C > 0$  independent of  $n$ , and use  $a_n = \tilde{O}(b_n)$  to hide the logarithmic terms of  $b_n$ . We also make use of the notation  $a_n \lesssim b_n$  ( $a_n \gtrsim b_n$ ) if  $a_n$  is less than (larger than)  $b_n$  up to a constant. We use  $a \wedge b$  and  $a \vee b$  to denote  $\min\{a, b\}$  and  $\max\{a, b\}$  respectively.

## 2 ALGORITHM

In this section, we present our SVRG-LD<sup>+</sup> algorithm, which is displayed in Algorithm 1.

The algorithm contains multiple epochs. At the beginning of the  $j$ -th epoch, we uniformly choose  $B$  samples from all training data and obtain a gradient estimator:

$$\tilde{\mathbf{g}}_j = \nabla f_{\mathcal{I}_j}(\tilde{\mathbf{x}}_j) = \frac{1}{|\mathcal{I}_j|} \sum_{i \in \mathcal{I}_j} \nabla f_i(\tilde{\mathbf{x}}_j), \quad (2.1)$$

where  $|\mathcal{I}_j| = B$ . At the  $l$ -th iteration in the  $j$ -th epoch, we define the semi-stochastic gradient as  $\mathbf{g}_k = \nabla f_{\tilde{\mathcal{I}}_k}(\mathbf{x}_k) - \nabla f_{\tilde{\mathcal{I}}_k}(\tilde{\mathbf{x}}_j) + \tilde{\mathbf{g}}_j$ , where  $k = jm + l$  is the total iteration of the algorithm,  $m$  is the length of each epoch, and  $\nabla f_{\tilde{\mathcal{I}}_k}(\mathbf{x}) = 1/|\tilde{\mathcal{I}}_k| \sum_{i \in \tilde{\mathcal{I}}_k} \nabla f_i(\mathbf{x})$ . Then we perform the following update

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \eta \mathbf{g}_k + \sqrt{2\eta} \boldsymbol{\epsilon}_k,$$

where  $\eta$  is the step size and  $\boldsymbol{\epsilon}_k \sim N(0, \mathbf{I}_{d \times d})$  is a Gaussian random vector.

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \eta \mathbf{g}_k + \sqrt{2\eta} \boldsymbol{\epsilon}_k,$$

where  $\eta$  is the step size and  $\boldsymbol{\epsilon}_k \sim N(0, \mathbf{I}_{d \times d})$  is a Gaussian random vector.

It is worth noting that the major difference between SVRG-LD<sup>+</sup> and SVRG-LD (Dubey et al., 2016) is that we replace the full gradient computation in the beginning of each epoch with a subsampled one. On one hand,

Table 1: Gradient complexity of gradient-based Monte Carlo algorithms in 2-Wasserstein distance for sampling from log-smooth, log-Hessian-Lipschitz and strongly log-concave distributions. For the ease of comparison, we follow Chatterji et al. (2018) that assumes  $n \lesssim d\sigma^2/(\mu^2\epsilon^2)$  and treats  $1/M$  and  $\mu$  as constants of order  $O(1)$ .

METHOD	GRADIENT COMPLEXITY	HESSIAN LIPSCHITZ
LMC (Dalalyan & Karagulyan, 2017)	$\tilde{O}\left(\frac{n\kappa^2 d^{1/2}}{\epsilon}\right)$	Yes
SGLD (Dalalyan, 2017)	$\tilde{O}\left(\frac{\kappa^2 d\sigma^2}{\epsilon^2}\right)$	No <sup>2</sup>
SAGA-LD (Chatterji et al., 2018) <sup>3</sup>	$\tilde{O}\left(n + \frac{\kappa^{3/2} n^{1/2} d^{1/2}}{\epsilon}\right)$	Yes
SVRG-LD (Chatterji et al., 2018)	$\tilde{O}\left(n + \frac{\kappa^{5/3} n^{2/3} d^{1/2}}{\epsilon}\right)$	Yes
SVRG-LD (this paper)	$\tilde{O}\left(n + \frac{\kappa^{3/2} n^{1/2} d^{1/2}}{\epsilon}\right)$	Yes
SVRG-LD <sup>+</sup> (this paper)	$\tilde{O}\left(\left(n + \frac{\kappa^{3/2} n^{1/2} d^{1/2}}{\epsilon}\right) \wedge \frac{\kappa^2 d\sigma^2}{\epsilon^2}\right)$	Yes

this leads to the consequence that the stochastic gradient  $\mathbf{g}_k$  is not an unbiased estimator of the true gradient  $\nabla f(\mathbf{x})$ , which introduces extra error that poses additional challenge in the analysis. On the other hand, compared with SVRG-LD, it saves gradient computations especially when the sample size  $n$  is large. Therefore, the crucial idea of SVRG-LD<sup>+</sup> is to make an appropriate trade-off between extra error and saving gradient computation, and the batch size  $B$  is a vital parameter which should be carefully designed.

---

#### Algorithm 1 SVRG-LD<sup>+</sup>

---

- 1: **input:** initial point  $\mathbf{x}_0$ , step size  $\eta$ , batch size  $B$ , mini-batch size  $b$ , epoch length  $m$
  - 2: **initialization:**  $\tilde{\mathbf{x}}_0 = \mathbf{x}_0$
  - 3: **for**  $j = 0, \dots, \lceil K/m \rceil$
  - 4:   Uniformly sample  $\mathcal{I}_j \subseteq [n]$  with  $|\mathcal{I}_j| = B$
  - 5:    $\tilde{\mathbf{g}}_j = \nabla f_{\mathcal{I}_j}(\tilde{\mathbf{x}}_j)$
  - 6:   **for**  $l = 0, \dots, m - 1$
  - 7:      $k = jm + l$
  - 8:     Uniformly sample  $\tilde{\mathcal{I}}_k \subseteq [n]$  where  $|\tilde{\mathcal{I}}_k| = b$
  - 9:      $\mathbf{g}_k = \nabla f_{\tilde{\mathcal{I}}_k}(\mathbf{x}_k) - \nabla f_{\tilde{\mathcal{I}}_k}(\tilde{\mathbf{x}}_j) + \tilde{\mathbf{g}}_j$
  - 10:      $\mathbf{x}_{k+1} = \mathbf{x}_k - \eta \mathbf{g}_k + \sqrt{2\eta}\epsilon_k$
  - 11:   **end for**
  - 12:    $\tilde{\mathbf{x}}_{j+1} = \mathbf{x}_{(j+1)m-1}$
  - 13: **end for**
  - 14: **output:**  $\mathbf{x}_K$
- 

### 3 MAIN THEORY

In this section, we are going to present our main theoretical results on the convergence rate of Algorithm 1 in

<sup>3</sup>Different from the definition in (1.3), the finite-sum function  $f$  is defined as  $f = \sum_{i=1}^n f_i(\mathbf{x})$  in Chatterji et al. (2018), which leads to a difference in the results by a factor of  $n$ . To make a fair comparison, we translate their results with the same definition in (1.3).

2-Wasserstein distance. We will first establish the convergence guarantees of Algorithm 1. Then, we will show that SVRG-LD<sup>+</sup> reduces to SVRG-LD when choosing  $B = n$ , and our analysis leads to a sharp convergence result of SVRG-LD that improves the recent result in Chaudhari et al. (2016).

For the target distribution  $\pi \propto e^{-f}$ , we first lay down the following assumptions on function  $f(\mathbf{x})$ , which are required in our analysis.

**Assumption 3.1** (Smoothness). There exists a positive constant  $M$  such that for each component function  $f_i(\mathbf{x})$ , the following holds for all  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ ,

$$\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\|_2 \leq M\|\mathbf{x} - \mathbf{y}\|_2.$$

Note that Assumption 3.1 immediately implies that the function  $f$  is also  $M$ -smooth, and consequently the target distribution  $\pi$  is  $M$ -log-smooth.

**Assumption 3.2** (Strong convexity). There exists a positive constant  $\mu$  such that for function  $f$ , the following holds for all  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ ,

$$f(\mathbf{x}) \geq f(\mathbf{y}) + \langle \nabla f(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle + \frac{\mu}{2}\|\mathbf{x} - \mathbf{y}\|_2^2.$$

The above assumption states that function  $f$  is strongly convex, which indicates that the distribution  $\pi \propto e^{-f}$  is strongly log-concave.

**Assumption 3.3** (Hessian Lipschitz). There exists a positive constant  $L$  such that for function  $f$ , the following holds for all  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ ,

$$\|\nabla^2 f(\mathbf{x}) - \nabla^2 f(\mathbf{y})\|_2 \leq L\|\mathbf{x} - \mathbf{y}\|_2.$$

This assumption is essential and useful for proving a faster convergence rate of Langevin Monte Carlo methods (Dalalyan & Karagulyan, 2017; Chatterji et al., 2018).

**Assumption 3.4** (Bounded Variance). There exists a constant  $\sigma$ , such that the following holds for all  $\mathbf{x} \in \mathbb{R}^d$ ,

$$\mathbb{E}_i[\|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x})\|_2^2] \leq d\sigma^2.$$

Assumption 3.4 is necessary and widely made in stochastic Langevin dynamics based methods such as SGLD (Dalalyan, 2017; Dalalyan & Karagulyan, 2017) and SGHMC (Cheng et al., 2017). However, it should be noted that this assumption is only required for the analysis of the SVRG-LD<sup>+</sup> algorithm but not required for SVRG-LD.

In what follows, we will present the convergence results of Algorithm 1. Following the literature (Dalalyan, 2017; Dalalyan & Karagulyan, 2017; Cheng & Bartlett, 2017; Zou et al., 2018; Chatterji et al., 2018), we will focus on the 2-Wasserstein distance between the target distribution  $\pi \propto e^{-f}$  and the distribution of the  $k$ -th iterate in Algorithm 1. Specifically, we have the following theorem for SVRG-LD<sup>+</sup>.

**Theorem 3.5.** Under Assumptions 3.1-3.4, let  $P(\mathbf{x}_k)$  denote the distribution of the  $k$ -th iterate  $\mathbf{x}_k$  in Algorithm 1. Set the step size  $\eta$  to satisfy

$$\eta \leq \min \left\{ \left( \frac{b\mu}{24M^4m^2} \right)^{1/3}, \frac{1}{6m(\sigma^2/B + M)} \right\}.$$

The 2-Wasserstein distance between  $P(\mathbf{x}_k)$  and  $\pi$  is bounded by

$$\begin{aligned} \mathcal{W}_2(P(\mathbf{x}_k), \pi) &\leq (1 - \eta\mu/4)^k \mathcal{W}_2(P(\mathbf{x}_0), \pi) + \frac{3\sigma d^{1/2}}{\mu B^{1/2}} \mathbf{1}(B \leq n) \\ &\quad + \frac{2\eta(Ld + M^{3/2}d^{1/2})}{\mu} \\ &\quad + \frac{4\eta M(md)^{1/2} \wedge 3\eta^{1/2}d^{1/2}\sigma}{(b\mu)^{1/2}}. \end{aligned}$$

It is worth noting that the mini-batch size  $b$  and the batch size  $B$  are two independent parameters in the algorithm that can be chosen separately. In practice, one typically chooses  $b \ll B$  (see for example, Harikandeh et al. (2015); Lei & Jordan (2016); Chen et al. (2017) in order to obtain a good convergence result. If we intentionally choose  $b > B$  in the algorithm, the evaluation of the semi-stochastic gradient will be even more expensive than that of the subsampled/full gradient in the outer loop, which makes variance reduction techniques no longer effective. The optimal choices of  $b$  and  $B$  in two different regimes of sample size  $n$  will be specified in the following corollaries.

Theorem 3.5 implies that in order to achieve  $\epsilon$  accuracy in 2-Wasserstein distance, the step size  $\eta$  should be set to

be sufficiently small, and the batch size  $B$  should be sufficiently large. To address these requirements, we present the following corollaries to show the optimal selections of  $\eta$  and  $B$ , and compute the gradient complexity of SVRG-LD<sup>+</sup> under different regimes.

We first consider the regime where  $n \gtrsim d\sigma^2/(\mu^2\epsilon^2)$ .

**Corollary 3.6.** Under the same assumptions as in Theorem 3.5, suppose the sample size satisfies  $n \gtrsim d\sigma^2/(\mu^2\epsilon^2)$ , if we set  $B = O(d\sigma^2\mu^{-2}\epsilon^{-2})$ ,  $b = O(1)$ ,  $m = O(B)$  and  $\eta = O(\mu\epsilon^2/(d\sigma^2))$ , Algorithm 1 achieves  $\epsilon$  accuracy in 2-Wasserstein distance after

$$T = \tilde{O}\left(\frac{d\sigma^2}{\mu^2\epsilon^2}\right) \quad (3.1)$$

stochastic gradient evaluations.

**Remark 3.7.** According to Corollary 3.6, if  $n \gtrsim d\sigma^2/(\mu^2\epsilon^2)$ , then the gradient complexity of SVRG-LD<sup>+</sup> in (3.1) matches that of SGLD (Dalalyan, 2017; Dalalyan & Karagulyan, 2017). Note that the gradient complexities of LMC, SAGA-LD and SVRG-LD are at least  $\tilde{O}(n)$  due to the use of full gradients, which indicates that SVRG-LD<sup>+</sup> achieves lower gradient complexity than LMC, SAGA-LD and SVRG-LD in this regime.

When the sample size satisfies  $n \lesssim d\sigma^2/(\mu^2\epsilon^2)$ , we choose the batch size  $B$  to be  $n$ , i.e., compute the full gradient in the beginning of each epoch in Algorithm 1. In this regime, Algorithm 1 reduces to SVRG-LD (Dubey et al., 2016), and its gradient complexity is characterized by the following corollary.

**Corollary 3.8.** Under the same assumptions as in Theorem 3.5, suppose the sample size satisfies  $n \lesssim d\sigma^2/(\mu^2\epsilon^2)$ , if we set  $b = 1$  and

$$\eta = \min \left\{ \frac{\mu\epsilon}{Ld + M^{2/3}d^{1/2}}, \frac{\mu^{1/2}\epsilon}{Md^{1/2}n^{1/2}} \right\},$$

SVRG-LD<sup>+</sup> achieves  $\epsilon$ -accuracy in 2-Wasserstein distance after

$$T = \tilde{O}\left(n + \frac{Ld + M^{3/2}d^{1/2}}{\mu^2\epsilon} + \frac{Md^{1/2}n^{1/2}}{\mu^{3/2}\epsilon}\right) \quad (3.2)$$

stochastic gradient evaluations.

**Remark 3.9.** According to Corollary 3.6, if  $n \lesssim d\sigma^2/(\mu^2\epsilon^2)$ , following Chatterji et al. (2018), if we further assume  $n \gtrsim L^2d/(M^2\mu) + \kappa$ , and treat  $1/M$  and  $\mu$  as constants of order  $O(1)$ , then the complexity in (3.2) can be simplified as

$$T = \tilde{O}\left(n + \frac{\kappa^{3/2}d^{1/2}n^{1/2}}{\epsilon}\right).$$

It is worth noting that in this regime, Algorithm 1 does not need Assumption 3.4. Moreover, combining the results in Corollaries 3.6 and 3.8, the gradient complexity of SVRG-LD<sup>+</sup> can be derived as follows

$$\tilde{O}\left(\left(n + \frac{\kappa^{3/2}n^{1/2}d^{1/2}}{\epsilon}\right) \wedge \frac{\kappa^2 d\sigma^2}{\epsilon^2}\right), \quad (3.3)$$

where  $O(1/\mu^2) = O(\kappa^2/M^2) = O(\kappa^2)$  as  $1/M = O(1)$ .

**Remark 3.10.** Corollary 3.8 essentially provides the gradient complexity for SVRG-LD, which is lower than that proved in Chatterji et al. (2018). Recall that their target distribution takes the form

$$\pi \propto \exp\left(-\sum_{i=1}^n f_i(\mathbf{x})\right) \triangleq \exp(-F(\mathbf{x})),$$

where the exponent term is different from our definition of  $f$  in (1.3) by a factor of  $1/n$ . In order to make their result comparable to ours, we translate their result to the same definition of  $f$  in (1.3), which gives rise to  $\tilde{O}(n + \kappa^{5/3}n^{2/3}d^{1/2}/\epsilon)$  gradient complexity of SVRG-LD (Chatterji et al., 2018). It is evident that our result improves the gradient complexity of SVRG-LD by a factor of  $(\kappa n)^{1/6}$ . Last but not the least, our proved gradient complexity of SVRG-LD matches that of SAGA-LD (Chatterji et al., 2018), which suggests that SVRG-LD and SAGA-LD enjoy the same performance.

## 4 PROOF OF THE MAIN THEORY

In this section we provide the proof for our main theory. We first define an operator  $\mathcal{L}$  derived from the Langevin dynamics. Specifically, let  $\mathbf{x}_0$  be any starting position, and we denote by  $\mathcal{L}_t\mathbf{x}_0$  the random position of the Markov process generated by Langevin dynamics (1.1) after time  $t$ . Let  $\mathbf{x}^\pi$  denote the random variable that satisfies the stationary distribution  $\pi \propto e^{-f}$ . In addition, we define  $\Delta_k = \mathcal{L}_\eta^k\mathbf{x}^\pi - \mathbf{x}_k$ , where  $\mathcal{L}_{k\eta} = \mathcal{L}_\eta \circ \mathcal{L}_\eta \circ \dots \circ \mathcal{L}_\eta = \mathcal{L}_\eta^k$  due to the Markov property of  $\mathcal{L}$ . Then the following holds trivially

$$\begin{aligned} \Delta_{k+1} &= \mathcal{L}_\eta^{k+1}\mathbf{x}^\pi - \mathbf{x}_{k+1} \\ &= \mathcal{L}_\eta^k\mathbf{x}^\pi - \mathbf{x}_k + \mathcal{L}_\eta^{k+1}\mathbf{x}^\pi - \mathcal{L}_\eta^k\mathbf{x}^\pi - (\mathbf{x}_{k+1} - \mathbf{x}_k). \end{aligned}$$

Consider two synchronously coupled Markov processes  $\mathcal{L}_\eta^k\mathbf{x}^\pi$  and  $\mathbf{x}_k$  which have shared Brownian motion term in updates  $\mathcal{L}_\eta^k\mathbf{x}^\pi \rightarrow \mathcal{L}_\eta^{k+1}\mathbf{x}^\pi$  and  $\mathbf{x}_k \rightarrow \mathbf{x}_{k+1}$ , we further have

$$\begin{aligned} \Delta_{k+1} &= \Delta_k + \eta\mathbf{g}_k - \int_0^\eta \nabla f(\mathcal{L}_{k\eta+t}\mathbf{x}^\pi)dt \\ &= \Delta_k + \eta(\mathbf{g}_k - \nabla f(\mathbf{x}_k)) - \eta(\nabla f(\mathcal{L}_\eta^k\mathbf{x}^\pi) - \nabla f(\mathbf{x}_k)) \\ &\quad - \int_0^\eta (\nabla f(\mathcal{L}_{k\eta+t}\mathbf{x}^\pi) - \nabla f(\mathcal{L}_{k\eta}\mathbf{x}^\pi))dt \\ &= \Delta_k + \eta\Phi_k - \eta\mathbf{U}_k - \mathbf{S}_k - \mathbf{V}_k, \end{aligned} \quad (4.1)$$

where we define

$$\begin{aligned} \Phi_k &= \mathbf{g}_k - \nabla f(\mathbf{x}_k), \\ \mathbf{U}_k &= \nabla f(\mathcal{L}_\eta^k\mathbf{x}^\pi) - \nabla f(\mathbf{x}_k), \\ \mathbf{S}_k &= \sqrt{2} \int_0^\eta \int_0^t \nabla^2 f(\mathcal{L}_{k\eta+s}\mathbf{x}^\pi)dB_s dt, \\ \mathbf{V}_k &= \int_0^\eta (\nabla f(\mathcal{L}_{k\eta+t}\mathbf{x}^\pi) - \nabla f(\mathcal{L}_{k\eta}\mathbf{x}^\pi))dt - \mathbf{S}_k. \end{aligned}$$

Note that in Algorithm 1, the semi-stochastic gradient  $\mathbf{g}_k$  has the following property

$$\begin{aligned} \mathbb{E}[\mathbf{g}_k|\tilde{\mathbf{x}}_j] &= \mathbb{E}[\nabla f_{\tilde{\mathcal{I}}_k}(\mathbf{x}_k) - \nabla f_{\tilde{\mathcal{I}}_k}(\tilde{\mathbf{x}}_j) + \nabla f_{\tilde{\mathcal{I}}_j}(\tilde{\mathbf{x}}_j)|\tilde{\mathbf{x}}_j] \\ &= \mathbb{E}[\nabla f(\mathbf{x}_k) - \nabla f(\tilde{\mathbf{x}}_j) + \nabla f_{\tilde{\mathcal{I}}_j}(\tilde{\mathbf{x}}_j)]. \end{aligned}$$

Then we can decompose  $\Phi_k$  as follows

$$\begin{aligned} \Phi_k &= \mathbf{g}_k - \nabla f(\mathbf{x}_k) \\ &= \underbrace{\nabla f_{\tilde{\mathcal{I}}_k}(\mathbf{x}_k) - \nabla f_{\tilde{\mathcal{I}}_k}(\tilde{\mathbf{x}}_j) - (\nabla f(\mathbf{x}_k) - \nabla f(\tilde{\mathbf{x}}_j))}_{\Psi_k} \\ &\quad + \underbrace{(\nabla f_{\tilde{\mathcal{I}}_j}(\tilde{\mathbf{x}}_j) - \nabla f(\tilde{\mathbf{x}}_j))}_{e_j}. \end{aligned} \quad (4.2)$$

Submitting the above equation into (4.1) yields

$$\Delta_{k+1} = \Delta_k - \eta\mathbf{U}_k + \eta\Psi_k + \eta e_j - \mathbf{S}_k - \mathbf{V}_k. \quad (4.3)$$

Now, we have already obtained the recursive update of  $\Delta_k$ . In what follows, we will upper bound the  $\ell_2$ -norm of each term on the R.H.S of (4.3). To begin with, we provide the following technical lemmas.

**Lemma 4.1.** (Dalalyan & Karagulyan, 2017) Under Assumptions 3.1 and 3.2, we have

$$\mathbb{E}[\|\Delta_k - \eta\mathbf{U}_k\|_2^2] \leq (1 - \eta\mu)^2\mathbb{E}[\|\Delta_k\|_2^2],$$

where  $\eta$  denotes the step size,  $\mu$  is the strongly convex parameter on function  $f(\mathbf{x})$ .

**Lemma 4.2.** Under Assumptions 3.1 and 3.2, we have the following upper bound on  $\|\Psi_k\|_2^2$ .

$$\begin{aligned} \mathbb{E}[\|\Psi_k\|_2^2] &\leq \frac{4d\sigma^2}{b} \wedge \frac{M^2}{b} (6m^2\eta^2 M^2\mathbb{E}[\|\Delta_{j_m}\|_2^2] + G_j)e^{2m^2M^2\eta^2}, \end{aligned} \quad (4.4)$$

where

$$G_j = 6m^2\eta^2(\mathbb{E}[\|e_j\|_2^2] + Md) + 2md\eta.$$

**Lemma 4.3.** Under Assumption 3.4,  $\|e_j\|_2$  is bounded as follows,

$$\mathbb{E}[\|e_j\|_2^2] = \mathbb{E}[\|\nabla f_{\tilde{\mathcal{I}}_j}(\tilde{\mathbf{x}}_j) - \nabla f(\tilde{\mathbf{x}}_j)\|_2^2] \leq \frac{d\sigma^2}{B}.$$

In addition, if  $B = n$ ,  $\mathbb{E}[\|e_j\|_2^2] = 0$ .

**Lemma 4.4.** (Dalalyan, 2017) Under Assumptions 3.1 and 3.3, regarding to terms  $\mathbf{S}_k$  and  $\mathbf{V}_k$  in (4.3), we have the following uniformly upper bound on their  $\ell_2$ -norms

$$\begin{aligned}\mathbb{E}[\|\mathbf{V}_k\|_2^2] &\leq \frac{\eta^4}{2}(L^2d^2 + M^3d), \\ \mathbb{E}[\|\mathbf{S}_k\|_2^2] &\leq \frac{\eta^3M^2d}{3},\end{aligned}$$

where  $M$  and  $L$  denotes the smoothness and Hessian Lipschitz parameters respectively.

Now, we are ready to present the proof for Theorem 3.5.

*Proof of Theorem 3.5.* Note that

$$\mathbb{E}[\boldsymbol{\Psi}_k | \mathbf{x}_k, \mathcal{L}_\eta^k \mathbf{x}^\pi, \mathcal{L}_\eta^{k+1} \mathbf{x}^\pi] = \mathbf{0},$$

which immediately implies

$$\begin{aligned}\mathbb{E}[\|\Delta_{k+1}\|_2^2] &= \mathbb{E}[\|\Delta_k - \eta\mathbf{U}_k + \eta\mathbf{e}_j - \mathbf{S}_k - \mathbf{V}_k\|_2^2] + \eta^2\mathbb{E}[\|\boldsymbol{\Psi}_k\|_2^2] \\ &\leq (1 + \alpha)\mathbb{E}[\|\Delta_k - \eta\mathbf{U}_k - \mathbf{S}_k\|_2^2] \\ &\quad + (1 + 1/\alpha)\mathbb{E}[\|\eta\mathbf{e}_j - \mathbf{V}_k\|_2^2] + \eta^2\mathbb{E}[\|\boldsymbol{\Psi}_k\|_2^2] \\ &= (1 + \alpha)\mathbb{E}[\|\Delta_k - \eta\mathbf{U}_k\|_2^2 + \|\mathbf{S}_k\|_2^2] \\ &\quad + (1 + 1/\alpha)\mathbb{E}[\|\eta\mathbf{e}_j - \mathbf{V}_k\|_2^2] + \eta^2\mathbb{E}[\|\boldsymbol{\Psi}_k\|_2^2], \\ &\leq (1 + \alpha)\mathbb{E}[\|\Delta_k - \eta\mathbf{U}_k\|_2^2 + \|\mathbf{S}_k\|_2^2] \\ &\quad + 2(1 + 1/\alpha)\mathbb{E}[\eta^2\|\mathbf{e}_j\|_2^2 + \|\mathbf{V}_k\|_2^2] + \eta^2\mathbb{E}[\|\boldsymbol{\Psi}_k\|_2^2],\end{aligned}$$

where  $\alpha > 0$  is an arbitrary chosen parameter, the first inequality is by Young's inequality, and the second equality follows from the fact  $\mathbb{E}[\mathbf{S}_k | \Delta_k, \mathbf{U}_k] = \mathbf{0}$ . Applying Lemmas 4.1 and 4.2, we have

$$\begin{aligned}\mathbb{E}[\|\Delta_{k+1}\|_2^2] &\leq (1 + \alpha)(1 - \eta\mu)^2\mathbb{E}[\|\Delta_k\|_2^2] + (1 + \alpha)\mathbb{E}[\|\mathbf{S}_k\|_2^2] \\ &\quad + 2(1 + 1/\alpha)\mathbb{E}[\eta^2\|\mathbf{e}_j\|_2^2 + \|\mathbf{V}_k\|_2^2] + \eta^2\mathbb{E}[\|\boldsymbol{\Psi}_k\|_2^2] \\ &\leq \left[ (1 + \alpha)(1 - \eta\mu)^2 + \frac{6\eta^4M^4m^2}{b}e^{2\eta^2m^2M^2} \right] \\ &\quad \times \max\{\mathbb{E}[\|\Delta_k\|_2^2], \mathbb{E}[\|\Delta_{jm}\|_2^2]\} + \Omega_1 + \Omega_2,\end{aligned}\tag{4.5}$$

where

$$\begin{aligned}\Omega_1 &= (1 + \alpha)\mathbb{E}[\|\mathbf{S}_k\|_2^2] \\ &\quad + 2(1 + 1/\alpha)\mathbb{E}[\eta^2\|\mathbf{e}_j\|_2^2 + \|\mathbf{V}_k\|_2^2], \\ \Omega_2 &= \frac{4d\sigma^2\eta^2}{b} \wedge \frac{M^2\eta^2G_j}{b}e^{2m^2M^2\eta^2}.\end{aligned}\tag{4.6}$$

Note that the step size  $\eta$  satisfies  $\eta \leq \min\{(b\mu/(24M^4m^2))^{1/3}, 1/(6m\sigma^2/B + 6mM)\}$ , we have  $\exp(2m^2M^2\eta^2) \leq 2$  and

$6\eta^4M^4m^2 \exp(2m^2M^2\eta^2)/b \leq \eta\mu/2$ . We choose  $\alpha = \eta\mu$ , which implies  $(1 + \alpha)(1 - \eta\mu)^2 \leq 1 - \eta\mu$ . Thus, (4.5) can be further rewritten as follows,

$$\mathbb{E}[\|\Delta_{k+1}\|_2^2] \leq (1 - \eta\mu/2) \max\{\mathbb{E}[\|\Delta_k\|_2^2], \mathbb{E}[\|\Delta_{jm}\|_2^2]\} + \Omega_1 + \Omega_2.\tag{4.7}$$

In order to obtain the upper bound of  $\mathbb{E}[\|\Delta_k\|_2^2]$ , we need to recursively call (4.7). Note that since  $jm \leq k$ , the number of calls to (4.7) must be smaller than  $k$ , thus we have

$$\mathbb{E}[\|\Delta_k\|_2^2] \leq (1 - \eta\mu/2)^k \mathbb{E}[\|\Delta_0\|_2^2] + \frac{\Omega_1 + \Omega_2}{\eta\mu/2}.\tag{4.8}$$

In what follows, we are going to upper bound  $\Omega_1$  and  $\Omega_2$ . Note that  $m \geq 1$  and  $M \geq \mu$ , we have  $\eta\mu \leq 1$ . Then by the application of Lemmas 4.3 and 4.4, we have

$$\begin{aligned}\Omega_1 &\leq \frac{(1 + \alpha)\eta^3M^2d}{3} + 2(1 + 1/\alpha) \\ &\quad \times \left( \frac{\eta^2d\sigma^2}{B} \mathbb{1}(B < n) + \frac{\eta^4(L^2d^2 + M^3d)}{2} \right) \\ &\leq \frac{2\eta^3M^2d}{3} + \frac{4\eta d\sigma^2}{B\mu} + \frac{2\eta^3(L^2d^2 + M^3d)}{\mu}, \\ \Omega_2 &\leq \frac{4d\sigma^2\eta^2}{b} \wedge \frac{6M^2md\eta^3}{b}.\end{aligned}$$

Then we substitute the above upper bounds of  $\Omega_1$  and  $\Omega_2$  into (4.8), and obtain

$$\begin{aligned}\mathbb{E}[\|\Delta_k\|_2^2] &\leq (1 - \eta\mu/2)^k \mathbb{E}[\|\Delta_0\|_2^2] + \frac{\Omega_1 + \Omega_2}{\eta\mu/2} \\ &\leq (1 - \eta\mu/2)^k \mathbb{E}[\|\Delta_0\|_2^2] + \frac{8d\sigma^2}{B\mu^2} \mathbb{1}(B < n) \\ &\quad + \frac{4\eta^2(L^2d^2 + M^3d)}{\mu^2} + \frac{14\eta^2mM^2d}{b\mu} \wedge \frac{8d\sigma^2\eta}{b\mu}.\end{aligned}$$

Based on the definition of 2-Wasserstein distance, we have  $\mathcal{W}_2^2(P(\mathbf{x}_k), \pi) \leq \mathbb{E}[\|\Delta_k\|_2^2]$ . Applying the inequality that  $x^2 + y^2 + z^2 \leq (|x| + |y| + |z|)^2$  for all  $x, y, z \in \mathbb{R}$ , we complete the proof of Theorem 3.5.  $\square$

## 5 EXPERIMENTS

In this section, we are going to verify our theoretical results and evaluate the performances of different Langevin based algorithms on both synthetic and real datasets.

### 5.1 SIMULATION ON SYNTHETIC DATA

We first validate our theoretical results based on synthetic data. In this simulation, we consider function  $f(\mathbf{x}) =$

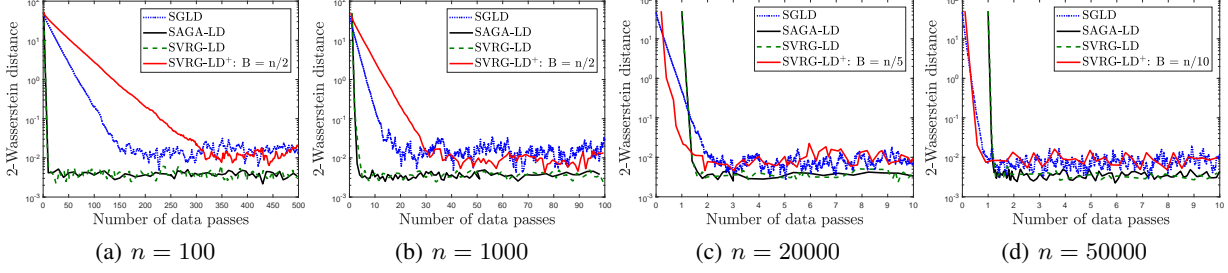


Figure 1: Comparison of different algorithms, where  $y$ -axis represents the 2-Wasserstein distance computed based on synthetic data, and  $x$ -axis is the number of data passes. (a) - (d) represent different sample sizes  $n$ .

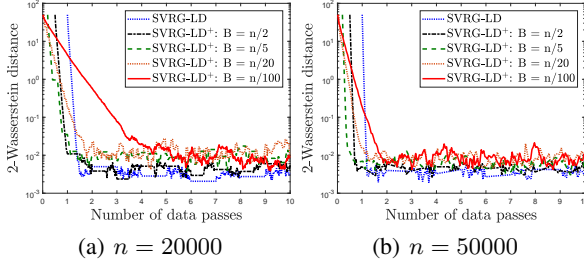


Figure 2: Comparison of SVRG-LD<sup>+</sup> with different batch size  $B$ , where  $y$ -axis represents the 2-Wasserstein distance computed based on synthetic data, and  $x$ -axis is the number of data passes. (a) and (b) represent different sample sizes  $n$ .

$1/n \sum_{i=1}^n f_i(\mathbf{x}) = 1/n \sum_{i=1}^n (\mathbf{x} - \boldsymbol{\theta}_i)^\top \boldsymbol{\Sigma} (\mathbf{x} - \boldsymbol{\theta}_i)/2$ , where  $\boldsymbol{\Sigma}$  is a symmetric matrix having largest eigenvalue  $M = 2$  and smallest eigenvalue  $\mu = 1/2$ , and  $\boldsymbol{\theta}_i$  is drawn from standard multivariate Gaussian distribution.

We first compare the convergence rates of four different algorithms (i.e., SGLD, SVRG-LD, SAGA-LD and SVRG-LD<sup>+</sup>) to the target distribution in 2-Wasserstein distance, which are reported in Figure 1. It can be seen that there is no obvious difference between the convergence rates of SAGA-LD and SVRG-LD, which verifies our theoretical result of SVRG-LD. Moreover, Figures 1(a) and 1(b) demonstrate that the best choice of  $B$  is  $B = n$  when the sample size  $n$  is small, while Figures 1(c) and 1(d) show that using subsampled gradient  $\tilde{\mathbf{g}}$  in SVRG-LD<sup>+</sup> (i.e.,  $B < n$ ) is able to improve the performance of SVRG-LD when  $n$  is relatively large. This is well aligned with our theoretical analysis that the optimal batch size  $B$  is in the order of  $O(d\sigma^2/(\epsilon\mu)^2 \wedge n)$ .

In Figure 2, we further compare different choices of batch size  $B$  in SVRG-LD<sup>+</sup> when  $n$  is large. Note that since the optimal batch size  $B = n$  when the sample size is small, we only perform this experiment on the synthetic datasets with big sample size  $n = 20000$  and  $n = 50000$ . It can be inferred from Figure 2 that if we set  $\epsilon = 10^{-2}$ , the optimal  $B$  in SVRG-LD<sup>+</sup> for datasets

with sample sizes  $n = 20000$  and  $n = 50000$  are both  $B = 20000/2 = 50000/5 = 10000$ . This phenomenon agrees with our theory that for a large  $n \gtrsim d\sigma^2/(\epsilon\mu)^2$ , the optimal batch size  $B = O(d\sigma^2/(\epsilon\mu)^2)$  is independent of  $n$ .

## 5.2 BAYESIAN LOGISTIC REGRESSION

We also collaborate our theoretical results with Bayesian logistic regression. Suppose we are given a dataset with  $n$  examples  $\{\mathbf{X}_i, \mathbf{y}_i\}_{i=1,2,\dots,n}$ , where  $\mathbf{X}_i \in \mathbb{R}^d$  denotes the  $d$ -dimensional feature of the  $i$ -th sample, and  $\mathbf{y}_i \in \{-1, 1\}$  denotes the corresponding binary label. In Bayesian logistic regression, we assume that the input examples are independent, then the probability distribution of  $\mathbf{y}_i$  given features  $\mathbf{X}_i$  and regression coefficients  $\boldsymbol{\beta} \in \mathbb{R}^d$  has the following form

$$p(\mathbf{y}_i | \mathbf{X}_i, \boldsymbol{\beta}) = \frac{1}{1 + e^{-\mathbf{y}_i \boldsymbol{\beta}^\top \mathbf{X}_i}}.$$

Moreover, the prior of  $\boldsymbol{\beta}$  is typically modelled as a Gaussian distribution with zero mean (Dubey et al., 2016; Chatterji et al., 2018), i.e.,  $\boldsymbol{\beta} \sim N(0, \lambda \mathbf{I}_{d \times d})$ . Then we apply the Langevin based method to sample from the posterior distribution of  $\boldsymbol{\beta}$ , i.e.,  $p(\boldsymbol{\beta} | \mathbf{X}, \mathbf{y}) \propto p(\boldsymbol{\beta}) \prod_{i=1}^n p(\mathbf{y}_i | \mathbf{X}_i, \boldsymbol{\beta})$ , which implies that the component function  $f_i(\boldsymbol{\beta})$  can be written as

$$f_i(\boldsymbol{\beta}) = n \log(1 + e^{-\mathbf{y}_i \boldsymbol{\beta}^\top \mathbf{X}_i}) + \frac{\|\boldsymbol{\beta}\|_2^2}{\lambda}.$$

We apply the Langevin based algorithm to four datasets: *pima*, *mushroom*, *a9a* and *ijcnn1*, which are available at UCI repository<sup>4</sup> and Libsvm website<sup>5</sup>. It is worth noting that *pima* and *mushroom* do not have test datasets like *a9a* and *ijcnn1*, thus we manually partition them into train and test datasets. The basic information of all the datasets is summarized in Table 2. Again, we evaluate the performance of four different algorithms: SGLD,

<sup>4</sup><https://archive.ics.uci.edu/ml/>

<sup>5</sup><https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/>

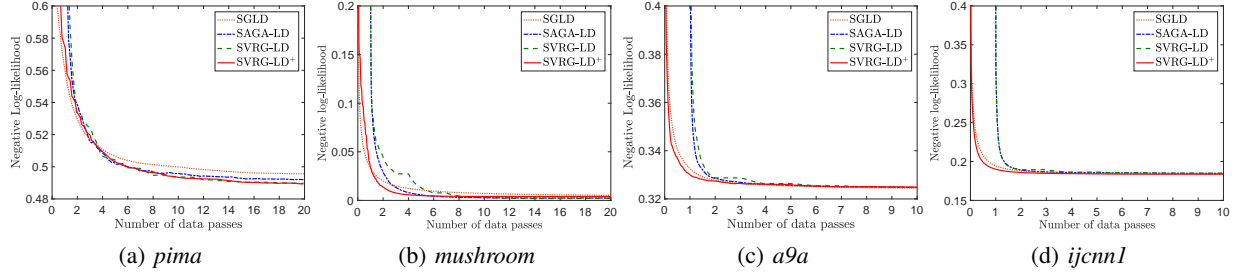


Figure 3: Comparison of different algorithms for Bayesian logistic regression, where  $y$  axis shows the negative log-likelihood on the test data, and  $x$  axis is the number of data passes. (a)-(d) correspond to 4 datasets.

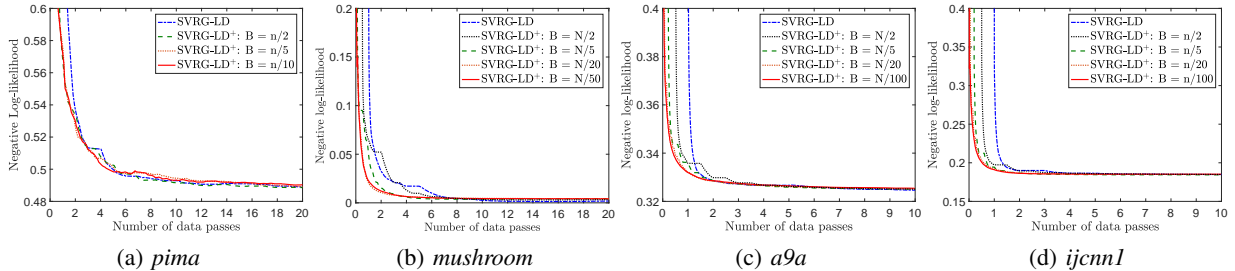


Figure 4: Bayesian Logistic regression results of SVRG-LD<sup>+</sup> using different batch size  $B$ , where  $y$  axis shows the negative log-likelihood on the test data, and  $x$  axis is the number of data passes. (a)-(d) correspond to 4 datasets.

SVRG-LD, SAGA-LD and SVRG-LD<sup>+</sup>, and perform sample path average to estimate the optimal  $\beta$ , where the minibatch size for each algorithm is set to be 1.

Figure 3 shows the negative log-likelihood of the test examples on these 4 datasets, where each algorithm has been run 10 times to calculate the averaged result. It can be seen that there exists a lag of one data pass for SAGA-LD and SVRG-LD, since they need to scan the entire dataset to compute a full gradient in the beginning. As we can see from the results in Figure 3, SVRG-LD<sup>+</sup> converges faster than the other methods, which validates the superior performance of SVRG-LD<sup>+</sup>. In detail, SVRG-LD<sup>+</sup> has a similar convergence rate as SAGA-LD and SVRG-LD when  $n$  is small, e.g. datasets *pima*, and performs close to SGLD for relatively large datasets, e.g., *a9a* and *ijcnn1*. This is also consistent with our theoretical results, since the convergence rate of SVRG-LD<sup>+</sup> matches that of SGLD when  $n \gtrsim d\sigma^2/(\epsilon\mu)^2$ .

gorithms when choosing different batch sizes  $B$ , which are reported in Figure 4. It can be observed that when the batch size is chosen appropriately, SVRG-LD<sup>+</sup> converges faster than SVRG-LD, but leading to a slightly higher error. Based on these observations, we can conclude that for Bayesian logistic regression, SVRG-LD<sup>+</sup> is more suitable than SVRG-LD when the dataset size is relatively large, and the required accuracy is moderate.

## 6 CONCLUSIONS

We propose the SVRG-LD<sup>+</sup> algorithm and analyze its convergence rate in 2-Wasserstein distance when the target distribution is log-smooth, strongly log-concave and log-Hessian-Lipschitz. Our result implies a sharper convergence analysis of SVRG-LD that improves the state-of-the-art. Experiments on synthetic and real data back up the theoretical results of this paper.

## Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments. This research was sponsored in part by the National Science Foundation IIS-1618948, IIS-1652539 and SaTC CNS-1717950. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

Table 2: Summary of datasets for Bayesian logistic regression.

Dataset	<i>pima</i>	<i>mushroom</i>	<i>a9a</i>	<i>ijcnn1</i>
# training	600	6000	32561	49990
# test	168	2124	16281	91701
$d$	8	112	123	22

Next, we evaluate the performance of SVRG-LD<sup>+</sup> al-



## References

- Ahn, Sungjin, Balan, Anoop Korattikara, and Welling, Max. Bayesian posterior sampling via stochastic gradient fisher scoring. In *ICML*, 2012.
- Allen-Zhu, Zeyuan and Hazan, Elad. Variance reduction for faster non-convex optimization. In *International Conference on Machine Learning*, pp. 699–707, 2016.
- Baker, Jack, Fearnhead, Paul, Fox, Emily B, and Nemeth, Christopher. Control variates for stochastic gradient MCMC. *arXiv preprint arXiv:1706.05439*, 2017.
- Chatterji, Niladri S, Flammarion, Nicolas, Ma, Yi-An, Bartlett, Peter L, and Jordan, Michael I. On the theory of variance reduction for stochastic gradient monte carlo. *arXiv preprint arXiv:1802.05431*, 2018.
- Chaudhari, Pratik, Choromanska, Anna, Soatto, Stefano, and LeCun, Yann. Entropy-sgd: Biasing gradient descent into wide valleys. *arXiv preprint arXiv:1611.01838*, 2016.
- Chen, Changyou, Ding, Nan, and Carin, Lawrence. On the convergence of stochastic gradient MCMC algorithms with high-order integrators. In *Advances in Neural Information Processing Systems*, pp. 2278–2286, 2015.
- Chen, Changyou, Wang, Wenlin, Zhang, Yizhe, Su, Qinliang, and Carin, Lawrence. A convergence analysis for a class of practical variance-reduction stochastic gradient MCMC. *arXiv preprint arXiv:1709.01180*, 2017.
- Chen, Tianqi, Fox, Emily, and Guestrin, Carlos. Stochastic gradient hamiltonian monte carlo. In *International Conference on Machine Learning*, pp. 1683–1691, 2014.
- Cheng, Xiang and Bartlett, Peter. Convergence of langevin MCMC in KL-divergence. *arXiv preprint arXiv:1705.09048*, 2017.
- Cheng, Xiang, Chatterji, Niladri S, Bartlett, Peter L, and Jordan, Michael I. Underdamped langevin MCMC: A non-asymptotic analysis. *arXiv preprint arXiv:1707.03663*, 2017.
- Chiang, Tzoo-Shuh, Hwang, Chii-Ruey, and Sheu, Shuenn Jyi. Diffusion for global optimization in  $\mathbb{R}^n$ . *SIAM Journal on Control and Optimization*, 25(3): 737–753, 1987.
- Dalalyan, Arnak S. Theoretical guarantees for approximate sampling from smooth and log-concave densities. *arXiv preprint arXiv:1412.7392*, 2014.
- Dalalyan, Arnak S. Further and stronger analogy between sampling and optimization: Langevin Monte Carlo and gradient descent. *arXiv preprint arXiv:1704.04752*, 2017.
- Dalalyan, Arnak S and Karagulyan, Avetik G. User-friendly guarantees for the langevin monte carlo with inaccurate gradient. *arXiv preprint arXiv:1710.00095*, 2017.
- Defazio, Aaron, Bach, Francis, and Lacoste-Julien, Simon. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, pp. 1646–1654, 2014.
- Dubey, Kumar Avinava, Reddi, Sashank J, Williamson, Sinead A, Póczos, Barnabas, Smola, Alexander J, and Xing, Eric P. Variance reduction in stochastic gradient langevin dynamics. In *Advances in Neural Information Processing Systems*, pp. 1154–1162, 2016.
- Durmus, Alain and Moulines, Eric. Sampling from strongly log-concave distributions with the unadjusted langevin algorithm. *arXiv preprint arXiv:1605.01559*, 2016.
- Harikandeh, Reza, Ahmed, Mohamed Osama, Virani, Alim, Schmidt, Mark, Konečný, Jakub, and Sallinen, Scott. Stopwasting my gradients: Practical SVRG. In *Advances in Neural Information Processing Systems*, pp. 2251–2259, 2015.
- Johnson, Rie and Zhang, Tong. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, pp. 315–323, 2013.
- Kloeden, Peter E and Platen, Eckhard. Higher-order implicit strong numerical schemes for stochastic differential equations. *Journal of statistical physics*, 66(1): 283–314, 1992.
- Lei, Lihua and Jordan, Michael I. Less than a single pass: Stochastically controlled stochastic gradient method. *arXiv preprint arXiv:1609.03261*, 2016.
- Lei, Lihua, Ju, Cheng, Chen, Jianbo, and Jordan, Michael I. Non-convex finite-sum optimization via scsg methods. In *Advances in Neural Information Processing Systems*, pp. 2345–2355, 2017.
- Li, Zhize, Zhang, Tianyi, and Li, Jian. Stochastic gradient hamiltonian monte carlo with variance reduction for bayesian inference. *arXiv preprint arXiv:1803.11159*, 2018.
- Ma, Yi-An, Chen, Tianqi, and Fox, Emily. A complete recipe for stochastic gradient MCMC. In *Advances in Neural Information Processing Systems*, pp. 2917–2925, 2015.
- Mattingly, Jonathan C, Stuart, Andrew M, and Higham, Desmond J. Ergodicity for sdes and approximations:

- locally lipschitz vector fields and degenerate noise. *Stochastic processes and their applications*, 101(2): 185–232, 2002.
- Nagapetyan, Tigran, Duncan, Andrew B, Hasenclever, Leonard, Vollmer, Sebastian J, Szpruch, Lukasz, and Zygalkis, Konstantinos. The true cost of stochastic gradient langevin dynamics. *arXiv preprint arXiv:1706.02692*, 2017.
- Neal, Radford M. MCMC using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2(11), 2011.
- Parisi, G. Correlation functions and computer simulations. *Nuclear Physics B*, 180(3):378–384, 1981.
- Raginsky, Maxim, Rakhlin, Alexander, and Telgarsky, Matus. Non-convex learning via stochastic gradient langevin dynamics: a nonasymptotic analysis. *arXiv preprint arXiv:1702.03849*, 2017.
- Reddi, Sashank J, Hefny, Ahmed, Sra, Suvrit, Póczos, Barnabas, and Smola, Alex. Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pp. 314–323, 2016.
- Roberts, Gareth O and Tweedie, Richard L. Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, pp. 341–363, 1996.
- Wang, Ziyu, Mohamed, Shakir, and Freitas, Nando. Adaptive hamiltonian and riemann manifold monte carlo. In *International Conference on Machine Learning*, pp. 1462–1470, 2013.
- Welling, Max and Teh, Yee W. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 681–688, 2011.
- Xu, Pan, Chen, Jinghui, Zou, Difan, and Gu, Quanquan. Global convergence of langevin dynamics based algorithms for nonconvex optimization. *arXiv preprint arXiv:1707.06618*, 2017.
- Ye, Nanyang, Zhu, Zhanxing, and Mantiuk, Rafal K. Langevin dynamics with continuous tempering for high-dimensional non-convex optimization. *arXiv preprint arXiv:1703.04379*, 2017.
- Zhang, Yuchen, Liang, Percy, and Charikar, Moses. A hitting time analysis of stochastic gradient langevin dynamics. *arXiv preprint arXiv:1702.05575*, 2017.
- Zou, Difan, Xu, Pan, and Gu, Quanquan. Stochastic variance-reduced hamilton monte carlo methods. *arXiv preprint arXiv:1802.04791*, 2018.

---

# Finite-state Controllers of POMDPs via Parameter Synthesis\*

---

Sebastian Junges<sup>1</sup>, Nils Jansen<sup>2</sup>, Ralf Wimmer<sup>3</sup>, Tim Quatmann<sup>1</sup>,  
Leonore Winterer<sup>3</sup>, Joost-Pieter Katoen<sup>1</sup>, and Bernd Becker<sup>3</sup>

<sup>1</sup>RWTH Aachen University, Aachen, Germany

<sup>2</sup>Radboud University, Nijmegen, The Netherlands

<sup>3</sup>Albert-Ludwigs-Universität Freiburg, Freiburg im Breisgau, Germany

## Abstract

We study finite-state controllers (FSCs) for partially observable Markov decision processes (POMDPs) that are provably correct with respect to given specifications. The key insight is that computing (randomised) FSCs on POMDPs is equivalent to—and computationally as hard as—synthesis for parametric Markov chains (pMCs). This correspondence allows to use tools for synthesis in pMCs to compute correct-by-construction FSCs on POMDPs for a variety of specifications. Our experimental evaluation shows comparable performance to well-known POMDP solvers.

## 1 INTRODUCTION

**Partially Observable MDPs.** We intend to provide guarantees for planning scenarios given by dynamical systems with uncertainties. In particular, we want to synthesise a *strategy* for an agent that ensures certain desired behaviour (Howard, 1960). A popular formal model for planning subject to stochastic behaviour are Markov decision processes (MDPs) (Puterman, 1994). An MDP is a nondeterministic model in which the agent chooses to perform an action under full knowledge of the environment it is operating in. The outcome of the action is a probability distribution over the system states. Many applications, however, allow only *partial observability* of the current system state (Kaelbling et al., 1998; Thrun et al., 2005; Wongpiromsarn and Frazzoli, 2012; Russell and Norvig, 2010). For such applications, MDPs are extended to *partially observable Markov decision processes* (POMDPs). While the agent acts within the environment, it encounters certain *observations*, according to which it

can infer the likelihood of the system being in a certain state. This likelihood is called the *belief state*. Executing an action leads to an update of the belief state according to new observations. The belief state together with an update function form a (typically uncountably infinite) MDP, referred to as the *belief MDP* (Shani et al., 2013).

**The POMDP Synthesis Problem.** For (PO)MDPs, a *randomised strategy* is a function that resolves the non-determinism by providing a probability distribution over actions at each time step. In general, strategies depend on the full history of the current evolution of the (PO)MDP. If a strategy depends only on the current state of the system, it is called *memoryless*. For MDPs, memoryless strategies suffice to induce optimal values according to our measures of interest (Puterman, 1994). Contrarily, POMDPs require strategies taking the full observation history into account (Ross, 1983), e. g. in case of infinite-horizon objectives. Moreover, strategies inducing *optimal* values are computed by assessing the entire belief MDP (Madani et al., 1999; Braziunas, 2003; Szer and Charpillat, 2005; Norman et al., 2017), rendering the problem undecidable (Chatterjee et al., 2016c).

POMDP strategies can be represented by *infinite-state controllers*. For computational tractability, strategies are often restricted to finite memory; this amounts to using *randomised finite-state controllers* (FSCs) (Meuleau et al., 1999). We often refer to strategies as FSCs. Already the computation of a memoryless strategy adhering to a specification is NP-hard, SQRT-SUM-hard, and in PSPACE (Vlassis et al., 2012). While optimal values cannot be guaranteed, small memory in combination with *randomisation* may supersede large memory in many cases (Chatterjee et al., 2004; Amato et al., 2010).

**Correct-by-Construction Strategy Computation.** In this paper, we synthesise FSCs for POMDPs. We require these FSCs to be provably correct for specifications such as indefinite-horizon properties like expected reward or reach-avoid probabilities. State-of-the-art POMDP

---

\*Supported by the DFG RTG 2236 “UnRAVeL”.

Table 1: Correspondence

POMDP under FSC	pMC
states $\times$ memory same observation strategy	states same parameter parameter instantiation

solvers mainly consider expected discounted reward measures (Walraven and Spaan, 2017), which are a subclass of indefinite horizon properties (Kolobov et al., 2012).

Our key observation is that for a POMDP the *set of all FSCs* with a fixed memory bound can be succinctly represented by a *parametric Markov chain* (pMC) (Daws, 2004). Transitions of pMCs are given by functions over a finite set of parameters rather than constant probabilities. The *parameter synthesis* problem for pMCs is to determine parameter instantiations that satisfy (or refute) a given specification. We show that the pMC parameter synthesis problem and the POMDP strategy synthesis problem are equally hard. This correspondence not only yields complexity results (Hutschenreiter et al., 2017), but particularly enables using a plethora of methods for parameter synthesis implemented in sophisticated and optimised parameter synthesis tools like PARAM (Hahn et al., 2010), PRISM (Kwiatkowska et al., 2011), and PROPHESY (Dehnert et al., 2015). They turn out to be competitive alternatives to dedicated POMDP solvers. Moreover, as we are solving slightly different problems, our methods are orthogonal to, e. g., PRISM-POMDP (Norman et al., 2017) and solve-POMDP (Walraven and Spaan, 2017).

We detail our contributions and the structure of the paper, which starts with necessary formalisms in Sect. 2. A longer version of the paper (Junges et al., 2017) contains some additional material.

**Section 3:** We establish the correspondence of POMDPs and pMCs, see Tab. 1. The product of a POMDP and an FSC yields a POMDP with state-memory pairs, which we map to states in the pMC. If POMDP states share *observations*, the corresponding pMC states share *parameters* at emanating transitions. A *strategy* of the POMDP corresponds to a *parameter instantiation* in the pMC.

**Section 4:** We show the opposite direction, namely a transformation from pMCs to POMDPs. This result establishes that the synthesis problems for POMDPs and pMCs are equally hard. Technically, we identify the practically relevant class of *simple pMCs*, which coincides with POMDPs under memoryless strategies.

**Section 5:** Typical restrictions on parameter instantiations concern whether parameters may be assigned the probability zero. We discuss effects of such restrictions to the resulting POMDP strategies.

**Section 6:** We evaluate the computation of correct-by-

construction FSCs using pMC synthesis techniques. To that end, we explain how particular parameter synthesis approaches deliver optimal or near-optimal FSCs. Then, we evaluate the approach on a range of typical POMDP benchmarks. We observe that often a small amount of memory suffices. Our approach is competitive to state-of-the-art POMDP solvers and is able to synthesise small, almost-optimal FSCs.

**Related Work.** In addition to the cited works, (Meuleau et al., 1999) uses a branch-&-bound method to find optimal FSCs for POMDPs. A SAT-based approach computes FSCs for qualitative properties (Chatterjee et al., 2016a). For a survey of decidability results and algorithms for broader classes of properties refer to (Chatterjee et al., 2016c,b). Work on parameter synthesis (Hutschenreiter et al., 2017; Filieri et al., 2011) might contain additions to the methods considered here.

## 2 PRELIMINARIES

A *probability distribution* over a finite or countably infinite set  $X$  is a function  $\mu: X \rightarrow [0, 1] \subseteq \mathbb{R}$  with  $\sum_{x \in X} \mu(x) = \mu(X) = 1$ . The set of all distributions on  $X$  is  $Distr(X)$ . The support of a distribution  $\mu$  is  $\text{supp}(\mu) = \{x \in X \mid \mu(x) > 0\}$ . A distribution is *Dirac* if  $|\text{supp}(\mu)| = 1$ .

Let  $V = \{p_1, \dots, p_n\}$  be a finite set of *parameters* over the domain  $\mathbb{R}$  and let  $\mathbb{Q}[V]$  be the set of multivariate polynomials over  $V$ . An *instantiation* for  $V$  is a function  $u: V \rightarrow \mathbb{R}$ . Replacing each parameter  $p$  in a polynomial  $f \in \mathbb{Q}[V]$  by  $u(p)$  yields  $f[u] \in \mathbb{R}$ .

Decision problems can be considered as languages describing all positive instances. A language  $L_1 \subseteq \{0, 1\}^*$  is *polynomial (many-one or Karp) reducible* to  $L_2 \subseteq \{0, 1\}^*$ , written  $L_1 \leq_P L_2$ , if there exists a polynomial-time computable function  $f: \{0, 1\}^* \rightarrow \{0, 1\}^*$  such that for all  $w \in \{0, 1\}^*$ ,  $w \in L_1 \iff f(w) \in L_2$ . Polynomial reductions are essential to define complexity classes, cf. (Papadimitriou, 1994).

### 2.1 PARAMETRIC MARKOV MODELS

**Definition 1 (pMDP)** A parametric Markov decision process (*pMDP*)  $M$  is a tuple  $M = (S, s_1, Act, V, \mathcal{P})$  with a finite (or countably infinite) set  $S$  of states, initial state  $s_1 \in S$ , a finite set  $Act$  of actions, a finite set  $V$  of parameters, and a transition function  $\mathcal{P}: S \times Act \times S \rightarrow \mathbb{Q}[V]$ .

The *available actions* in  $s \in S$  are  $A(s) = \{a \in Act \mid \exists s' \in S : \mathcal{P}(s, a, s') \neq 0\}$ . W.l. o. g. we assume  $\forall s, s' \in$

$S. \forall a \in Act. P(s, a, s') \neq 0 \wedge P(s, a, s') \neq 1 \Rightarrow \exists s'' \neq s'. P(s, a, s'') \neq 0$ . We assume that pMDP  $M$  contains no deadlock states, i. e.  $A(s) \neq \emptyset$  for all  $s \in S$ . A *path* of a pMDP  $M$  is an (in)finite sequence  $\pi = s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} \dots$ , where  $s_0 = s_I$ ,  $s_i \in S$ ,  $a_i \in A(s_i)$ , and  $\mathcal{P}(s_i, a_i, s_{i+1}) \neq 0$  for all  $i \in \mathbb{N}$ . For finite  $\pi$ ,  $\text{last}(\pi)$  denotes the last state of  $\pi$ . The set of (in)finite paths of  $M$  is  $\text{Paths}_{fin}^M$  ( $\text{Paths}^M$ ).

**Definition 2 (MDP)** A Markov decision process (MDP) is a pMDP where  $\mathcal{P}: S \times Act \times S \rightarrow [0, 1] \subseteq \mathbb{R}$  and for all  $s \in S$  and  $a \in A(s)$ ,  $\sum_{s' \in S} \mathcal{P}(s, a, s') = 1$ .

A (*parametric*) *discrete-time Markov chain* ((p)MC) is a (p)MDP with  $|A(s)| = 1$  for all  $s \in S$ . For a (p)MC  $D$ , we may omit the actions and use the notation  $D = (S, s_I, V, P)$  with a transition function  $P$  of the form  $P: S \times S \rightarrow \mathbb{Q}[V]$ .

Applying an *instantiation*  $u: V \rightarrow \mathbb{R}$  to a pMDP or pMC  $M$ , denoted  $M[u]$ , replaces each polynomial  $f$  in  $M$  by  $f[u]$ .  $M[u]$  is also called the *instantiation* of  $M$  at  $u$ . Instantiation  $u$  is *well-defined* for  $M$  if the replacement yields probability distributions, i. e. if  $M[u]$  is an MDP or an MC, respectively.

**Strategies.** To resolve the nondeterministic action choices in MDPs, so-called *strategies* determine at each state a distribution over actions to take. This decision may be based on the *history* of the current path.

**Definition 3 (Strategy)** A strategy  $\sigma$  for (p)MDP  $M$  is a function  $\sigma: \text{Paths}_{fin}^M \rightarrow \text{Distr}(Act)$  s. t.  $\text{supp}(\sigma(\pi)) \subseteq Act(\text{last}(\pi))$  for all  $\pi \in \text{Paths}_{fin}^M$ . The set of all strategies of  $M$  is  $\Sigma^M$ .

A strategy  $\sigma$  is *memoryless* if  $\text{last}(\pi) = \text{last}(\pi')$  implies  $\sigma(\pi) = \sigma(\pi')$  for all  $\pi, \pi' \in \text{Paths}_{fin}^M$ . It is *deterministic* if  $\sigma(\pi)$  is a Dirac distribution for all  $\pi \in \text{Paths}_{fin}^M$ . A strategy that is not deterministic is *randomised*.

A strategy  $\sigma$  for an MDP  $M$  resolves all nondeterministic choices, yielding an *induced Markov chain*  $M^\sigma$ , for which a *probability measure* over infinite paths is defined by the cylinder set construction (Baier and Katoen, 2008).

**Definition 4 (Induced Markov Chain)** For an MDP  $M = (S, s_I, Act, \mathcal{P})$  and a strategy  $\sigma \in \Sigma^M$ , the MC induced by  $M$  and  $\sigma$  is given by  $M^\sigma = (\text{Paths}_{fin}^M, s_I, P^\sigma)$  where:

$$P^\sigma(\pi, \pi') = \begin{cases} \mathcal{P}(\text{last}(\pi), a, s') \cdot \sigma(\pi)(a) & \text{if } \pi' = \pi a s' \\ 0 & \text{otherwise.} \end{cases}$$

## 2.2 PARTIAL OBSERVABILITY

**Definition 5 (POMDP)** A partially observable MDP (POMDP) is a tuple  $\mathcal{M} = (M, Z, O)$ , with  $M = (S, s_I, Act, \mathcal{P})$  the underlying MDP of  $\mathcal{M}$ ,  $Z$  a finite set of observations and  $O: S \rightarrow Z$  the observation function.

We require that states with the same observations have the same set of enabled actions, i. e.  $O(s) = O(s')$  implies  $A(s) = A(s')$  for all  $s, s' \in S$ . We define  $A(z) = A(s)$  if  $O(s) = z$ . More general observation functions (Roy et al., 2005; Shani et al., 2013) take the last action into account and provide a distribution over  $Z$ . There is a transformation of the general case to the POMDP definition used here that blows up the state space polynomially (Chatterjee et al., 2016b). In Fig. 1(a), a fragment of the underlying MDP of a POMDP has two different observations, indicated by the state colouring.

We lift the observation function to paths: For  $\pi = s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} \dots s_n \in \text{Paths}_{fin}^M$ , the associated *observation sequence* is  $O(\pi) = O(s_0) \xrightarrow{a_0} O(s_1) \xrightarrow{a_1} \dots O(s_n)$ . Several paths in the underlying MDP may yield the same observation sequence. Strategies have to take this restricted observability into account.

**Definition 6** An observation-based strategy  $\sigma$  for a POMDP  $\mathcal{M}$  is a strategy for the underlying MDP  $M$  such that  $\sigma(\pi) = \sigma(\pi')$  for all  $\pi, \pi' \in \text{Paths}_{fin}^M$  with  $O(\pi) = O(\pi')$ .  $\Sigma^{\mathcal{M}}$  is the set of observation-based strategies for  $\mathcal{M}$ .

An observation-based strategy selects actions based on observations along a path and the past actions. Applying the strategy to a POMDP yields an induced MC as in Def. 4, resolving all nondeterminism and partial observability. To represent observation-based strategies with finite memory, we define *finite-state controllers* (FSCs). A randomised observation-based strategy for a POMDP  $\mathcal{M}$  with (finite)  $k$  memory is represented by an FSC  $\mathcal{A}$  with  $k$  memory nodes. If  $k = 1$ , the FSC describes a *memoryless strategy*. We often refer to observation-based strategies as FSCs.

**Definition 7 (FSC)** A finite-state controller (FSC) for a POMDP  $\mathcal{M}$  is a tuple  $\mathcal{A} = (N, n_I, \gamma, \delta)$ , where  $N$  is a finite set of memory nodes,  $n_I \in N$  is the initial memory node,  $\gamma$  is the action mapping  $\gamma: N \times Z \rightarrow \text{Distr}(Act)$ , and  $\delta$  is the memory update  $\delta: N \times Z \times Act \rightarrow \text{Distr}(N)$ . The set  $\text{FSC}_k^{\mathcal{M}}$  denotes the set of FSCs with  $k$  memory nodes, called  $k$ -FSCs. Let  $\sigma_{\mathcal{A}} \in \Sigma^{\mathcal{M}}$  denote the observation-based strategy represented by  $\mathcal{A}$ .

From a node  $n$  and the observation  $z$  in the current state of the POMDP, the next action  $a$  is chosen from  $A(z)$

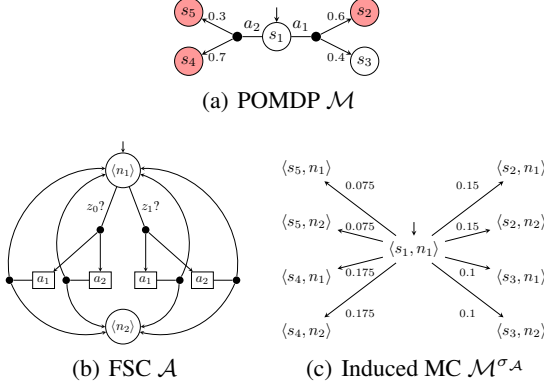


Figure 1: (a) The POMDP  $\mathcal{M}$  with observations  $O(s_1) = O(s_3) = z_0$  (white) and  $O(s_2) = O(s_4) = O(s_5) = z_1$  (red). (b) The associated (partial) FSC  $\mathcal{A}$  has two memory nodes. (c) A part of MC  $\mathcal{M}^{\sigma\mathcal{A}}$  induced by  $\mathcal{M}$  and  $\mathcal{A}$ .

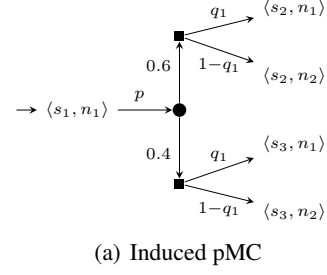
randomly as given by  $\gamma(n, z)$ . Then, the successor node of the FSC is determined randomly via  $\delta(n, z, a)$ .

**Example 1** Fig. 1(b) shows an excerpt of an FSC  $\mathcal{A}$  with two memory nodes. From node  $n_1$ , the action mapping distinguishes observations  $z_0$  and  $z_1$ . The solid dots indicate a probability distribution from  $\text{Distr}(\text{Act})$ . For readability, all distributions are uniform and we omit the action mapping for node  $n_2$ .

Now recall the POMDP  $\mathcal{M}$  from Fig. 1(a). The induced MC  $\mathcal{M}^{\sigma\mathcal{A}}$  is shown in Fig. 1(c). Assume  $\mathcal{M}$  is in state  $s_1$  and  $\mathcal{A}$  in node  $n_1$ . Based on the observation  $z_0 := O(s_1)$ ,  $\sigma_{\mathcal{A}}$  chooses action  $a_1$  with probability  $\delta(n_1, z_0)(a_1) = 0.5$  leading to the probabilistic branching in the POMDP. With probability 0.6,  $\mathcal{M}$  evolves to state  $s_2$ . Next, the FSC  $\mathcal{A}$  updates its memory node; with probability  $\delta(n_1, z_0, a_1)(n_1) = 0.5$ ,  $\mathcal{A}$  stays in  $n_1$ . The corresponding transition from  $\langle s_1, n_1 \rangle$  to  $\langle s_2, n_1 \rangle$  in  $\mathcal{M}^{\sigma\mathcal{A}}$  has probability  $0.5 \cdot 0.6 \cdot 0.5 = 0.15$ .

## 2.3 SPECIFICATIONS

For a POMDP  $\mathcal{M}$ , a set  $G \subseteq S$  of goal states, a set  $B \subseteq S$  of bad states, and a threshold  $\lambda \in [0, 1)$ , we consider quantitative reach-avoid specifications  $\varphi = \mathbb{P}_{>\lambda}(\neg B \cup G)$ . The specification  $\varphi$  is satisfied for a strategy  $\sigma \in \Sigma^{\mathcal{M}}$  if the probability  $\Pr^{\mathcal{M}^{\sigma}}(\neg B \cup G)$  of reaching a goal state in  $\mathcal{M}^{\sigma}$  without entering a bad state in between exceeds  $\lambda$ , denoted by  $\mathcal{M}^{\sigma} \models \varphi$ . The task is to compute such a strategy provided that one exists. For an MDP  $M$ , there is a memoryless deterministic strategy inducing the maximal probability  $\Pr_{\max}^M(\neg B \cup G)$  (Condon, 1992). For a POMDP  $\mathcal{M}$ , however, observation-based strategies with infinite memory as in Def. 6 are



Act	$\mathcal{P}$	Node	Result
$a_1 : p$	0.6	$n_1 : q_1$	$0.6 \cdot p \cdot q_1$
		$n_2 : 1 - q_1$	$0.6 \cdot p \cdot (1 - q_1)$
	0.4	$n_1 : q_1$	$0.4 \cdot p \cdot q_1$
		$n_2 : 1 - q_1$	$0.4 \cdot p \cdot (1 - q_1)$
$a_2 : 1 - p$	0.7	$n_1 : q_2$	$0.7 \cdot (1 - p) \cdot q_2$
		$n_2 : 1 - q_2$	$0.7 \cdot (1 - p) \cdot (1 - q_2)$
	0.3	$n_1 : q_2$	$0.3 \cdot (1 - p) \cdot q_2$
		$n_2 : 1 - q_2$	$0.3 \cdot (1 - p) \cdot (1 - q_2)$

(b) Parameterised transition probabilities

Figure 2: Induced parametric Markov chain for FSCs.

necessary (Ross, 1983) to attain  $\Pr_{\max}^{\mathcal{M}}(\neg B \cup G)$ . The problem of proving the satisfaction of  $\varphi$  is therefore undecidable (Chatterjee et al., 2016c). In our experiments, we also use *undiscounted expected reachability reward properties* (Baier and Katoen, 2008).

## 3 FROM POMDPs TO PMCS

Our goal is to make pMC synthesis methods available for POMDPs. In this section we provide a transformation from a POMDP  $\mathcal{M}$  to a pMC  $D$ . We consider the following decision problems.

**Problem 1 ( $\exists k$ -FSC)** Given a POMDP  $\mathcal{M}$ , a specification  $\varphi$ , and a (unary encoded) memory bound  $k > 0$ , is there a  $k$ -FSC  $\mathcal{A}$  with  $\mathcal{M}^{\sigma\mathcal{A}} \models \varphi$ ?

**Problem 2 ( $\exists \text{INST}$ )** For a pMC  $D$  and a specification  $\varphi$ , does a well-defined instantiation  $u$  exist s.t.  $D[u] \models \varphi$ ?

**Theorem 1**  $\exists k\text{-FSC} \leq_P \exists \text{INST}$ .

The remainder of the section outlines the proof, the converse direction is addressed in Sect. 4. Consider a POMDP  $\mathcal{M}$ , a specification  $\varphi$ , and a memory bound  $k > 0$  for which  $\exists k$ -FSC is to be solved. The degrees of freedom to select a  $k$ -FSC are given by the possible choices for  $\gamma$  and  $\delta$ . For each  $\gamma$  and  $\delta$ , we get a different induced MC, but these MCs are *structurally similar* and can be represented by a single pMC.

**Example 2** Recall Fig. 1 and Ex. 1. The action mapping  $\gamma$  and the memory update  $\delta$  have arbitrary but fixed

probability distributions. For  $a_1$ , we represent the probability  $\gamma(n_1, z_0)(a_1) =: p$  by  $p \in [0, 1]$ . The memory update yields  $\delta(n_1, z_0, a_1)(n_1) =: q_1 \in [0, 1]$  and  $\delta(n_1, z_0, a_1)(n_2) =: 1 - q_1$ , respectively. Fig. 2(a) shows the induced pMC for action choice  $a_1$ . For instance, the transition from  $\langle s_1, n_1 \rangle$  to  $\langle s_2, n_1 \rangle$  is labelled with polynomial  $p \cdot 0.6 \cdot q_1$ .

We collect all polynomials for observation  $z_0$  in Fig. 2(b). The result column describes a parameterised distribution over tuples of states and memory nodes. Thus, instantiations for these polynomials need to sum up to one.

As the next step, we define the pMC that results from combining a  $k$ -FSC with a POMDP. The idea is to assign parameters as arbitrary probabilities to action choices. Each observation has one *remaining action* given by a mapping  $Remain: Z \rightarrow Act$ .  $Remain(z) \in A(z)$  is the action to which, after choosing probabilities for all other actions in  $A(z)$ , the remaining probability is assigned. A similar principle holds for the remaining memory node.

**Definition 8 (Induced pMC for a  $k$ -FSC on POMDPs)**  
Let  $\mathcal{M} = (M, Z, O)$  be a POMDP with  $M = (S, s_I, Act, \mathcal{P})$  and let  $k > 0$  be a memory bound. The induced pMC  $D_{\mathcal{M},k} = (S_{\mathcal{M},k}, s_{I,\mathcal{M},k}, V_{\mathcal{M},k}, P_{\mathcal{M},k})$  is defined by:

- $S_{\mathcal{M},k} = S \times \{0, \dots, k-1\}$ ,  $s_{I,\mathcal{M},k} = \langle s_I, 0 \rangle$ ,
- $V_{\mathcal{M},k} = \{p_a^{z,n} \mid z \in Z, n \in \{0, \dots, k-1\}, a \in A(z), a \neq Remain(z)\} \cup \{q_{a,n'}^{z,n} \mid z \in Z, n, n' \in \{0, \dots, k-1\}, n' \neq k-1, a \in A(z)\}$ ,
- $P_{\mathcal{M},k}(s, s') = \sum_{a \in A(s)} H(s, s', a)$  for all  $s, s' \in S'$ ,

where  $H: S_{\mathcal{M},k} \times S_{\mathcal{M},k} \times Act \rightarrow \mathbb{R}$  is for  $z = O(s)$  defined by  $H(\langle s, n \rangle, \langle s', n' \rangle, a) =$

$$\mathcal{P}(s, a, s') \cdot \left\{ \begin{array}{ll} p_a^{z,n}, & \text{if } a \neq Remain(z) \\ 1 - \sum_{b \neq a} p_b^{z,n}, & \text{if } a = Remain(z) \end{array} \right\} \cdot \left\{ \begin{array}{ll} q_{a,n'}^{z,n}, & \text{if } n' \neq k-1 \\ 1 - \sum_{\bar{n} \neq n'} q_{a,\bar{n}}^{z,n}, & \text{if } n' = k-1 \end{array} \right\}.$$

Intuitively,  $H(s, s', a)$  describes the probability mass from  $s$  to  $s'$  in the induced pMC that is contributed by action  $a$ . The three terms correspond to the terms as seen in the first three columns of Tab. 2(b).

**Example 3** Consider the POMDP in Fig. 3(a) and let  $k = 1$ . The induced pMC is given in Fig. 3(b). The three actions from  $s_0$  have probability  $p_1, p_2$ , and  $1 - p_1 - p_2$  for the remaining action  $a_3$ . From the indistinguishable states  $s_1, s_3$ , actions have probability  $q$  and  $1 - q$ , respectively.

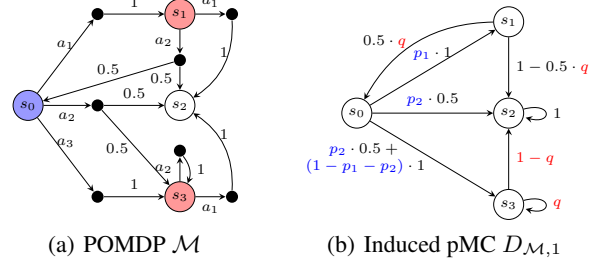


Figure 3: From POMDPs to pMCs ( $k = 1$ )

By construction, the induced pMC describes the set of all induced MCs:

**Theorem 2 (Correspondence Theorem)** For POMDP  $\mathcal{M}$ , memory bound  $k$ , and the induced pMC  $D_{\mathcal{M},k}$ :

$$\{D_{\mathcal{M},k}[u] \mid u \text{ well-defined}\} = \{\mathcal{M}^{\sigma_A} \mid \mathcal{A} \in FSC_k^{\mathcal{M}}\}.$$

In particular, every well-defined instantiation  $u$  describes an FSC  $\mathcal{A}_u \in FSC_k^{\mathcal{M}}$ .

By the correspondence, we can thus evaluate an instantiation of the induced pMC to assess whether the corresponding  $k$ -FSC satisfies a given specification.

**Corollary 1** Given an induced pMC  $D_{\mathcal{M},k}$  and a specification  $\varphi$ : For every well-defined instantiation  $u$  of  $D_{\mathcal{M},k}$  and the corresponding  $k$ -FSC  $\mathcal{A}_u$  we have:

$$\mathcal{M}^{\sigma_{\mathcal{A}_u}} \models \varphi \iff D_{\mathcal{M},k}[u] \models \varphi.$$

The number of parameters in the induced pMC  $D_{\mathcal{M},k}$  is given by  $\mathcal{O}(|Z| \cdot k^2 \cdot \max_{z \in Z} |A(z)|)$ .

## 4 FROM PMCS TO POMDPS (AND BACK AGAIN)

In the previous section we have shown that  $\exists k$ -FSC is at least as hard as  $\exists INST$ . We now discuss whether both problems are equally hard: The open question is whether we can reduce  $\exists INST$  to  $\exists k$ -FSC.

A straightforward reduction maintains the states of the pMC in the POMDP, or even yields a POMDP with the same graph structure (the topology) as the pMC. The next example shows that this naive reduction is impossible.

**Example 4** In the pMC in Fig. 4 the parameter  $p$  occurs in two different distributions (at  $s_0$  and  $s_2$ ). For defining a reduction where the resulting POMDP has the same set of states, there are two options for the observation

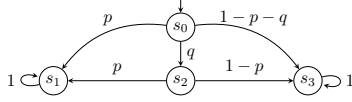


Figure 4: Non-simple pMC

function at the states  $s_0$  and  $s_2$ : Either  $O(s_0) = O(s_2)$  or  $O(s_0) \neq O(s_2)$ . The intuition is that every (parametric) transition in the pMC corresponds to an action choice in a POMDP. Then  $O(s_0) = O(s_2)$  is impossible as  $s_0$  and  $s_2$  have a different number of outgoing transitions (outdegree). Adding a self-loop to  $s_2$  does not alleviate the problem. Moreover,  $O(s_0) \neq O(s_2)$  is impossible, as a strategy could distinguish  $s_0$  and  $s_2$  and assign different probabilities to  $p$ .

The pMC in the example is problematic as the parameters occur at the outgoing transitions of states in different combinations. We restrict ourselves to an important subclass<sup>1</sup> of pMCs which we call *simple pMCs*. A pMC is simple if for all states  $s, s', P(s, s') \in \mathbb{Q} \cup \{p, 1-p \mid p \in V\}$ . Consequently, we can map states to parameters, and use this map to define the observations. Then, the transformation from a POMDP to a pMC is the reverse of the transformation from Def. 8. In the remainder, we detail this correspondence. The correspondence also establishes a construction to compute  $k$ -FSCs via parameter synthesis on *simple* pMCs. Current tool-support (cf. Sect. 6) for simple pMCs is more mature than for the more general pMCs obtained via Def. 8.

Let *simple- $\exists$ INST* be the restriction of  $\exists$ INST to simple pMCs. Similarly, let *simple- $\exists$ 1-FSC* be a variant of  $\exists$ 1-FSC that only considers *simple* POMDPs.

**Definition 9 (Binary/Simple POMDP)** A POMDP is binary, if  $|A(s)| \leq 2$  for all  $s \in S$ . A binary POMDP is simple, if for all  $s \in S$

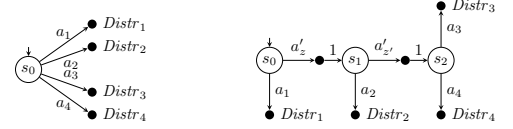
$$|A(s)| = 2 \implies \forall a \in A(s) \exists s' \in S : P(s, a, s') = 1.$$

We establish the following relation between the POMDP and pMC synthesis problems, which asserts that the problems are equivalently hard.

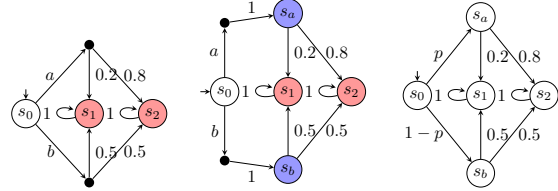
**Theorem 3** For any  $L_1, L_2 \in \{ \exists k\text{-FSC}, \exists 1\text{-FSC}, \text{simple-}\exists 1\text{-FSC}, \text{simple-}\exists \text{INST} \}$ ,  $L_1 \leq_P L_2$ .

The proof is a direct consequence of the Lemmas 1-4 below, as well as the facts that every 1-FSC is a  $k$ -FSC, and every simple POMDP is a POMDP.

<sup>1</sup>All pMC benchmarks from the PARAM webpage (PARAM Website, 2015) are simple pMCs.



(a) Four actions in a POMDP (b) Four actions in a binary POMDP



(c) Binary POMDP (d) Simple POMDP (e) Simple pMC

Figure 5: POMDP  $\leftrightarrow$  simple pMC

The induced pMC  $D_{\mathcal{M},1}$  of a simple POMDP  $\mathcal{M}$  is also simple. Consequently, Sect. 3 yields:

**Lemma 1** *simple- $\exists$ 1-FSC*  $\leq_P$  *simple- $\exists$ INST*.

## 4.1 FROM SIMPLE PMCS TO POMDPS

**Theorem 4** Every simple pMC  $D$  with  $n$  states and  $m$  parameters is isomorphic to  $D_{\mathcal{M},1}$  for some simple POMDP  $\mathcal{M}$  with  $n$  states and  $m$  observations.

We refrain from a formal proof: The construction is the reverse of Def. 8, with observations  $\{z_p \mid p \in V_D\}$ . In a simple pMC, the outgoing transitions are either all parameter free, or of the form  $p, 1-p$ . The parameter-free case is transformed into a POMDP state with a single action (and any observation). The parametric case is transformed into a state with two actions with Dirac-distributions attached. As observation we use  $z_p$ .

**Lemma 2** *simple- $\exists$ INST*  $\leq_P$  *simple- $\exists$ 1-FSC*.

## 4.2 MAKING POMDPS SIMPLE

We present a reduction from  $\exists 1\text{-FSC}$  to *simple- $\exists$ 1-FSC* by translating a (possibly not simple) POMDP into a *binary* POMDP and subsequently into a *simple* POMDP. Examples are given in Fig. 5(a-e). We emphasise that our construction only preserves the expressiveness of 1-FSCs. Details are given in (Junges et al., 2017).

**Lemma 3**  $\exists 1\text{-FSC} \leq_P \text{simple-}\exists 1\text{-FSC}$ .



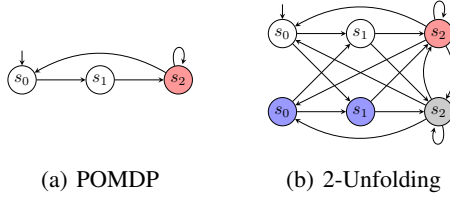


Figure 6: Unfolding a POMDP for two memory states

### 4.3 FROM $k$ -FSCS TO 1-FSCS

For a POMDP  $\mathcal{M}$  and memory bound  $k > 1$ , we construct a POMDP  $\mathcal{M}_k$  such that  $\mathcal{M}$  satisfies a specification  $\varphi$  under some  $k$ -FSC iff  $\mathcal{M}_k$  satisfies  $\varphi$  under some 1-FSC.

**Definition 10 ( $k$ -Unfolding)** Let  $\mathcal{M} = (M, Z, O)$  be a POMDP with  $M = (S, s_I, Act, \mathcal{P})$ , and  $k > 1$ . The  $k$ -unfolding of  $\mathcal{M}$  is the POMDP  $\mathcal{M}_k = (M_k, Z_k, O_k)$  with  $M_k = (S_k, s_{I,k}, Act_k, \mathcal{P}_k)$  defined by:

- $S_k = S \times \{0, \dots, k-1\}$ ,  $s_{I,k} = \langle s_I, 0 \rangle$ ,
- $Act_k = Act \times \{0, \dots, k-1\}$
- $\mathcal{P}_k(\langle s, n \rangle, \langle a, \bar{n} \rangle, \langle s', n' \rangle) = \begin{cases} \mathcal{P}(s, a, s') & n' = \bar{n}, \\ 0 & \text{else.} \end{cases}$

and  $Z_k = Z \times \{0, \dots, k-1\}$ ,  $O_k(\langle s, n \rangle) = \langle O(s), n \rangle$ .

Intuitively,  $\mathcal{M}_k$  stores the current memory node into its state space. At state  $\langle s, n \rangle$  of  $\mathcal{M}_k$ , a 1-FSC can not only choose between the available actions  $A(s)$  in  $\mathcal{M}$  but also between different successor memory nodes.

Fig. 6 shows this process for  $k = 2$ . All states of the POMDP are copied once. Different observations allow to determine in which copy of a state – and therefore, which memory cell – we currently are. Additionally, all actions are duplicated to model the option for a strategy to switch the memory cell.

The induced pMC  $D_{\mathcal{M}_k,1}$  of the  $k$ -unfolding of  $\mathcal{M}$  has the same topology as the induced pMC  $D_{\mathcal{M},k}$  of  $\mathcal{M}$  with memory bound  $k$ . In fact, both pMCs have the same instantiations.

**Proposition 1** For POMDP  $\mathcal{M}$  and memory bound  $k$ :

$$\{D_{\mathcal{M}_k,1}[u] \mid u \text{ well-def.}\} = \{D_{\mathcal{M},k}[u] \mid u \text{ well-def.}\}.$$

The intuition is that in both pMCs the parameter instantiations reflect arbitrary probability distributions over the same set of successor states. In the transition probability function of the induced pMC  $D_{\mathcal{M},k}$  of  $\mathcal{M}$  we can also substitute the multiplications of parameters  $p_a^{z,n}$  and  $q_a^{z,n}$ ,

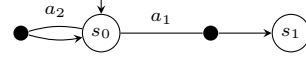


Figure 7: MDP  $\mathcal{M}$

by single parameters. This transformation yields a *substituted induced pMC* which is isomorphic to the induced pMC  $D_{\mathcal{M}_k,1}$  of the  $k$ -unfolding of  $\mathcal{M}$ . Details are given in (Junges et al., 2017).

Proposition 1 and Thm. 2 imply that induced MCs of  $\mathcal{M}$  under  $k$ -FSCs coincide with induced MCs of  $\mathcal{M}_k$  under 1-FSCs:  $\{\mathcal{M}^{\sigma_A} \mid \mathcal{A} \in FSC_k^{\mathcal{M}}\} = \{\mathcal{M}_k^{\sigma_A} \mid \mathcal{A} \in FSC_1^{\mathcal{M}_k}\}$ .

**Lemma 4**  $\exists k\text{-FSC} \leq_P \exists 1\text{-FSC}$ .

## 5 STRATEGY RESTRICTIONS

Two simplifying restrictions on the parameters are usually made in parameter synthesis for pMCs:

- Each transition is assigned a strictly positive probability (*graph-preserving*).
- Each transition is assigned at least probability  $\varepsilon > 0$  ( $\varepsilon$ -*preserving*).

For simple pMCs, the restrictions correspond to parameters instantiations over  $(0, 1)$  or  $[\varepsilon, 1 - \varepsilon]$ , respectively.

Accordingly, we define restrictions to POMDP strategies that correspond to such restricted parameter instantiations.

**Definition 11 (Non-zero Strategies)** A strategy  $\sigma$  is non-zero if  $\sigma(\pi)(a) > 0$  for all  $\pi \in \text{Paths}_{\min}^M$ ,  $a \in A(\text{last}(\pi))$ , and min- $\varepsilon$  if additionally  $\sigma(\pi)(a) \geq \varepsilon > 0$ .

Non-zero strategies enforce  $\text{supp}(\sigma(s)) = A(s)$ . Example 5 shows the impact on reachability probabilities.

**Example 5** The MDP  $M$  in Fig. 7 has a choice between actions  $a_1$  and  $a_2$  at state  $s_0$ . If action  $a_1$  is chosen with probability zero, the probability to reach  $s_1$  from  $s_0$  becomes zero, and the corresponding parameter instantiation is not graph-preserving. Contrarily, if  $a_1$  is chosen with any positive probability, as would be enforced by a non-zero strategy, the probability to reach  $s_1$  is one.

**Proposition 2** Let  $\mathcal{M}$  be a POMDP. An instantiation  $u$  on  $D_{\mathcal{M},1}$  is graph-preserving ( $\varepsilon$ -preserving), iff  $\sigma_{\mathcal{A}_u}$  is non-zero (min- $\varepsilon$ ).

Still, for the considered specifications, we can, w. l. o. g., restrict ourselves to FSCs that induce non-zero strategies.

**Theorem 5** Let  $\mathcal{M}$  be a POMDP,  $k$  a memory bound and  $\varphi = \mathbb{P}_{>\lambda}(\neg B \cup G)$ . Either  $\forall \mathcal{A} \in k\text{-FSC} : \mathcal{M}^{\sigma_{\mathcal{A}}} \not\models \varphi$  or  $\exists \mathcal{A}' \in k\text{-FSC} : \mathcal{M}^{\sigma_{\mathcal{A}'}} \models \varphi$  with  $\sigma_{\mathcal{A}'}$  non-zero.

The statement is shown in (Junges et al., 2017) by considering the corresponding pMC.

## 6 EMPIRICAL EVALUATION

We established the correspondence between the synthesis problems for POMDPs and pMCs. Now, we discuss the available methods for pMC parameter synthesis, and how they may be exploited or adapted to synthesise FSCs. We distinguish three key problems:

1. *Find a correct-by-construction strategy for a POMDP and a specification.* To construct such a strategy, one needs to find a parameter valuation for the pMC that provably satisfies the specification. Most solution techniques focused on pMCs with a few parameters, rendering the problem at hand infeasible. Recently, efficient approaches emerged that are either based on particle swarm optimisation (PSO) (Chen et al., 2013) or on convex optimisation (Amato et al., 2010; Cubuktepe et al., 2017), in particular using quadratically-constrained quadratic programming (QCQP) (Cubuktepe et al., 2018). We employ PSO and QCQP for our evaluation.

2. *Prove that no FSC exists for a POMDP and a specification.* Proving the absence of an FSC with the given memory bound allows us to show  $\varepsilon$ -optimality of a previously synthesised strategy. Two approaches exist: An approximative technique called *parameter lifting* (Quatmann et al., 2016) and a method based on SAT-modulo-theories (SMT) solving (de Moura and Bjørner, 2008).

3. *Provide a closed-form solution for the underlying measure of a specification in form of a function over the induced parameters of an FSC.* The function may be used for further analysis, e. g. of the sensitivity of decisions or parameter values, respectively. To compute this function, all of the parameter synthesis tools PARAM (Hahn et al., 2010), PRISM (Kwiatkowska et al., 2011), Storm (Dehnert et al., 2017), and PROPhESY (Dehnert et al., 2015) employ a technique called *state elimination* (Daws, 2004).

**Implementation and Setup.** We extended the tool Storm (Dehnert et al., 2017) to parse and store POMDPs, and implemented several transformation options to pMCs. Most notably, Storm supports  $k$ -unfolding, the product with several restricted FSCs such as counters that can be incremented at will, and several types of transformation to (simple) pMCs.

We evaluate on a HP BL685C G7 with 48 2GHz cores, a 16GB memory limit, and 1800 seconds time

Table 2: Benchmarks

Id	Name	Tp.	POMDP $\mathcal{M}$			PRISM-POMDP		SolvePOMDP		MDP
			States	Bran.	Obs.	Result	Time	Result	Time	Res
1	NRP (8)	$\mathbb{P}$	125	161	41	[.125, .24]	20		TO	1.0
2	Grid (4)	$\mathbb{E}$	17	62	3	[3.97, 4.13]	1038	4.13	0.4	3.2
3	Netw (3,4,8)	$\mathbb{E}$	2729	4937	361				TO	0.83
4	Crypt (5)	$\mathbb{P}$	4885	11733	890				MO	1.0
5	Maze (2)	$\mathbb{E}$	16	58	8	[5.11, 5.23]	3.9	5.23	16	4.0
6	Load (8)	$\mathbb{E}$	16	28	5	[10.5, 10.5]	1356	10.5	7.6	10.5
7	Slippery (4)	$\mathbb{P}$	17	59	4			0.93	95	1.0

limit. The compared methods are single-threaded. We took the POMDPs from PRISM-POMDP (Norman et al., 2017), and additional maze, load/unload examples from (Meuleau et al., 1999), and a slippery gridworld with traps inspired by (Russell and Norvig, 2010). Table 2 gives details. The specifications (Tp.) are either the minimisation of expected costs from an initial state until reaching a specified target set ( $\mathbb{E}$ ), or the maximisation the probability of reaching from an initial state a target set without hitting a bad state before ( $\mathbb{P}$ ). We list the number of states, branches ( $\sum |A(s)|$ ), and observations in each POMDP. As a baseline, we provide the results and run time of the model-checking tool PRISM-POMDP, and the point-based solver SolvePOMDP (Walraven and Spaan, 2017), obtained with default settings. Both tools compute optimal memory-unbounded strategies and are prototypes. The last column contains the result on the underlying, fully observable MDP. The experiments contain minimal expected rewards, which are analysed by a straightforward extension of maximal reachability probabilities. All pMCs computed are simple pMCs, as PROPhESY typically benefits from the simpler structure. PROPhESY has been invoked with the default set-up.

### 6.1 FINDING STRATEGIES

We evaluate how quickly a strategy that satisfies the specification is synthesised. We vary the threshold used in the specification, as well as the structure of the FSC.

**Results.** We summarise the obtained results in Tab. 3. For each instance (Id), we define three thresholds (Ts), ordered from challenging (i. e. close to the optimum) to less challenging. For different types of FSCs (FSC, F=full, C=counter) and memory bounds ( $k$ ), we obtain pMCs with the given number of states, transitions and parameters. Full-FSCs are fully connected, in counter-FSCs memory state  $m$  is succeeded by either  $m$  or  $m + 1$ . For each threshold (T1, T2, T3), we report the run time of the two methods PSO and QCQP, respectively. T1 is chosen to be nearly optimal for all benchmarks. A dash indicates a combination of memory and threshold that is not realisable according to the results in Sect. 6.2. TO/MO denote violations of the time/memory limit, respectively.

Table 3: Synthesizing strategies

Id	Ts	FSC/k	States	Trans	Pars	T1		T2		T3	
						pso	qcqp	pso	qcqp	pso	qcqp
1	.124/.11/.09	F/1	75	118	8	<1	<1	<1	<1	<1	<1
		F/2	205	420	47	2	<1	2	<1	2	<1
		F/4	921	1864	215	9	2	9	2	10	2
		F/8	3889	7824	911	43	15	42	14	42	14
2	4.15/4.5/5.5	F/1	47	106	3	-	-	-	-	Err	<1
		F/2	183	390	15	7.4	11	4	9	2	<1
		F/4	719	1486	63	TO	64	39	91	14	8
		F/8	2845	5788	255	TO	700	TO	946	254	69
3	9/10/15	F/1	3268	13094	276	TO	TO	TO	43	22	4
		F/2	16004	46153	1783	TO	TO	TO	877	152	28
		C/2	11270	36171	1168	TO	TO	TO	358	100	62
		C/4	27183	82145	2940	TO	MO	TO	MO	476	MO
4	.249/.2/.15	F/1	3366	6534	364	18	25	18	15	18	12
		F/2	25713	51608	3907	330	MO	350	MO	326	MO
5	5.2/15/25	F/1	30	64	8	-	-	TO	TO	<1	TO
		F/2	137	294	49	TO	TO	14	TO	2	TO
		F/4	587	1214	219	93	TO	TO	TO	26	TO
		F/8	2421	4924	919	TO	TO	1034	TO	115	TO
		C/2	99	212	33	TO	TO	3.7	TO	<1	TO
C/4	231	476	81	7	TO	6	TO	3	TO		
6	10.6/10.9/82.5	F/1	16	33	1	-	-	-	-	<1	TO
		F/2	77	160	11	9	TO	6	TO	<1	TO
		F/4	354	721	63	20	TO	21	63	3	TO
7	.929/.928/.927	F/1	87	184	3	TO	TO	<1	1	<1	<1
		F/2	285	592	15	4	TO	4	20	3	22
		F/4	1017	2080	63	76	767	71	205	67	187
		F/8	3825	7744	255	TO	TO	TO	TO	TO	TO

**Evaluation.** Strategies for thresholds which are suboptimal (T3) are synthesised faster. If the memory bound is increased, the number of parameters quickly grows and the performance of the methods degrades. Additional experiments showed that the number of states has only a minor effect on the performance. The simpler FSC topology for a counter alleviates the blow-up of the pMC and is successfully utilised to find strategies for larger instances.

Trivially, a  $k$ -FSC is also a valid  $(k+i)$ -FSC for some  $i \in \mathbb{N}$ . Yet, the larger number of parameters make searching for  $(k+i)$ -FSCs significantly more difficult. We furthermore observe that the performance of PSO and QCQP is incomparable, and both methods have their merits.

Summarising, many of the POMDPs in the benchmarks allow good performance via FSCs with small memory. **We find nearly-optimal, and small, FSCs for POMDP benchmarks with thousands of states within seconds.**

## 6.2 PROVING $\varepsilon$ -OPTIMALITY

We now focus on evaluating how fast pMC techniques prove the absence of a strategy satisfying the specification. In particular, we consider proving that for a specific threshold, no strategy induces a better value. Such a proof allows us to draw conclusions about the ( $\varepsilon$ -)optimality of a strategy synthesised in Sect. 6.1.

**Results.** Table 4(a) shows the run times to prove that for the POMDP in column *Id*, there exists no strategy of type FSC with  $k$  memory that performs better than threshold  $T$ . The row indicated by \* was obtained with SMT. All

Table 4:  $\varepsilon$ -optimality and closed-form computation

(a) Proving absence				(b) Closed-form sol.		
Id	FSC/k	T	time	Id	FSC/k	time
2	F/1	5	<1	1	F/1	<1
3	F/1	5	8	1	F/2	97
3	F/4	5	183	2	F/1	155
4	F/1	0.25	2*	3	F/1	464
5	F/1	10	3	4	F/1	<1
5	F/2	5	TO	5	F/1	116
6	F/1	82	<1	6	F/1	<1
6	F/8	10.5	1	7	F/1	TO
7	F/1	0.94	5			

other results were obtained with parameter lifting.

**Evaluation.** The methods generally prove tight bounds for  $k=1$ . For  $k>1$ , the high number of parameters yields a mixed impression, the performance depends on the benchmark. **We find proofs for non-trivial bounds up to  $k=8$ , even if the pMC has hundreds of parameters.**

## 6.3 CLOSED-FORM SOLUTIONS

**Results.** Table 4(b) indicates running times to compute a closed-form solution, i. e. a rational function that maps  $k$ -FSCs to the induced probability.

**Evaluation.** Closed form computation is limited to small memory bounds. The rational functions obtained vary wildly in their structure. For (4), the result is a constant function which is trivial to analyse, while for (3), we obtained rational functions with roughly one million terms, rendering further evaluation expensive.

## 7 CONCLUSION

This paper connects two active research areas, namely verification and synthesis for POMDPs and parameter synthesis for Markov models. We see benefits for both areas. On the one hand, the rich application area for POMDPs in, e. g. robotics, yields new challenging benchmarks for parameter synthesis and can drive the development of more efficient methods. On the other hand, parameter synthesis tools and techniques extend the state-of-the-art approaches for POMDP analysis. Future work will also concern a thorough investigation of *permissive schedulers*, that correspond to regions of parameter instantiations, in concrete motion planning scenarios.

## References

- Amato, C., Bernstein, D. S., and Zilberstein, S. (2010). Optimizing fixed-size stochastic controllers for POMDPs and decentralized POMDPs. *Autonomous Agents and Multi-Agent Systems*, 21(3):293–320.
- Baier, C. and Katoen, J.-P. (2008). *Principles of Model Checking*. MIT Press.
- Braziunas, D. (2003). POMDP solution methods. *University of Toronto*.
- Chatterjee, K., Chmelik, M., and Davies, J. (2016a). A symbolic SAT-based algorithm for almost-sure reachability with small strategies in POMDPs. In *AAAI*, pages 3225–3232. AAAI Press.
- Chatterjee, K., Chmelik, M., Gupta, R., and Kanodia, A. (2016b). Optimal cost almost-sure reachability in POMDPs. *Artif. Intell.*, 234:26–48.
- Chatterjee, K., Chmelik, M., and Tracol, M. (2016c). What is decidable about partially observable Markov decision processes with  $\omega$ -regular objectives. *Journal of Computer and System Sciences*, 82(5):878–911.
- Chatterjee, K., De Alfaro, L., and Henzinger, T. A. (2004). Trading memory for randomness. In *QEST*. IEEE.
- Chen, T., Hahn, E. M., Han, T., Kwiatkowska, M. Z., Qu, H., and Zhang, L. (2013). Model repair for Markov decision processes. In *TASE*, pages 85–92. IEEE CS.
- Condon, A. (1992). The complexity of stochastic games. *Inf. Comput.*, 96(2):203–224.
- Cubuktepe, M., Jansen, N., Junges, S., Katoen, J., Pappas, I., Poonawala, H. A., and Topcu, U. (2017). Sequential convex programming for the efficient verification of parametric MDPs. In *TACAS (2)*, volume 10206 of *LNCS*, pages 133–150.
- Cubuktepe, M., Jansen, N., Junges, S., Katoen, J.-P., and Topcu, U. (2018). Synthesis in pMDPs: A tale of 1001 parameters. *CoRR*, abs/1803.02884.
- Daws, C. (2004). Symbolic and parametric model checking of discrete-time Markov chains. In *ICTAC*, volume 3407 of *LNCS*, pages 280–294. Springer.
- de Moura, L. M. and Björner, N. (2008). Z3: An efficient SMT solver. In *TACAS*, volume 4963 of *LNCS*, pages 337–340. Springer.
- Dehnert, C., Junges, S., Jansen, N., Corzilius, F., Volk, M., Bruintjes, H., Katoen, J.-P., and Ábrahám, E. (2015). PROPhESY: A probabilistic parameter synthesis tool. In *CAV*, volume 9206 of *LNCS*, pages 214–231. Springer.
- Dehnert, C., Junges, S., Katoen, J., and Volk, M. (2017). A Storm is coming: A modern probabilistic model checker. In *CAV (2)*, volume 10427 of *LNCS*, pages 592–600. Springer.
- Filieri, A., Ghezzi, C., and Tamburrelli, G. (2011). Runtime efficient probabilistic model checking. In *ICSE*, pages 341–350. ACM.
- Hahn, E. M., Hermanns, H., and Zhang, L. (2010). Probabilistic reachability for parametric Markov models. *Software Tools for Technology Transfer*, 13(1):3–19.
- Howard, R. A. (1960). *Dynamic Programming and Markov Processes*. The MIT Press.
- Hutschenreiter, L., Baier, C., and Klein, J. (2017). Parametric Markov chains: PCTL complexity and fraction-free Gaussian elimination. In *GandALF*, volume 256 of *EPTCS*, pages 16–30.
- Junges, S., Jansen, N., Wimmer, R., Quatmann, T., Winterer, L., Katoen, J., and Becker, B. (2017). Permissive finite-state controllers of POMDPs using parameter synthesis. *CoRR*, abs/1710.10294.
- Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artif. Intell.*, 101(1):99–134.
- Kolobov, A., Mausam, and Weld, D. S. (2012). A theory of goal-oriented MDPs with dead ends. In *UAI*, pages 438–447. AUAI Press.
- Kwiatkowska, M., Norman, G., and Parker, D. (2011). PRISM 4.0: Verification of probabilistic real-time systems. In *CAV*, volume 6806 of *LNCS*, pages 585–591. Springer.
- Madani, O., Hanks, S., and Condon, A. (1999). On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems. In *AAAI*, pages 541–548. AAAI Press.
- Meuleau, N., Kim, K.-E., Kaelbling, L. P., and Cassandra, A. R. (1999). Solving POMDPs by searching the space of finite policies. In *UAI*, pages 417–426. Morgan Kaufmann Publishers Inc.
- Norman, G., Parker, D., and Zou, X. (2017). Verification and control of partially observable probabilistic systems. *Real-Time Systems*, 53(3):354–402.
- Papadimitriou, C. H. (1994). *Computational complexity*. Addison-Wesley.
- PARAM Website (2015). <http://depend.cs.uni-sb.de/tools/param/>.
- Puterman, M. L. (1994). *Markov Decision Processes*. John Wiley and Sons.
- Quatmann, T., Dehnert, C., Jansen, N., Junges, S., and Katoen, J. (2016). Parameter synthesis for Markov models: Faster than ever. In *ATVA*, volume 9938 of *LNCS*, pages 50–67. Springer.
- Ross, S. M. (1983). *Introduction to Stochastic Dynamic Programming*. Academic Press, Inc.
- Roy, N., Gordon, G. J., and Thrun, S. (2005). Finding approximate POMDP solutions through belief compression. *J. Artif. Intell. Res.*, 23:1–40.
- Russell, S. J. and Norvig, P. (2010). *Artificial Intelligence – A Modern Approach (3. ed.)*. Pearson Education.
- Shani, G., Pineau, J., and Kaplow, R. (2013). A survey of point-based POMDP solvers. *Autonomous Agents and Multi-Agent Systems*, 27(1):1–51.
- Szer, D. and Charpillet, F. (2005). An optimal best-

- first search algorithm for solving infinite horizon DEC-POMDPs. In *ECML*, pages 389–399. Springer.
- Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics*. The MIT Press.
- Vlassis, N., Littman, M. L., and Barber, D. (2012). On the computational complexity of stochastic controller optimization in POMDPs. *ACM Trans. on Computation Theory*, 4(4):12:1–12:8.
- Walraven, E. and Spaan, M. T. J. (2017). Accelerated vector pruning for optimal POMDP solvers. In *AAAI*, pages 3672–3678. AAAI Press.
- Wongpiromsarn, T. and Frazzoli, E. (2012). Control of probabilistic systems under dynamic, partially known environments with temporal logic specifications. In *CDC*, pages 7644–7651. IEEE.

---

# Identification of Personalized Effects Associated With Causal Pathways

---

**Ilya Shpitser**

Department of Computer Science  
Johns Hopkins University  
Baltimore, MD

**Eli Sherman**

Department of Computer Science  
Johns Hopkins University  
Baltimore, MD

## Abstract

Unlike classical causal inference, where the goal is to estimate average causal effects within a *population*, in settings such as personalized medicine, the goal is to map a unit's characteristics to a treatment tailored to maximize the expected outcome for that unit. Obtaining high-quality mappings of this type is the goal of the dynamic treatment regime literature. In healthcare settings, optimizing policies with respect to a particular causal pathway is often of interest as well. In the context of average treatment effects, estimation of effects associated with causal pathways is considered in the mediation analysis literature.

In this paper, we combine mediation analysis and dynamic treatment regime ideas and consider how unit characteristics may be used to tailor a treatment strategy that maximizes an effect along specified sets of causal pathways. In particular, we define counterfactual responses to such policies, give a general identification algorithm for these counterfactuals, and prove completeness of the algorithm for unrestricted policies. A corollary of our results is that the identification algorithm for responses to policies given in [16] is complete for arbitrary policies.

## 1 INTRODUCTION

Establishing causal relationships between actions and outcomes is fundamental to rational decision-making. The gold standard for establishing causal relationships is the randomized controlled trial (RCT), which may be used to establish average causal effects within a population. Causal inference is a branch of statistics that seeks

to predict effects of RCTs from observational data, where treatment assignment is not randomized. Such data is often gathered from observational studies, surveys given to patients during follow up, and in-hospital electronic medical records.

While average treatment effects reported from implemented RCTs, or hypothetical RCTs emulated by causal inference methods using observational data establish whether a particular action is helpful *on average*, optimal decision making must tailor decisions to specific situations. In the context of causal inference this involves finding a map between characteristics of an experimental unit, such as baseline features, to an action that optimizes some outcome for that unit. Methods for finding such maps are studied in the dynamic treatment regime literature [3], and in off-policy reinforcement learning [2].

If an action is known to have a beneficial effect on some outcome, it is often desirable to understand the causal mechanism behind this effect. A popular type of mechanism analysis is *mediation analysis*, which seeks to decompose average treatment effects into direct and indirect components, or more generally into components associated with specific causal pathways. These components of the average causal effect are known as direct, indirect, and path-specific effects, and are also defined as population averages [1, 8, 12].

In this paper, we define counterfactual outcomes necessary to personalize effects associated with causal pathways, give an algorithm for non-parametric identification of these outcomes and prove that it is complete for arbitrary policies. We consider estimation methods for identified outcomes of this type in a companion paper [7].

### Why Personalize Effects Along Causal Pathways?

It often makes sense to structure decision-making such that the *overall* effect of an action on the outcome is maximized for a given unit. However, in some cases it is ap-

propriate to choose an action such that only a part of the effect of an action on the outcome is maximized. Consider management of HIV patients' care. Since HIV is a chronic disease, care for HIV patients involves designing a long-term treatment plan to minimize the chance of viral failure (an undesirable outcome). In designing such a plan, an important choice is when to initiate primary therapy, and when to switch to a second line therapy. Initiating or switching too early risks unneeded side effects and "wasting" treatment efficacy, while initiating or switching too late risks viral failure [4].

In the context of HIV, however, *treatment adherence* is an important component of the overall effect of the drug on the outcome. Patients who do not take prescribed doses compromise the efficacy of the drug, and different drugs may have different levels of adherence. Thus, for HIV patients, the overall effect of the drug can be viewed as a combination of the chemical effect and the adherence effect [6]. Therefore, choosing an action that maximizes the overall effect of HIV treatment on viral failure entangles these two very different causal mechanisms. One approach to tailoring treatments to patients in a way that disentangles these mechanisms is to find a policy that optimizes a part of the effect, say the chemical (direct) effect of the drug, while hypothetically keeping the adherence levels to some reference level. Finding such a policy yields information on how best to assign drugs to maximize their chemical efficacy in settings where adherence levels can be controlled to that of a reference treatment – even if the only data available is one where patients have differential adherence.

## 2 PRELIMINARIES

We proceed as follows. We first give graph theoretic preliminaries, and define graphical causal models that equate counterfactual responses to interventions (setting variables to values, contrary to fact) with truncated factorizations of the observed data distribution [11]. Next, we describe the more general *edge intervention* that sets variables to different values for different outgoing edges in a graph. Edge interventions are used to formulate direct, indirect, and path-specific effects in mediation analysis. Then, we define counterfactual responses to policies that set variables not to *constant* values but to values that potentially depend on other sets of variables. Extending these notions, we describe counterfactuals that generalize both responses to edge interventions, and responses to policies, namely responses to *edge-specific policies*. We briefly describe identification theory for these counterfactuals in causal models with no hidden variables, and note this theory is based on variations of a truncated factorization known as the g-formula [11].

We next consider identification theory for all counterfactuals we described in hidden variable causal models. This theory is more complex, and is based on the ID algorithm [14, 17]. We rephrase the algorithm and its necessary variations in a single line formula based on the fixing operator described in [10]. This reformulation allows us to express any functional corresponding to a counterfactual distribution identifiable in a hidden variable causal model as a single truncated factorization formula, just as identifiable counterfactual distributions in fully observed models are expressed via the g-formula. Finally, we describe a completeness result for the identification algorithm for responses to unrestricted edge-specific policies in hidden variable causal models.

While our primary contributions lie in the presentation of counterfactuals and identification theory for edge-specific policies, we include some discussion of prior theory to build up to our result, and show how identification theory of edge-specific policies generalizes identification theory for edge-specific effects and policy interventions.

### Graph Theory

We will define statistical and causal models as sets of distributions defined by restrictions associated with graphs. We will use vertices and variables interchangeably – capital letters for a vertex or variable ( $V$ ), bold capital letter for a set ( $\mathbf{V}$ ), lowercase letters for values ( $v$ ), and bold lowercase letters for sets of values ( $\mathbf{v}$ ). By convention, each graph is defined on a vertex set  $\mathbf{V}$ .

For a set of values  $\mathbf{a}$  of  $\mathbf{A}$ , and a subset  $\mathbf{A}^\dagger \subseteq \mathbf{A}$ , define  $\mathbf{a}_{\mathbf{A}^\dagger}$  to be a restriction of  $\mathbf{a}$  to elements in  $\mathbf{A}^\dagger$ . The state space of  $A$  will be denoted by  $\mathfrak{X}_A$ , and the (Cartesian product) state space of  $\mathbf{A}$  will be denoted by  $\mathfrak{X}_{\mathbf{A}}$ .

For a graph mixed graph  $\mathcal{G}$  with directed and bidirected edges, and any  $V \in \mathbf{V}$ , we define the following genealogic sets: parents, children, ancestors, descendants, and districts as:  $\text{pa}_{\mathcal{G}}(V) \equiv \{W \in \mathbf{V} \mid W \rightarrow V\}$ ,  $\text{ch}_{\mathcal{G}}(V) \equiv \{W \in \mathbf{V} \mid V \rightarrow W\}$ ,  $\text{an}_{\mathcal{G}}(V) \equiv \{W \in \mathbf{V} \mid W \rightarrow \dots \rightarrow V\}$ ,  $\text{de}_{\mathcal{G}}(V) \equiv \{W \in \mathbf{V} \mid V \rightarrow \dots \rightarrow W\}$ ,  $\text{dis}_{\mathcal{G}}(V) \equiv \{W \in \mathbf{V} \mid V \leftrightarrow \dots \leftrightarrow W\}$ . By convention,  $\text{an}_{\mathcal{G}}(V) \cap \text{de}_{\mathcal{G}}(V) \cap \text{dis}_{\mathcal{G}}(V) = \{V\}$ . These sets generalize to  $\mathbf{V}^\dagger \subseteq \mathbf{V}$  disjunctively. For example,  $\text{pa}_{\mathcal{G}}(\mathbf{V}^\dagger) \equiv \bigcup_{V \in \mathbf{V}^\dagger} \text{pa}_{\mathcal{G}}(V)$ . For  $\mathbf{A} \subseteq \mathbf{V}$ , define  $\text{pa}_{\mathcal{G}}^{\mathbf{A}}(\mathbf{A}) \equiv \text{pa}_{\mathcal{G}}(\mathbf{A}) \setminus \mathbf{A}$ , the parents of a set  $\mathbf{A}$ .

The non-descendants of  $V$  are denoted  $\text{nd}_{\mathcal{G}}(V) \equiv \mathbf{V} \setminus \text{de}_{\mathcal{G}}(V)$ . The set of districts forms a partition of vertices in  $\mathcal{G}$  and is denoted  $\mathcal{D}(\mathcal{G})$ . Finally, given a graph  $\mathcal{G}$  and  $\mathbf{A} \subseteq \mathbf{V}$ , the subgraph of  $\mathcal{G}$  containing only vertices in  $\mathbf{A}$  and edges between these vertices is denoted  $\mathcal{G}_{\mathbf{A}}$ .

## Statistical And Causal Models Of A Dag

A directed acyclic graph (DAG), or Bayesian network, is a graph  $\mathcal{G}$  with vertex set  $\mathbf{V}$  connected by directed edges and such that there are no directed cycles in the graph (i.e. no sequences of edges and vertices  $V \rightarrow \dots W$  and edge  $W \rightarrow V$ ). A statistical model of a DAG  $\mathcal{G}$  is the set of distributions  $p(\mathbf{V})$  such that  $p(\mathbf{V}) = \prod_{V \in \mathbf{V}} p(V | \text{pa}_{\mathcal{G}}(V))$ . Such a  $p(\mathbf{V})$  is said to be Markov relative to  $\mathcal{G}$ .

Causal models of a DAG are also sets of distributions, but on counterfactual random variables. Given  $Y \in \mathbf{V}$  and  $\mathbf{A} \subseteq \mathbf{V} \setminus \{Y\}$ , a counterfactual variable, or ‘potential outcome’, written as  $Y(\mathbf{a})$ , represents the value of  $Y$  in a hypothetical situation where  $\mathbf{A}$  were set to values  $\mathbf{a}$  by an *intervention operation* [9]. Given a set  $\mathbf{Y}$ , define  $\mathbf{Y}(\mathbf{a}) \equiv \{\mathbf{Y}\}(\mathbf{a}) \equiv \{Y(\mathbf{a}) \mid Y \in \mathbf{Y}\}$ . The distribution  $p(\mathbf{Y}(\mathbf{a}))$  is sometimes written as  $p(\mathbf{Y}|\text{do}(\mathbf{a}))$  [9].

Causal models of a DAG  $\mathcal{G}$  consist of distributions defined on counterfactual random variables of the form  $V(\mathbf{a})$  where  $\mathbf{a}$  are values of  $\text{pa}_{\mathcal{G}}(V)$ . In this paper we assume Pearl’s functional model for a DAG  $\mathcal{G}$  with vertices  $\mathbf{V}$  which is the set containing any joint distribution over all potential outcome random variables where the sets of variables

$$\{\{V(\mathbf{a}_V) \mid \mathbf{a}_V \in \mathfrak{X}_{\text{pa}_{\mathcal{G}}(V)}\} \mid V \in \mathbf{V}\}$$

are mutually independent [9]. The *atomic counterfactuals* in the above set model the relationship between  $\text{pa}_{\mathcal{G}}(V)$ , representing direct causes of  $V$ , and  $V$  itself. From these, all other counterfactuals may be defined using recursive substitution. For any  $\mathbf{A} \subseteq \mathbf{V} \setminus \{V\}$ ,

$$V(\mathbf{a}) \equiv V(\mathbf{a}_{\text{pa}_{\mathcal{G}}(V) \cap \mathbf{A}}, \{\text{pa}_{\mathcal{G}}(V) \setminus \mathbf{A}\}(\mathbf{a})). \quad (1)$$

For example, in the DAG in Fig. 1 (a),  $Y(a)$  is defined to be  $Y(a, M(a, W), W)$ .

A causal parameter is said to be *identified* in a causal model if it is a function of the observed data distribution  $p(\mathbf{V})$ . Otherwise the parameter is said to be *non-identified*. In all causal models of a DAG  $\mathcal{G}$ , all interventional distributions  $p(\{\mathbf{V} \setminus \mathbf{A}\}(\mathbf{a}))$  are identified by the *g-formula* [11]:

$$p(\{\mathbf{V} \setminus \mathbf{A}\}(\mathbf{a})) = \prod_{V \in \mathbf{V} \setminus \mathbf{A}} p(V | \text{pa}_{\mathcal{G}}(V)) \Big|_{\mathbf{A}=\mathbf{a}} \quad (2)$$

Not all interventional distributions are identified when there are hidden variables present in the causal model. We discuss identification theory in hidden variable DAGs later in this paper.

### Edge Interventions

A more general type of intervention in a graphical causal model is the *edge intervention* [15], which maps a set

of directed edges in  $\mathcal{G}$  to values of their source vertices. Edge interventions have a natural interpretation in cases where a treatment variable has multiple components that a) influence the outcome in different ways, b) occur or do not occur together in observed data, and c) may in principle be intervened on separately. For instance, smoking leads to poor health outcomes due to two components: smoke inhalation and exposure to nicotine. A smoker would be exposed to both of these components, while a non-smoker to neither. However, one might imagine exposing someone selectively only to nicotine but not smoke inhalation (via a nicotine patch), or only smoke inhalation but not nicotine (via smoking plant matter not derived from tobacco leaves). These types of hypothetical experiments correspond precisely to edge interventions, and have been used to conceptualize direct and indirect effects [8, 12], often on the mean difference scale.

Formally, we will write the mapping of a set of edges to values of their source vertices using the following shorthand:  $(a_1 W_1)_{\rightarrow}, (a_2 W_2)_{\rightarrow}, \dots, (a_k W_k)_{\rightarrow}$  to mean that edge  $(A_1 W_1)_{\rightarrow}$  is assigned to value  $a_1$ ,  $(A_2 W_2)_{\rightarrow}$  is assigned to value  $a_2$ , and so on until  $(A_k W_k)_{\rightarrow}$  is assigned to value  $a_k$ . Alternatively, we will write  $\mathbf{a}_{\alpha}$  to mean edges in  $\alpha$  are mapped to values in the *multiset*  $\mathbf{a}$  (since multiple edges may share the same source vertex, and be assigned to different values). For a subset  $\beta \subseteq \alpha$ , and an assignment  $\mathbf{a}_{\alpha}$  denote  $\mathbf{a}_{\beta}$  to be a restriction of  $\mathbf{a}_{\alpha}$  to edges in  $\beta$ .

We will write counterfactual responses to edge interventions as  $Y(\mathbf{a}_{\alpha})$  or, for simple cases, as:  $Y((aY)_{\rightarrow}, (a'M)_{\rightarrow})$  meaning the response to  $Y$  where  $A$  is set to value  $a$  for the purposes of the edge  $(AY)_{\rightarrow}$  and to  $a'$  for the purposes of the edge  $(AM)_{\rightarrow}$ . An edge intervention that sets a set of edges  $\alpha$  to values in the multiset  $\mathbf{a}$  is defined via the following generalization of recursive substitution (1):

$$Y(\mathbf{a}_{\alpha}) \equiv Y(\mathbf{a}_{\{(ZY)_{\rightarrow} \in \alpha\}}, \{\text{pa}_{\mathcal{G}}^{\bar{\alpha}}(Y)\}(\mathbf{a}_{\alpha})), \quad (3)$$

where  $\text{pa}_{\mathcal{G}}^{\bar{\alpha}}(Y) \equiv \{W \mid (WY)_{\rightarrow} \notin \alpha\}$ . For example, in the DAG in Fig. 1 (a),  $Y((a'Y)_{\rightarrow}, (aM)_{\rightarrow})$  is defined as  $Y(a', M(a, W), W)$ .

For simplicity of presentation, we will restrict attention to edge interventions with the property that if  $(AW)_{\rightarrow} \in \alpha$ , then for any  $V \in \text{ch}_{\mathcal{G}}(A)$ ,  $(AV)_{\rightarrow} \in \alpha$ . These types of edge interventions set values for all causal pathways for a set of treatment variables. This is the convention in the majority of existing mediation literature as these interventions are most relevant in practical mediation analysis problems. Specifically, in our HIV example, we are interested in the effect of a drug along all pathways that start with a particular edge, while the effect of the drug via pathways that begin with other edges is kept to a reference level. This assumption may be relaxed, at the price of complicating the theory [15].



Edge interventions are used to define direct and indirect effects. For example, in the model given by the DAG in Fig 1 (a), the direct effect of  $A$  on  $Y$  is defined as  $\mathbb{E}[Y((aY)_{\rightarrow}, (aM)_{\rightarrow})] - \mathbb{E}[Y((a'Y)_{\rightarrow}, (aM)_{\rightarrow})]$  which is equal to  $\mathbb{E}[Y(a)] - \mathbb{E}[Y(a', M(a))]$ . The indirect effect may be defined similarly as  $\mathbb{E}[Y((a'Y)_{\rightarrow}, (aM)_{\rightarrow})] - \mathbb{E}[Y((a'Y)_{\rightarrow}, (a'M)_{\rightarrow})]$ , which is equal to  $\mathbb{E}[Y(a', M(a))] - \mathbb{E}[Y(a')]$ . The direct and indirect effects add up to the ACE.

Note that while direct, indirect, and path-specific effects may be defined directly as nested counterfactuals [8, 13], this notation quickly becomes unreadable for complicated interventions applied at multiple time points. The edge intervention notation may be viewed as a generalization of the  $\text{do}(\cdot)$  operator notation of Pearl to mediation problems, which avoids having to specify the entire nested counterfactual, and instead directly ties interventions and sets of causal pathways to which these interventions apply (as represented by the first edge shared by all pathways in the set).

Identification of edge interventions in graphical causal models without hidden variables corresponds quite closely with identification of regular (node) interventions, as follows. Let  $\mathbf{A}_\alpha \equiv \{A \mid (AB)_{\rightarrow} \in \alpha\}$ . Consider an edge intervention given by the mapping  $\alpha_\alpha$ . Then, under the functional model of a DAG  $\mathcal{G}$ , the joint distribution of counterfactual responses  $p(\{\mathbf{V} \setminus \mathbf{A}_\alpha\}(\alpha_\alpha))$  is identified via the the following generalization of (2) called the *edge g-formula*:

$$\prod_{V \in \mathbf{V} \setminus \mathbf{A}_\alpha} p(V \mid \alpha_{\{(ZV)_{\rightarrow} \in \alpha\}}, \text{pa}_{\mathcal{G}}^{\bar{\alpha}}(V)). \quad (4)$$

For example, in Fig 1 (a),  $p(Y((aY)_{\rightarrow}, (a'M)_{\rightarrow})) = \sum_{W, M} p(Y \mid a, M, W) p(M \mid a', W) p(W)$ , which is obtained by marginalizing  $W, M$  from the edge g-formula.

Edge interventions represent a special case of the more general notion of a *path intervention* [15]. Responses to both of these interventions are used to define *path-specific effects* [8], however responses to edge interventions are precisely those that are always identified under the functional model of a DAG, via (3). Responses to path interventions that cannot be rephrased as responses to edge interventions are not identified even in a DAG model, including the functional model, due to the presence of *recanting witnesses* [1]. For this reason, in this paper we restrict attention only to edge interventions and responses to edge-specific policies.

### Responses To Treatment Policies

In personalized medicine settings, counterfactual responses to conditional interventions that set treatment values in response to other variables via a known function are of interest. As an example, assume the graph

in Fig. 1 (b) represents an observational study of cancer patients where  $W_0$  represents baseline patient metrics,  $A_1$  is the primary therapy,  $W_1$  is the measured intermediate response to the primary therapy,  $A_2$  is a decision to either continue primary therapy or switch to a secondary therapy in the event of a poor response to  $A_1$ , and  $W_2$  is the outcome of interest. In this setting, we might be interested in evaluating policies in the set  $\{f_{A_1} : \mathfrak{X}_{W_0} \mapsto \mathfrak{X}_{A_1}, f_{A_2} : \mathfrak{X}_{\{W_0, W_1\}} \mapsto \mathfrak{X}_{A_2}\}$  that map patient characteristics to decisions about therapies  $A_1$  and  $A_2$ . We evaluate the efficacy of these policies via the counterfactual variable  $W_2(f_{A_1}, f_{A_2})$ , representing patient outcomes had treatment decisions been made according to those policies.

These types of variables are defined via a generalization of (1), where instead of setting values of parents in  $A_1, A_2$  to values fixed by the intervention, values of parents in  $A$  are instead set according to  $f_{A_1}$  and  $f_{A_2}$ . In particular,  $W_2(f_{A_1}, f_{A_2})$  is defined as

$$W_2[f_{A_2}(W_1[f_{A_1}(W_0), W_0], W_0), W_1[f_{A_1}(W_0), W_0], f_{A_1}(W_0), W_0]. \quad (5)$$

The distribution of this variable is identified under the functional model via the natural generalization of (2) as

$$\sum_{W_0, W_1} p(W_2 \mid W_0, f_{A_1}(W_0), W_1, f_{A_2}(W_0, W_1)) \times p(W_1 \mid W_0, f_{A_1}(W_0)) p(W_0). \quad (6)$$

More generally, given a DAG  $\mathcal{G}$ , a topological ordering  $\prec$ , and a set  $\mathbf{A} \subseteq \mathbf{V}$ , for each  $A \in \mathbf{A}$ , define  $\mathbf{W}_A$  to be some subset of predecessors of  $A$  according to  $\prec$ . Then, given a set of functions  $\mathbf{f}_A$  of the form  $f_A : \mathfrak{X}_{\mathbf{W}_A} \mapsto \mathfrak{X}_A$ , define  $Y(\mathbf{f}_A)$ , the counterfactual response  $Y \in \mathbf{V}$  to  $\mathbf{A}$  being intervened on via  $\mathbf{f}_A \equiv \{f_A \mid A \in \mathbf{A}\}$ , as

$$Y(\{f_A(\mathbf{W}_A(\mathbf{f}_A)) \mid A \in \text{pa}_{\mathcal{G}}(Y) \cap \mathbf{A}\}, \{\text{pa}_{\mathcal{G}}(Y) \setminus \mathbf{A}\}(\mathbf{f}_A)). \quad (7)$$

In a functional model of a DAG  $\mathcal{G}$ , the effect of  $\mathbf{f}_A$  on the set of variables not being intervened upon,  $\mathbf{V} \setminus \mathbf{A}$ , represented by the distribution  $p(\{\mathbf{V} \setminus \mathbf{A}\}(\mathbf{f}_A))$ , is identified by the following modification of (2) [16]:

$$\prod_{V \in \mathbf{V} \setminus \mathbf{A}} p(V \mid \{f_A(\mathbf{W}_A)\} \mid A \in \mathbf{A} \cap \text{pa}_{\mathcal{G}}(V), \text{pa}_{\mathcal{G}}(V) \setminus \mathbf{A}). \quad (8)$$

## 3 EDGE-SPECIFIC POLICIES

We now give a general definition of counterfactual responses to edge-specific policies that generalize both responses to edge interventions (where a variable is set to

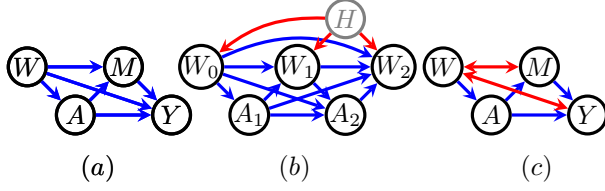


Figure 1: (a) A simple causal DAG, with a treatment  $A$ , an outcome  $Y$ , a vector  $W$  of baseline variables, and a mediator  $M$ . (b) A more complex causal DAG with two treatments  $A_1, A_2$ , an intermediate outcome  $W_1$ , and the final outcome  $W_2$ .  $H$  is a hidden common cause of the  $W$  variables. (c) A graph where  $p(Y(a, M(a)))$  is identified, but  $p(Y(f_A(W), M(a)))$  is not.

different constants for different outgoing edges) and responses to policies, where a variable is set according to a single known function for all causal pathways at once.

As an example, we can view Fig. 1 (a) as representing a cross-sectional study of HIV patients of the kind described in [6], where  $W$  is a set of baseline characteristics,  $A$  is one of a set of possible antiretroviral treatments,  $M$  is adherence to treatment, and  $Y$  is a binary outcome variable signifying viral failure. In this type of study, we may wish to find  $f_A(W)$  that maximizes the expected outcome  $Y$  had  $A$  been set according to  $f_A(W)$  for the purposes of the direct effect of  $A$  on  $Y$ , and  $A$  were set to some reference level  $a$  for the purposes of the effect of  $A$  on  $M$ . In other words, we may wish to find  $f_A(W)$  to maximize the counterfactual mean  $\mathbb{E}[Y(f_A(W), M(a, W), W)]$ . This would correspond to finding a treatment policy that maximizes the direct (chemical) effect, if it were possible to keep adherence to a level  $M(a)$  as if a reference (easy to adhere to) treatment  $a$  were given.

We now give a general definition for responses to such edge-specific policies. Fix a set of directed edges  $\alpha$ , and define  $\mathbf{A}_\alpha \equiv \{A \mid (AB)_{\rightarrow} \in \alpha\}$ . As before, we assume if  $(AW)_{\rightarrow} \in \alpha$ , then for all  $V \in \text{ch}_{\mathcal{G}}(A)$ ,  $(AV)_{\rightarrow} \in \alpha$ . Define  $\mathfrak{f}_\alpha \equiv \{f_A^{(AW)_{\rightarrow}} : \mathfrak{X}_{\mathbf{W}_A} \mapsto \mathfrak{X}_A \mid (AW)_{\rightarrow} \in \alpha\}$  as the set of policies associated with edges in  $\alpha$ . Note that  $\mathfrak{f}_\alpha$  may contain multiple policies for a given treatment variable  $A$ .

Define  $Y(\mathfrak{f}_\alpha)$ , the counterfactual response of  $Y$  to the set of edge-specific policies  $\mathfrak{f}_\alpha$ , as the following generalization of (3) and (7):

$$Y(\{f_A^{(AY)_{\rightarrow}}(\mathbf{W}_A(\mathfrak{f}_\alpha)) \mid (AY)_{\rightarrow} \in \alpha\}, \{\text{pa}_{\mathcal{G}}^\alpha(Y)\}(\mathfrak{f}_\alpha)) \quad (9)$$

In our earlier example, if  $\mathfrak{f}_{\{(AY)_{\rightarrow}, (AM)_{\rightarrow}\}} \equiv \{f_A^{(AY)_{\rightarrow}}(W), \tilde{f}_A^{(AM)_{\rightarrow}}\}$ , where  $\tilde{f}_A$  assigns  $A$  to a constant value  $a$ , then  $Y(\mathfrak{f}_{\{(AY)_{\rightarrow}, (AM)_{\rightarrow}\}}) \equiv Y(f_A(W), M(a, W), W)$ .

The joint counterfactual distribution for responses to edge-specific policies,  $p(\{V(\mathfrak{f}_\alpha) \mid V \in \mathbf{V} \setminus \mathbf{A}_\alpha\})$ , is identified under the functional model, and generalizes (4) and (6) as follows:

$$\prod_{V \in \mathbf{V} \setminus \mathbf{A}_\alpha} p(V \mid \{f_A^{(AV)_{\rightarrow}}(\mathbf{W}_A) \mid (AV)_{\rightarrow} \in \alpha\}, \text{pa}_{\mathcal{G}}^\alpha(V)). \quad (10)$$

This is a consequence of the fact that (4) holds regardless of how edge interventions are set. In Fig. 1 (a), for example,  $p(Y(f_A(W), M(a, W), W)) = \sum_{W, M} p(Y \mid f_A(W), M, W) p(M \mid a, W) p(W)$ .

## 4 IDENTIFICATION IN HIDDEN VARIABLE DAG MODELS

In a causal model of a DAG where some variables are hidden, not every causal parameter is a function of the observed data distribution. It is well known, however, that any two hidden variable DAGs which share a special mixed graph called a *latent projection* [9] share identification theory (see [10] for a proof).

Given a DAG  $\mathcal{G}(\mathbf{V} \cup \mathbf{H})$ , where  $\mathbf{V}$  are observed and  $\mathbf{H}$  are hidden variables, define a latent projection  $\mathcal{G}(\mathbf{V})$  to be an acyclic directed mixed graph (ADMG) with the vertex set  $\mathbf{V}$  and  $\rightarrow$  and  $\leftrightarrow$  edges. An edge  $A \rightarrow B$  exists in  $\mathcal{G}(\mathbf{V})$  if there is a directed path from  $A$  to  $B$  in  $\mathcal{G}(\mathbf{V} \cup \mathbf{H})$  with all intermediate vertices in  $\mathbf{H}$ . Similarly, an edge  $A \leftrightarrow B$  exists in  $\mathcal{G}(\mathbf{V})$  if there is a path without consecutive edges  $\rightarrow \circ \leftarrow$  from  $A$  to  $B$  with the first edge on the path of the form  $A \leftarrow$  and the last edge on the path of the form  $\rightarrow B$ , and all intermediate vertices on the path in  $\mathbf{H}$ . For example, the graph in Fig. 2 (b) is the latent projection of Fig. 2 (a).

We will describe identification results on latent projections directly. General algorithms for identification of interventional distributions were given in [14, 17], for responses to edge interventions in [13], and for policies in [16]. Here we reformulate these results as one line formulas using the fixing operator described in [10]. We do so to explicate the connection between these earlier results, and our new identification algorithm.

### Reformulation Of The ID Algorithm

A complete algorithm, called the ID algorithm, for identifying interventional distributions of the form  $p(\mathbf{Y} \mid \text{do}(\mathbf{a}))$ , or  $p(\mathbf{Y}(\mathbf{a}))$ , for  $\mathbf{Y} \subseteq \mathbf{V} \setminus \mathbf{A}$  was given in [17] and simplified in [14]. We now illustrate how this algorithm may be further simplified into a one line formula, which can be viewed as a generalization of the g-formula from the fully observed DAG to the hidden variable DAG case. We then show how this formula may

be generalized appropriately to yield identification algorithms for edge interventions, and edge-specific policies in hidden variable causal models, just as g-formula was generalized to these cases in fully observed DAGs.

The version of the ID algorithm in [14], shown in Fig. 1 in the Appendix, proceeds as follows. Lines 2 and 3 reformulate the original query  $p(\mathbf{Y}(\mathbf{a}))$  as  $\sum_{\mathbf{Y}^* \setminus \mathbf{Y}} p(\mathbf{Y}^*(\mathbf{a}^*))$ , where  $\mathbf{Y}^*, \mathbf{A}^*$  partition  $\text{an}_{\mathcal{G}}(\mathbf{Y})$ , and  $\mathbf{Y}^* \equiv \text{an}_{\mathcal{G}_{\mathbf{V} \setminus \mathbf{A}}}(\mathbf{Y})$ . In line 4, the distribution  $p(\mathbf{Y}^*(\mathbf{a}^*))$  is factorized into terms corresponding to districts  $\mathbf{D}$  in the subgraph  $\mathcal{G}_{\mathbf{Y}^*}$ , with the ID algorithm called recursively on each term. These terms correspond to interventional distributions  $p(\mathbf{D} \mid \text{do}(\mathbf{V} \setminus \mathbf{D} = \mathbf{c}_{\mathbf{V} \setminus \mathbf{D}}))$ , where  $\mathbf{c}_{\mathbf{V} \setminus \mathbf{D}}$  is any set of values of  $\mathbf{V} \setminus \mathbf{D}$  consistent with  $\mathbf{a}$ . In subsequent recursive calls, lines 2, 6 and 7 are iterated for each term until it is identified, or the failure condition is reached. Here line 2 corresponds to marginalizing out irrelevant variables, and lines 6 and 7 correspond to identifying a part of the set of intervened on variables in  $\mathbf{V} \setminus \mathbf{D}$  via the g-formula.

Consider Fig. 2 (b), where  $A$  represents a binary treatment,  $Y$  an outcome of interest,  $W_0$  a vector of baseline confounding factors, and  $M, W_1$  variables mediating the causal effect of  $A$  on  $Y$ . We are interested in identifying the counterfactual distribution  $p(Y(a))$  as a function of the observed data distribution  $p(W_0, A, M, W_1, Y)$ . Here  $\text{an}_{\mathcal{G}}(Y) = \{Y, M, W_1, W_0, A\}$  is partitioned into  $\mathbf{Y}^* \equiv \{Y, M, W_1, W_0\}$  and  $\mathbf{A}^* \equiv \{A\}$ , with  $\mathcal{G}_{\mathbf{Y}^*}$  shown in Fig. 2 (c). There are three districts in this graph,  $\{W_0, M\}$ ,  $\{W_1\}$ , and  $\{Y\}$ . Thus, the ID algorithm attempts to identify  $p(W_0, M \mid \text{do}(w_1, y, a))$ ,  $p(W_1 \mid \text{do}(w_0, m, y, a))$  and  $p(Y \mid \text{do}(w_0, m, w_1, a))$ .

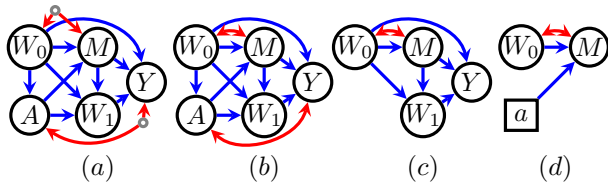


Figure 2: (a) A causal model with a treatment  $A$  and outcome  $Y$ . (b) A latent projection of the DAG in (a). (c) The graph derived from (b) corresponding to  $\mathcal{G}_{\mathbf{Y}^*} = \mathcal{G}_{\{Y, M, W_1, W_0\}}$ . (d) A CADMG corresponding to  $p(M, W_0 \mid \text{do}(a))$ .

As an example, identifying  $p(W_0, M \mid \text{do}(w_1, y, a))$  entails the following steps. First,  $Y$  and  $W_1$ , as irrelevant variables that do not cause  $W_0$  and  $M$ , are marginalized out via line 2, leading to a subproblem where  $p(W_0, M \mid \text{do}(a))$  is identified from  $p(W_0, A, M)$  with the subgraph corresponding to this subproblem shown in Fig. 2 (d). In this subproblem,  $p(W_0, M \mid \text{do}(a))$  is iden-

tified as  $p(M \mid a, W_0)p(W_0)$  via the g-formula in line 6. The recursion alternates steps that marginalize and apply the g-formula can be unified via a fixing operator applied to graphs and distributions that arise in the intermediate steps of the ID algorithm. We now define these graphs and distributions formally.

## CADMGs And Kernels

A *kernel*  $q_{\mathbf{V}}(\mathbf{V} \mid \mathbf{W})$  is a mapping from  $\mathfrak{X}_{\mathbf{W}}$  to normalized densities over  $\mathbf{V}$ . Conditioning and marginalization are defined in kernels in the usual way:

$$q_{\mathbf{V}}(\mathbf{A} \mid \mathbf{W}) \equiv \sum_{\mathbf{V} \setminus \mathbf{A}} q_{\mathbf{V}}(\mathbf{V} \mid \mathbf{W}); \quad q_{\mathbf{V}}(\mathbf{V} \setminus \mathbf{A} \mid \mathbf{A} \cup \mathbf{W}) \equiv \frac{q_{\mathbf{V}}(\mathbf{V} \mid \mathbf{W})}{q_{\mathbf{V}}(\mathbf{A} \mid \mathbf{W})},$$

for  $\mathbf{A} \subseteq \mathbf{V}$ . A conditional distribution is one type of kernel, but others are possible. The functional  $p(M \mid a, W_0)p(W_0) = p(W_0, M \mid \text{do}(a))$  in the previous example is a kernel,  $q(M, W_0 \mid a)$ , that is not in general equal to the conditional distribution  $p(M, W_0 \mid a)$ .

A conditional ADMG (CADMG)  $\mathcal{G}(\mathbf{V}, \mathbf{W})$  is a type of ADMG where nodes are partitioned into two sets. The set  $\mathbf{W}$  corresponds to *fixed* constants, and the set  $\mathbf{V}$  corresponds to *random variables*. A CADMG has the property that no edges with an arrowhead into an element of  $\mathbf{W}$  may exist. Intuitively, a CADMG represents a situation where some variables have already been intervened on. Pearl introduced a similar concept called the ‘mutilated graph’ in [9]. For example, the graph in Fig. 2 (d) is a CADMG  $\mathcal{G}(\{W_0, M\}, \{A\})$  corresponding to the situation where  $W_0, M$  are random variables and  $A$  is fixed to a constant. Just as a distribution may be associated with a DAG via factorization, so may a kernel be associated with a CADMG in a particular way [10]. For example, the CADMG in Fig. 2 (d) may be associated with  $p(W_0, M \mid \text{do}(a)) = p(M \mid a, W_0)p(W_0)$ . Genealogic definitions, such as  $\text{pa}_{\mathcal{G}}(\cdot)$ , carry over identically to CADMGs. Districts in a CADMG are defined as subsets of  $\mathbf{V}$ .

## The Fixing Operator And The ID Algorithm

Given a CADMG  $\mathcal{G}(\mathbf{V}, \mathbf{W})$ , a variable  $V \in \mathbf{V}$  is *fixable* if  $\text{deg}_{\mathcal{G}}(V) \cap \text{dis}_{\mathcal{G}}(V) = \emptyset$ . For example, in Fig. 2 (b),  $M$  is fixable, while  $W_0$  is not. Intuitively,  $V$  is fixable in a CADMG  $\mathcal{G}(\mathbf{V}, \mathbf{W})$  if, in a causal graph representing a hypothetical situation  $p(\mathbf{V} \mid \text{do}(\mathbf{w}))$ , where variables in  $\mathbf{W}$  were already intervened on,  $p(\mathbf{V} \setminus \{V\} \mid \text{do}(\mathbf{w}, v))$  is identified by the application of the g-formula to  $p(\mathbf{V} \mid \text{do}(\mathbf{w}))$ . Whenever a variable  $V$  is fixable, a fixing operator may be applied to both the CADMG and the kernel to yield a new causal graph and a new kernel representing the situation where  $V$  is also intervened on.

Given  $V \in \mathbf{V}$  fixable in a CADMG  $\mathcal{G}(\mathbf{V}, \mathbf{W})$ , the fixing operator  $\phi_V(\mathcal{G})$  yields a new CADMG  $\tilde{\mathcal{G}}(\mathbf{V} \setminus \{V\}, \mathbf{W} \cup \{V\})$ , where all vertices and edges in  $\mathcal{G}(\mathbf{V}, \mathbf{W})$  are kept, *except*  $V$  is viewed as fixed, and all edges with arrowheads into  $V$  are removed. Given  $V \in \mathbf{V}$  fixable in a CADMG  $\mathcal{G}(\mathbf{V}, \mathbf{W})$ , and a kernel  $q_V(\mathbf{V}|\mathbf{W})$  associated with  $\mathcal{G}$ , the fixing operator  $\phi_V(q_V; \mathcal{G})$  yields a new kernel  $\tilde{q}_{\mathbf{V} \setminus \{V\}}(\mathbf{V} \setminus \{V\}|\mathbf{W} \cup \{V\}) \equiv q_V(\mathbf{V}|\mathbf{W})/q_V(V|\mathbf{W} \cup \text{nd}_{\mathcal{G}}(V))$ , where the denominator is defined as above by marginalization and conditioning within the kernel  $q_V$ . If  $\text{ch}_{\mathcal{G}}(V) = \emptyset$ , division by  $q_V(V|\text{nd}_{\mathcal{G}}(V))$  is equivalent to marginalizing  $V$  from  $q_V$ . In this way, the fixing operator unifies applications of the g-formula in lines 6 and 7 of the ID algorithm, and marginalization of irrelevant variables in line 2 of the ID algorithm, and the recursive operation of the ID algorithm can be expressed concisely as repeated invocations of the operator. This allows us to concisely express functionals returned by ID algorithm and its variations, including our new algorithm for identifying responses to edge-specific policies, as one line formulas.

A set  $\mathbf{V}^\dagger \subseteq \mathbf{V}$  is said to be *fixable* in a latent projection  $\mathcal{G}(\mathbf{V})$  if there is a *valid* sequence  $\langle V_1, V_2, \dots, V_k \rangle$  of variables in  $\mathbf{V}^\dagger$  such that  $V_1$  is fixable in  $\mathcal{G}$ ,  $V_2$  is fixable in  $\phi_{V_1}(\mathcal{G})$ , and so on. If  $\mathbf{V}^\dagger$  is fixable,  $\mathbf{V} \setminus \mathbf{V}^\dagger$  is called a *reachable* set. If  $p(\mathbf{V})$  is a marginal of a distribution  $p(\mathbf{V} \cup \mathbf{H})$  Markov relative to a DAG  $\mathcal{G}(\mathbf{V} \cup \mathbf{H})$ , and  $\mathcal{G}(\mathbf{V})$  is a latent projection, then CADMG/kernel pairs obtained from  $\mathcal{G}(\mathbf{V})$  and  $p(\mathbf{V})$  by any valid sequence in  $\mathbf{V}^\dagger$  is the same [10, 17]. As a result, for any fixable set  $\mathbf{V}^\dagger$  in  $\mathcal{G}$ , writing  $\phi_{\mathbf{V}^\dagger}(\mathcal{G})$  or  $\phi_{\mathbf{V}^\dagger}(q_V; \mathcal{G})$  is well-defined, and means “apply the fixing operator to elements of  $\mathbf{V}^\dagger$  in some valid sequence,” with the understanding that any such sequence will yield the same result.

The existence of a valid fixing sequence for each district in  $\mathcal{G}_{\mathbf{Y}^*}$  implies corresponding terms may be identified via lines 2, 6, and 7 of the ID algorithm, and the overall algorithm can be rephrased as:

$$\begin{aligned} p(\mathbf{Y}|\text{do}(\mathbf{a})) &= \sum_{\mathbf{Y}^* \setminus \mathbf{Y}} \prod_{\mathbf{D} \in \mathcal{D}(\mathcal{G}_{\mathbf{Y}^*})} p(\mathbf{D}|\text{do}(\mathbf{V} \setminus \mathbf{D}))|_{\mathbf{A}=\mathbf{a}} \quad (11) \\ &= \sum_{\mathbf{Y}^* \setminus \mathbf{Y}} \prod_{\mathbf{D} \in \mathcal{D}(\mathcal{G}_{\mathbf{Y}^*})} \phi_{\mathbf{V} \setminus \mathbf{D}}(p(\mathbf{V}); \mathcal{G}(\mathbf{V}))|_{\mathbf{A}=\mathbf{a}}, \end{aligned}$$

which yields the following identifying formula for  $p(Y|\text{do}(a))$  in our example in Fig. 2 (a):

$$p(Y(a)) = \sum_{W_0, A, M, W_1} p(W_1|M, A=a, W_0) \times \quad (12)$$

$$p(M|A=a, W_0)p(W_0) \sum_{W_0, A} p(Y|W_1, M, A, W_0)p(W_0, A).$$

We omit the full derivation in the interest of space. See the section on identification of edge-specific policy inter-

ventions and the appendix for a complete example. Observe that this equation is a generalized version of Pearl’s front-door formula [9].

Whenever  $\mathbf{V} \setminus \mathbf{D}$  for every  $\mathbf{D}$  is fixable, the formula (11) yields the correct expression for  $p(\mathbf{Y}|\text{do}(\mathbf{a}))$  in terms of the observed data. If some  $\mathbf{V} \setminus \mathbf{D}$  is not fixable, the algorithm fails, and  $p(\mathbf{Y}|\text{do}(\mathbf{a}))$  is not identified. See [10] for a detailed proof.

## Edge Interventions

Identification of path-specific effects where each path is associated with one of two possible value sets  $\mathbf{a}, \mathbf{a}'$  was given a general characterization in [13] via the *recanting district criterion*. Here, we reformulate this result in terms of the fixing operator in a way that generalizes (11), and applies to the response of any edge intervention, including those that set edges to multiple values rather than two. This result can also be viewed as a generalization of *node consistency* of edge interventions in DAG models, found in [15].

Given  $\mathbf{A}_\alpha \equiv \{A \mid (AB)_\rightarrow \in \alpha\}$ , and an edge intervention given by the mapping  $\alpha_\alpha$ , define  $\mathbf{Y}^* \equiv \text{an}_{\mathcal{G}_{\mathbf{V} \setminus \mathbf{A}_\alpha}}(\mathbf{Y})$ . The joint distribution of the counterfactual response  $p(\{\mathbf{V} \setminus \mathbf{A}_\alpha\}(\mathbf{a}_\alpha))$  is identified if  $p(\{\mathbf{V} \setminus \mathbf{A}_\alpha\}(\mathbf{a}))$  is identified via (11), and for every  $\mathbf{D} \in \mathcal{D}(\mathcal{G}_{\mathbf{Y}^*})$ , for every  $A \in \mathbf{A}_\alpha$ ,  $\alpha_\alpha$  has the same value assignment for every directed edge out of  $A$  into  $\mathbf{D}$ . Under these assumptions, we have the following result.

**Theorem 1**  $p(\mathbf{Y}(\alpha_\alpha))$  is identified and equal to

$$\sum_{\mathbf{Y}^* \setminus \mathbf{Y}} \prod_{\mathbf{D} \in \mathcal{D}(\mathcal{G}_{\mathbf{Y}^*})} \phi_{\mathbf{V} \setminus \mathbf{D}}(p(\mathbf{V}); \mathcal{G})|_{\alpha_{\{(AD)_\rightarrow \in \alpha \mid D \in \mathbf{D}, A \in \mathbf{A}_\alpha\}}} \quad (13)$$

*Proof:* This follows directly from results in [13] and [10]. Identifying edge interventions entails identifying  $\prod_{\mathbf{D} \in \mathcal{D}(\mathcal{G}_{\mathbf{Y}^*})} p(\mathbf{D}|\text{do}(\mathbf{a}_\mathbf{D}))$ , where  $\mathbf{a}_\mathbf{D}$  is an assignment for  $\text{pa}_{\mathcal{G}}^s(\mathbf{D})$ , and  $\mathbf{a}_\mathbf{D}$  possibly assigns different values to elements of  $\mathbf{A}$  with respect to different districts. The fact that this identification algorithm can be rephrased as (13) follows directly by Theorem 60 in [10].  $\square$

Consider again the example in Fig. 2 (a). Now assume we set  $A = a$  for the edge  $(AM)_\rightarrow$  and  $A = a'$  for the edge  $(AW_1)_\rightarrow$ . The identifying functional for  $p(Y((aW_1)_\rightarrow, (a'M)_\rightarrow))$  has the same form as (12), but some terms are evaluated at  $A = a$ , and some at  $A = a'$ :

$$\sum_{W_0, A, M, W_1} p(W_1|M, A=a, W_0) \quad (14)$$

$$p(M|A=a', W_0)p(W_0) \sum_{W_0, A} p(Y|W_0, A, M, W_1)p(W_0, A)$$

### Policy Interventions (Dynamic Treatment Regimes)

A general algorithm for identification of responses to a set of policies  $\mathbf{f}_A$  was given in [16]. We again reformulate this algorithm in terms of the fixing operator. Define a graph  $\mathcal{G}_{\mathbf{f}_A}$  to be a graph obtained from  $\mathcal{G}$  by removing all edges into  $\mathbf{A}$ , and adding for any  $A \in \mathbf{A}$ , directed edges from  $\mathbf{W}_A$  to  $A$ . By definition of  $\mathbf{W}_A$ ,  $\mathcal{G}_{\mathbf{f}_A}$  is guaranteed to be acyclic. Define  $\mathbf{Y}^* \equiv \text{an}_{\mathcal{G}_{\mathbf{f}_A}}(\mathbf{Y}) \setminus \mathbf{A}$ . Assume  $p(\mathbf{Y}^*(\mathbf{a}))$  is identified in  $\mathcal{G}$ . Then, under the above assumptions, we have the following result.

**Theorem 2**  $p(\mathbf{Y}(\mathbf{f}_A))$  is identified in  $\mathcal{G}$ . Moreover, the identification formula is

$$\sum_{(\mathbf{Y}^* \cup \mathbf{A}) \setminus \mathbf{Y} \mathbf{D} \in \mathcal{D}(\mathcal{G}_{\mathbf{Y}^*})} \prod \phi_{\mathbf{V} \setminus \mathbf{D}}(p(\mathbf{V}); \mathcal{G}) \Big|_{\tilde{\mathbf{a}}_{\text{pa}_{\mathcal{G}}(\mathbf{D}) \cap \mathbf{A}}} \quad (15)$$

where  $\tilde{\mathbf{a}}_{\text{pa}_{\mathcal{G}}(\mathbf{D}) \cap \mathbf{A}}$  is defined as

$$\begin{cases} \{A = f_A(\mathbf{W}_A) \mid A \in \text{pa}_{\mathcal{G}}(\mathbf{D}) \cap \mathbf{A}\} & \text{pa}_{\mathcal{G}}(\mathbf{D}) \cap \mathbf{A} \neq \emptyset \\ \emptyset & \text{otherwise} \end{cases}$$

*Proof:* This follows from the fact that identification of  $p(\mathbf{Y}(\mathbf{f}_A))$  can be rephrased as identification of  $p(\mathbf{Y}^*(\mathbf{a}))$ , with values  $\mathbf{a}$  set according to  $\{\mathbf{W}_A \mid A \in \mathbf{A}\}$ , where all  $\mathbf{W}_A$  in the set are subsets of  $\mathbf{Y}^*$ . Identification of  $p(\mathbf{Y}^*(\mathbf{a}))$  may be rephrased as (15) follows by Theorem 60 in [10].  $\square$

The outer sum over  $\mathbf{A}$  in (15) is vacuous if  $\mathbf{f}_A$  is a set of deterministic policies. To illustrate (15), in our example in Fig. 2 (b),  $p(Y(A = f_A(W_0)))$  is identified as

$$\begin{aligned} & \sum_{W_0, A, M, W_1} p(W_1 | M, A = f(W_0), W_0) \\ & p(M | A = f(W_0), W_0) p(W_0) \sum_{W_0, A} p(Y | W_1, M, A, W_0) p(W_0, A) \end{aligned} \quad (16)$$

### Identification Of Edge-Specific Policies

Having reformulated existing identification results on responses to policies (15) and responses to edge interventions arising in mediation analysis (13) in terms of the fixing operator, we generalize these results for identification of responses to edge-specific policies.

Given  $\mathbf{A}_\alpha \equiv \{A \mid (AB)_{\rightarrow} \in \alpha\}$ , and a set of edge-specific policies given by the set of mappings  $\mathbf{f}_\alpha$ , define the graph  $\mathcal{G}_{\mathbf{f}_\alpha}$  to be one where all edges with arrowheads into  $\mathbf{A}_\alpha$  are removed, and directed edges from any vertex in  $\mathbf{W}_A$  to  $A \in \mathbf{A}_\alpha$  added. Fix a set  $\mathbf{Y}$  of outcomes of interest, and define  $\mathbf{Y}^*$  equal  $\text{an}_{\mathcal{G}_{\mathbf{f}_\alpha}}(\mathbf{Y}) \setminus \mathbf{A}_\alpha$ . We have the following result.

**Theorem 3**  $p(\mathbf{Y}(\mathbf{f}_\alpha))$  is identified if  $p(\mathbf{Y}^*(\mathbf{a}))$  is identified, and for every  $\mathbf{D} \in \mathcal{D}((\mathcal{G}_{\mathbf{f}_\alpha})_{\mathbf{Y}^*})$ ,  $\mathbf{f}_\alpha$  yields the same policy assignment for every edge from  $A \in \mathbf{A}_\alpha$  to  $\mathbf{D}$ . Moreover, the identifying formula is

$$\sum_{(\mathbf{Y}^* \cup \mathbf{A}_\alpha) \setminus \mathbf{Y} \mathbf{D} \in \mathcal{D}(\mathcal{G}_{\mathbf{Y}^*})} \prod \phi_{\mathbf{V} \setminus \mathbf{D}}(p(\mathbf{V}); \mathcal{G}) \Big|_{\tilde{\mathbf{a}}_{\text{pa}_{\mathcal{G}}(\mathbf{D}) \cap \mathbf{A}_\alpha}} \quad (17)$$

where  $\tilde{\mathbf{a}}_{\text{pa}_{\mathcal{G}}(\mathbf{D}) \cap \mathbf{A}_\alpha}$  is defined to be  $\{A = f_A(\mathbf{W}_A) \in \mathbf{f}_\alpha \mid A \in \text{pa}_{\mathcal{G}}(\mathbf{D}) \cap \mathbf{A}_\alpha\}$ , if  $\text{pa}_{\mathcal{G}}(\mathbf{D}) \cap \mathbf{A}_\alpha \neq \emptyset$ , and is defined to be the  $\emptyset$  otherwise.

*Proof:* This is a straightforward generalization of the proofs of Theorems 1 and 2.  $\square$

Responses to edge-specific policies are identified in strictly fewer cases compared to responses to edge interventions. This is because  $\mathbf{Y}^*$  is a larger set in the former case. As an example, consider the graph in Fig. 1 (c), where we are interested either in the counterfactual  $p(Y(a, M(a')))$ , used to define pure direct effects, or the counterfactual  $p(Y(f_A(W), M(a')))$ .

For the former counterfactual, we have  $\mathbf{Y}^* = \{Y, M\}$ , and  $p(Y(a, M(a')))$  equal to

$$\sum_m \left( \frac{\sum_w p(Y, m | a, w) p(w)}{\sum_w p(m | a, w) p(w)} \right) \sum_w p(m | a', w) p(w)$$

We omit the detailed derivation in the interest of space. For the latter counterfactual, however, the set  $\mathbf{Y}^* = \{Y, M, W\}$  forms a single district in  $\mathcal{G}_{\mathbf{Y}^*}$ , and the edge-specific policy set  $\mathbf{f}_{\{(AM)_{\rightarrow}, (AY)_{\rightarrow}\}}$  sets edges from  $A$  to this district to different policies. As a result, Theorem 3 is insufficient to conclude identification.

Generalizations of the example in Fig. 1 (b) are the most relevant in practice, as their causal structure corresponds to longitudinal observational studies, of the kind considered in [11], and many other papers. However, we illustrate complications that may arise in identifiability of responses to edge-specific policies with our running example in Fig. 2 (b), where we are interested in the response of  $Y$  to edge-specific policies  $\mathbf{f}_{\{(AM)_{\rightarrow}, (AW_1)_{\rightarrow}\}} = \{f_A^{(AM)_{\rightarrow}}(W_0), f_A^{(AW_1)_{\rightarrow}}(W_0)\}$ . Theorem 3 yields the following identifying formula:

$$\begin{aligned} & \sum_{W_0, A, M, W_1} \left[ p(W_1 | M, A = f_A^{(AM)_{\rightarrow}}(W_0), W_0) \right] \quad (18) \\ & \times \left[ p(M | A = f_A^{(AW_1)_{\rightarrow}}(W_0), W_0) p(W_0) \right] \\ & \times \left[ \sum_{W_0, A} p(Y | W_1, M, A, W_0) p(W_0, A) \right]. \end{aligned}$$

Note that (18) generalizes both (14), which sets  $A$  to different constants in different terms, and (16), which sets  $A$  to the output of a function that depends on  $W_0$ . We give a detailed derivation of this functional in the appendix.

## 5 ON COMPLETENESS

An identification algorithm for a class of parameters is said to be *complete* relative to a class of causal models if, whenever the algorithm fails to identify a parameter within a model class, the parameter is in fact not identified within that class.

The ID algorithm is known to be complete for the class of interventional distributions in the class of functional models [5, 14]. We restate this result here, and give a sequence of increasingly general completeness results for the identification algorithms described so far. Completeness results on policies and edge-specific policies are new. For completeness results pertaining to policies, we assume a completely unrestricted class of policies. If the set of policies of interest,  $\mathbf{f}_A$  or  $\mathbf{f}_\alpha$  is restricted, or alternatively if the causal model has parametric restrictions, completeness results we present may no longer hold.

**Theorem 4** *Given disjoint subsets  $\mathbf{Y}, \mathbf{A}$  of  $\mathbf{V}$  in an ADMG  $\mathcal{G}$ , define  $\mathbf{Y}^* \equiv \text{an}_{\mathcal{G}_{\mathbf{V} \setminus \mathbf{A}}}(\mathbf{Y})$ . Then  $p(\mathbf{Y}(\mathbf{a}))$  is not identified if there exists  $\mathbf{D} \in \mathcal{D}(\mathcal{G}_{\mathbf{Y}^*})$  that is not a reachable set in  $\mathcal{G}$ .*

**Corollary 1** *The algorithm for identification of  $p(\mathbf{Y}(\mathbf{a}))$ , as phrased in (11), is complete.*

**Theorem 5** *Given  $\mathbf{A}_\alpha \equiv \{A \mid (AB)_\rightarrow \in \alpha\}$ , and an edge intervention given by the mapping  $\mathbf{a}_\alpha$ , define  $\mathbf{Y}^* \equiv \text{an}_{\mathcal{G}_{\mathbf{V} \setminus \mathbf{A}_\alpha}}(\mathbf{Y})$ . The joint distribution of the counterfactual response  $p(\{\mathbf{V} \setminus \mathbf{A}_\alpha\}(\mathbf{a}_\alpha))$  is not identified if  $p(\{\mathbf{V} \setminus \mathbf{A}_\alpha\}(\mathbf{a}))$  is not identified, or there exists  $\mathbf{D} \in \mathcal{D}(\mathcal{G}_{\mathbf{Y}^*})$  and  $A \in \mathbf{A}_\alpha$ , such that  $\mathbf{a}_\alpha$  has the different value assignments for a pair of directed edges out of  $A$  into  $\mathbf{D}$ .*

**Corollary 2** *The algorithm for identification of  $p(\mathbf{Y}(\mathbf{a}_\alpha))$ , as phrased in (13), is complete.*

**Theorem 6** *Define  $\mathcal{G}_{\mathbf{f}_A}$  to be a graph obtained from  $\mathcal{G}$  by removing all edges into  $\mathbf{A}$ , and adding for any  $A \in \mathbf{A}$ , directed edges from  $\mathbf{W}_A$  to  $A$ . Define  $\mathbf{Y}^* \equiv \text{an}_{\mathcal{G}_{\mathbf{f}_A}}(\mathbf{Y}) \setminus \mathbf{A}$ . Then if  $p(\mathbf{Y}^*(\mathbf{a}))$  is not identified in  $\mathcal{G}$ ,  $p(\mathbf{Y}(\mathbf{f}_A))$  is not identified in  $\mathcal{G}$  if  $\mathbf{f}_A$  is the unrestricted class of policies.*

**Corollary 3** *The algorithm for identification of  $p(\mathbf{Y}(\mathbf{f}_A))$ , as phrased in (15), is complete for unrestricted policies.*

**Theorem 7** *Define the graph  $\mathcal{G}_{\mathbf{f}_\alpha}$  to be one where all edges with arrowheads into  $\mathbf{A}_\alpha$  are removed, and directed edges from any vertex in  $\mathbf{W}_A$  to  $A \in \mathbf{A}_\alpha$  added. Fix a set  $\mathbf{Y}$  of outcomes of interest, and define  $\mathbf{Y}^*$  equal  $\text{an}_{\mathcal{G}_{\mathbf{f}_\alpha}}(\mathbf{Y}) \setminus \mathbf{A}_\alpha$ . Then if  $p(\mathbf{Y}^*(\mathbf{a}))$  is not identified, or*

*there exists  $\mathbf{D} \in \mathcal{D}((\mathcal{G}_{\mathbf{f}_\alpha})_{\mathbf{Y}^*})$ , such that  $\mathbf{f}_\alpha$  yields different policy assignments for two edges from  $A \in \mathbf{A}_\alpha$  to  $\mathbf{D}$ ,  $p(\mathbf{Y}(\mathbf{f}_\alpha))$  is not identified.*

**Corollary 4** *The algorithm for identification of  $p(\mathbf{Y}(\mathbf{f}_\alpha))$ , as phrased in (17), is complete for unrestricted policies.*

Detailed proofs of these results are in the Appendix. Corollaries are immediate consequences of the preceding Theorems.

## 6 CONCLUSION

In this paper, we defined counterfactual responses to policies that set treatment values in such a way that they affect outcomes with respect to certain causal pathways only. Such counterfactuals arise when we wish to personalize only some portion of the causal effect of a treatment, while keeping other portions set to some reference values. An example might be optimizing the chemical effect of a drug, while keeping drug adherence to a reference value.

We gave a general algorithm for identifying these responses from data, which generalizes similar algorithms due to [16, 13] for dynamic treatment regimes, and edge-specific effects, respectively. Further, we showed that given an unrestricted class of policies the algorithm is complete. As a corollary, this established that the identification algorithm for dynamic treatment regimes in [16] is complete for unrestricted policies.

Given a fixed set of policies associated with a set of causal pathways, and assuming (17) yields a functional containing only conditional densities, as is the case in the functional (18), the counterfactual mean under those policies  $\mathbb{E}[Y(\mathbf{f}_\alpha)]$  may be estimated using the maximum likelihood plug-in estimator. Such an estimator can be viewed as a generalization of the parametric g-formula [11] to edge-specific policies. More general estimation strategies, and approaches to learning the optimal set of policies are the subject of our companion paper [7].

### Acknowledgments

This research was supported in part by the NIH grants R01 AI104459-01A1 and R01 AI127271-01A1. We thank the anonymous reviewers for their insightful comments that greatly improved this manuscript.

## References

- [1] C. Avin, I. Shpitser, and J. Pearl. Identifiability of path-specific effects. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05)*, volume 19, pages 357–363. Morgan Kaufmann, San Francisco, 2005.
- [2] D. P. Bertsekas and J. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Publishing, 1996.
- [3] B. Chakraborty and E. E. M. Moodie. *Statistical Methods for Dynamic Treatment Regimes (Reinforcement Learning, Causal Inference, and Personalized Medicine)*. Springer, New York, 2013.
- [4] M. A. Hernan, E. Lanoy, D. Costagliola, and J. M. Robins. Comparison of dynamic treatment regimes via inverse probability weighting. *Basic and Clinical Pharmacology and Toxicology*, 98:237–242, 2006.
- [5] Y. Huang and M. Valtorta. Pearl’s calculus of interventions is complete. In *Twenty Second Conference On Uncertainty in Artificial Intelligence*, 2006.
- [6] C. Miles, I. Shpitser, P. Kanki, S. Melone, and E. J. Tchetgen Tchetgen. Quantifying an adherence path-specific effect of antiretroviral therapy in the nigeria pefar program. *Journal of the American Statistical Association*, 2017.
- [7] R. Nabi and I. Shpitser. Estimation of personalized effects associated with causal pathways. In *Proceedings of the Thirty Fourth Conference on Uncertainty in Artificial Intelligence (UAI)*, 2018.
- [8] J. Pearl. Direct and indirect effects. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 411–420. Morgan Kaufmann, San Francisco, 2001.
- [9] J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2 edition, 2009.
- [10] T. S. Richardson, R. J. Evans, J. M. Robins, and I. Shpitser. Nested Markov properties for acyclic directed mixed graphs, 2017. Working paper.
- [11] J. M. Robins. A new approach to causal inference in mortality studies with sustained exposure periods – application to control of the healthy worker survivor effect. *Mathematical Modeling*, 7:1393–1512, 1986.
- [12] J. M. Robins and S. Greenland. Identifiability and exchangeability of direct and indirect effects. *Epidemiology*, 3:143–155, 1992.
- [13] I. Shpitser. Counterfactual graphical models for longitudinal mediation analysis with unobserved confounding. *Cognitive Science (Rumelhart special issue)*, 37:1011–1035, 2013.
- [14] I. Shpitser and J. Pearl. Identification of joint interventional distributions in recursive semi-Markovian causal models. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI-06)*. AAAI Press, Palo Alto, 2006.
- [15] I. Shpitser and E. J. Tchetgen Tchetgen. Causal inference with a graphical hierarchy of interventions. *Annals of Statistics*, 44(6):2433–2466, 2016.
- [16] J. Tian. Identifying dynamic sequential plans. In *Proceedings of the Twenty-Fourth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-08)*, pages 554–561, Corvallis, Oregon, 2008. AUAI Press.
- [17] J. Tian and J. Pearl. On the testable implications of causal models with hidden variables. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI-02)*, volume 18, pages 519–527. AUAI Press, Corvallis, Oregon, 2002.

---

# Fast Counting in Machine Learning Applications

---

**S. Karan**

University at Buffalo  
skaran@buffalo.edu

**M. Eichhorn**

University at Buffalo  
maeichho@buffalo.edu

**B. Hurlburt**

University at Buffalo  
blakehur@buffalo.edu

**G. Iraci**

University at Buffalo  
grantira@buffalo.edu

**J. Zola**

University at Buffalo  
jzola@buffalo.edu

## Abstract

We propose scalable methods to execute counting queries in machine learning applications. To achieve memory and computational efficiency, we abstract counting queries and their context such that the counts can be aggregated as a stream. We demonstrate performance and scalability of the resulting approach on random queries, and through extensive experimentation using Bayesian networks learning and association rule mining. Our methods significantly outperform commonly used ADtrees and hash tables, and are practical alternatives for processing large-scale data.

## 1 INTRODUCTION

Counting data records with instances that support some specific configuration of the selected variables is one of the basic operations utilized by machine learning (ML) algorithms. For example, when learning Bayesian network structure from data counting is necessary to evaluate a scoring function, or to assess constraints (e.g., via mutual information) [1]. In association rule mining, counting over binary data representing transactions is required to assess support and confidence for a given association rule [2]. Other examples include problems ranging from classification [3, 4] through deep learning [5, 6] to information retrieval [7].

While counting is typically viewed as a black-box procedure, and implemented using simple and not necessarily efficient strategies, e.g., contingency tables, in many practical applications it accounts for over 90% of the total execution time (we show several practical cases in Sec. 4). Consequently, improving performance of counting can directly translate into better performance of these applications. At the same time, popular specialized approaches

based on data indexes, such as ADtrees [8], have limited applicability due to the significant preprocessing and memory overheads, which easily exceed the capability of current computational servers. This holds true for a broad spectrum of problem sizes and applications, with cases involving anywhere from tens to hundreds of variables, and thousands to millions of instances. As the size of the data analyzed by ML codes increases, there is a clear need for easy-to-adopt, efficient and scalable counting strategies.

In this paper, we address the above challenge by designing simple, yet fast and memory efficient counting strategies. Our methods are derived from the standard techniques like bitmap set representation and radix sorting, which can be efficiently implemented in a software. We describe an intuitive and convenient programming interface that leverages properties of the operators used in ML to separate the counting process from how the counts are utilized. This interface enables us to aggregate counts in a stream-like fashion. We encapsulate our methods in an open source software, and demonstrate its performance on random queries, Bayesian networks learning and association rule mining. Through extensive experiments on multiple popular benchmark data, we show that our strategies are orders of magnitude faster than the commonly used methods, such as ADtrees and hash tables.

## 2 PRELIMINARIES

Consider a set of  $n$  categorical random variables  $\mathcal{X} = \{X_1, X_2, \dots, X_n\}$ , where the domain of variable  $X_i$  is represented by states  $[x_{i1}, \dots, x_{ir_i}]$ . Alternatively, we can think of  $X_i$  as a symbolic feature with arity  $r_i$ , and for convenience we can represent its states by integers  $[1, \dots, r_i]$ . Let  $D = [D_1, D_2, \dots, D_n]$  be a complete database of instances of  $\mathcal{X}$ , where  $D_i$ ,  $|D_i| = m$ , records observed states of  $X_i$ . Given  $D$ , and a set of input variables  $\{X_i, X_j, \dots\} \subseteq \mathcal{X}$ , the counting query  $Count((X_i = x_i) \wedge (X_j = x_j) \wedge \dots)$  returns the size of the support in  $D$  for the specific assignment  $[x_i, x_j, \dots]$



of variables  $[X_i, X_j, \dots]$ . For example, for the database in Fig. 1, the query  $Count((X_1 = 3) \wedge (X_2 = 2) \wedge (X_3 = 1))$  would return 2, as there are 2 instances matching the query condition. We note that the above formulation of counting is a special and simple case of the general counting problem in conjunctive queries, known from database theory [9] (we provide more details in Section 5).

Counting queries are the basic operations performed when learning statistical models from data. In some ML applications, such as association rule mining or learning probabilistic graphical models, they may account for over 90% of the total execution time. In the most basic form, the queries can be issued without shared context. For example, to estimate joint probability  $p(X_i = x_i, X_j = x_j)$  from  $D$ , we could use just one query:  $\hat{p}(x_i, x_j|D) = \frac{Count((X_i=x_i) \wedge (X_j=x_j))}{m}$ . However, in the most common use scenarios, a group of consecutive queries is executed over the same set of variables (i.e., the queries share context). For instance, consider log-likelihood score frequently used in Bayesian networks learning [10]:

$$\mathcal{L}(X_i|Pa(X_i)) = \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} \log \left( \frac{N_{ijk}}{N_{ij}} \right), \quad (1)$$

where  $Pa(X_i) \subseteq \mathcal{X} - \{X_i\}$  is a set of predictor variables for  $X_i$ ,  $j$  enumerates all possible  $q_i = \prod_{X_j \in Pa(X_i)} r_j$  states of variables in  $Pa(X_i)$ , and  $N_{ij}$  and  $N_{ijk}$  are respectively the counts of instances in  $D$  such that variables in  $Pa(X_i)$  are in state  $j$ , and the counts of instances such that variables in  $Pa(X_i)$  are in state  $j$  and  $X_i$  is in state  $k$ . To compute  $\mathcal{L}$  we require multiple counting queries over the same group of variables  $Pa(X_i) \cup \{X_i\}$ , testing different configurations of their states. Moreover, we care only about queries that return non-zero counts  $N_{ijk}$  (note that non-zero  $N_{ijk}$  implies non-zero  $N_{ij}$ ), since only those contribute to the final sum.

The standard approach to handle queries that share context is to either directly scan the database  $D$  to construct  $r_i \times q_i$  contingency table of counts (or its high-dimensional variant such as data cube [11]), or to first create an ADtree index to cache all sufficient statistics from  $D$ , and then to materialize contingency table on demand. Here materialization is done by retrieving the required counts via fast traversal over the index [8, 12]. However, both these approaches have significant limitations.

To use a contingency table we have to either maintain a lookup table with  $r_i \cdot q_i$  entries, or to use a dictionary (e.g., hash table) with keys over the states of  $Pa(X_i)$  and values being vectors of counts for the corresponding states of  $X_i$ . While lookup table may offer very fast memory accesses during construction and querying phases, it becomes computationally impractical, since usually it is very sparse. This is because even for large  $m$ , most of the time  $D$  will

not contain all  $q_i$  possible configurations for the majority of sets  $Pa(X_i)$ . Consequently, lookup tables become a feasible choice only when we are dealing with a small number of variables, each with very small arity. Dictionaries address the problem of sparsity, as they store only configurations that are observed in  $D$ . However, they impose non-trivial overheads owing to the cost of hashing in a hash table dictionary or traversing scattered memory in a search tree dictionary. Moreover, when large number of high-arity variables are considered, a dictionary quickly becomes memory intensive easily exceeding capacity of a typical cache memory.

The alternative approach is to use one of many published variants of the ADtree index, e.g., [8, 12, 13, 14]. Here the idea is to first invest (significant) time and memory to enumerate and cache counts of all configurations found in  $D$ , and then reference those counts to answer subsequent queries. However, even with various optimizations, the space complexity of ADtrees is exponential in the number of variables, and even for modestly sized  $D$  it may exceed the available main memory. Moreover, by caching all counts indiscriminately, ADtrees often store entries that are never referenced in a given application, creating unnecessary memoization and searching overhead. Finally, ADtrees still require that a contingency table is materialized to deliver retrieved counts, and hence they pose a significant challenge in balancing memory and computations.

### 3 PROPOSED APPROACH

Given the database  $D$ , our goal is to provide memory and computationally efficient mechanism to answer counting queries with shared context. The memory efficiency is critical, since many ML algorithms, especially in classification and probabilistic graphical modeling, already have significant memory constraints (see for example [15]). If the memory has to be devoted to handling queries instead of being used by the actual algorithm, it would clearly constrain the applicability of the algorithm. At this point it is worth noting that ML applications fall into a gray zone in terms of the size of the input data on which they typically operate. On the one hand, the size of the input is too small to benefit from many excellent optimizations known from database theory (some we review in Sec. 5), as those are targeting cases in which volume of the data necessitates concurrent use of both persistent and main memory. On the other hand, the data is too large to warrant efficient execution using direct techniques like simple contingency tables.

To address this situation, we first define an intuitive programming interface to abstract the query context, including how counts are utilized by the target application. Then,

we overlay the interface on top of two simple, yet very efficient, query execution strategies, where instead of storing counts we *consume* them in a stream-like fashion.

### 3.1 Programming Interface

In a typical application, counts provided by queries with shared context are iteratively aggregated via some associative and commutative operator. One simple example with the summation operator is given in Eq. (1). A more complex example could be Dirichlet priors with the product of gamma functions [16]. From the computational point of view, this assumption is very helpful as it provides ample opportunities for optimization. We note also that while it may look very narrow, it actually accurately captures surprisingly many ML applications, which primarily involve estimating conditional probabilities. Examples include classifiers and regression, feature extraction, different variants of probabilistic graphical models, etc..

Following notation in Sec. 2, let us consider a set of variables  $(Pa(X_i) \cup \{X_i\}) \subseteq \mathcal{X}$  and their corresponding counts  $N_{ij}$  and  $N_{ijk}$ , for some specific configuration  $j$  of  $Pa(X_i)$  and  $k$  of  $X_i$ . Here we are distinguishing between counts for  $(Pa(X_i) \cup \{X_i\})$  and  $Pa(X_i)$  to simplify computing conditional probabilities while maintaining generality – by passing  $Pa(X_i) = \emptyset$  we can execute queries over single variable  $X_i$ , and by considering only  $N_{ijk}$  we get joint queries  $Pa(X_i) \cup \{X_i\}$ . The key observation is that we can leverage associativity and commutativity, and instead of first gathering all counts and then performing aggregation, we can create a stream of counts corresponding to all unique and relevant configurations found in  $D$ , and perform the aggregation directly on the stream. This enables us to push computations to data, mitigating memory overheads due to counts caching. To achieve this, we abstract the computations via a function object (a concept supported by all modern programming languages), which is then repeatedly invoked over the stream. The example function object corresponding to Eq. (1) is given in Fig. 2. In the essence, the object receives  $N_{ijk}$  and  $N_{ij}$  via the function call operator (line 3), performs the required intermediate computations, and then aggregates the result into internal state. This internal state can be then inspected (line 7) to retrieve the final result of the aggregation. From the user perspective, the function call operator acts as an interface, and is directly invoked by a routine responsible for enumerating, and emitting, all unique configurations for the variables of interest (see parameter  $F$  in Algs. 1 and 2 in the following sections). Thus the interface provides a convenient encapsulation, and the end-user who defines the function object (e.g., implementing a scoring function in BN learning) can focus solely on expressing computations (i.e., high-level

logic and correctness), and does not have to worry about potentially complex logic of low-level details (e.g., how counts are enumerated). Additionally, because function object behaves like a function, but has the advantage of possessing an internal state, it is a convenient mechanism to express even the most demanding computations.

While the proposed interface stems from a relatively simple observation, it has several immediate advantages. First, by separating functionality (i.e., data traversing from computations) we gain flexibility to rapidly investigate different data traversal schemes to extract counts, or even alternate between different strategies depending on the query context (e.g., how many variables are involved, their domain, etc.). Second, since counts are aggregated into an isolated state represented by a function object, and multiple objects can coexist independently, multiple groups of queries, each group with individual context, can be executed concurrently and in parallel, e.g., by different threads. Collectively, this makes the proposed design extremely flexible, efficient and easy to use, as we demonstrate in the experimental results section.

### 3.2 Bitmap Strategy

For the specific  $X_i$  and  $Pa(X_i)$  our task now is to enumerate counts  $N_{ij}$  and  $N_{ijk}$  for all configurations  $j$  and  $k$  found in  $D$ , and then pass the counts to a function object for aggregation. The idea behind the Bitmap strategy is to represent each variable  $X_i$  via a set of  $r_i$  bitmaps of size  $m$ , where each bitmap indicates instances for which  $X_i$  is in the corresponding state (see Fig. 1a). Then, the entire process of enumerating counts can be reduced to performing logical AND on bitmaps, equivalent of set intersection, and to bit counting, equivalent of computing set cardinality. This is summarized in Alg. 1, with example in Fig. 1b (for convenience, in the algorithm we use set notation instead of directly representing bitmaps).

---

#### Algorithm 1 QUERY( $X_i, Pa, F, b$ )

---

```

1 if  $|Pa| = 0$  then
2    $N_{ij} \leftarrow |b|$ 
3   for  $v \in [1, \dots, r_i]$  do
4      $b_v \leftarrow \{p \mid D_i[p] = v\}$ 
5      $N_{ijk} \leftarrow |b \cap b_v|$ 
6     if  $N_{ijk} > 0$  then
7        $F(N_{ijk}, N_{ij}) \triangleleft$  emit new configuration
8 else
9    $X_h \leftarrow \text{HEAD}(Pa)$ 
10  for  $v \in [1, \dots, r_h]$  do
11     $b_v \leftarrow \{p \mid D_h[p] = v\}$ 
12    if  $|b \cap b_v| > 0$  then
13      QUERY( $X_i, \text{TAIL}(Pa), F, b \cap b_v$ )

```

---

To execute counting queries for  $X_i$  and  $Pa(X_i)$  (abbreviated to  $Pa$ ), and function object  $F$ , we perform Depth

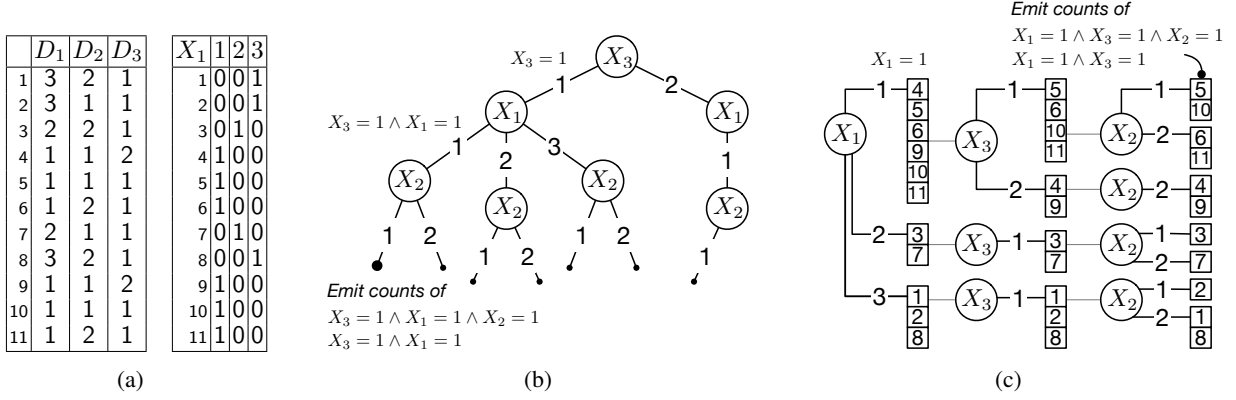


Figure 1: (a) Database  $D$  with three variables, and the corresponding bitmap representation of  $X_1$ . (b) Example of executing  $Query(X_2, \{X_1, X_3\})$  over  $D$  using Bitmap strategy, and (c) Radix strategy.

```

1 class L {
2 public:
3     void operator()(int Nijk, int Nij) {
4         double p = 1.0 * Nijk / Nij;
5         score_ += (Nijk * log2(p));
6     }
7     double score() const { return score_; }
8 private:
9     double score_ = 0.0;
10 };

```

Figure 2: Example C++ code packaging Eq. (1) into our programming interface.

First Traversal (DFS) over the tree whose leaves represent all possible  $r_i \cdot q_i$  states of interest (recall that  $q_i = \prod_{X_j \in Pa(X_i)} r_j$ ). The bottom layer of the tree is induced by the states of  $X_i$ , and the top layers correspond to variables in  $Pa$ . When moving down the tree (lines 9-13), we compute intersection between the set of instances supporting variables' configurations seen thus far (in the algorithm denoted by  $b$ , which initially consists of all  $m$  instances), and the set of instances supporting current configuration of the considered variable from  $Pa$  (in the algorithm denoted by  $b_v$ ). We continue traversal only if the size of the intersection is greater than zero, implying non-zero count for given joint configuration of variables. Once we reach a leaf of the tree (lines 1-7), we compute the final counts  $N_{ijk}$  and  $N_{ij}$  for the corresponding configurations  $j$  and  $k$ , and emit those via call to  $F$ .

The depth of the tree depends on the number of variables involved in the query, and the number of leaves is bounded by  $O(\min(q_i, m))$ , with each step in the traversal involving  $O(m)$  cost of computing intersection and cardinality. While the tree could potentially involve exponential (in the number of query variables) number of

nodes, it is never explicitly stored in the memory, and even for  $D$  with large number of instances many configurations have zero count, allowing for their corresponding sub-trees to be pruned. To further leverage this property, we order  $Pa$  such that variables with lowest entropy estimated from  $D$  are at the top of the tree. Since variables with low entropy are likely to have configurations for which there will be only few supporting instances, they are more likely to trigger zero counts and hence lead to a smaller tree to traverse. For example, consider executing  $Query(X_2, \{X_1, X_3\})$  outlined in Fig. 1b. There are total of 7 configurations which we should enumerate, and if we traverse the tree starting from variable  $X_3$ , which has lower entropy than  $X_1$ , then we will have to consider 6 intermediate states. If we were to start with variable  $X_1$ , then this number would increase to 9. This optimization performs extremely well in practice, and, as we show in the experimental results section, for certain ranges of  $n$  and  $m$ , Bitmap outperforms other strategies.

In the practical terms, the strategy can be efficiently implemented using streaming extensions (SIMD) in current processors. Bitmaps for individual variables can be pre-computed and laid out in the memory instead of  $D$ , with acceptable memory overhead (i.e.,  $m \cdot r_i$  vs.  $m \cdot \log_2(r_i)$  bits), and the relative ordering of variables in  $D$ , based on their entropy, can be established beforehand as well.

### 3.3 Radix Strategy

While the Bitmap strategy is amenable to very efficient implementation, its scalability may still suffer when datasets with very large number of instances are exercised by queries with many variables, or variables with high arity. This is because in such cases the DFS tree will have fewer nodes to prune, and the advantage of fast bit-wise operations will be offset by the poor asymptotic behavior. To address these cases, we consider an alternative

approach, which we refer to as Radix strategy. The strategy is derived from the classic radix sort algorithm, and it involves recursively partitioning instances in  $D$  such that single partition at given level captures all instances corresponding to one specific configuration of the query variables. This approach is summarized in Algs. 2 and 3, with example in Fig. 1c.

---

**Algorithm 2** QUERY( $X_i, Pa, F$ )
 

---

```

1  $B \leftarrow \llbracket 1, \dots, m \rrbracket$ 
2 if  $|Pa| \neq 0$  then
3    $B \leftarrow \text{BUCKETS}(\text{HEAD}(Pa), \text{TAIL}(Pa), \text{HEAD}(B))$ 
4 for  $b \in B$  do
5    $N_{ij} \leftarrow |b|$ 
6   if  $N_{ij} > 0$  then
7      $B' \leftarrow \text{BUCKETS}(X_i, \llbracket \rrbracket, b)$ 
8     for  $b' \in B'$  do
9        $N_{ijk} \leftarrow |b'|$ 
10      if  $N_{ijk} > 0$  then
11         $F(N_{ijk}, N_{ij}) \triangleleft$  emit new configuration

```

---



---

**Algorithm 3** BUCKETS( $X_p, Pa, b$ )
 

---

```

1  $B' \leftarrow \llbracket \rrbracket$ 
2 for  $q \in \llbracket 1, \dots, |b| \rrbracket$  do
3    $x_p \leftarrow D_p[b[q]]$ 
4    $B'[x_p].\text{APPEND}(b[q])$ 
5 if  $\text{TAIL}(Pa) = \llbracket \rrbracket$  then
6   return  $B'$ 
7  $B'' \leftarrow \llbracket \rrbracket$ 
8 for  $b' \in B'$  do
9    $B''.\text{APPEND}(\text{BUCKETS}(\text{HEAD}(Pa), \text{TAIL}(Pa), b'))$ 
10 return  $B''$ 

```

---

The algorithm follows the Most Significant Digit (MSD) radix, with the left most digits being states of individual variables in  $Pa$ , and the least significant digit representing states of  $X_i$  (Alg. 2, line 3). At each level, the number of newly created partitions is proportional to the arity of the considered variable, and the size of the partition is the support in  $D$  for the particular configuration. The order in which variables from  $Pa$  are processed is not significant, since the cost of identifying empty partitions does not induce overheads. Because the actual instances in  $D$  are not to be sorted, but only partitioned, it is sufficient that we maintain a list (in algorithms denoted by  $B$ ) of partition descriptors containing indexes of the constituent instances and partition size (Alg. 3, lines 1-4). As soon as all partitions prescribed by  $Pa$  are identified we can proceed to emitting counts (Alg. 2, lines 4-11), which must be preceded by the final round of partitioning with respect to  $X_i$  (Alg. 2, line 7).

The algorithm requires that for each variable  $X_p \in Pa$  its corresponding data vector  $D_p$  is completely scanned, leading to the overall  $O(|Pa| \cdot m)$  complexity. In practice, the entire method is efficiently implemented by first orga-

nizing the database  $D$  in the column-major format, and then maintaining a FIFO queue of partition descriptors, with  $O(m)$  auxiliary space to keep track of the assignment of indexes to partitions. Moreover, partitioning for individual variables can be precomputed in advance, further bootstrapping the first step of the algorithm.

To conclude the presentation, we note that both Bitmap and Radix strategies can be further augmented such that instead of enumerating all counts (i.e., executing queries with shared context) they deliver counts just for the specific assignment of the query variables. To achieve this, it is sufficient to process only a single path from the root to the leaf with the target assignment in the DFS tree of the Bitmap strategy, and to find the partition corresponding to the assignment, instead of all partitions, in Radix.

## 4 EXPERIMENTAL VALIDATION

We implemented both proposed strategies as a C++ software library, which we complemented with Python bindings for the ease of use. At its core, the library uses standard SSE SIMD intrinsics to implement basic bitmap operations (i.e., logical AND, and bit counting), and it exposes all functionality via the interface described in Sec. 3.1. The resulting open source package, which we call SABNatk, is available from: <https://gitlab.com/SCoRe-Group/SABNatk>.

We deployed SABNatk on a server with two Intel Xeon E5-2650 2.30 GHz 10-core CPUs, and 64 GB of RAM. To test the performance, we ran a series of experiments using popular ML benchmark datasets (see data summary in Tab. 1). For reference, we used hash table from the C++ standard library, and the sparse ADtree data index [8]. The hash table represented contingency table created by directly scanning the input database, with keys encoding specific assignment of variables in  $Pa(X_i)$ , and values representing vectors of counts for specific assignment of  $X_i$ . To maximize cache memory usage, the strategy operated on the database stored in the row-major order. Finally, to make the comparison fair and avoid biases due to the differences in programming languages, we developed an efficient ADtree implementation in C++. We note that other available implementations, for example [17], turned out to be substantially slower than our version.

In the following, we discuss in detail several key results obtained using the above setup. More extensive results (including additional test cases), together with the data that can be used to reproduce our experiments, are available from: <https://gitlab.com/SCoRe-Group/SABNatk-Benchmarks>.

Before we proceed with the results discussion, we note that in order to use ADtree, the input database has to be

Table 1: Benchmark data used in experiments.

Dataset	$n$	Range of $r_i$	Average $r_i$
Child	20	2-6	3
Insur(ance)	27	2-5	3.3
Mild(ew)	35	3-99	16.4
Alarm	37	2-4	2.83
Barley	46	3-67	9.02
Hail(finder)	56	2-11	3.98
Win95(pts)	74	2-2	2
Path(finder)	104	2-63	4.2

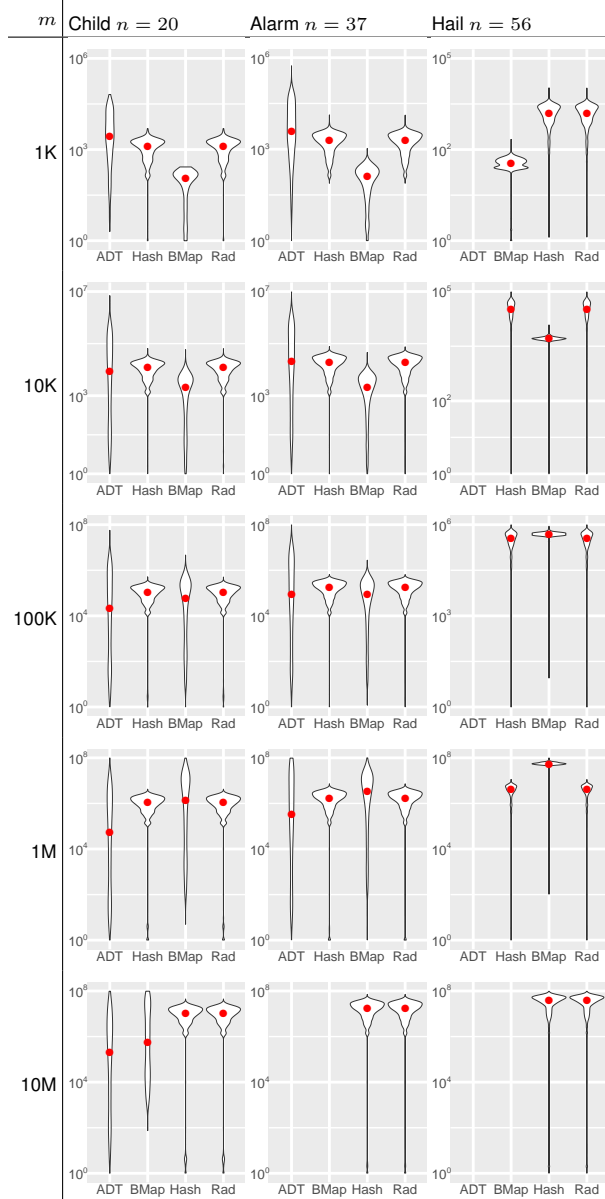


Figure 3: Comparison ADT, Hash, BMap and Rad strategies on the stream of uniformly random queries. The plots show the distribution of response time in microseconds, computed from the same sample of 1,000 queries for different number of instances  $m$ . Y-axis is in  $\log_{10}$ -scale.

first indexed. In all our experiments, we considered only the query time with index already loaded into memory. Moreover, ADtree provides a hyper-parameter  $\ell$  to configure the size of the leaf-lists [8]. We experimented with several values of the parameter, to fine-tune the trade-off between query performance and memory consumption, and we settled with  $\ell = 16$ , which we use throughout the paper. The results obtained for other ADtree configurations exhibited similar patterns to those reported in the paper, and are available online.

#### 4.1 Random Queries

In the first set of experiments we tested how ADtree (ADT), HashTable (Hash), Bitmap (BMap), and Radix (Rad) strategies respond to a stream of random queries. The idea here is to understand average performance of each strategy in case where we have no prior information about specific query execution patterns. For each benchmark database, we generated 100,000 queries of the form  $Query(X_i, Pa(X_i))$  as follows. First, the size of  $Pa(X_i)$  was sampled uniformly from the range  $[1, \dots, n - 1]$ , and then variables were assigned to  $X_i$  and  $Pa(X_i)$  by randomly sampling without replacement from  $\mathcal{X}$ . To measure time taken to execute the query, we used a simple function object that consumes and immediately discards the counts. In this way, the overhead of performing computations on the counts was negligible, and did not offset the actual time spent by each strategy to enumerate the counts. Each query was executed five times to obtain the average response time (with negligible variance), and exactly the same stream of queries was processed by each strategy. The results of this experiment are summarized in Figs. 3 and 4. Here, we note that plots are in  $\log_{10}$  scale, and should be interpreted with care.

Figure 3 shows that depending on the number of input variables  $n$ , and the number of instances  $m$ , different strategies perform better in terms of the mean response time. When the number of instances is relatively small, Bitmap strategy significantly outperforms other methods. This is explained by two factors: first, for small scale data, Bitmap benefits from continuous memory accesses, and acceleration via SIMD instructions, second, because in small datasets many possible variable configurations are unobserved, Bitmap is able to prune significant portions of the DFS tree, taking advantage of the entropy-based data reordering (see Sec. 3.2). However, as the number of instances in the input database increases these advantages diminish, to the extent where the average time taken by a query becomes unacceptable (longer than several seconds, a threshold we set to make experiments computationally feasible). The Radix and HashTable strategies perform steadily across all datasets, and are able to handle even the

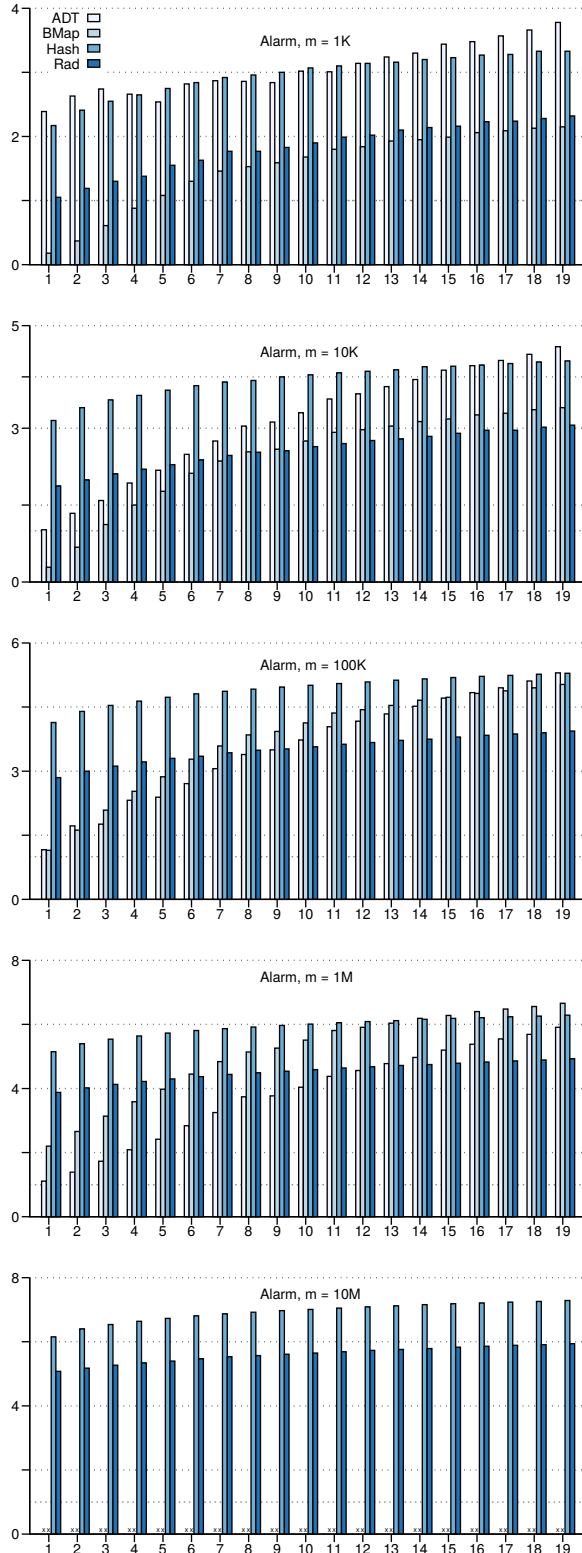


Figure 4: Comparison of ADT, Hash, BMap, and Rad for different sizes of  $Pa$  (x-axis) in the sample of 1,000 uniformly random queries. The plot shows the average query response time in microseconds. Y-axis is in  $\log_{10}$ -scale.

most demanding test cases. This is expected, since both strategies involve similar data access pattern (i.e., scanning selected columns of the input database). However, Rad is on average 20 times faster than Hash (not captured in the figure due to log-scale), as it does not require costly hashing and scattered memory accesses.

The ADtree strategy exhibits the best mean response for problems with few variables and large number of instances, but it significantly underperforms in all remaining test cases. In fact, as the number of variables increases, ADtree fails to index the database and cannot be used to answer the queries. This is because the exponential growth of the number of configurations, which have to be cached, leads to the exhaustion of the main memory. Recall also that we do not include ADtree preprocessing time, which for datasets with more than 100K instances exceeds several hours, much longer than the time required to answer all 100K queries.

To further dissect performance of random queries, in Fig. 4 we show how the response time varies with the number of query variables, for an example database. When processing small queries ( $|Pa| < 4$ ), ADtree is generally outperformed by BMap, and when handling large queries ( $|Pa| > 10$ ) it is slower than Rad. Moreover, the cost of Radix strategy is linear with the query size, compared to the exponential growth of ADtree.

Based on the tests with random queries, we conclude that Bitmap and Radix strategies significantly outperform ADtree and HashTable, except of a small set of scenarios in which small queries are executed over databases with few variables and millions of instances, if we exclude the preprocessing time.

## 4.2 Queries in Bayesian Networks Learning

Counting queries with shared context are the key operations performed in score-based Bayesian networks structure learning and Markov blankets discovery [1]. Both problems depend on the parent set assignment as a subroutine [18], and for given  $X_i$  can be solved exactly by traversing a lattice with  $n$  levels formed by the partial order *set inclusion* on the power set of  $\mathcal{X} - \{X_i\}$ . For given  $\mathcal{X}$  and  $D$ , queries of the form  $Query(X_i, Pa, F)$  are performed for each  $X_i$ , where  $Pa$  iterates over all possible subsets of  $\mathcal{X} - \{X_i\}$ , starting from empty set. Hence, at level  $i = 0, \dots, n - 1$  we have that  $|Pa| = i$ , and there are total  $\binom{n-1}{i}$  queries to execute, creating interesting pattern of queries that grow in size as computations progress. The function object  $F$  implements decomposable scoring function, e.g., MDL [19], BDeu [16], etc., that evaluates the assignment of  $Pa$  as parents of  $X_i$ .

We used all tested strategies to implement count-

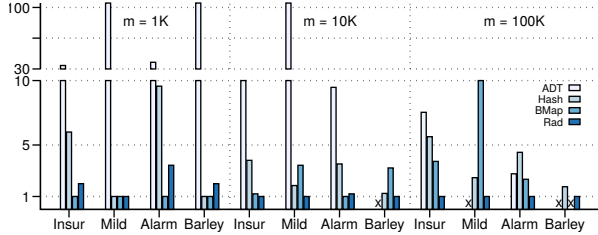


Figure 5: The total execution time of the parent set assignment solver with ADT, Hash, BMap and Rad strategies, normalized with respect to the fastest method. The solver was executed up to the level where  $|Pa| = 6$ .

ing queries in the open source parent set assignment solver [20]. The solver uses MDL scoring function, deploys several optimizations to eliminate some of the queries based on the results seen thus far, and because it effectively explores large combinatorial search space it has significant memory requirements. It also leverages OpenMP to execute multiple queries in parallel. As such, it serves as a practical benchmark for the query strategies. In our experiments, instead of considering all possible parent set sizes, as required by the exact solver, we limited the solver to  $|Pa| \leq 6$ , to make tests computationally feasible. This corresponds to a heuristic in which we make an assumption that no variable in the final Bayesian network can have more than six parents.

Figure 5 shows the total execution time of the solver for different input databases and query strategies. From the plots, we can see that our proposed strategies significantly outperform ADtree and HashTable, across all benchmarks. In fact, for datasets with high-arity variables, i.e., Mildew and Barley, the Radix strategy is 100 times faster than ADtree. This is explained by very large number of states that are to be expected in such datasets (and are costly to manage by ADtree), and by the pattern of how queries are generated by the solver. Because the size of the queries and their number grow together, there are only a few small queries that benefit ADtree, and increasing number of queries that are easily handled by the Radix strategy.

To illustrate how critical is the performance of counting queries for parent sets assignment, in Tab. 2 we report the total execution time of the solver, together with the fraction of the time taken by the queries, when running on databases with 100K instances. In all cases, the execution is dominated by database querying that accounts for 90%-99% of the total time. Interestingly, this fraction is smaller for ADtree than for other strategies, even though ADtree is slower (we observed this pattern in all test cases). We believe that this is because BMap and Rad are memory friendly, and have minimal effect on

Table 2: Execution time of the parent set assignment.

	Insur	Mild	Alarm	Barley
ADT	35m20s 90.2%	–	119m 98.2%	–
Hash	26m26s 98.2%	92m3s 99.1%	190m47s 99.4%	276m16s 98.2%
BMap	17m28s 98.8%	1107m57s 99.8%	100m55s 99.9%	–
Rad	4m41s 99.9%	37m28s 99.9%	43m4s 99.9%	156m32s 99.9%

memory utilization by the solver, thus minimizing cache update overheads, which in turn could slow down the solver. This is not the case for ADtree, which requires gigabytes of memory to run, and hence influences performance of the solver, affecting the ratio between the query and the solver time.

### 4.3 Queries in Association Rule Mining

Association rule mining is the classic method for establishing implication rules between a set of items in a database [2, 21]. Given a set of binary variables  $\mathcal{X}$ , and a database of transactions  $D$ , where  $D_i$  shows in which transactions item represented by  $X_i$  was involved (i.e.,  $X_i$  is in state 1), we want to identify rules of the form  $Pa(X_i) \Rightarrow X_i$  with support, i.e., how frequently  $Pa(X_i) \cup \{X_i\}$  are set together in  $D$ , and confidence, i.e., how frequently the rule is true in  $D$ , above some pre-defined thresholds. In the most direct form, the problem can be solved by traversing the power set lattice over  $\mathcal{X}$ , a query pattern similar to the one used by the parent set assignment solver. However, compared to the parent set assignment, the actual queries are simpler, since we only require the counts of query variables being in state 1. For example, to assess the rule  $\{X_1, X_2\} \Rightarrow X_3$  we would perform queries  $Count(X_1 = 1, X_2 = 1, X_3 = 1)$  and  $Count(X_1 = 1, X_2 = 1)$ . Consequently, instead of considering the query context, it is sufficient that our strategies search for one specific configuration of the query variables (as explained at the end of Sec. 3.3).

We used all four tested strategies to implement simple association rule mining engine based on the bottom-up search [21]. With the engine, we processed several large databases to enumerate rules with the support above 0.2 and confidence above 0.3, but of size less than seven. We selected these thresholds empirically to retrieve association rules with more than four variables, which allowed us to reliably measure the execution times (for smaller rules, the solver ran extremely fast). Results of this experiment are summarized in Fig. 6.

The figure shows that the Bitmap strategy significantly outperforms other approaches across all tested databases.

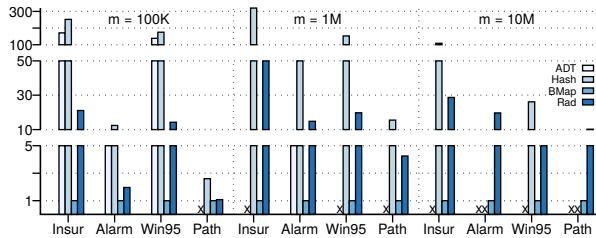


Figure 6: The total execution time of association rule mining with ADT, Hash, BMap and Rad strategies, normalized with respect to the fastest method. The solver was executed up to the level where  $|Pa| = 6$ .

Since in the Bitmap, processing queries with specific assignment amounts to a series of  $|Pa(X_i)| + 1$  bitmap intersections, followed by bit counting, the cost of queries becomes linear in the size of the query and the number of instances in the database. Moreover, because all bit-wise operations are implemented via SIMD extensions, BMap becomes much faster than Rad, which has the same asymptotic behavior but involves less cache friendly operations. Finally, the overheads of traversing ADtree and handling its MCV subtrees (see [12] for details) leads to its poor overall performance. At this point we should note that originally ADtree was designed for learning of association rules, however the design did not account for the memory and SIMD capabilities of modern processors.

## 5 RELATED WORK

As we mentioned through out the paper, counting queries in machine learning applications are often handled via some variant of the ADtree data index. The sparse ADtree [8, 12], which we used in our experiments, pre-computes and caches counts for all possible variable configurations. The counts are organized into a tree of *vary nodes*, encoding the choice of variables to facilitate fast searching, and *AD nodes* that store the actual query counts. To partially mitigate the excessive memory use, ADtrees do not explicitly represent most commonly occurring counts, and instead of creating AD nodes for counts lower than certain threshold, they resort to on-demand counting when such nodes are accessed. These base ideas have been extended by multiple researchers to account for dynamic data (i.e., updates to the database) [22], and to improve performance on high-arity data [13, 14]. However, as the core functionality in these data structures remains exactly the same, they suffer from the same limitations that we demonstrated in our experiments (expensive preprocessing, large memory footprint, significant traversing overheads).

Support for counting queries is a primary component in

any database management system. In such systems, the query mechanism must support conjunctive queries over multiple tables, and with a variety of possible query predicates [9]. Moreover, the queries are typically executed over tables that cannot be fully materialized in the main memory. Our Bitmap strategy can be viewed as a practical realization of the Leapfrog Trie Join [23] with an unary relation, under assumption that the entire database resides in the main memory.

The idea of using bitmaps to represent sets and their operations (e.g., intersection, cardinality, etc.) is frequent in software and databases design. This is because it allows to reduce memory, storage or network bandwidth, while maintaining the basic sets functionality [24]. In these applications, bitmaps are typically compressed following methods like for example RLE encoding or Roaring [25, 26]. The compressed bitmaps are orthogonal to our approach, and in fact we could use them to improve memory profile of our Bitmap strategy. However, as the compression induces computational overheads, and the size of the databases we consider practical is relatively small, currently we do not use compression.

## 6 CONCLUSIONS

In this paper, we describe efficient strategies for handling counting queries in machine learning applications. By combining convenient programming interface with memory efficient data traversing algorithms we are able to scale to large data instances, which we confirm via extensive experiments. The proposed solutions outperform and can substitute popular ADtree index. Moreover, to maintain best possible performance across different data instances, they can be selectively applied at the runtime depending on the properties of the queries.

While our approach is presented as a method for static databases, we note that it can be easily adopted to the cases where the input database expands with new instances during processing. This would amount to a simple update to the bitmaps in the Bitmap strategy, and is automatically handled in the Radix strategy.

### Acknowledgements

Authors wish to acknowledge hardware and technical support provided by the Center for Computational Research at the University at Buffalo. This work has been supported in part by the Department of Veterans Affairs under grant 7D0084, and by the National Institutes of Health under grant 1UL1 TR001412-01.



## References

- [1] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.
- [2] R. Agrawal, T. Imielinski, and A. Swami, “Mining association rules between sets of items in large databases,” in *ACM SIGMOD International Conference on Management of Data*, vol. 22, no. 2, 1993, pp. 207–216.
- [3] R. Kohavi, “Scaling up the accuracy of Naive-Bayes classifiers: A decision-tree hybrid,” in *International Conference on Knowledge Discovery and Data Mining*, 1996, pp. 202–207.
- [4] J. Quinlan, “Bagging, boosting, and C4.5,” in *AAAI Innovative Applications of Artificial Intelligence Conferences*, 1996, pp. 725–730.
- [5] H. Lee, R. Grosse, R. Ranganath, and A. Ng, “Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations,” in *International Conference on Machine Learning*. ACM, 2009, pp. 609–616.
- [6] R. Salakhutdinov and G. Hinton, “Deep Boltzmann machines,” in *International Conference on Artificial Intelligence and Statistics*, 2009, pp. 448–455.
- [7] J. Ramos, “Using TF-IDF to determine word relevance in document queries,” in *Instructional Conference on Machine Learning*, 2003, pp. 133–142.
- [8] A. Moore and M. Lee Soon, “Cached sufficient statistics for efficient machine learning with large datasets,” *Journal of Artificial Intelligence Research*, vol. 8, pp. 67–91, 1998.
- [9] G. Greco and F. Scarcello, “Counting solutions to conjunctive queries,” in *ACM Symposium on Principles of Database Systems*, 2014, pp. 132–143.
- [10] L. de Campos, “A scoring function for learning Bayesian networks based on mutual information and conditional independence tests,” *Journal of Machine Learning Research*, vol. 7, pp. 2149–2187, 2006.
- [11] J. Han, J. Pei, and M. Kamber, *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, 2011.
- [12] B. Anderson and A. Moore, “ADtrees for fast counting and for fast learning of association rules,” in *Conference on Knowledge Discovery and Data Mining*, 1998.
- [13] R. van Dam, I. Langkilde-Geary, and D. Ventura, “Adapting ADtrees for high arity features,” in *AAAI Conference on Artificial Intelligence*, 2008.
- [14] —, “Adapting ADtrees for improved performance on large datasets with high-arity features,” *Knowledge and Information Systems*, vol. 35, 2013.
- [15] C. Yuan, B. Malone, and X. Wu, “Learning optimal Bayesian networks using A\* search,” in *International Joint Conference on Artificial Intelligence*, 2011, pp. 2186–2191.
- [16] G. Cooper and E. Herskovits, “A Bayesian method for the induction of probabilistic networks from data,” *Machine Learning*, vol. 9, no. 4, pp. 309–347, 1992.
- [17] “Sparse AD-Tree package in Python,” <https://github.com/uraplutonium/adtree-py>.
- [18] M. Koivisto, “Parent assignment is hard for the MDL, AIC, and NML costs,” in *International Conference on Computational Learning Theory*, 2006, pp. 289–303.
- [19] G. Schwarz, “Estimating the dimension of a model,” *The Annals of Statistics*, vol. 6, pp. 461–464, 1978.
- [20] “SABNA – Scalable Accelerated Bayesian Network Analytics,” <https://gitlab.com/SCoRe-Group/SABNA>.
- [21] M. Zaki, “Scalable algorithms for association mining,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 12, no. 3, pp. 372–390, 2000.
- [22] J. Roure and A. Moore, “Sequential update of ADtrees,” in *International Conference on Machine Learning*, 2006, pp. 769–776.
- [23] T. Veldhuizen, “Leapfrog triejoin: A simple, worst-case optimal join algorithm,” *arXiv preprint arXiv:1210.0481*, 2012.
- [24] O. Kaser and D. Lemire, “Compressed bitmap indexes: Beyond unions and intersections,” *Software: Practice and Experience*, vol. 46, no. 2, pp. 167–198, 2016.
- [25] D. Lemire, O. Kaser, and K. Aouiche, “Sorting improves word-aligned bitmap indexes,” *Data and Knowledge Engineering*, vol. 69, no. 1, pp. 3–28, 2010.
- [26] D. Lemire, O. Kaser, N. Kurz, L. Deri, C. O’Hara, F. Saint-Jacques, and G. Ssi-Yan-Kai, “Roaring bitmaps: Implementation of an optimized software library,” *Software: Practice and Experience*, vol. 48, no. 4, pp. 867–895, 2018.

---

# A Dual Approach to Scalable Verification of Deep Networks

---

Krishnamurthy (Dj) Dvijotham\*, Robert Stanforth, Sven Gowal, Timothy Mann, Pushmeet Kohli  
DeepMind  
London, N1C 4AG, UK

## Abstract

This paper addresses the problem of formally verifying desirable properties of neural networks, *i.e.*, obtaining provable guarantees that neural networks satisfy specifications relating their inputs and outputs (robustness to bounded norm adversarial perturbations, for example). Most previous work on this topic was limited in its applicability by the size of the network, network architecture and the complexity of properties to be verified. In contrast, our framework applies to a general class of activation functions and specifications on neural network inputs and outputs. We formulate verification as an optimization problem (seeking to find the largest violation of the specification) and solve a Lagrangian relaxation of the optimization problem to obtain an upper bound on the worst case violation of the specification being verified. Our approach is *anytime* *i.e.* it can be stopped at any time and a valid bound on the maximum violation can be obtained. We develop specialized verification algorithms with provable tightness guarantees under special assumptions and demonstrate the practical significance of our general verification approach on a variety of verification tasks.

## 1 INTRODUCTION

Neural networks and deep learning have revolutionized machine learning achieving state of the art performance on a wide range of complex prediction tasks [Krizhevsky et al., 2012, Goodfellow et al., 2016]. However, in recent years, researchers have observed that even state of the art networks can be easily fooled into changing their

predictions by making small but carefully chosen modifications to the input data (known as *adversarial perturbations*) [Szegedy et al., 2013, Kurakin et al., 2016, Carlini and Wagner, 2017a, Goodfellow et al., 2014, Carlini and Wagner, 2017b]. While modifications to neural network training algorithms have been proposed to mitigate this phenomenon Madry et al. [2018], a comprehensive solution that is fully robust to adversarial attacks remains elusive [Carlini and Wagner, 2017b, Uesato et al., 2018].

Neural networks are typically tested using the standard machine learning paradigm: If the performance (accuracy) of the network is sufficiently high on a holdout (test) set that the network did not have access to while training, the network is deemed acceptable. This is justified by statistical arguments based on an *i.i.d.* assumption on the data generating mechanism, that is each input output pair is generated independently from the same (unknown) data distribution. However, this evaluation protocol is not sufficient in domains with critical safety constraints [Marston and Baca, 2015]. In these cases, we may require a stronger test: for example, we may require that the network is robust against adversarial perturbations within certain bounds.

**Adversarial evaluation.** In the context of adversarial examples, a natural idea is to test neural networks by checking if it is possible to generate an adversarial attack to change the label predicted by the neural network [Kurakin et al., 2016] and train them to be robust to these examples Madry et al. [2018]. Generating adversarial examples is a challenging computational task itself, and the attack generated by a specific attack algorithm may be far from optimal. This may lead one to falsely conclude that a given model is robust to attacks even though a stronger adversary may have broken the robustness. Recent work [Athalye et al., 2018, Uesato et al., 2018] has shown that evaluating models against weak adversaries can lead to incorrect conclusions regarding the robustness of the model. Thus, there is a need to go beyond evaluation us-

---

\*dvij@cs.washington.edu

ing specific adversarial attacks and find approaches that provide provable guarantees against attacks by *any adversary*.

**Towards verifiable models.** Verification of neural networks has seen significant research interest in recent years. In the formal verification community, Satisfiability Modulo Theory (SMT) solvers have been adapted for verification of neural networks [Ehlers, 2017, Huang et al., 2017, Katz et al., 2017]. While SMT solvers have been successfully applied to several domains, applying them to large neural networks remains a challenge due to the scale of the resulting SMT problem instances. Furthermore, these approaches have been largely limited to networks with piecewise linear activation functions since most SMT solvers are unable to deal efficiently with nonlinear arithmetic. More recently, researchers have proposed a set of approaches that make use of branch and bound algorithms either directly or via mixed-integer programming solvers [Bunel et al., 2017, Cheng et al., 2017, Tjeng and Tedrake, 2017]. While these approaches achieve strong results on smaller networks, scaling them to large networks remains an open challenge. These approaches also rely heavily on the piecewise linear structure of networks where the only nonlinearities are max-pooling and ReLUs.

**Towards scalable verification of general models.** In this paper, we develop a novel approach to neural network verification based on optimization and duality. The approach consists of formulating the verification problem as an optimization problem that tries to find the largest violation of the property being verified. If the largest violation is smaller than zero, we can conclude that the property being verified is true. By using ideas from duality in optimization, we can obtain bounds on the optimal value of this problem in a computationally tractable manner. Note that this approach is *sound but incomplete*, in that there may be cases where the property of interest is true, but the bound computed by our algorithm is not tight enough to prove the property. This strategy has been used in prior work as well [Kolter and Wong, 2018, Raghunathan et al., 2018]. However, our results improve upon prior work in the following ways:

1. Our verification approach applies to *arbitrary* feed-forward neural networks with *any architecture* and *any activation function* and our framework recovers previous results [Ehlers, 2017] when applied to the special case of piecewise linear activation functions.
2. We can handle verification of systems with discrete inputs and combinatorial constraints on the input space, including cardinality constraints.

3. The computation involved only requires solving an unconstrained convex optimization problem (of size linear in the number of neurons in the network), which can be done using a subgradient method efficiently. Further, our approach is *anytime*, in the sense that the computation can be stopped at any time and a valid bound on the verification objective can be obtained.
4. For the special case of single hidden layer networks, we develop specialized verification algorithms with provable tightness guarantees.
5. We attain state of the art verified bounds on adversarial error rates on image classifiers trained on MNIST and CIFAR-10 under adversarial perturbations in the infinity norm.

## 2 Related Work

**Certifiable training and verification of neural networks** A separate but related thread of work is on *certifiable training*, ie, training neural networks so that they are guaranteed to satisfy a desired property (for example, robustness to adversarial examples within a certain radius) [Kolter and Wong, 2018, Raghunathan et al., 2018]. These approaches use ideas from convex optimization and duality to construct bounds on an optimization formulation of verification. However, these approaches are limited to either a class of activation functions (piecewise linear models) or architectures (single hidden layer, as in Raghunathan et al. [2018]). Further, in Kolter and Wong [2018], the dual problem starts with a constrained convex formulation but is then converted into an unconstrained but nonconvex optimization problem to allow for easy optimization via a backprop-style algorithm. In contrast, our formulation allows for an unconstrained dual convex optimization problem so that for *any choice of dual variables*, we obtain a valid bound on the adversarial objective and this dual problem can be solved efficiently using subgradient methods.

We also note that the ultimate goals of [Kolter and Wong, 2018, Raghunathan et al., 2018] are different from our paper: they modify the training procedure of the neural network so that the network is trained to be easily verifiable. In contrast, our work focuses on extending verification algorithms to apply to a broader class of architectures, activation functions and - in this sense, we view our work as complementary to [Kolter and Wong, 2018, Raghunathan et al., 2018]. In fact, since the objective of our dual optimization is differentiable with respect to the network weights, we can extend our approach to training verifiable networks easily by simultaneously optimizing the network weights and dual variables to minimize the

dual objective. We leave the study of such an extension for future work.

**Theoretical analysis of robustness** Another related line of work has to do with theoretical analysis of adversarial examples. It has been shown that feedforward ReLU networks cannot learn to distinguish between points on two concentric spheres without necessarily being vulnerable to adversarial examples within a small radius [Gilmer et al., 2018]. Under a different set of assumptions, the existence of adversarial examples with high probability is also established in Fawzi et al. [2018]. In Wang et al. [2017], the authors study robustness of nearest neighbor classifiers to adversarial examples. As opposed to these theoretical analyses, our approach searches computationally for proofs of existence or non-existence of adversarial examples. The approach does not say anything a-priori about the existence of adversarial examples, but can be used to investigate their existence for a given network and compare strategies to guard against adversarial attacks.

### 3 VERIFICATION AS OPTIMIZATION

#### 3.1 NOTATION

Our techniques apply to general feedforward architectures and recurrent networks, but we focus on layered architectures for the development in this paper. The input layer is numbered 0, the hidden layers are numbered  $1, \dots, L - 1$  and the output layer is numbered  $L$ . The size of layer  $l$  is denoted  $n_l$

We denote by  $x^{in}$  the input to the neural network, by  $z^l$  the pre-activations of neurons at layer  $l$  before application of the activation function and by  $x^l$  the vector of neural activations after application of the activation function (to  $z^{l-1}$ ). For convenience, we define  $x^0 = x^{in}$ . We use  $x^l(x^{in}), z^l(x^{in})$  to denote the activations at the  $l$ -th layer as a function of the input  $x^{in}$ . Upper and lower bounds on the pre/post activations are denoted by  $\underline{x}^l, \bar{x}^l, \underline{z}^l, \bar{z}^l$  respectively. The activation function at layer  $l$  is denoted  $h^l$  and is assumed to be applied component-wise, ie,

$$[h^l(z^l)]_k = h_k^l(z_k^l)$$

Note that max-pooling is an exception to this rule - we discuss how max-pooling is handled separately in the Appendix section 6.2.1. The weights of the network at layer  $l$  are denoted  $W^l$  and the bias is denoted  $b^l$ ,  $z^l = W^l x^l + b^l$ .

#### 3.2 VERIFICATION PROBLEM

As mentioned earlier, *verification* refers to the process of checking that the output of the neural network sat-

isfies a certain desirable property for all choices of the input within a certain set. Formally, this can be stated as follows:

$$\forall x^{in} \in \mathcal{S}_{in}(x^{nom}) \quad x^L(x^{in}) \in \mathcal{S}_{out} \quad (1)$$

where  $x^{in}$  denotes the input to the network,  $x^{nom}$  denotes a nominal input,  $\mathcal{S}_{in}(x^{nom})$  defines the constrained subset of inputs induced by the nominal input, and  $\mathcal{S}_{out}$  denotes the constraints on the output that we would like to verify are true for all inputs in  $\mathcal{S}_{in}(x^{nom})$ . In the case of adversarial perturbations in image classification,  $x^{nom}$  would refer to the nominal (unperturbed image),  $\mathcal{S}_{in}(x^{nom})$  would refer to all the images that can be obtained by adding bounded perturbations to  $x^{nom}$ , and  $x^{in}$  would refer to a perturbed image.

In this paper, we will assume that:  $\mathcal{S}_{out}$  is always a described by a finite set of linear constraints on the values of final layer ie.  $\mathcal{S}_{out} = \cap_{i=1}^m \{x^L : (c^i)^T x^L + d^i \leq 0\}$ , and  $\mathcal{S}_{in}(x^{nom})$  is any bounded set such that any linear optimization problem of the form

$$\max_{x^{in} \in \mathcal{S}_{in}(x^{nom})} c^T x$$

can be solved efficiently. This includes convex sets and also sets describing combinatorial structures like spanning trees, cuts in a graph and cardinality constraints.

See the following examples for a concrete illustration of the formulation of the problem:

**Robustness to targeted adversarial attacks.** Consider an adversarial attack that seeks to perturb an input  $x^{nom}$  to an input  $x^{in}$  subject to a constraint on the perturbation  $\|x^{in} - x^{nom}\| \leq \epsilon$  to change the label from the true label  $i$  to a target label  $j$ . We can map this to (1) as follows:

$$\mathcal{S}_{in}(x^{nom}) = \{x^{in} : \|x^{in} - x^{nom}\| \leq \epsilon\}, \quad (2a)$$

$$\mathcal{S}_{out} = \{z : c^T z \leq 0\} \quad (2b)$$

where  $c$  is a vector with  $c_j = 1, c_i = -1$  and all other components 0. Thus,  $\mathcal{S}_{out}$  denotes the set of outputs for which the true label  $i$  has a higher logit value than the target label  $j$  (implying that the targeted adversarial attack did not succeed).

**Monotonic predictors.** Consider a network with a single real valued output and we are interested in ensuring that the output is monotonically increasing wrt each dimension of the input  $x^{in}$ . We can state this as a verifica-

tion problem:

$$\mathcal{S}_{in}(x^{nom}) = \{x^{in} : x^{in} \geq x^{nom}\} \quad (3a)$$

$$\mathcal{S}_{out} = \{x^L : x^L(x^{nom}) - x^L \leq 0\} \quad (3b)$$

Thus,  $\mathcal{S}_{out}$  denotes the set of outputs which are large than the network output at  $x^{nom}$ . If this is true for each value of  $x^{nom}$ , then the network is monotone.

**Cardinality constraints.** In several cases, it makes sense to constrain a perturbation not just in norm but also in terms of the number of dimensions of the input that can be perturbed. We can state this as:

$$\begin{aligned} \mathcal{S}_{in}(x^{nom}) = \\ \{x^{in} : \|x^{in} - x^{nom}\|_0 \leq k, \|x^{in} - x^{nom}\|_\infty \leq \epsilon\} \end{aligned} \quad (4a)$$

$$\mathcal{S}_{out} = \{z : c^T z \leq 0\} \quad (4b)$$

where  $\|x\|_0$  denotes the number of non-zero entries in  $x$ . Thus,  $\mathcal{S}_{out}$  denotes the set of outputs which are larger than the network output at  $x^{nom}$ . If this is true for each value of  $x^{nom}$ , then the network is monotone.

### 3.3 OPTIMIZATION PROBLEM FOR VERIFICATION

Once we have a verification problem formulated in the form (1), we can easily turn the verification procedure into an optimization problem. This is similar to the optimization based search for adversarial examples [Szegedy et al., 2013] when the property being verified is adversarial robustness. For brevity, we only consider the case where  $\mathcal{S}_{out}$  is defined by a single linear constraint  $c^T z + d \leq 0$ . If there are multiple constraints, each one can be verified separately.

$$\max_{\substack{z^0, \dots, z^{L-1} \\ x^0, \dots, x^L}} c^T x^L + d \quad (5a)$$

$$\text{s.t. } x^{l+1} = h^l(z^l), l = 0, 1, \dots, L-1 \quad (5b)$$

$$z^l = W^l x^l + b^l, l = 0, 1, \dots, L-1 \quad (5c)$$

$$x^0 = x^{in}, x^{in} \in \mathcal{S}_{in}(x^{nom}) \quad (5d)$$

If the optimal value of this problem is smaller than 0 (for each  $c, d$  in the set of linear constraints defining  $\mathcal{S}_{out}$ ), we have verified the property (1). This is a nonconvex optimization problem and finding the global optimum in general is NP-hard (see Appendix section 6.4.1 for a proof). However, if we can compute *upper bounds* on the value of the optimization problem and the upper bound is smaller than 0, we have successfully verified the property. In the following section, we describe our main approach for computing bounds on the optimal value of (5).

### 3.4 BOUNDING THE VALUE OF THE OPTIMIZATION PROBLEM

We assume that bounds on the activations  $z^l, x^l, l = 0, \dots, L-1$  are available. Section 6.1 discusses details of how such bounds may be obtained given the constraints on the input layer  $\mathcal{S}_{in}(x^{nom})$ . We can bound the optimal value of (5) using a Lagrangian relaxation of the constraints:

$$\begin{aligned} \max_{\substack{z^0, \dots, z^{L-1} \\ x^0, x^1, \dots, x^{L-1}}} c^T (h^{L-1}(z^{L-1})) + d \\ + \sum_{l=0}^{L-1} (\mu^l)^T (z^l - W^l x^l - b^l) \\ + \sum_{l=0}^{L-2} (\lambda^l)^T (x^{l+1} - h^l(z^l)) \end{aligned} \quad (6a)$$

$$\text{s.t. } \underline{z}^l \leq z^l \leq \bar{z}^l, l = 0, 1, \dots, L-1 \quad (6b)$$

$$\underline{x}^l \leq x^l \leq \bar{x}^l, l = 0, 1, \dots, L-1 \quad (6c)$$

$$x^0 \in \mathcal{S}_{in}(x^{nom}) \quad (6d)$$

Note that any feasible solution for the original problem (5) is feasible for the above problem, and for any such solution, the terms involving  $\lambda, \mu$  become 0 (since the terms multiplying  $\lambda, \mu$  are 0 for every feasible solution). Thus, for any choice of  $\lambda, \mu$ , the above optimization problem provides a valid upper bound on the optimal value of (5) (this property is known as weak duality [Vandenberghe and Boyd, 2004]).

We now look at solving the above optimization problem. Since the objective and constraints are separable in the layers, the variables in each layer can be optimized independently. For  $l = 1, \dots, L-1$ , we have

$$\begin{aligned} f_l(\lambda^{l-1}, \mu^l) = \\ \max_{x^l \in [\underline{x}^l, \bar{x}^l]} \left( \lambda^{l-1} - (W^l)^T \mu^l \right)^T x^l - (b^l)^T \mu^l \end{aligned}$$

which can be solved trivially by setting each component of  $x^l$  to its upper or lower bound depending on whether the corresponding entry in  $\lambda^{l-1} - (W^l)^T \mu^l$  is non-negative. Thus,

$$\begin{aligned} f_l(\lambda^{l-1}, \mu^l) = \\ \left[ \lambda^{l-1} - (W^l)^T \mu^l \right]_+^T \bar{x}^l \\ + \left[ \lambda^{l-1} - (W^l)^T \mu^l \right]_-^T \underline{x}^l - (b^l)^T \mu^l \end{aligned}$$

where  $[x]_+ = \max(x, 0)$ ,  $[x]_- = \min(x, 0)$  denote the positive and negative parts of  $x$ .

Similarly, collecting the terms involving  $z^l$ , we have, for  $l = 0, \dots, L-1$

$$\tilde{f}_l(\lambda^l, \mu^l) = \max_{z^l \in [\underline{z}^l, \bar{z}^l]} \mu^{lT} z^l - (\lambda^l)^T h^l(z^l)$$

where  $\lambda_{L-1} = -c$ .

Since  $h^l$  is a component-wise nonlinearity, each dimension of  $z^l$  can be optimized independently. For the  $k$ -th dimension, we obtain

$$\tilde{f}_{l,k}(\lambda_k^l, \mu_k^l) = \max_{z_k^l \in [\underline{z}_k^l, \bar{z}_k^l]} \mu_k^l z_k^l - \lambda_k^l h_k^l(z_k^l)$$

This is a one-dimensional optimization problem and can be solved easily- for common activation functions (ReLU, tanh, sigmoid, maxpool), it can be solved analytically, as discussed in appendix section 6.2. Finally, we need to solve

$$f_0(\mu^0) = \max_{x^0 \in \mathcal{S}_{in}(x^{nom})} \left( -(W^0)^T \mu_0 \right)^T x^0 - (b^0)^T \mu^0$$

which can also be solved easily given the assumption on  $\mathcal{S}_{in}$ . We work out some concrete cases in 6.3.

Once these problems are solved, we can construct the dual optimization problem:

$$\begin{aligned} \min_{\lambda, \mu} & \sum_{k=0}^{n_{L-1}} \tilde{f}_{L-1,k}(-c_k, \mu_k^L) + \sum_{l=0}^{L-2} \sum_{k=0}^{n_l} \tilde{f}_{l,k}(\lambda_k^l, \mu_k^l) \\ & + \sum_{l=1}^{L-1} f_l(\lambda^{l-1}, \mu^l) + f_0(\mu^0) + d \end{aligned} \quad (7)$$

This seeks to choose the values of  $\lambda, \mu$  so as to minimize the upper bound on the verification objective, thereby obtaining the tightest bound on the verification objective.

This optimization can be solved using a subgradient method on  $\lambda, \mu$ .

**Theorem 1.** *For any values of  $\lambda, \mu$ , the objective of (7) is an upper bound on the optimal value of (5). Hence, the optimal value of (7) is also an upper bound. Further, (7) is a convex optimization problem in  $(\lambda, \mu)$ .*

*Proof.* The upper bound property follows from weak duality [Vandenberghe and Boyd, 2004]. The fact that (7) is a convex optimization problem can be seen as each term  $f_l, \tilde{f}_{l,k}$  is expressed as a maximum over a set of linear functions of  $\lambda, \mu$  [Vandenberghe and Boyd, 2004].  $\square$

**Theorem 2.** *If each  $h$  is a ReLU function, then (7) is equivalent to the dual of the LP described in Ehlers [2017].*

*Proof.* See section 6.4.  $\square$

The LP formulation from Ehlers [2017] is also used in Kolter and Wong [2018]. The dual of the LP is derived in Kolter and Wong [2018] - however this dual is different from (7) and ends up with a constrained optimization formulation for the dual (the details of this can be found in appendix section 6.4.2). To allow for an unconstrained formulation, this dual LP is transformed to a backpropagation-like computation. While this allows for folding the verification into training, it also introduces nonconvexity in the verification optimization - our formulation of the dual differs from Kolter and Wong [2018] in that we directly solve an unconstrained dual formulation, allowing us to circumvent the need to solve a non-convex optimization for verification.

### 3.5 TOWARDS THEORETICAL GUARANTEES FOR VERIFICATION

The bounds computed by solving (7) could be loose in general, since (5) is an NP-hard optimization problem (section 6.4.1). SMT solvers and MIP solvers are guaranteed to find the exact optimum for piecewise linear neural networks, however, they may take exponential time to do so. Thus, an open question remains: Are there cases where it is possible to perform *exact verification* efficiently? If not, can we approximate the verification objective to within a certain factor (known a-priori)? We develop results answering these questions in the following sections.

*Prior work:* For any linear classifier, the scores of each label are a linear function of the inputs  $w_i^T x + b_i$ . Thus, the difference between the predictions of two classes  $j$  (target class for an adversary) and class  $i$  (true label) is  $(w_i - w_j)^T x$ . Maximizing this subject to  $\|x - x^0\|_2 \leq \epsilon$  can be solved analytically to obtain the value  $(w_i - w_j)^T x^0 + \|w_i - w_j\|_2 \epsilon$ . This observation formed the basis for algorithms in [Ragunathan et al., 2018] and [Hein and Andriushchenko, 2017]. However, once we move to nonlinear classifiers, the situation is not so simple and computing the worst case adversarial example, even in the 2-norm case, becomes a challenging task. In [Hein and Andriushchenko, 2017], the special case of kernel methods and single hidden layer classifiers are considered, but the approaches developed are only upper bounds on the verification objective (just like those computed by our dual relaxation approach). Similarly, in Ragunathan et al. [2018], a semidefinite programming approach is developed to compute bounds on the verification objective for the special case of adversarial perturbations on the infinity norm. However, none of these approaches come with a-priori guarantees on the quality of the bound, that is, before actually running the verification algorithm, one cannot predict how tight the

bound on the verification objective would be. In this section, we develop novel theoretical results that quantify when the verification problem (5) can be solved *either exactly or with a-priori approximation guarantees*. Our results require strong assumptions and do not immediately apply to most practical situations. However, we believe that they shed some understanding on the conditions under which exact verification can be performed tractably and lead to specialized verification algorithms that merit further study.

We assume the following for all results in this section:

- 1) We study networks with a single hidden layer, *i.e.*  $L = 2$ , with activation function  $h^0 = h$  and a linear mapping from the penultimate to the output layer  $x^2 = h^1(z^1) = z^1 = W^1 x^1 + b^1$ .
- 2) The network has a differentiable activation function  $h$  with Lipschitz-continuous derivatives denoted  $h'$  (tanh, sigmoid, ELU, polynomials satisfy this requirement).
- 3)  $\mathcal{S}_{in}(x^{nom}) = \{x^{in} : \|x^{in} - x^{nom}\|_2 \leq \epsilon\}$ .

Since the output layer is a linear function of the penultimate layer  $x^1$ , we have

$$\begin{aligned} c^T x^2 &= \left( (W^1)^T c \right)^T x^1 + c^T b^1 \\ &= \left( (W^1)^T c \right)^T h^0(z^0) + c^T b^1 \end{aligned}$$

For brevity, we simply denote  $(W^1)^T c$  as  $c$ , drop the constant term  $c^T b^1$  and let  $W = W^0$ ,  $b = b^0$ ,  $z^{nom} = W x^{nom} + b$  and  $W_i$  denote the  $i$ -th row of  $W$ . Then, (5) reduces to:

$$\max_{x^{in}: \|x^{in} - x^{nom}\|_2 \leq \epsilon} \sum_i c_i h_i(W_i x^{in} + b_i) \quad (8)$$

**Theorem 3.** *Suppose that  $h$  has a Lipschitz continuous first derivative:*

$$h'_i(t) - h'_i(\tilde{t}) \leq \gamma_i |t - \tilde{t}|$$

Let

$$\begin{aligned} \nu &= \|\text{diag}(c) W^T h'(z^{nom})\|_2 \\ L &= \sigma_{\max}(\text{diag}(c) W^T) \sigma_{\max}(\text{diag}(\gamma) W) \end{aligned}$$

Then  $\forall \epsilon \in (0, \frac{\nu}{2L})$ , the iteration:

$$x^{k+1} \leftarrow x^{nom} + \epsilon \left( \frac{W^T \text{diag}(c) h'(W x^k + b)}{\|W^T \text{diag}(c) h'(W x^k + b)\|_2} \right)$$

starting at  $x^0 = x^{nom}$  converges to the global optimum  $x^*$  of (8) at the rate  $\|x^k - x^*\| \leq \left( \frac{\epsilon L}{\nu - \epsilon L} \right)^k$

*Proof.* Section 6.4 □

Thus, when  $\epsilon$  is small enough, a simple algorithm exists to find the global optimum of the verification objective. However, even when  $\epsilon$  is larger, one can obtain a good approximation of the verification objective. In order to do this, consider the following quadratic approximation of the objective from (8):

$$\begin{aligned} \max \sum_i c_i (h_i(z_i^{nom}) + h'_i(z_i^{nom})(W_i z)) \\ + \sum_i \frac{c_i}{2} h''_i(z_i^{nom})(W_i z)^2 \end{aligned} \quad (9a)$$

$$\text{s.t. } \|z\|_2 \leq \epsilon \quad (9b)$$

This optimization problem corresponds to a trust region problem that can be solved to global optimality using semidefinite programming [Yakubovic, 1971]:

$$\begin{aligned} \max_{z, Z} \sum_i c_i (h_i(z_i^{nom}) + h'_i(z_i^{nom})(W_i z)) \\ + \sum_i \frac{c_i}{2} h''_i(z_i^{nom}) \text{tr}(W_i^T W_i Z) \end{aligned} \quad (10a)$$

$$\text{s.t. } \text{tr}(Z) \leq \epsilon, \begin{pmatrix} 1 & z^T \\ z & Z \end{pmatrix} \succeq 0 \quad (10b)$$

where  $X \succeq 0$  denotes that  $X$  is constrained to be a positive semidefinite matrix. While this can be solved using general semidefinite programming solvers, several special purpose algorithms exist for this trust region problem that can exploit its particular structure for efficient solution. [Hazan and Koren, 2016]

**Theorem 4.** *Suppose that  $h$  is thrice-differentiable with a globally bounded third derivative. Let*

$$\zeta_i = \|W_i\|_2, \eta_i = \sup_t |h_i'''(t)|, \kappa = \frac{1}{6} \left( \sum_i \eta_i c_i \zeta_i^3 \right)$$

For each  $\epsilon > 0$ , the difference between the optimal values of (10), (8) is at most  $\kappa \epsilon^3$ .

*Proof.* See Section 6.4 □

## 4 EXPERIMENTS

In this section, we present numerical studies validation our approach on three sets of verification tasks:

*Image classification on MNIST and CIFAR:* We use our approach to obtain guaranteed lower bounds on the accuracy of image classifiers trained on MNIST and CIFAR-10 under adversarial attack with varying sizes of the perturbation radius. We compare the bounds obtained by our method with prior work (in cases where prior work is applicable) and also with the best attacks found by various approaches.

*Classifier stability on GitHub data:* We train networks on sequences of commits on GitHub over a collection of 10K repositories - the prediction task consists of predicting whether a given repository will reach more than 40 commits within 250 days given data observed until a certain day. Input features consist a value between 0 and 1 indicating the number of days left until the 250th day, as well as another value indicating the progress of commits towards the total of 40. As the features evolve, the prediction of the classifier changes (for example, predictions should become more accurate as we move closer to the 250th day). In this situation, it is desirable that the classifier provides consistent predictions and that the number of times its prediction switches is as small as possible. It is also desirable that this switching frequency cannot be easily be changed by perturbing input features. We use our verification approach combined with dynamic programming to compute a bound on the maximum number of switches in the classifier prediction over time.

*Digit sum task:* We consider a more complex verification task here: Given a pair of MNIST digits, the goal is to bound how much the sum of predictions of a classifier can differ from the true sum of those digits under adversarial perturbation subject to a total budget on the perturbation across the two digits.

#### 4.1 IMAGE CLASSIFICATION: MNIST AND CIFAR

We study adversarial attacks subject to an  $l_\infty$  bound on the input perturbation. An adversarial example (AE) is a perturbation of an input of the neural network such that the output of the neural network differs from the correct label for that input. An AE is said to be within radius  $\epsilon$  if the  $l_\infty$  norm of the difference between the AE and the original input is smaller than  $\epsilon$ . We are interested in the *adversarial error rate*, that is,

$$\frac{\# \text{ Test examples that have an AE within radius } \epsilon}{\text{Size of test set}}$$

Computing this quantity precisely requires solving the NP-hard problem (5) for each test example, but we can obtain upper bounds on it using our (and other) verification methods and lower bounds using a fixed attack algorithm (in this paper we use a bound constrained LBFGS algorithm similar to [Carlini and Wagner, 2017b]). Since theorem 2 shows that for the special case of piecewise linear neural networks, our approach reduces to the basic LP relaxation from Ehlers [2017] (which also is the basis for the algorithms in Bunel et al. [2017] and Kolter and Wong [2018]), we focus on networks with smooth nonlinearities like tanh and sigmoid. We compare our approach with The SDP formulation from Raghunathan et al. [2018] (note that this approach only works for sin-

gle hidden layer networks, so we just show it as producing vacuous bounds for other networks).

Each approach gets a budget of 300 s per verification problem (choice of test example and target label). Since the SDP solver from [Raghunathan et al., 2018] only needs to be run once per label pair (and not per test example), its running time is amortized appropriately.

*Results on smooth activation functions:* Figures 1a,1b show that our approach is able to compute nearly tight bounds (bounds that match the upper bound) for small perturbation radii (up to 2 pixel units) and our bounds significantly outperform those from the SDP approach [Raghunathan et al., 2018] ( which is only able to compute nontrivial bounds for the smallest model with 20 hidden units).

*Results on models trained adversarially:* We use the adversarial training approach of [Uesato et al., 2018] and train models on MNIST and CIFAR that are robust to perturbations from the LBFGS-style attack on the training set. We then apply our verification algorithm to these robust models and obtain bounds on the adversarial error rate on the test set. These models are all multilayer models, so the SDP approach from [Raghunathan et al., 2018] does not apply and we do not plot it here. We simply plot the attack versus the bound from our approach. The results for MNIST are plotted in figure 2b and for CIFAR in figure 2a. On networks trained using a different procedure, the approaches from Raghunathan et al. [2018] and Kolter and Wong [2018] are able to achieve stronger results for larger values of  $\epsilon$  (they work with  $\epsilon = .1$  in real units, which corresponds to  $\epsilon = 26$  in pixel units we use here). However, we note that our verification procedure is agnostic to the training procedure, and can be used to obtain fairly tight bounds for any network and any training procedure. In comparison, the results in Kolter and Wong [2018] and Raghunathan et al. [2018] rely on the training procedure optimizing the verification bound. Since we do not rely on a particular adversarial training procedure, we were also able to obtain the first non-trivial verification bounds on CIFAR-10 (to the best of our knowledge) shown in figure 2a. While the model quality is rather poor, the results indicate that our approach could scale to more complicated models.

#### 4.2 GITHUB CLASSIFIER STABILITY

We allow the adversary to modify input features by up to 3% (at each timestep) and our goal is to bound the maximum number of prediction switches induced by each attack over time. We can model this within our verification framework as follows: Given a sequence of input features, we compute the maximum number of switches



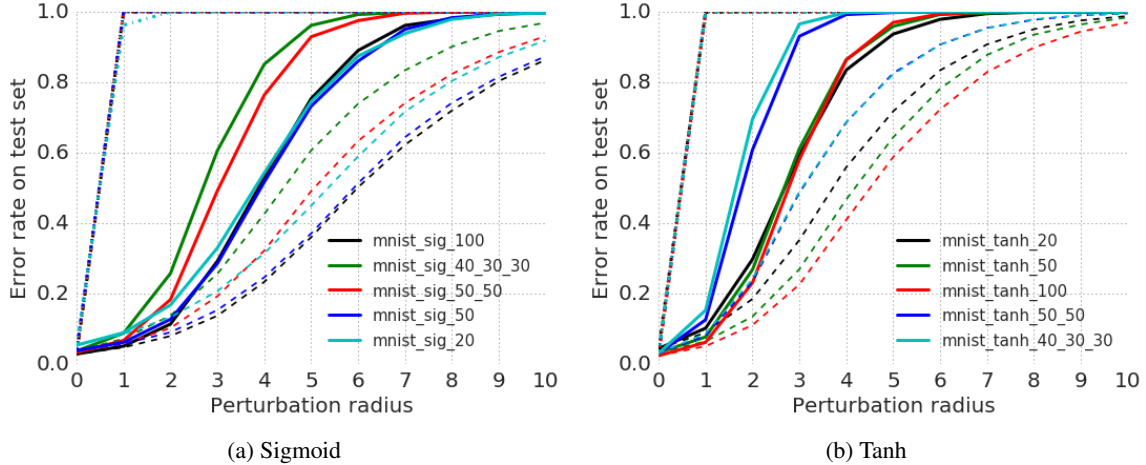


Figure 1: Figures show three curves per model - Dashed line: Lower bound from LBFSG attack. Solid line: Our verified upper bound. Dash-dot line: SDP verified upper bound from [Raghunathan et al., 2018]. Each color represents a different network. The dashed lines at the bottom are lower bounds on the error rate computed using the best attack found using the LBFSG algorithm.

achievable by first computing the target classes that are reachable through an adversarial attack at each timestep (using (7)), and then running a dynamic program to compute the choices of target classes over time (from within the reachable target classes) to maximize the number of switches over time.

Figure 3a shows how initially predictions are easily attackable (as little information is available to make predictions), and also shows how the gap between our approach and the best attack found using the LBFSG algorithm evolves over time.

### 4.3 COMPLEX VERIFICATION TASK: DIGIT SUM

In order to test our approach on a more complex specification, we study the following task: Given a pair of MNIST digits, we ask the question: Can an attacker perturb each image, subject to a constraint on the total perturbation across both digits, such that the sum of the digits predicted by the classifier differs from the true sum of those digits by as large an amount as possible? Answering this question requires solving the following optimization problem:

$$\begin{aligned} & \max_{\substack{x_a^{in}, x_b^{in} \\ \epsilon_a, \epsilon_b}} |\operatorname{argmax}(x^L(x_a^{in})) + \operatorname{argmax}(x^L(x_b^{in})) - s| \\ & \text{s.t. } \|x_a^{in} - x_a^{nom}\| \leq \epsilon_a, \|x_b^{in} - x_b^{nom}\| \leq \epsilon_b \\ & \quad \epsilon_a + \epsilon_b \leq \epsilon \end{aligned}$$

where  $s$  is the true sum of the two digits. Thus, the adversary has to decide on both the perturbation to each digit,

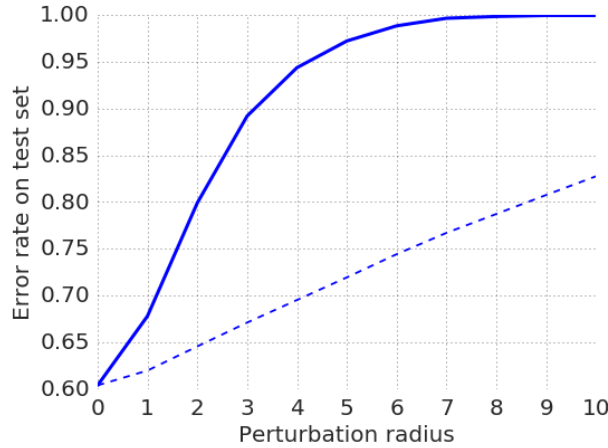
as well as the size of the perturbation. We can encode this within our framework (we skip the details here). The upper bound on the maximum error in the predicted sum from the verification and the lower bound on the maximum error computed from an attack for this problem (on an adversarially trained two hidden layer sigmoid network) is plotted in figure 3b. The results show that even on this rather complex verification task, our approach is able to compute tight bounds.

## 5 CONCLUSIONS

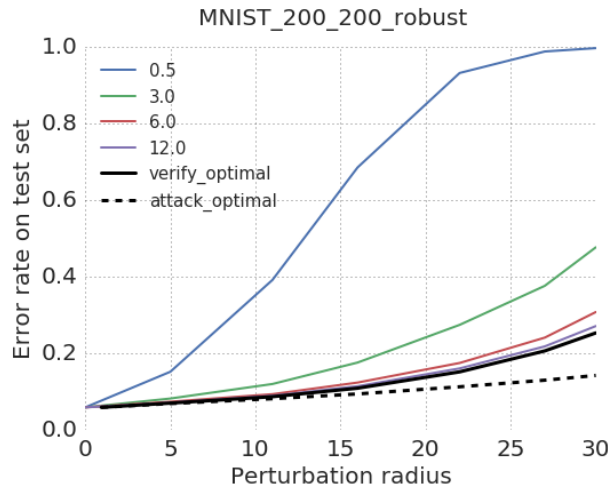
We have presented a novel framework for verification of neural networks. Our approach extends the applicability of verification algorithms to arbitrary feedforward networks with any architecture and activation function and to more general classes of input constraints than those considered previously (like cardinality constraints). The verification procedure is both efficient (given that it solves an unconstrained convex optimization problem) and practically scalable (given its anytime nature only required gradient like steps). We proved the first known (to the best of our knowledge) theorems showing that under special assumptions, nonlinear neural networks can be verified tractably. Numerical experiments demonstrate the practical performance of our approach on several classes of verification tasks.

## ACKNOWLEDGEMENTS

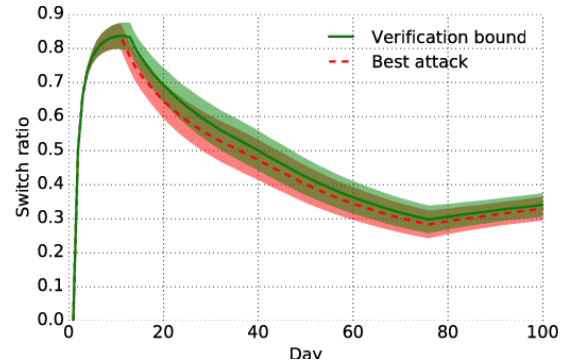
The authors would like to thank Brendan O’Donoghue, Csaba Szepesvari, Rudy Bunel, Jonathan Uesato and



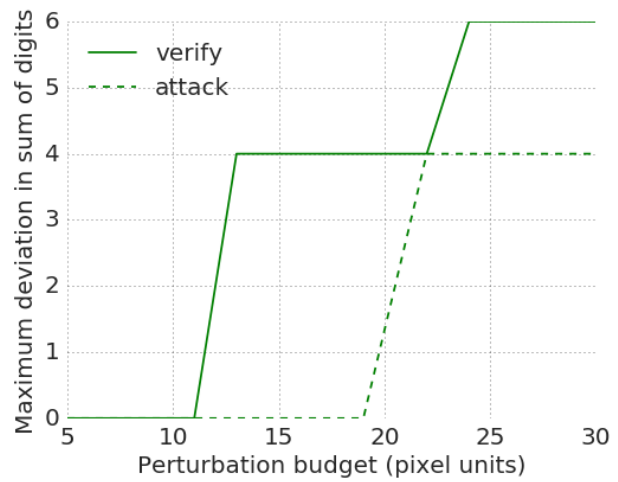
(a) Verification of robust CIFAR-10 model as a function of perturbation radius  $\epsilon$ .



(b) Verification of robust models as a function of perturbation radius  $\epsilon$ . Solid lines:verified upper bounds and dashed lines: attack lower bounds.



(a) Maximum number of switches (over number of days evaluated so far) over different days (upper bound from our verification approach and lower bound from the best attack found). Results are averaged over 100 unseen repositories. Shaded areas are 95% confidence intervals.



(b) Maximum difference between predicted and actual sum of digits vs the perturbation budget for a pair of randomly chosen MNIST images.

Shane Legg for helpful comments and feedback on this paper. Jonathan Uesato's help on adversarial training of neural networks is also gratefully acknowledged.

## References

A. Athalye, N. Carlini, and D. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, 2018.

R. Bunel, I. Turkaslan, P. H. Torr, P. Kohli, and M. P. Kumar. Piecewise linear neural network verification: A comparative study. *arXiv preprint arXiv:1711.00455*, 2017.

N. Carlini and D. Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In

*Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 3–14. ACM, 2017a.

N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE, 2017b.

C.-H. Cheng, G. Nührenberg, and H. Ruess. Maximum resilience of artificial neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, pages 251–268. Springer, 2017.

R. Ehlers. Formal Verification of Piece-Wise Linear Feed-Forward Neural Networks. *ArXiv e-prints*, May 2017.

A. Fawzi, H. Fawzi, and O. Fawzi. Adversarial vulnerability for any classifier. *arXiv preprint arXiv:1802.08686*, 2018.

- J. Gilmer, L. Metz, F. Faghri, S. S. Schoenholz, M. Raghu, M. Wattenberg, and I. Goodfellow. Adversarial spheres. *ArXiv e-prints*, Jan. 2018.
- I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.
- E. Hazan and T. Koren. A linear-time algorithm for trust region problems. *Mathematical Programming*, 158(1-2):363–381, 2016.
- M. Hein and M. Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *Advances in Neural Information Processing Systems*, pages 2263–2273, 2017.
- X. Huang, M. Kwiatkowska, S. Wang, and M. Wu. Safety verification of deep neural networks. In *International Conference on Computer Aided Verification*, pages 3–29. Springer, 2017.
- G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In R. Majumdar and V. Kunčak, editors, *Computer Aided Verification*, pages 97–117, Cham, 2017. Springer International Publishing. ISBN 978-3-319-63387-9.
- J. Z. Kolter and E. Wong. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, 2018.
- A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.
- A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.
- M. Marston and G. Baca. Acas-xu initial self-separation flight tests. *NASA Armstrong Flight Research Center Flight Report*, 2015.
- B. Polyak. The convexity principle and its applications. *Bulletin of the Brazilian Mathematical Society*, 34(1): 59–75, 2003.
- A. Raghunathan, J. Steinhardt, and P. Liang. Certified defenses against adversarial examples. In *International Conference on Learning Representations*, 2018.
- URL <https://openreview.net/forum?id=Bys4ob-Rb>.
- C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.
- V. Tjeng and R. Tedrake. Verifying neural networks with mixed integer programming. *arXiv preprint arXiv:1711.07356*, 2017.
- M. Udell and S. Boyd. Maximizing a sum of sigmoids. *Optimization and Engineering*, 2013.
- J. Uesato, B. O’Donoghue, A. van den Oord, and P. Kohli. Adversarial risk and the dangers of evaluating against weak attacks. In *International Conference on Machine Learning*, 2018.
- L. Vandenberghe and S. Boyd. *Convex optimization*, volume 1. Cambridge University Press Cambridge, 2004.
- Y. Wang, S. Jha, and K. Chaudhuri. Analyzing the robustness of nearest neighbors to adversarial examples. *arXiv preprint arXiv:1706.03922*, 2017.
- V. Yakubovic. S-procedure in nonlinear control theory. *Vestnik Leningrad Univ.*, 1:62–77, 1971.

---

# Understanding Measures of Uncertainty for Adversarial Example Detection

---

**Lewis Smith**

Department of Engineering Science  
University of Oxford  
Oxford, United Kingdom

**Yarin Gal**

Department of Computer Science  
University of Oxford  
Oxford, United Kingdom

## Abstract

Measuring uncertainty is a promising technique for detecting adversarial examples, crafted inputs on which the model predicts an incorrect class with high confidence. There are various measures of uncertainty, including predictive entropy and mutual information, each capturing distinct types of uncertainty. We study these measures, and shed light on why mutual information seems to be effective at the task of adversarial example detection. We highlight failure modes for MC dropout, a widely used approach for estimating uncertainty in deep models. This leads to an improved understanding of the drawbacks of current methods, and a proposal to improve the quality of uncertainty estimates using probabilistic model ensembles. We give illustrative experiments using MNIST to demonstrate the intuition underlying the different measures of uncertainty, as well as experiments on a real-world Kaggle dogs vs cats classification dataset.

## 1 INTRODUCTION

Deep neural networks are state of the art models for representing complex, high dimensional data such as natural images. However, neural networks are not robust: there exist small perturbations to the input of the network which produce erroneous and over-confident classification results. These perturbed inputs, known as adversarial examples (Szegedy et al., 2013), are a major hurdle for the use of deep networks in safety-critical applications, or those for which security is a concern.

One possible hypothesis for the existence of adversarial examples is that such images lie off the manifold of natural images, occupying regions where the model makes unconstrained extrapolations. If this hypothesis were to hold true, then one could detect adversarial perturbation

by measuring the distance of the perturbed input to the image manifold.

Hypothetically, such distances could be measured using nearest neighbour approaches, or by assessing the probability of the input under a density model on image space. However, approaches based on geometric distance are a suboptimal choice for images, as pixel-wise distance is a poor metric for perceptual similarity; similarly, density modelling is difficult to scale to the high dimensional spaces found in image recognition.

Instead, we may consider proxies to the distance from the image manifold. For example, the model uncertainty of a *discriminative* Bayesian classification model should

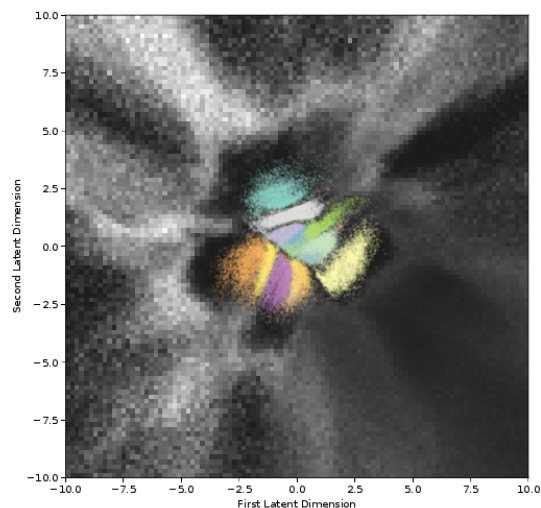


Figure 1: Uncertainty of a standard dropout network trained on MNIST, as measured by *mutual information*, visualized in the latent space obtained from a variational autoencoder. Colours are classes for each encoded training image. The background shows uncertainty, calculated by decoding each latent point into image space, and evaluating the mutual information between the decoded image and the model parameters. A lighter background corresponds to higher uncertainty.

be high for points far away from the training data, for a high-capacity model such as a deep network. Under the hypothesis that adversarial examples lie far from the image manifold, i.e. the training data, such uncertainty could be used to identify an input as adversarial.

The uncertainty of such models is not straightforward to obtain. Numerical methods for integrating the posterior, such as Markov Chain Monte Carlo, are difficult to scale to large datasets (Gal, 2016). As a result, approximations have been studied extensively. For example, approximate inference in Bayesian neural networks using dropout is a computationally tractable technique (Gal & Ghahramani, 2016) which has been widely used in the literature (Leibig et al., 2017; Gal, 2016). Dropout based model uncertainty can be used for the detection of adversarial examples, with moderate success (Li & Gal, 2017; Feinman et al., 2017; Rawat et al., 2017).

However, existing research has mostly overlooked the effect of the chosen *measure for uncertainty quantification*. Many such measures exist, including mutual information, predictive entropy and softmax variance. (Li & Gal, 2017) for example use expected entropy, (Rawat et al., 2017) use mutual information, whereas (Feinman et al., 2017) estimate the variance of multiple draws from the predictive distribution (obtained using dropout). Further, to date, research for the identification of adversarial examples using model uncertainty has concentrated on toy problems such as MNIST, and has not been shown to extend to more realistic data distributions and larger models such as ResNet (He et al., 2015).

In this paper we examine the differences between the various measures of uncertainty used for adversarial example detection, and in the process provide further evidence for the hypothesis that model uncertainty could be used to identify an input as adversarial. More specifically, we illustrate the differences between the measures by projecting the uncertainty onto lower dimensional spaces (see for example Fig. 1). We show that the softmax variance can be seen as an approximation to the mutual information (section 3.2), explaining the effectiveness of this rather ad-hoc technique. We show that some measures of uncertainty do not distinguish between non-adversarial off-manifold images (for example image interpolations) and adversarial inputs. We highlight ways in which dropout fails to capture the full Bayesian uncertainty by visualizing gaps in model uncertainty in the latent space (Section 4.2), and use this insight to propose a simple extension to dropout schemes to be studied in future research. We finish by demonstrating the effectiveness of dropout on the real-world ASIRRA (Elson et al., 2007) cats and dogs classification dataset (Section 4.3). Code for the experi-

ments described in this paper is available online<sup>1</sup>.

## 2 BACKGROUND

### 2.1 BAYESIAN DEEP LEARNING

A deep neural network (with a given architecture) defines a function  $f : \mathcal{X} \mapsto \mathcal{Y}$  parametrised by a set of weights and biases  $\omega = \{\mathbf{W}_l, \mathbf{b}_l\}_{l=1}^L$ . These parameters are generally chosen to minimize some loss function  $E : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}$  on the model outputs and the target outputs over some dataset  $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$  with  $\mathbf{x} \in \mathcal{X}$  and  $\mathbf{y} \in \mathcal{Y}$ . Since neural networks are highly flexible models with many degrees of freedom, a regulariser is often added to the loss, giving

$$\hat{\omega} = \arg \min_{\omega} \sum_i E(f(\mathbf{x}_i; \omega), \mathbf{y}_i) + \lambda \sum_l \|\mathbf{W}_l\|^2 \quad (1)$$

for the common choice of an  $L_2$  regulariser with weight decay  $\lambda$ .

In Bayesian approaches, rather than thinking of the weights as fixed parameters that are optimised over, we treat them as random variables, and so we place a prior distribution  $p(\omega)$  over the weights of the network. If we also have a likelihood function  $p(\mathbf{y} | \mathbf{x}, \omega)$  that gives the probability of  $\mathbf{y} \in \mathcal{Y}$  given the model parameters and an input to the network, we can conduct inference given a dataset by marginalizing the parameters. Such models are known as *Bayesian neural networks*.

If the prior on the weights is a zero mean Gaussian with diagonal covariance, and the loss of the network is the negative log likelihood (so  $p(\mathbf{y} | \omega, \mathbf{x}) = e^{-E(f(\mathbf{x}), \mathbf{y})}$ ) then the optimised solution in equation 1 corresponds to a mode of the posterior over the weights.

Ideally we would integrate out our uncertainty by taking the expectation of the predictions over the posterior, rather than using this point estimate. For neural networks this can only be done approximately. Here we discuss one practical approximation, variational inference with dropout approximating distributions.

### 2.2 VARIATIONAL INFERENCE

*Variational inference* is a general technique for approximating complex probability distributions. The idea is to approximate the intractable posterior  $p(\omega | \mathcal{D})$  with a simpler approximating distribution  $q_{\theta}(\omega)$ . By applying Jensen’s inequality to the Kullback-Leibler divergence between the approximating distribution and the true pos-

<sup>1</sup><https://github.com/lsgos/uncertainty-adversarial-paper>

terior, we obtain the log-evidence lower bound  $\mathcal{L}_{VI}$

$$\mathcal{L}_{VI} := \int q_{\theta}(\omega) \log p(\mathcal{D} | \omega) d\omega - D_{KL}(q_{\theta} || p(\omega)).$$

Since the model evidence is a constant independent of the parameters of  $q_{\theta}$ , maximizing  $\mathcal{L}_{VI}$  with respect to  $\theta$  will minimize the KL divergence between  $q$  and the model posterior. The key advantage of this from a computational perspective is that we replace an integration problem with an optimisation problem, maximising a parametrised function, which can be approached by standard gradient based techniques.

For neural networks, a common approximating distribution is *dropout* (Srivastava et al., 2014) and it’s variants. In the variational framework, this means the weights are drawn from

$$\mathbf{W}_l = \mathbf{M}_l \cdot \text{diag}([\mathbf{z}_{l,j}]_{j=1}^{K_l})$$

where  $\mathbf{z}_{l,j} \sim \text{Bernoulli}(p_l)$ ,  $l = 1..L, j = 1..K_{l-1}$

for a network with  $L$  layers, where the dimension of each layer is  $K_i \times K_{i-1}$ , and the parameters of  $q$  are  $\theta = \{\mathbf{M}_l, p_l | l = [1..L]\}$ . Informally, this corresponds to randomly setting the outputs of units in the network to zero (or zeroing the rows of the fixed matrix  $\mathbf{M}_l$ ). Often the layer dropout probabilities  $p_i$  are chosen as constant and not varied as part of the variational framework, but it is possible to learn these parameters as well (Gal et al., 2017). Using variational inference, the expectation over the posterior can be evaluated by replacing the true posterior with the approximating distribution. The dropout distribution is still challenging to marginalise, but it is readily sampled from, so expectations can be approximated using the Monte Carlo estimator

$$\begin{aligned} \mathbb{E}_{p(\omega|\mathcal{D})}[f^{\omega}(x)] &= \int p(\omega|\mathcal{D})f^{\omega}(x)d\omega \\ &\simeq \int q_{\theta}(\omega)f^{\omega}(x)d\omega \\ &\simeq \frac{1}{T} \sum_{i=1}^T f^{\omega_i}(x), \omega_{1..T} \sim q_{\theta}(\omega). \end{aligned} \quad (2)$$

### 2.3 ADVERSARIAL EXAMPLES

Works by (Szegedy et al., 2013) and others, demonstrating that state-of-the-art deep image classifiers can be fooled by small perturbations to input images, have initiated a great deal of interest in both understanding the reasons for why such adversarial examples occur, and devising methods to resist and detect adversarial attacks. So far, attacking has proven more successful than defence; a recent survey of detection methods by (Carlini & Wag-

ner, 2017a) found that, with the partial exception of the method based on dropout uncertainty analysed by (Feinman et al., 2017), all other investigated methods could be defeated straightforwardly.

There is no precise definition of when an example qualifies as ‘adversarial’. The most common definition used is of an input  $\mathbf{x}_{adv}$  which is close to a real data point  $\mathbf{x}$  as measured by some  $L_p$  norm, but is classified wrongly by the network with high score. Speaking more loosely, an adversarially perturbed input is one which a human observer would assign a certain class, but for which the network would predict a different class with a high score.

It is notable that there exists a second, related, type of images which have troubling implications for the robustness of deep models, namely meaningless images which are nevertheless classified confidently as belonging to a particular class (see, for example, Nguyen et al. (2015)). That such images can be found reveals another shortcoming of neural networks from the point of view of uncertainty, since they are far from all training data by any reasonable metric (based on either pixel-wise or perceptual distance). We refer to these as ‘rubbish class examples’ or ‘fooling images’ following (Nguyen et al., 2015) and (Goodfellow et al., 2014).

Several possible explanations for the existence of adversarial examples have been proposed in the literature (Akhtar & Mian, 2018). These include the idea, proposed in the original paper by (Szegedy et al., 2013), that the set of adversarial examples are a dense, low probability set like the rational numbers on  $\mathbb{R}$ , with the discontinuous boundary somehow due to the strong non-linearity of neural networks. Contrary to that, (Goodfellow et al., 2014) proposed that adversarial examples are partially due to the intrinsically linear response of neural network layers to their inputs. (Tanay & Griffin, 2016) have proposed that adversarial examples are possible when the decision boundaries are strongly tilted with respect to the vector separating the means of the class clusters.

Many of these ideas are consistent with the idea that the training data of the model lies on a low dimensional manifold in image space, the hypothesis we build upon in this paper.

### 2.4 MEASURES OF UNCERTAINTY

There are two major sources of uncertainty a model may have:

1. *epistemic* uncertainty is uncertainty due to our lack of knowledge; we are uncertain because we lack understanding. In terms of machine learning, this corresponds to a situation where our model parameters are poorly determined due to a lack of data, so

our posterior over parameters is broad.

2. *aleatoric* uncertainty is due to genuine stochasticity in the data. In this situation, an uncertain prediction is the best possible prediction. This corresponds to *noisy* data; no matter how much data the model has seen, if there is inherent noise then the best prediction possible may be a high entropy one (for example, if we train a model to predict coin flips, the best prediction is the max-entropy distribution  $P(\text{heads}) = P(\text{tails})$ ).

In the classification setting, where the output of a model is a conditional probability distribution  $P(y|x)$  over some discrete set of outcomes  $\mathcal{Y}$ , one straight-forward measure of uncertainty is the entropy of the predictive distribution

$$H[P(y|x)] = - \sum_{y \in \mathcal{Y}} P(y|x) \log P(y|x). \quad (3)$$

However, the predictive entropy is not an entirely satisfactory measure of uncertainty, since it does not distinguish between epistemic and aleatoric uncertainties. However, it may be of interest to do so; in particular, we want to capture when an input lies in a region of data space where the model is poorly constrained, and distinguish this from inputs near the data distribution with noisy labels.

An attractive measure of uncertainty able to distinguish epistemic from aleatoric examples is the information gain between the model parameters and the data. Recall that the mutual information (MI) between two random variables  $X$  and  $Y$  is given by

$$\begin{aligned} I(X, Y) &= H[P(X)] - \mathbb{E}_{P(y)} H[P(X | Y)] \\ &= H[P(Y)] - \mathbb{E}_{P(x)} H[P(Y | X)]. \end{aligned}$$

The amount of information we would gain about the model parameters if we were to receive a label  $y$  for a new point  $x$ , given the dataset  $\mathcal{D}$  is then given by

$$I(\omega, y | \mathcal{D}, x) = H[p(y | x, \mathcal{D})] - \mathbb{E}_{p(\omega | \mathcal{D})} H[p(y | x, \omega)] \quad (4)$$

Being uncertain about an input point  $x$  implies that if we knew the label at that point we would gain information. Conversely, if the parameters at a point are already well determined, then we would gain little information from obtaining the label. Thus, the MI is a measurement of the model’s *epistemic* uncertainty.

In the form presented above, it is also readily approximated using the Bayesian interpretation of dropout. The first term we will refer to as the ‘predictive entropy’; this is just the entropy of the predictive distribution, which we have already discussed. The second term is the mean of the entropy of the predictions given the parameters over the posterior distribution  $p(\omega | \mathcal{D})$ , and we thus refer to it as the expected entropy.

These quantities are not tractable analytically for deep

nets, but using dropout inference and equation (2), the predictive distribution, entropy and the MI are readily approximated; (Gal, 2016):

$$p(y | \mathcal{D}, \mathbf{x}) \simeq \frac{1}{T} \sum_{i=1}^T p(y | \omega_i, \mathbf{x}) \quad (5)$$

$$:= p_{MC}(y | \mathbf{x})$$

$$H[p(y | \mathcal{D}, \mathbf{x})] \simeq H[p_{MC}(y | \mathcal{D}, \mathbf{x})] \quad (6)$$

$$I(\omega, y | \mathcal{D}, x) \simeq H[p_{MC}(y | \mathcal{D}, \mathbf{x})] \quad (7)$$

$$- \frac{1}{T} \sum_{i=1}^T H[p(y | \omega_i, \mathbf{x})] \quad (8)$$

where  $\omega_i \sim q(\omega | \mathcal{D})$  are samples from the dropout distribution.

Other, measures of uncertainty include the empirical variance of the softmax probabilities  $p(y = c | \omega_i, \mathbf{x})$  (with the variance calculated over  $i$ ), and variation ratios (Gal, 2016), with the former commonly used in previous research on adversarial examples.

### 3 UNCERTAINTY FOR ADVERSARIAL EXAMPLE DETECTION

We start by explaining the type of uncertainty relevant for adversarial example detection under the hypothesis that adversarial images lie off the manifold of natural images, occupying regions where the model makes unconstrained extrapolations. We continue by relating the *softmax variance* measure of uncertainty to mutual information.

#### 3.1 WHAT KIND OF UNCERTAINTY?

Both the MI and predictive entropy should increase on inputs which lie far from the image manifold. Under our hypothesis, we expect both to be effective in highlighting such inputs. However, predictive entropy could also be high *near* the image manifold, on inputs which have inherent ambiguity. Such inputs could be ambiguous images, such as an image that contains both a cat and a dog, or more generally interpolations between classes, such as a digit that could be either a 1 or a 7. Such inputs would have high predictive probability for more than one class even in the limit of infinite data, yielding high predictive entropy (but low MI). Such inputs are clearly not adversarial, but would falsely trigger a hypothetical automatic detection system<sup>2</sup>. We demonstrate this experimentally in the next section.

Algorithms to find adversarial examples seek to create an example image with a different class to the original, typically by either minimising the predicted probability of

<sup>2</sup>We speculate that previous research using predictive entropy has not encountered this phenomenon due to insufficient coverage of the test cases.

the current class for an untargeted attack, or maximising the predicted probability of a target class. This has the side-effect of minimising the entropy of the predictions, a simple consequence of the normalisation of the probability. It is interesting to highlight that this also affects the uncertainty as measured by MI; since both the mutual information and entropy are strictly positive, the mutual information is bounded above by the predictive entropy (see equation 4). Therefore, the model giving low entropy predictions at a point is a sufficient condition for the mutual information to be low as well. Equally, the mutual information bounds the entropy from below; it is not possible for a model to give low entropy predictions when the MI is high. It is important to realise that this means that adversarial example algorithms implicitly seek low uncertainty examples: detecting adversarial examples, at least via model uncertainty, is *not* independent of being able to fool the model without explicit detection methods.

### 3.2 MI AND SOFTMAX VARIANCE

Some works in the literature estimate the epistemic uncertainty of a dropout model using the estimated variance of the MC samples, rather than the mutual information (Leibig et al., 2017; Feinman et al., 2017; Carlini & Wagner, 2017a). This is somewhat arbitrary for classification, but seems to work fairly well in practice. We suggest a possible explanation of the effectiveness of this measure, arguing that the softmax variance can be seen as a proxy to the mutual information.

One way to see the relation between the two measures of uncertainty is to observe that the variance is the leading term in the series expansion of the mutual information. For brevity, we denote the sampled distributions  $p(y | \omega_i, \mathbf{x})$  as  $p_i$  and the mean predictive distribution  $p_{MC}(y | \mathbf{x})$  as  $\hat{p}$ . These are in general distribution over  $C$  classes, and we denote the probability of the  $j^{th}$  class as  $\hat{p}_j$  and  $p_{ij}$  for the mean and  $i^{th}$  sampled distribution respectively. The variance score is the mean variance across the classes

$$\begin{aligned} \hat{\sigma}^2 &= \frac{1}{C} \sum_{j=1}^C \frac{1}{T} \sum_{i=1}^T (p_{ij} - \hat{p}_j)^2 \\ &= \frac{1}{C} \left( \sum_{j=1}^C \left( \frac{1}{T} \sum_{i=1}^T p_{ij}^2 \right) - \hat{p}_j^2 \right) \end{aligned} \quad (9)$$

And the mutual information score is

$$\begin{aligned} \hat{I} &= H(\hat{p}) - \frac{1}{T} \sum_i H(p_i) \\ &= \sum_j \left( \frac{1}{T} \sum_i p_{ij} \log p_{ij} \right) - \hat{p}_j \log \hat{p}_j \end{aligned}$$

Using a Taylor expansion of the logarithm,

$$\begin{aligned} \hat{I} &= \sum_j \left( \frac{1}{T} \sum_i p_{ij} (p_{ij} - 1) \right) - \hat{p}_j (\hat{p}_j - 1) + \dots \\ &= \sum_j \left( \frac{1}{T} \sum_i p_{ij}^2 \right) - \hat{p}_j^2 - \left( \frac{1}{T} \sum_i p_{ij} \right) + \hat{p}_j + \dots \\ &= \sum_j \left( \frac{1}{T} \sum_i p_{ij}^2 \right) - \hat{p}_j^2 + \dots \end{aligned} \quad (10)$$

we see that the first term in the series is identical up to a multiplicative constant to the mean variance of the samples.

This relation between the softmax variance and the mutual information measure could explain the effectiveness of the variance in detecting adversarial examples encountered by (Feinman et al., 2017). MI increases on images far from the image manifold and not on image interpolations (on which the predictive variance increases as well), with the variance following similar trends.

## 4 EMPIRICAL EVALUATION

In the next section we demonstrate the effectiveness of various measures of uncertainty as proxies to distance from the image manifold. We demonstrate the difference in behaviour between the predictive entropy and mutual information on image interpolations, for interpolations in the latent space as well as interpolations in image space. We continue by visualising the various measures of uncertainty, highlighting the differences discussed above. This is further developed by highlighting shortcomings with current approaches for uncertainty estimation, to which we suggest initial ideas on how to overcome and suggest new ideas for attacks (to be explored further in future research). We finish by assessing the ideas discussed in this paper on a real world dataset of cats vs dogs image classification.

### 4.1 UNCERTAINTY ON INTERPOLATIONS

We start by assessing the behaviour of the measures of uncertainty on image interpolations, comparing interpolations via convex combination  $(\lambda x_1 + (1 - \lambda)x_2, \lambda \in [0, 1], x_i \in \mathcal{D})$  in latent space to those in image space. A convex combination in image space will clearly produce off manifold images, while we assume that moving in latent space approximates the manifold of the data fairly closely. That model uncertainty can capture what we want in practice is demonstrated in Figures 2 and 3. We see that the MI distinguishes between these on-manifold and off-manifold images, whereas the entropy fails to do so. This is necessary for the hypothesis proposed in the introduction; if we are able to accurately capture the MI,



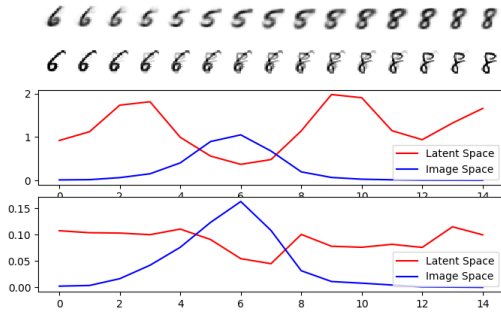


Figure 2: Entropy (middle) and the MI (bottom) vary along a convex interpolation between two images in latent space and image space (top). The entropy is high for regions along both interpolations, wherever the class of the image is ambiguous. In contrast, the MI is roughly constant along the interpolation in latent space, since these images have aleatoric uncertainty (they are ambiguous), but the model has seen data that resembles them. On the other hand, the MI has a clear peak as the pixel space interpolation produces out-of-sample images between the classes

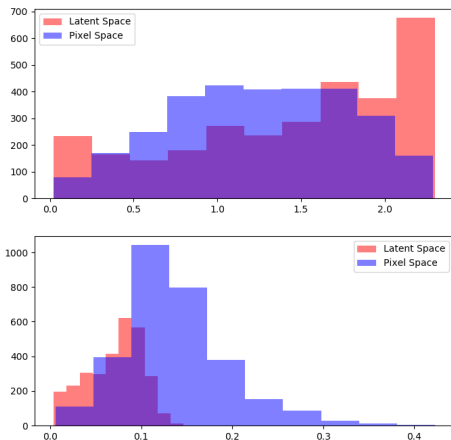


Figure 3: The entropy (top) and mutual information (bottom) of the interpolation halfway between 3000 random points of different classes in the MNIST test set, showing that the two modes of interpolation have very different statistical properties with respect to the model uncertainty, as shown for a single example in figure 2.

it would serve well as a proxy for whether an images belongs to the learned manifold or not.

## 4.2 VISUALIZATION IN LATENT SPACE

We wish to gain intuition into how the different measures of uncertainty behave. In order to do so, we use a variational autoencoder (Kingma & Welling, 2013) to compress the MNIST latent space. We choose a latent space of two dimensions so we can use this to visualise the dataset. By decoding the image that corresponds to a point in latent space, we can classify the decoded image

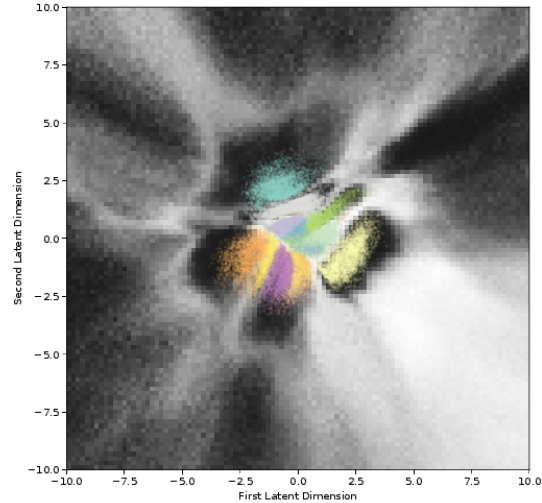


Figure 4: The predictive entropy of the same network as in figure 1. Note the differences with the MI, which is low everywhere close to the data in the centre of the plot, but the entropy is high between the classes here. These points correspond to images which resemble digits, but which are inherently ambiguous. Note however that there are large regions of latent space where the predictive entropy is high and the MI low, despite being far from any training data.

and evaluate the network uncertainty, thus providing a two dimensional map of the input space. Figure 1 shows the latent space with the uncertainty measured using the MI, calculated using dropout. Similarly, Figure 4 shows the predictive entropy. Note the differences in uncertainty near the class cluster boundaries (corresponding to image interpolations) – the MI has low uncertainty in these regions, whereas the predictive entropy is high.

Another question of interest in this context is how well the dropout approximation captures uncertainty. The approximating distribution is fairly crude, and variational inference schemes are known to underestimate the uncertainty of the posterior, tending to fit an approximation to a local mode rather than capturing the full posterior<sup>3</sup>.

As seen from the figures, the network does a reasonable job of capturing uncertainty close to the data. However, the network’s uncertainty has ‘holes’ – regions where the predictions of the model are very confident, despite the images generated by the decoder here being essentially nonsense (see Figure 5). This suggests that, while the uncertainty estimates generated by MC dropout are useful,

<sup>3</sup>There are two reasons for this behaviour: firstly, that the approximating distribution  $q$  may not have sufficient capacity to represent the full posterior, and secondly, the asymmetry of the KL divergence, which penalizes  $q$  placing probability mass where the support of  $p$  is small far more heavily than the reverse.

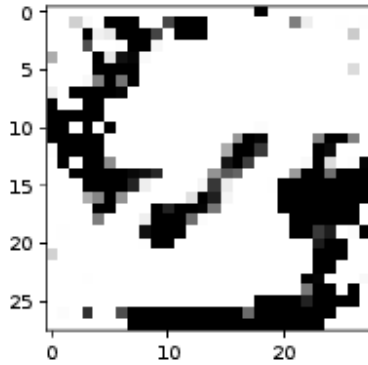


Figure 5: A typical garbage class example from the ‘holes’ in latent space. This is classified as a 2 with high confidence.

they do not capture the full posterior, instead capturing local behaviour near one of its modes, since we would expect the uncertainty to be high for a neural network everywhere where it is not constrained by the training data due to the high capacity of the model.

This may offer an explanation as to why MC dropout nets are still vulnerable to adversarial attack; despite their treatment of uncertainty, there are still large regions where they are mistakenly overconfident due to the approximations used, which adversarial attack algorithms can exploit. It may be possible to deliberately find and exploit these ‘holes’ to create adversarial examples. This intuition suggests a simple fix; since a single dropout model averages over a single mode of the posterior, we can capture the posterior using an ensemble of dropout models using different initializations, assuming that these will converge to different local modes. We find that even a small ensemble can qualitatively improve this behaviour (Figure 6).

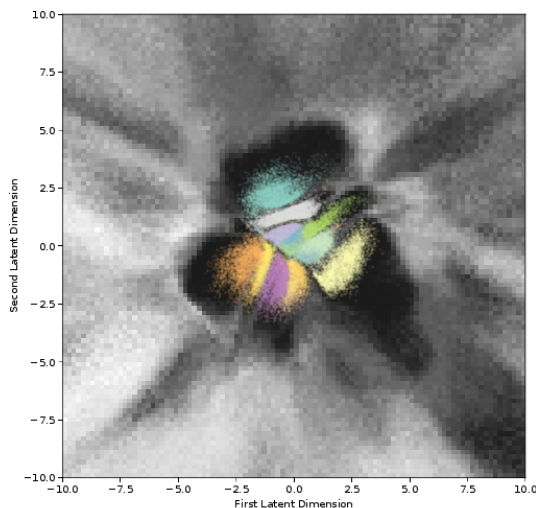


Figure 6: The MI calculated using an ensemble of dropout models, treating all of their predictions as Monte Carlo samples from the posterior. This mitigates some of the spuriously confident regions in latent space

It should be noted, though, that there is no guarantee that an ensemble of dropout models is a better approximation to the true posterior. It will approximate it well only if the posterior is concentrated in many local modes, all of roughly equal likelihood (since all the models in the ensemble are weighted equally), and a randomly initialized variational dropout net trained with some variant of gradient descent will converge to all of these modes with roughly equal probability<sup>4</sup>. Investigating possible theoretical justification for this ensembling procedure for variational models is a possible direction for future research.

### 4.3 EVALUATION ON CATS AND DOGS DATASET

It has been observed by (Carlini & Wagner, 2017a) that many proposed defences against adversarial examples fail to generalize from MNIST. Therefore, we also evaluate the various uncertainty measures on a more realistic dataset; the ASSIRA cats and dogs dataset (see Figure 7 for example images). The task is to distinguish pic-

<sup>4</sup>This description does coincide with common beliefs about neural network loss surfaces, for which there is some justification in the literature; see, for example, Choromanska et al. (2015)

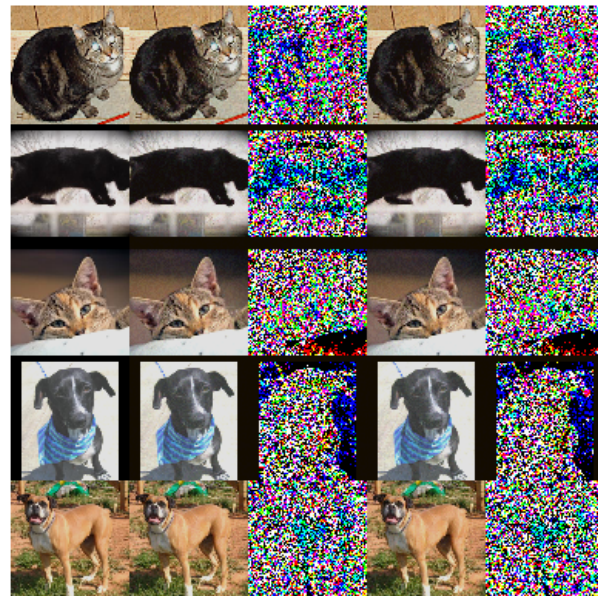


Figure 7: Example adversarial images generated by the Momentum iterative method at  $\epsilon = 10$ , with original images on the left, adversarial images on the deterministic model in the second column, and those for the MC dropout model in the fourth column. The difference between the adversarial image and the original is shown on the right of each image.

Table 1: The AUC for the adversarial discrimination task described in the experiments section. Fields marked with (S) denote this quantity evaluated on a version of the dataset with unsuccessful adversarial examples (that do not change the label) removed. The success rate of each attack in changing the label is given as a measure of each attacks effectiveness on this dataset.

	ENTROPY	MI	ENTROPY (S)	MI (S)	SUCCESS RATE
BIM $\epsilon = 5$					
DETERMINISTIC	0.322	N.A	0.293	N.A	0.757
MC	0.0712	<b>0.728</b>	0.0617	<b>0.733</b>	0.900
FGM $\epsilon = 5$					
DETERMINISTIC MODEL	0.439	N.A	<b>0.490</b>	N.A	0.517
MC MODEL	0.426	<b>0.557</b>	0.465	<b>0.497</b>	0.563
MIM $\epsilon = 5$					
DETERMINISTIC MODEL	0.347	N.A	0.319	N.A	0.743
MC MODEL	0.0476	<b>0.657</b>	0.0410	<b>0.669</b>	0.917
BIM $\epsilon = 10$					
DETERMINISTIC MODEL	0.302	N.A	0.285	N.A	0.753
MC MODEL	0.0686	<b>0.708</b>	0.0719	<b>0.723</b>	0.917
FGM $\epsilon = 10$					
DETERMINISTIC MODEL	0.502	N.A	<b>0.550</b>	N.A	0.487
MC MODEL	0.480	<b>0.529</b>	0.514	0.491	0.547
MIM $\epsilon = 10$					
DETERMINISTIC MODEL	0.350	N.A	0.319	N.A	0.763
MC MODEL	0.0527	<b>0.661</b>	0.0442	<b>0.665</b>	0.907

tures of cats and dogs. While this is not a state of the art problem, these are realistic, high resolution images. We finetune a ResNet model (He et al., 2015), pre-trained on Imagenet, replacing the final layer with a dropout layer followed by a new fully connected layer. We use 20 forward passes for the Monte Carlo dropout estimates. We use dropout only on the layers we retrain, treating the pre-trained convolutions as deterministic.

We compare the receiver operating characteristic (ROC) of the predictive entropy of the deterministic network, the predictive entropy of the dropout network (equation 7), and the MI of the dropout network (the MI is always zero if the model is deterministic; this corresponds to the approximating distribution  $q$  being a delta function). Note that we compare with the *same set of weights* (trained with dropout) – the only difference is whether we use dropout at test time. For each measure of uncertainty we generate the ROC plot by thresholding the uncertainty at different values, using the threshold to decide whether an input is adversarial or not.

The receiver operating characteristic is evaluated on a synthetic dataset consisting of images drawn at random from the test set and images from the test set corrupted by Gaussian noise, which comprise the negative examples, as well as adversarial examples generated with the Basic Iterative Method (Kurakin et al., 2016), Fast Gradient

method (Goodfellow et al., 2014), and Momentum Iterative Method (Dong et al., 2017). We test with the final attack because it is notably strong, winning the recent NIPS adversarial attack competition, and is simpler to adapt to stochastic models than the other strong attacks in the literature, such as that of Carlini and Wagner (Carlini & Wagner, 2017b).

We find that only the mutual information gets a useful AUC on adversarial examples. In fact, most other measures of uncertainty seem to be worse than random guessing; this suggests that this dataset has a lot of examples the model considers to be ambiguous (high aleatoric uncertainty), which mean that the entropy has a high false positive rate. The fact the AUC of the entropy is low suggests that the model is actually *more* confident about adversarial examples than natural ones under this measure.

An interesting quirk of this particular model is that the accuracy of using Monte Carlo estimation is lower than the point estimates, even though the uncertainty estimates are sensible. Possibly this is because the dropout probability is quite high; only a subset of the features in the later layers of a convnet are relevant to cat and dog discrimination, so this may be a relic of our transfer learning procedure; dropout does not normally have an adverse effect on the accuracy of fully trained models (Gal, 2016).

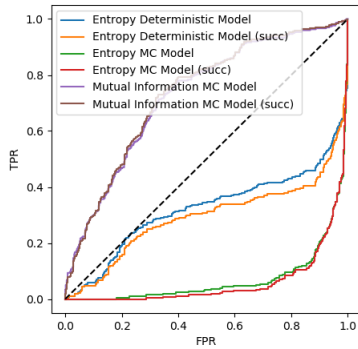


Figure 8: BIM with  $\epsilon = 5$

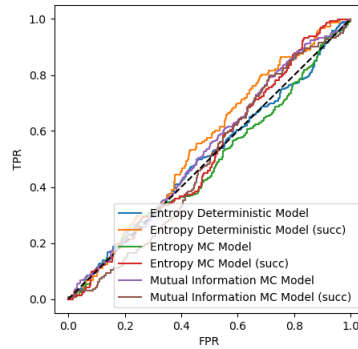


Figure 9: FGM with  $\epsilon = 5$

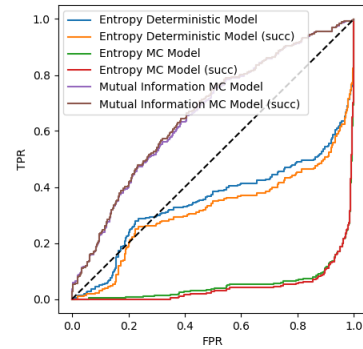


Figure 10: MIM with  $\epsilon = 5$

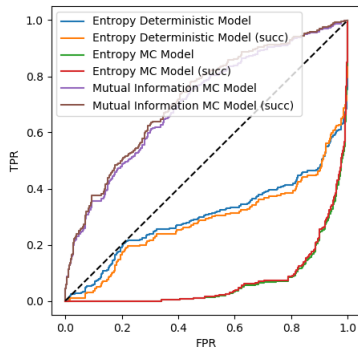


Figure 11: BIM with  $\epsilon = 10$

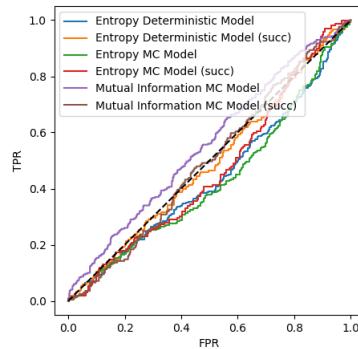


Figure 12: FGM with  $\epsilon = 10$

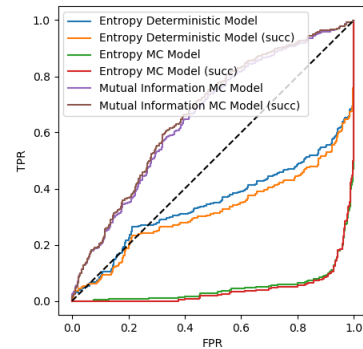


Figure 13: MIM with  $\epsilon = 10$

Figure 14: ROC plots for adversarial example detection with different measures of uncertainty and different attacks. From left to right: basic iterative method (BIM), fast gradient method (FGM), and momentum iterative method (MIM). Top row uses  $\epsilon$  of 5, bottom row uses  $\epsilon$  of 10. All use infinity norm. (succ) denotes the quantity evaluated only for successful adversarial examples. We suspect that the low FGM attack success rate is related to the difficulty we observe in identifying these using model uncertainty, however further investigation is required.

## 5 DISCUSSION & CONCLUSION

We have examined various measures of uncertainty for detecting adversarial examples, and provided both theoretical and experimental evidence that measuring the epistemic uncertainty with the mutual information is the most appropriate and effective for this task.

We do not claim, however, that using dropout provides a very convincing defence against adversarial attack. Our results (in agreement with previous literature on the subject) show that dropout networks are *more difficult* to attack than their deterministic counterparts, but attacks against them can still succeed while remaining imperceptible to the human eye, at least in the white-box setting we investigated.

It is worth noting, however, that these techniques for quantifying uncertainty can be derived without any explicit reference to the adversarial setting, and no assumptions are made about the distribution of adversarial examples.

By improving model robustness and dealing with uncertainty more rigorously, models become harder to fool as a side effect; model robustness and good uncertainty estimates are not independent, as discussed in section 3. We think the fact that dropout models can still be defeated by adversarial attack is at least partly because dropout is a fairly crude approximation that underestimates the uncertainty significantly, as we have demonstrated here. Looking for scalable ways to improve on the uncertainty quality captured by dropout is an important avenue for future research.

## ACKNOWLEDGEMENTS

LS is supported by EPSRC. This work was supported by the Alan Turing Institute’s Defence and Security programme.

## References

Akhtar, N., & Mian, A. (2018). Threat of adversarial

- attacks on deep learning in computer vision: A survey. *arXiv preprint arXiv:1801.00553*.
- Carlini, N., & Wagner, D. (2017a). Adversarial examples are not easily detected: Bypassing ten detection methods. *arXiv preprint arXiv:1705.07263*.
- Carlini, N., & Wagner, D. (2017b). Towards evaluating the robustness of neural networks. In *Security and privacy (sp), 2017 IEEE Symposium on* (pp. 39–57).
- Choromanska, A., Henaff, M., Mathieu, M., Arous, G. B., & LeCun, Y. (2015). The loss surfaces of multilayer networks. In *Artificial intelligence and statistics* (pp. 192–204).
- Dong, Y., Liao, F., Pang, T., Su, H., Hu, X., Li, J., & Zhu, J. (2017). Boosting adversarial attacks with momentum. *arXiv preprint arXiv:1710.06081*.
- Elson, J., Douceur, J. J., Howell, J., & Saul, J. (2007, October). Asirra: A captcha that exploits interest-aligned manual image categorization. In *Proceedings of 14th ACM conference on computer and communications security (ccs)*. Association for Computing Machinery, Inc.
- Feinman, R., Curtin, R. R., Shintre, S., & Gardner, A. B. (2017). Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*.
- Gal, Y. (2016). *Uncertainty in deep learning* (Unpublished doctoral dissertation). PhD thesis, University of Cambridge.
- Gal, Y., & Ghahramani, Z. (2016). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning* (pp. 1050–1059).
- Gal, Y., Hron, J., & Kendall, A. (2017). Concrete dropout. *arXiv preprint arXiv:1705.07832*.
- Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.
- He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep residual learning for image recognition. corr abs/1512.03385 (2015)*.
- Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.
- Kurakin, A., Goodfellow, I., & Bengio, S. (2016). Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*.
- Leibig, C., Allken, V., Ayhan, M. S., Berens, P., & Wahl, S. (2017). Leveraging uncertainty information from deep neural networks for disease detection. *Scientific Reports*, 7(1), 17816. Retrieved from <https://doi.org/10.1038/s41598-017-17876-z>  
doi: 10.1038/s41598-017-17876-z
- Li, Y., & Gal, Y. (2017). Dropout inference in bayesian neural networks with alpha-divergences. *arXiv preprint arXiv:1703.02914*.
- Nguyen, A., Yosinski, J., & Clune, J. (2015). Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 427–436).
- Rawat, A., Wistuba, M., & Nicolae, M.-I. (2017). Adversarial phenomenon in the eyes of bayesian deep learning. *arXiv preprint arXiv:1711.08244*.
- Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1), 1929–1958.
- Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.
- Tanay, T., & Griffin, L. (2016). A boundary tilting perspective on the phenomenon of adversarial examples. *arXiv preprint arXiv:1608.07690*.

---

# An Upper Bound for Random Measurement Error in Causal Discovery

---

Tineke Blom<sup>a,1</sup>

Anna Klimovskaia<sup>b,2</sup>

Sara Magliacane<sup>c,3</sup>

Joris M. Mooij<sup>a,4</sup>

<sup>a</sup>Informatics Institute, University of Amsterdam, The Netherlands

<sup>b</sup>Institute of Molecular Systems Biology, ETH Zürich, Switzerland

<sup>c</sup>IBM Research, Yorktown Heights, USA

E-mails: <sup>1</sup>t.blom2@uva.nl, <sup>2</sup>klimovskaia@imsb.biol.ethz.ch,

<sup>3</sup>sara.magliacane@gmail.com, <sup>4</sup>j.m.mooij@uva.nl

## Abstract

Causal discovery algorithms infer causal relations from data based on several assumptions, including notably the absence of measurement error. However, this assumption is most likely violated in practical applications, which may result in erroneous, irreproducible results. In this work we show how to obtain an upper bound for the variance of random measurement error from the covariance matrix of measured variables and how to use this upper bound as a correction for constraint-based causal discovery. We demonstrate a practical application of our approach on both simulated data and real-world protein signaling data.

## 1 INTRODUCTION

The discovery of causal relations is a fundamental objective in science, and the interest in causal discovery algorithms has increased rapidly since they were first established in the 1990s [Pearl, 2000, Spirtes et al., 2000]. In practice, it may happen that their predictions are not reproducible in independent experiments. In this article we show that the presence of measurement error may be a possible explanation for incorrect and inconsistent output and we propose a solution aimed to mitigate its ramifications.

The presence of measurement error complicates causal discovery, because measured quantities are typically not causes of one another, even when the variables that they represent are. Consider the example in Figure 1, and suppose that exercise  $E$  is a variable that can be controlled in an experiment, weight loss  $W$  can be measured very precisely, but the amount of burned calories  $C$  cannot be observed directly. Suppose we do have a measured quantity  $\tilde{C} = C + M_C$  with  $M_C$  a measurement error. Even

though exercise and weight loss are independent conditional on burned calories, they are not when we condition on the measurement  $\tilde{C}$ . If  $M_C$  is large, one might even find that the measurements of the calories are independent of exercise conditional on the weight loss. A researcher who is unaware of the measurement error could then draw incorrect conclusions (e.g. weight loss causes the burning of calories).

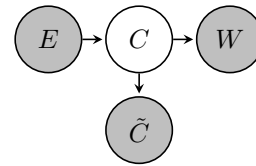


Figure 1: Example of causal discovery in the presence of measurement error. Gray shaded nodes are observed variables, white nodes are latent variables.

The example in Figure 1 illustrates the crucial difference between measurement error and disturbance terms that are usually considered in causal models. In particular, the fluctuations that are due to measurement error do not propagate to effect variables (e.g. measurement noise  $M_C$  in  $\tilde{C}$  cannot be seen in  $E$ ), whereas the effects of unmodeled causes do.

Following [Scheines and Ramsey, 2016, Zhang et al., 2017, Pearl, 2010] and [Kuroki and Pearl, 2014], we focus on *random* measurement error, an independent random variable that adds noise to the measurement of one variable in a model. We present a method that identifies an upper bound for the variance of random measurement error. This result builds on previous work where the identification of sets of variables that are d-separated by a common latent variable using vanishing tetrad constraints is considered, see [Silva et al., 2006, Pearl, 2010, Bollen, 1989, Sullivant et al., 2010]. Uncertainty regarding the size of the measurement error can be propagated to an uncertainty in the partial correlations of the latent variables that are yet unperturbed by measurement error,

see also [Harris and Drton, 2013]. This uncertainty can then be taken into account when performing statistical tests so that we have outputs: dependent, independent, or unknown. Although these types of outputs for independence tests have been already used in previous work, e.g. [Triantafillou et al., 2017], in that case the thresholds for the different decisions were hyperparameters of the algorithm, while we provide an adaptive and more principled way to set them. Similarly to previous work, our approach relies on strong faithfulness [Spirtes et al., 2000, Kalisch and Bühlmann, 2007, Maathuis et al., 2010] but crucially it does not require causal sufficiency, i.e. the absence of unmeasured confounders, as Zhang et al. [2017] do.

In this work, we propose a practical correction method for measurement error in the context of constraint-based causal discovery. We demonstrate the effectiveness of our approach in identifying causal structures using Local Causal Discovery (LCD) [Cooper, 1997] both on simulated data and real-world protein signaling data. Although we focus on one particular causal discovery algorithm, our ideas can be applied to other constraint-based causal discovery algorithms as well, but we consider this to be outside of the scope of this paper.

## 2 PRELIMINARIES

For the remainder of this paper, variables will be denoted by capital letters and sets of variables by bold capital letters. We will assume that the data-generating processes described here can be modeled by a causal graph  $\mathcal{G}$  with nodes  $V$  and directed and bidirected edges  $E$ , where some of the variables in  $V$  may be latent. When there is a directed edge from a variable  $X$  to a variable  $Y$ , we say that  $X$  is a direct cause of  $Y$ . When there is a sequence of directed edges from  $X$  to  $Y$  with all arrowheads pointing towards  $Y$  we call it a directed path, and we say that  $X$  is an ancestor of  $Y$ . Bidirected edges between two variables  $X$  and  $Y$  are used to represent hidden confounders. Conditional independence between  $X$  and  $Y$  while controlling for variables in  $Z$  is denoted by  $X \perp\!\!\!\perp Y \mid Z$ . If  $Z$  d-separates  $X$  from  $Y$ , we denote this as  $X \perp Y \mid Z$ .

In the absence of measurement error, the following commonly made assumptions allow us to relate conditional (in)dependences between disjoint sets of variables  $X, Y$ , and  $Z$  to d-separation in an underlying causal graph  $\mathcal{G}$  [Pearl, 2000, Spirtes et al., 2000]. Throughout the remainder of this paper we will assume that the common assumptions hold.

**Assumption 1** (Common Assumptions).

1. *There are no directed cycles in the causal graph.*

2. *Causal Markov Property: For all disjoint sets of variables  $X, Y, Z$ :  $X \perp Y \mid Z \implies X \perp\!\!\!\perp Y \mid Z$ .*
3. *Causal Faithfulness: For all disjoint sets of variables  $X, Y, Z$ :  $X \perp\!\!\!\perp Y \mid Z \implies X \perp Y \mid Z$ .*
4. *No selection bias is present.*

**Local causal discovery** The LCD (Local Causal Discovery) algorithm is a straight-forward and efficient search method to detect one specific causal structure from experimental data using dependence relations between variables in  $V$  [Cooper, 1997].<sup>1</sup> LCD uses both (conditional) independences and background knowledge to recover causal relations from data.

The LCD algorithm looks for triples of variables  $(X, Y, Z)$  for which (a)  $X$  is not caused by any observed variable and (b) the following (in)dependences hold:  $X \not\perp\!\!\!\perp Y$ ,  $Y \not\perp\!\!\!\perp Z$ , and  $X \perp\!\!\!\perp Z \mid Y$ . We henceforth call such triples *LCD triples*. Under the common assumptions, the causal model that corresponds to this independence pattern is shown in Figure 2.

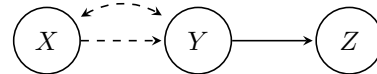


Figure 2: An LCD triple has the above causal structure, with at least one of the dashed arrows present.

**Conditional (in)dependence testing** In practice, constraint-based causal discovery algorithms rely on a statistical test to assess the (in)dependence relationships between variables. For data that has a multivariate Gaussian distribution, a (conditional) independence corresponds to a vanishing (partial) correlation coefficient. For random variables  $(X_1, \dots, X_D) \sim \mathcal{N}(\mu, \Sigma)$ , the Pearson partial correlation can be calculated from the inverse covariance matrix, which we will denote by  $\Lambda = \Sigma^{-1}$ .

Conventionally, one calculates a p-value  $p_T$  for the (conditional) dependence between variables, so that dependence relations can be determined by

$$\begin{cases} X \not\perp\!\!\!\perp Y \mid Z & \text{if } p_T < \alpha \\ X \perp\!\!\!\perp Y \mid Z & \text{if } p_T > \beta, \end{cases} \quad (1)$$

where  $\alpha$  and  $\beta$  are thresholds for dependence and independence respectively. The nature of the relation is undecided when  $\alpha \leq p_T \leq \beta$ . Usually only a single threshold  $\alpha = \beta = 0.01$  or  $\alpha = \beta = 0.05$  is used.

<sup>1</sup>Triantafillou et al. [2017] give a conservative variant of LCD with an application to protein signaling data.

### 3 CAUSAL DISCOVERY UNDER MEASUREMENT ERROR

In this section we illustrate some possible negative effects of random measurement error on constraint-based causal discovery. To that end, we analyze the behavior of partial correlations for increasing measurement error in a simple model. We consider *random* measurement error, which is a vector of independent noise variables  $\mathbf{M} = (M_1, \dots, M_n)$ . The measurements of the random vector  $\mathbf{X} = (X_1, \dots, X_n)$  are then given by  $\tilde{\mathbf{X}} = (\tilde{X}_1, \dots, \tilde{X}_2) = \mathbf{X} + \mathbf{M}$ . This means that a measurement node  $\tilde{X}_i$  is always child-less and has precisely two parents:  $X_i$  and the measurement error source  $M_i$ .

In many practical applications, it is reasonable to assume that the measurement noise has a Gaussian distribution. For instance, when the measurement noise is the sum of many small independent sources of error, the measurement error approximates a normal distribution because of the central limit theorem. In this article we consider the case where the measurement error is Gaussian so that the measurement noise variables are given by  $\mathbf{M} = (M_1, \dots, M_n) \sim \mathcal{N}(0, \Sigma_M)$ , where  $\Sigma_M$  is a diagonal matrix.

#### 3.1 MOTIVATIONAL EXAMPLE

We illustrate the effects of measurement error on the following structural causal model:

$$\begin{aligned} X_1 &= E_1 \\ X_2 &= \beta_{12}X_1 + E_2 \\ X_3 &= \beta_{23}X_2 + E_3 \\ \tilde{X}_2 &= X_2 + M_2 \end{aligned}$$

where  $X_1$  and  $X_3$  are not affected by measurement error. In this model  $E_1, E_2$ , and  $E_3$  are normally distributed noise variables and  $M_2$  is a normally distributed random measurement error. The observed variables are  $X_1, \tilde{X}_2$ , and  $X_3$ , where the second represents the corrupted measurement of  $X_2$ . The corresponding causal graph is displayed in Figure 3.

Note that in the random measurement error model,  $(X_1, X_2, X_3)$  has the causal structure of an LCD triple, but  $(X_1, \tilde{X}_2, X_3)$  does not. Therefore  $X_1 \perp\!\!\!\perp X_3 \mid X_2$  and the partial correlation for the latent unmeasured variables satisfies  $\rho_{13|2} = 0$ . Let  $\tilde{\Sigma}$  be the covariance matrix of  $(X_1, \tilde{X}_2, X_3)$  and  $\tilde{\Lambda}$  its inverse. Then we have that

$$\tilde{\rho}_{13|2} = -\frac{\tilde{\Lambda}_{13}}{\sqrt{\tilde{\Lambda}_{11}\tilde{\Lambda}_{33}}} = \frac{-\beta_{12}\beta_{23}\tilde{\Sigma}_{11}\text{var}(M_2)}{|\tilde{\Sigma}|\sqrt{\tilde{\Lambda}_{11}\tilde{\Lambda}_{33}}} \neq 0,$$

for non-zero parameters, so that  $X_1 \not\perp\!\!\!\perp X_3 \mid \tilde{X}_2$ .

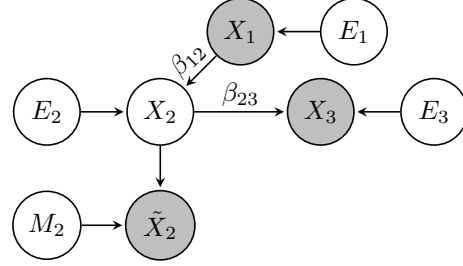


Figure 3: Causal graph of a model with random measurement error on  $X_2$ . Gray shaded nodes are observed variables (the others are latent), and coefficients alongside the arrows represent the coefficients in the model.

**Remark 1.** A statistical test with conventional thresholds would conclude that  $X_1$  and  $X_3$  are conditionally dependent conditional on the measurement  $\tilde{X}_2$ , if the measurement error is large enough. If we would incorrectly assume that there is no measurement error, so that  $X_2 = \tilde{X}_2$ , then the Markov assumption would appear to be violated.

#### 3.2 EMPIRICAL STUDY

For a better understanding of the impact of measurement error on causal discovery, we consider the effect of varying the measurement error variance  $\text{var}(M_2)$  relative to the total variance of the measurement  $\tilde{X}_2$  on the partial correlations in the motivational example.

Figure 4 shows the effect of increasing relative random measurement error on different partial correlations, where the dotted lines represent the  $\alpha = 0.05$  threshold at different sample sizes. It can be seen that for zero measurement error (so that  $\tilde{X}_2 = X_2$ ), only the yellow line is below the red and black dotted lines. In that case a conventional statistical test would indicate that all variables are marginally dependent and  $X_1 \perp\!\!\!\perp X_3 \mid \tilde{X}_2$ , so that  $(X_1, \tilde{X}_2, X_3)$  is an LCD triple, and the directed edge from  $\tilde{X}_2$  to  $X_3$  can be detected. For relative measurement errors larger than  $\sim 0.25$  this conditional independence is no longer detected (because the yellow line is above the black-dotted line).

In Figure 4 we can also observe that for sample size 100 and a relative measurement error larger than  $\sim 0.3$ , a conventional statistical test would indicate that  $X_1 \perp\!\!\!\perp \tilde{X}_2 \mid X_3$  since the partial correlation  $\tilde{\rho}_{12|3} \approx 0$  (i.e. below the red dotted line) and all other (partial) correlations indicate a dependence (i.e. above the red dotted line). Causal discovery algorithms cannot recover the correct causal structure from these constraints. In fact, the LCD algorithm would conclude that  $(X_1, X_3, \tilde{X}_2)$  is an LCD triple so that there must be a directed edge in the *reversed* direction.



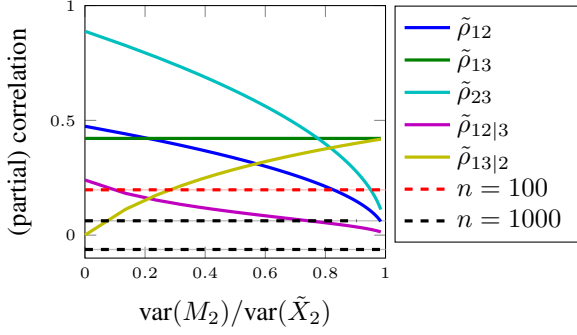


Figure 4: Partial correlations in the random measurement error model in Figure 3. The dotted lines represent the critical values for the correlation at a significance level of  $\alpha = 5\%$  for different sample sizes. The parameter settings were  $\beta_{12} = 0.6$ ,  $\beta_{23} = 1.2$  and all noise variables had variance 1.0.

**Remark 2.** *The results of constraint-based causal discovery may depend on the sample size. This can be better understood by observing that the dependences that are identified by a statistical test, depend both on the size of the measurement error and the sample size. This may lead to inconsistent causal discoveries, which cannot be reproduced on new datasets.*

This example shows how measurement error interferes with detecting the correct causal structures, which may lead to edge deletions, insertions or reversals. Note that although we focused on the LCD algorithm here, the conclusions that we draw are more generally applicable to constraint-based causal discovery algorithms.

**Remark 3.** *For relative measurement error of  $\sim 0.25$  a conflicting set of (in)dependences arises for  $n = 100$ . Since both the yellow and purple line are below the red dotted line, a statistical test would indicate that  $X_1 \perp\!\!\!\perp X_3 \mid \tilde{X}_2$  and  $X_1 \perp\!\!\!\perp \tilde{X}_2 \mid X_3$ , while all variables are marginally dependent. But there is no model that satisfies the common assumptions and these (in)dependences.*

## 4 ERROR BOUND DETECTION

Recall that the true covariances of  $D$  random variables  $\Sigma$ , measurements  $\tilde{\Sigma}$  and random measurement errors  $\Sigma_M$  are related as follows:

$$\Sigma_M = \tilde{\Sigma} - \Sigma = \text{diag}(m_1, \dots, m_D)$$

where  $m_1, \dots, m_D > 0$  are the variances of the random measurement error associated with each variable. In this section we show how, under certain conditions, an upper bound for the variance of random measurement error can be obtained from observational data with random measurement error.

**Remark 4.** *Given an (unbiased) estimate of  $\Sigma_M$ , we can simply adjust the covariance matrix  $\tilde{\Sigma}$  as suggested by Pearl [2010]. In practice such an estimate of the covariance matrix of measurement error may not be available.*

We consider latent random variables  $X_1, \dots, X_4$  and their corresponding measurements  $\tilde{X}_1, \dots, \tilde{X}_4 \in \mathbf{V}$  with true covariance matrices  $\Sigma$  and  $\tilde{\Sigma}$  respectively. Our upper bound result relies on Lemma 1 which is due to Silva et al. [2006] and gives conditions<sup>2</sup> under which there exists a latent variable that d-separates the measured variables  $\tilde{X}_1, \dots, \tilde{X}_4$ .

**Lemma 1.** *Let  $X_1, \dots, X_4$  be variables in a linear-Gaussian model and let  $\tilde{X}_1, \dots, \tilde{X}_4$  be their measurements with random measurement error. If the correlations satisfy  $\tilde{\rho}_{ij} \neq 0$  for all  $i, j \in \{1, \dots, 4\}$  and  $\tilde{\Sigma}_{12}\tilde{\Sigma}_{34} = \tilde{\Sigma}_{13}\tilde{\Sigma}_{24} = \tilde{\Sigma}_{14}\tilde{\Sigma}_{23}$ , then there exists a node  $L$  in the true underlying DAG such that  $\tilde{X}_i \perp\!\!\!\perp \tilde{X}_j \mid L$  for all  $i \neq j \in \{1, \dots, 4\}$ .*

*Proof.* The proof can be found in Silva et al. [2006].  $\square$

When there exists a node  $L$  that d-separates  $\tilde{X}_1, \dots, \tilde{X}_4$ , then the causal graph and latent structure are represented by the causal graph in Figure 5. This follows from the fact that the variables with measurement error  $\tilde{X}_i$  can only have incoming arrows from  $X_i$  and  $M_i$  and never have any outgoing arrows. Because  $L$  d-separates all  $\tilde{X}_i$  there can be no collider at  $L$ .

Before we present our upper bound result, we introduce an adjusted covariance matrix:

$$\tilde{\Sigma}(u, j) = \tilde{\Sigma} - u \text{diag}(e_j),$$

where  $j \in \{1, 2, 3, 4\}$  and  $e_j$  is a standard basis vector. For all  $u$  such that  $\tilde{\Sigma}(u, j)$  is a valid covariance matrix, the adjusted partial correlations  $\tilde{\rho}_{ik|j}^u$  may be calculated from  $\tilde{\Lambda}(u, j) = (\tilde{\Sigma}(u, j))^{-1}$  as follows:

$$\tilde{\rho}_{ik|j}^u = -\frac{(\tilde{\Lambda}(u, j))_{ik}}{\sqrt{(\tilde{\Lambda}(u, j))_{ii}(\tilde{\Lambda}(u, j))_{kk}}}. \quad (2)$$

Theorem 1 shows how the adjusted partial correlation is related to the underlying causal graph in Figure 5. Corollary 1 shows how we can use adjusted partial correlations to find an upper bound for the measurement error on one variable.

<sup>2</sup>These conditions are known as *tetrad conditions* in the literature, see Bollen [1989], Sullivant et al. [2010], Drton et al. [2008], Sullivant et al. [2010]

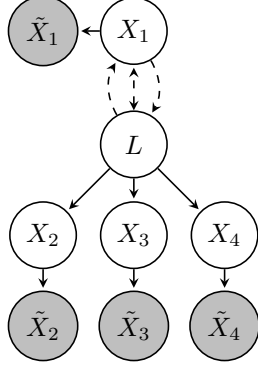


Figure 5: Causal graph of upper bound pattern for model with random measurement error, where at least one of the dashed edges is present. The indexes  $1, \dots, 4$  can be permuted. Noise variables  $E_1, \dots, E_4$  may be present but are not drawn. Measurement errors  $M_1, \dots, M_4$  are present but not drawn.

**Theorem 1.** Let  $X_1, \dots, X_4, \tilde{X}_1, \dots, \tilde{X}_4$  and  $\tilde{\rho}_{ij}$  be as in Lemma 1. The true underlying DAG is as in Figure 5 if and only if there exists  $u > 0$  such that  $\tilde{\rho}_{13|2}^u = \tilde{\rho}_{14|2}^u = \rho_{34|2}^u = 0$ .

*Proof.* The proof can be obtained by explicitly calculating the adjusted partial correlations and applying Lemma 1. A complete proof can be found in the supplementary material.  $\square$

**Corollary 1.** Let  $m_2$  be the variance of the measurement error on  $X_2$ . If  $\tilde{\rho}_{13|2}^{u^*} = \tilde{\rho}_{14|2}^{u^*} = \tilde{\rho}_{34|2}^{u^*} = 0$  for some  $u^* > 0$  then  $m_2 \leq u^*$ .

*Proof.* Follows from the proof of Theorem 1.  $\square$

These results can also be applied in a practical, more general setting. If data is generated from a random measurement error model for variables  $V$ , we consider subsets of four variables. If we can find an adjustment  $u^*$  on the covariance matrix of this subset of variables so that the adjusted partial correlations in Theorem 1 vanish, then Corollary 1 implicates that this adjustment is an upper bound for the variance of random measurement error. To ensure that the causal structure of these four variables is as in Figure 5, we can test for the constraints in Lemma 1 (see [Bollen, 1989, Silva et al., 2006, Thoemmes et al., 2018]). When all variables are measured in a similar manner, it may be reasonable to assume that the variance of the measurement error is the same for all variables. Under this assumption, the upper bound for the measurement error can be extended to an upper bound for the measurement error variance on all variables.

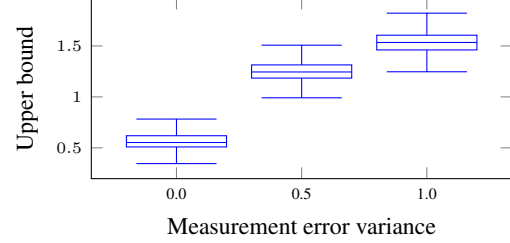


Figure 6: Simulation results for measurement error upper bound detection.

**Data simulations** To empirically test the performance of the upper bound, we simulated 10000 datapoints for 10000 random models with causal structures as in Figure 5 with parameters chosen uniformly from the interval  $[-1, 1]$  and error variances chosen uniformly from the interval  $[0.5, 1]$ . We added random measurement error to each variable (the same variance was used for all variables) and minimized the sum of adjusted partial correlations in Corollary 1 to obtain an upper bound. The result in Figure 6 shows that this leads to a correct upper bound on the variance of the measurement error.

## 5 STRONG FAITHFULNESS

In this section we prove that conditional independences cannot be reliably detected in the presence of measurement error. We then discuss the *strong* faithfulness assumption and its repercussions. In the next section we will present our error correction method, which relies on an upper bound for measurement error and the strong faithfulness assumption.

Lemma 2 shows that for two dependent (sets of) variables, a conditional independence between these variables can never be detected if the conditioning set is subject to measurement error, *unless* the faithfulness assumption is violated.

**Lemma 2.** Let  $X, Y$  and  $\tilde{Z}$  be three sets of (disjoint) variables. If  $\tilde{Z}$  has measurement error with non-zero variance, then the (in)dependences

$$X \not\perp\!\!\!\perp Y \quad X \perp\!\!\!\perp Y | \tilde{Z},$$

must be due to a violation of the faithfulness assumption.

*Proof.* A faithfulness violation occurs when  $X \perp\!\!\!\perp Y | \tilde{Z}$  but  $\tilde{Z}$  does not d-separate  $X$  and  $Y$ . Since  $X$  and  $Y$  are dependent in the data there must be an open path between them by the Markov assumption. By definition of random measurement error the variables in  $\tilde{Z}$  are leaf nodes. Therefore  $\tilde{Z}$  cannot block the path between  $X$  and  $Y$ , so that  $X \not\perp\!\!\!\perp Y | \tilde{Z}$ .  $\square$

Under the assumption that all variables in the model have the same measurement error variance (e.g. because they are subject to the same source of measurement error), the variance of the measurement error must be zero whenever a marginal dependence and a conditional independence is detected, as shown in Proposition 1.

**Proposition 1.** *Let  $\tilde{X}, \tilde{Y}$  and  $\tilde{Z}$  be three sets of (disjoint) variables with measurement errors that have equal (possibly zero) variances. Under the faithfulness assumption, if  $\tilde{X} \not\perp\!\!\!\perp \tilde{Y}$  and  $\tilde{X} \perp\!\!\!\perp \tilde{Y} | \tilde{Z}$ , then the measurement error on all variables has zero variance.*

*Proof.* Follows directly from Lemma 2.  $\square$

Since constraint-based causal discovery algorithms rely both on the faithfulness assumption and on the results of conditional independence tests, poor performance is to be expected when variables are measured with error. In this article, we consider the *strong faithfulness* assumption [Spirtes et al., 2000] instead.

**Assumption 2.** (Strong faithfulness) *We assume that the data of the unobserved measurement-error-free variables is  $\lambda$ -strong faithful to the true underlying causal graph that generated it. That is, for all disjoint sets of variables  $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ :*

$$|\rho_{\mathbf{X}, \mathbf{Y} | \mathbf{Z}}| < \lambda \implies \mathbf{X} \perp\!\!\!\perp \mathbf{Y} | \mathbf{Z}.$$

The example in Figure 4 illustrates how the strong faithfulness assumption may alleviate some of the negative effects of measurement error, but may aggravate the risk of detecting wrong conditional independences. If the data is  $\lambda$ -strong faithful, then it is also  $\mu$ -strong faithful, where  $0 < \mu \leq \lambda$ , and  $\mu$  can then be treated as a tuning parameter. In Figure 4, for zero relative measurement error, the data is  $\mu$ -strong faithful for any  $\mu$  up to  $\lambda \sim 0.25$ . For  $\mu = 0.25$  we find from the partial correlations that  $X_1 \perp\!\!\!\perp \tilde{X}_3 | \tilde{X}_2$  upto a relative measurement error of approximately 0.3, but for large enough measurement error we may also wrongly detect that  $X_1 \perp\!\!\!\perp \tilde{X}_2 | X_3$ .<sup>3</sup>

The tuning parameter thus represents a trade-off between detecting as many as possible of the true conditional independences and wrongfully detecting conditional independences. For the identification of LCD triples this means that for small  $\mu$  and data that is corrupted by measurement error, we cannot detect the true LCD triples, while for large  $\mu$  we may detect false LCD triples, because we detect conditional independences between variables that are actually dependent.

<sup>3</sup>Small enough correlations correspond to d-separations in the underlying graph by the strong faithfulness assumption. By the causal Markov assumption, d-separations correspond to conditional independences.

## 6 ERROR PROPAGATION

In this section we consider propagation of an error bound on random measurement error to partial correlations. If the strong faithfulness assumption holds, the effectiveness of tuning the threshold parameter  $\lambda$  depends on the size of the measurement error. By taking measurement error into account, we aim to alleviate the adverse effect of wrongfully detecting conditional independences by including the possibility to adaptively assign ‘unknown’ to a statistical test result. In that case we could get the best of both worlds: detect the correct conditional independences and assign ‘unknown’ or ‘dependent’ to the conditional dependences.

We start by defining an adjusted covariance matrix for three variables. Let  $\mathbf{m} = (m_1, m_2, m_3)$  be the variances of the random measurement errors ( $M_1, M_2, M_3$ ) on the latent (unmeasured) variables ( $X_1, X_2, X_3$ ), and suppose that  $\mathbf{u}^* = (u_1^*, u_2^*, u_3^*)$  is an upper bound such that  $\mathbf{m} \preceq \mathbf{u}^*$ .<sup>4</sup> Suppose that  $\tilde{\Sigma}$  is the true covariance matrix of the measured variables  $\tilde{X}_1, \tilde{X}_2, \tilde{X}_3 \in \mathbf{V}$ . The adjusted covariance matrix is given by

$$\tilde{\Sigma}(\mathbf{u}) = \tilde{\Sigma} - \mathbf{u}^T I, \quad (3)$$

where  $I$  denotes the identity matrix, when  $\tilde{\Sigma}(\mathbf{u})$  has an inverse, otherwise  $\tilde{\Sigma}(\mathbf{u}) = \tilde{\Sigma}$ .

For  $0 \preceq \mathbf{u} \preceq \mathbf{u}^*$  we can find minimal and maximal absolute values of partial correlations based on  $\tilde{\Lambda}(\mathbf{u}) = (\tilde{\Sigma}(\mathbf{u}))^{-1}$ . We define

$$\tilde{\rho}_{12|3}^{\min} = \arg \min_{0 \preceq \mathbf{u} \preceq \mathbf{u}^*} \left| \frac{(\tilde{\Lambda}(\mathbf{u}))_{12}}{\sqrt{(\tilde{\Lambda}(\mathbf{u}))_{11}(\tilde{\Lambda}(\mathbf{u}))_{22}}} \right|, \quad (4)$$

$$\tilde{\rho}_{12|3}^{\max} = \arg \max_{0 \preceq \mathbf{u} \preceq \mathbf{u}^*} \left| \frac{(\tilde{\Lambda}(\mathbf{u}))_{12}}{\sqrt{(\tilde{\Lambda}(\mathbf{u}))_{11}(\tilde{\Lambda}(\mathbf{u}))_{22}}} \right|. \quad (5)$$

Under the  $\lambda$ -strong faithfulness assumption, the conditional (in)dependence relations can be determined as follows:

$$\begin{cases} X_1 \not\perp\!\!\!\perp X_2 | X_3 & \text{if } \tilde{\rho}_{12|3}^{\min} > \lambda \\ X_1 \perp\!\!\!\perp X_2 | X_3 & \text{if } \tilde{\rho}_{12|3}^{\max} < \lambda, \end{cases} \quad (6)$$

The nature of the relation is undecided when  $\tilde{\rho}_{12|3}^{\min} < \lambda$  and  $\tilde{\rho}_{12|3}^{\max} > \lambda$ .<sup>5</sup>

Although we consider a measurement error correction in cases where only one variable is conditioned upon, our

<sup>4</sup> $\preceq$  is the component-wise inequality between two vectors.

<sup>5</sup>In practical applications the covariance matrix  $\tilde{\Sigma}$  is estimated from data. The added uncertainty can be taken into account by using bootstrapping to obtain confidence intervals for  $\tilde{\rho}_{12|3}^{\min}$  and  $\tilde{\rho}_{12|3}^{\max}$ .

ideas can be trivially extended to accommodate larger conditioning sets when an upper bound on the measurement error is known for all variables involved<sup>6</sup>.

## 7 DATA SIMULATIONS

We now evaluate the effects of a measurement error correction on simulated data. For detailed descriptions of the simulation settings, we refer to the supplementary material.

### 7.1 CONDITIONAL INDEPENDENCE TESTING

To illustrate the effectiveness of the measurement error correction in identifying conditional (in)dependence relations, we generated data for three variables  $(X_1, X_2, X_3)$  from linear-Gaussian acyclic causal structures, possibly with latent confounders. We only considered triples that satisfied the  $\lambda$ -strong faithfulness assumption for  $\lambda = 0.1$  and  $X_1 \not\perp\!\!\!\perp X_2$  and  $X_2 \not\perp\!\!\!\perp X_3$ .

We simulated 2000 models where half of the models satisfied  $X_1 \perp\!\!\!\perp X_3 \mid X_2$ . From each model we generated 10000 samples and added normally distributed random measurement error to each variable with varying variance. The conditional (in)dependence between  $\tilde{X}_1 \perp\!\!\!\perp \tilde{X}_3 \mid \tilde{X}_2$  was tested in various ways: using a threshold on the p-value  $\alpha = 0.05$ , using a threshold  $\lambda = 0.1$  on the partial correlation, and using the same threshold with a measurement error correction with an upper bound on the measurement error of  $t$  times the true variance. We then calculated the error rate as the number of incorrect classifications relative to the total number of tests. Note that the amount of conditional (in)dependence relations that are assigned ‘unknown’ increases with the size of the measurement error and the tightness of the upper bound. For an evaluation of the amount of ‘unknown’ classifications we refer to the supplementary material.

Figure 7a shows that the measurement error correction slightly reduces the error rate for conditional dependences, and 7b shows that the error rate of detecting incorrect conditional dependences is greatly reduced.

### 7.2 APPLICATION TO LCD

Typically, when data is  $\lambda$ -strong faithful to the true underlying causal graph, the value of  $\lambda$  is not known and  $\lambda$  is therefore used as a tuning parameter instead. We generated triples  $(X_1, X_2, X_3)$  as in the previous simulation, but only selected triples where  $X_1$  was not caused

<sup>6</sup>In that case one considers a larger adjusted covariance matrix, and since the partial correlations are calculated from the covariance matrix one can use the same scheme to find minimal and maximal values for the absolute partial correlation.

by  $X_2$  and  $X_3$ . We added measurement error with a fixed variance. We then applied the LCD algorithm, testing conditional independences as in the previous section. We evaluated the results by checking whether the causal structure of the triples was correctly identified. Figure 7c shows that the precision of the algorithm with the measurement error corrected test results outperforms the standard methods.

We also consider the more realistic case where multiple variables are measured, the upper bound for the variance of the measurement error is not known in advance, and the data is not necessarily  $\lambda$ -strong faithful. To that end we simulated 10000 datapoints from a random acyclic model with 15 variables, where one variable was not caused by any of the other variables. We added measurement error to each variable with a fixed variance.

For the upper bound detection, we first tested whether the tetrad constraints vanished using Wishart’s test [Wishart, 1928] at the 5% level, and then used the result in Corollary 1 to obtain an upper bound for the measurement error. When we found multiple upper bounds (for multiple variables) we chose the median as an upper bound for the measurement error on all variables. Finally we applied the LCD algorithm, testing marginal (in)dependences with a t-test at the 5% level and conditional (in)dependences as in the previous experiments and using the detected upper bound. If we were not able to detect an upper bound, we assigned ‘unknown’ to every test result. We checked how often a correct causal structure was identified. In 200 repetitions of the experiment the upper bound was incorrect in only 3 cases and no upper bound was detected in 39 cases. Figure 7d shows that all methods score significantly better than the random baseline and that the precision for detecting LCD triples increases significantly when we use the measurement error correction.

## 8 PROTEIN SIGNALING NETWORKS

We present an application of our ideas to real-world protein signaling data that could be corrupted by measurement error. We used a dataset concerning the influence of protein abundances on the properties of a protein signaling network in human kidney cells [Lun et al., 2017], and obtained an upper bound for the variance of random measurement error from this data. In absence of a reliable ground truth for this experiment, we validated the results of a measurement error correction applied to the LCD algorithm by comparing it to a baseline derived from interventions in the data.

**Data description** For conditions  $j = 1, \dots, 20$  the abundance of a different protein labeled  $(GFP)_j$  was

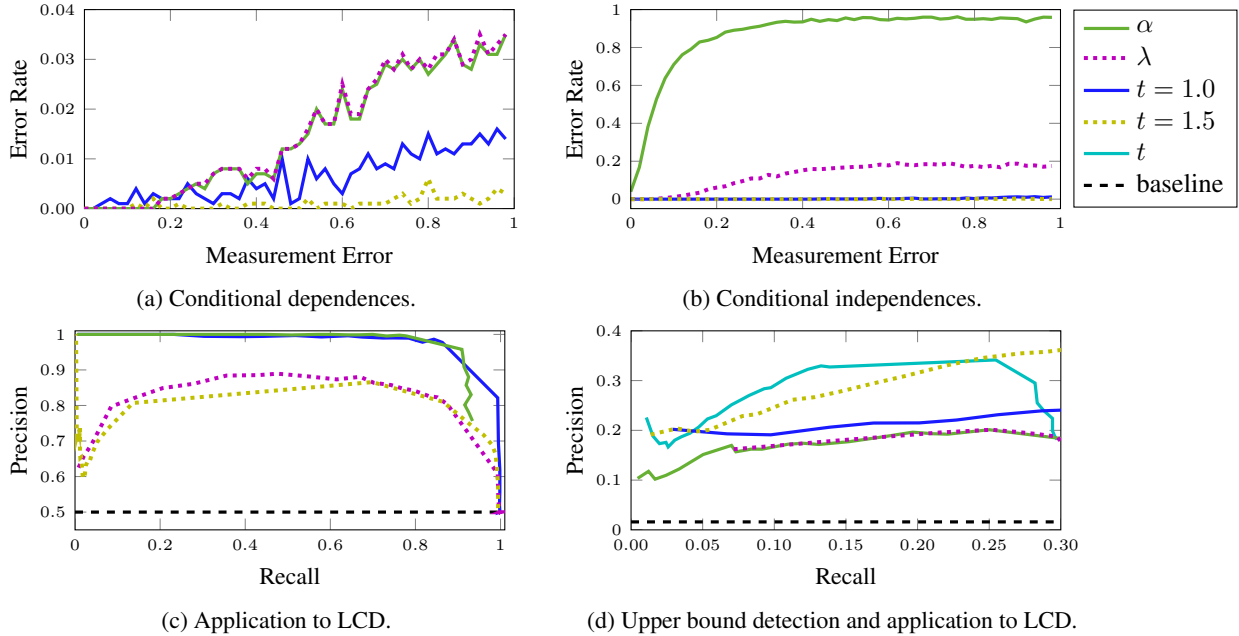


Figure 7: Simulation results. Figures a and b show the error rate for detecting conditional dependences and independences in the presence of measurement error for  $\lambda$ -strong faithful data. It is assumed that  $\lambda = 0.1$  is known, and  $\alpha = 0.05$ . Figure c shows the precision-recall curve for detecting LCD triples from  $\lambda$ -strong faithful data subject to measurement error with fixed variance and a given upper bound, where  $\lambda$  and  $\alpha$  are used as tuning parameters. Figure d shows the precision-recall curve for simulations of 15 variables, where we first apply the upper bound detection and then the measurement error correction and  $\alpha$  and  $\lambda$  are treated as tuning parameters. The baseline is at 0.016.

over-expressed and then measured [Lun et al., 2017]. The abundances of an additional 34 *phosphorylated* proteins  $P_i$  were measured after stimulation of the network. We relabeled conditions  $j$  so that over-expression of a protein  $(GFP)_j$  corresponds to the measured phosphorylated abundance  $P_j$ .

The abundance of an over-expressed protein typically differed between cells and not every cell was affected [Lun et al., 2017]. Because of the experimental design,  $(GFP)_j$  is not caused by the abundance or phosphorylation of the other proteins, which allowed us to treat the abundance of  $(GFP)_j$  as an intervention variable.

Typically  $\sim 10000$  single cells were measured for each condition. We assume that the data-generating process can be approximated by a linear-Gaussian model after pre-processing. For details about data pre-processing we refer to the supplementary material.

**Upper bound detection** We considered all proteins under over-expression of the SRC protein, for which strong signaling relations were present (see also [Lun et al., 2017]). For all 4-tuples  $((GFP)_{SRC}, P_i, P_j, P_k)$  that were all marginally dependent at the 1% level (using a t-test), we tested whether all three tetrads vanished using Wishart’s test at the 5% level. We found that these

constraints were satisfied for the 4-tuple  $((GFP)_{SRC}, pS6K, pMAPKAPK2, pMAP2K3)$ .

This allowed us to apply the results presented in Section 4 to obtain an upper bound. The upper bounds for the variance of measurement error that we found were 0.10 for pS6K, 0.15 for pMAPKAPK2, and 0.14 for pMAP2K3.<sup>7</sup> Since all proteins were measured with the same device, we assumed that the variance of the measurement error is the same for each variable, so that 0.14 is a suitable upper bound for the measurement error on any variable.

Although the detected upper bound was large for weak signals, the proteins with stronger signals typically had variances  $> 1$ , so that the relative amount of measurement error for proteins with strong signaling relations amounted to less than 10%.

**Baseline** To validate the results of LCD, we created a baseline from the interventions (corresponding to over-expression of certain proteins) in the dataset. A reasonable assumption is that  $(GFP)_j$  is a direct cause of  $P_j$ ,

<sup>7</sup>Each of the detected other bounds corresponds to adjusting the corresponding variable, as in Corollary 1. Other triples that satisfied the constraints gave similar or (much) higher upper bounds for the measurement error.

because the higher the abundance of a protein, the more it can be phosphorylated. Under the assumption that over-expression of a protein  $P_j$  does not alter the network structure [Lun et al., 2017] and that  $(\text{GFP})_j$  does not directly cause any of the other proteins  $P_i$ , with  $i \neq j$ , we have that  $P_j$  is a cause of  $P_k$ , whenever  $(\text{GFP})_j$  and  $P_k$  are dependent.

We constructed a baseline for cause-effect pairs  $(P_j, P_k)$ , where we considered 7 phosphorylated proteins  $P_j$  that were over-expressed in one of the conditions as cause variables and all 34 phosphorylated proteins as effect variables. The subset of proteins that was used to construct the baseline follows the recommendations in Lun et al. [2017]. We considered a pair  $(P_j, P_k)$  a causal pair, if a t-test indicated that  $(\text{GFP})_j$  and  $P_k$  were dependent at a level of  $10^{-4}$ . This resulted in 231 possible cause-effect pairs, 71% of which were cause-effect pairs in the baseline.

**Methods and results** We applied the LCD algorithm to the data to identify causal pairs  $(P_j, P_k)$  by treating  $(\text{GFP})_i$  as an intervention variable for conditions  $i \in \{1, \dots, 20\}$ , with  $i \neq j$  and  $i \neq k$ . Since central proteins in the network were over-expressed, true causal pairs were expected to appear under multiple conditions. To make our results more robust, we only made a positive prediction if a causal pair was predicted for at least 2 conditions. We applied three methods of conditional (in)dependence testing in combination with the LCD algorithm: a threshold  $\alpha$  on the p-value of t-tests, a threshold  $\lambda$  on the absolute value of partial correlations, and a threshold  $\lambda$  on partial correlations with a measurement error correction using the upper bound  $u = 0.14$ .

By treating  $\lambda$  and  $\alpha$  as tuning parameters and taking the baseline as ground truth, we calculated a precision-recall curve for each method of conditional (in)dependence testing. The results are displayed in Figure 8, which shows that  $\alpha$  and  $\lambda - u$  have comparable pr-curves. For this dataset there seems to exist a threshold  $\alpha$  that is already able to distinguish between conditional independences and dependences. We can see that both  $\lambda - u$  and  $\alpha$  significantly outperform random guessing, but  $\lambda$  does not. Although the differences between the methods are not significant, this seems to indicate that a measurement-error correction improves the precision at low recall for conditional independence testing with a fixed threshold on (partial) correlations.

## 9 CONCLUSION

In this paper we demonstrated that measurement error, when not taken into account, can fool causal discovery methods into wrongfully inserting, deleting or reversing

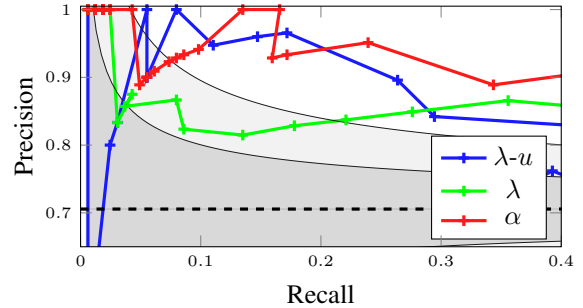


Figure 8: LCD applied to protein signaling data with  $\alpha$  or  $\lambda$  as tuning parameter and a measurement-error correction. The results are compared with the random baseline, the gray-shaded areas represent one and two standard deviations from the random baseline.

edges in the predicted causal graph. We showed that regular statistical tests with conventional thresholds would fail to detect conditional independences between the uncorrupted variables from the measurement data when measurement error is present. We also proposed a correction method aimed at mitigating the negative effects of measurement error.

The key result that we presented in this work is that, under certain conditions, we can find an upper bound for the variance of the measurement error from data that has been corrupted by measurement error. We show that this uncertainty can be propagated into an uncertainty regarding the partial correlations to correct for measurement error. We showed a successful application of our approach on simulated data.

We also applied our ideas to a real-world protein signaling dataset, and we found an upper bound for the variance of the measurement error in this dataset. We found that our approach gave significantly higher precision than a random baseline. However, also the conventional method without correction for measurement error seems to work well on this dataset. Nevertheless, it is our belief that taking measurement error into account is a promising step towards successful real-world applications of (constraint-based) causal discovery.

## Acknowledgements

This work was supported by the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement 639466). We thank Ioannis Tsamardinos, Sofia Triantafillou and Karen Sachs for providing useful feedback on initial drafts of this work.

## References

- J. Pearl. *Causality: models, reasoning, and inference*. Cambridge University Press, 2000.
- P. Spirtes, C. Glymour, and R. Scheines. *Causation, prediction, and search*. MIT Press, 2000.
- R. Scheines and J. Ramsey. Measurement error and causal discovery. *CEUR workshop proceedings*, 1792: 1–7, 2016.
- K. Zhang, M. Gong, J. Ramsey, K. Batmanghelich, P. Spirtes, and C. Glymour. Causal discovery in the presence of measurement error: identifiability conditions. In *UAI workshop on causality*, 2017.
- J. Pearl. On measurement bias in causal inference. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, 2010.
- M. Kuroki and J. Pearl. Measurement bias and effect restoration in causal inference. *Biometrika*, 101(2): 423–437, 2014.
- R. Silva, R. Scheines, C. Glymour, and P. Spirtes. Learning the structure of linear latent variable models. *Journal of Machine Learning Research*, 7:191–246, 2006.
- K.A. Bollen. *Structural equations with latent variables*. John Wiley & Sons, Inc., 1989.
- S. Sullivant, K. Talaska, and J. Draisma. Trek separation for Gaussian graphical models. *The Annals of Statistics*, 38(3):1665–1685, 2010.
- N. Harris and M. Drton. PC-algorithm for nonparanormal graphical models. *Journal of Machine Learning Research*, 14:3365–3383, 2013.
- S. Triantafyllou, V. Lagani, C. Heinze-Deml, A. Schmidt, J. Tegner, and I. Tsamardinos. Predicting causal relationships from biological data: applying automated causal discovery on mass cytometry data of human immune cells. *Scientific reports*, 7(1):12724, 2017.
- M. Kalisch and P. Bühlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *Journal of Machine Learning Research*, 8: 613–636, 2007.
- M. Maathuis, D. Colombo, M. Kalisch, and P. Bühlmann. Predicting causal effects in large-scale systems from observational data. *Nature Methods*, 7:247–248, 2010.
- G.F. Cooper. A simple constraint-based algorithm for efficiently mining observational databases for causal relationships. *Data Mining and Knowledge Discovery*, 1(2):203–224, 1997.
- M. Drton, H. Massam, and I. Olkin. Moments of minors of Wishart matrices. *The Annals of Statistics*, 36(5): 2261–2283, 2008.
- F. Thoemmes, Y. Rosseel, and J. Textor. Local fit evaluation of structural equation models using graphical criteria. *Psychological Methods*, 23(1):27–41, 2018.
- J. Wishart. Sampling errors in the variance of two factors. *British Journal of Psychology*, 19:180–187, 1928.
- X. Lun, V.R.T. Zanutelli, J.D. Wade, D. Schapiro, M. Tognetti, N. Dobberstein, and B. Bodenmiller. Influence of node abundance on signaling network state and dynamics analyzed by mass cytometry. *Nature Biotechnology*, 35(2):164–172, 2017.

---

# IDK Cascades: Fast Deep Learning by Learning not to Overthink

---

Xin Wang, Yujia Luo, Daniel Crankshaw, Alexey Tumanov, Fisher Yu, Joseph E. Gonzalez

Department of Electrical Engineering and Computer Sciences

University of California, Berkeley

{xinw, yujialuo, crankshaw, atumanov, fy, jgonzal}@berkeley.edu

## Abstract

Advances in deep learning have led to substantial increases in prediction accuracy but have been accompanied by increases in the cost of rendering predictions. We conjecture that for a majority of real-world inputs, the recent advances in deep learning have created models that effectively “over-think” on simple inputs. In this paper we revisit the classic question of building model cascades that primarily leverage class asymmetry to reduce cost. We introduce the “*I Don’t Know*” (IDK) prediction cascades framework, a general framework to systematically compose a set of pre-trained models to accelerate inference without a loss in prediction accuracy. We propose two search based methods for constructing cascades as well as a new cost-aware objective within this framework. The proposed IDK cascade framework can be easily adopted in the existing model serving systems without additional model re-training. We evaluate the proposed techniques on a range of benchmarks to demonstrate the effectiveness of the proposed framework.

## 1 INTRODUCTION

Advances in deep learning have enabled substantial recent progress on challenging machine learning benchmarks. As a consequence, deep learning is being deployed in real-world applications, ranging from automated video surveillance, to voice-powered personal assistants, to self-driving cars. In these applications, accurate predictions must be delivered in *real-time* (e.g, under 200ms) under *heavy query load* (e.g., processing millions of streams) with *limited resources* (e.g., limited GPUs and power).

The need for accurate, low-latency, high-throughput, and

low-cost predictions has forced the machine learning community to explore a complex trade-off space spanning model and system design. For example, several researchers have investigated techniques for performing deep learning model compression [1, 2, 3]. However, model compression primarily reduces model memory requirements so as to fit on mobile devices or in other energy-bounded settings. There is a limit to how far compression-based techniques can be pushed to reduce latency at inference time while retaining state-of-the-art accuracy across all inputs.

We conjecture that in the pursuit of improved classification accuracy the machine learning community has developed models that effectively “*overthink*” on an increasing fraction of queries. To support this conjecture we show that while the cost of computing predictions has increased by an order of magnitude over the past 5 years, the accuracy of predictions on a large fraction of the ImageNet 2012 validation images has remained constant (see Fig. 2). This observation suggests that *if* we could distinguish between easy and challenging inputs (e.g., images) and only apply more advanced models when necessary, we could reduce computational costs without impacting accuracy. In this paper we study the design of *prediction cascades* as a mechanism to *exploit this conjecture* by combining fast models with accurate models to increase throughput and reduce mean latency without a loss in accuracy.

Though prediction cascades are well established in the machine learning literature [4, 5, 6, 7], the classic approaches focus on detection tasks and developed cascades for early rejection of negative object or region proposals – leveraging the class asymmetry of detection tasks. In this paper, we revisit the question of how to effectively build model cascades with little training overhead to trade off between prediction accuracy and cost, extending to any multi-class classification task.

We introduce *IDK prediction cascades*, a general framework to accelerate inference without reducing prediction



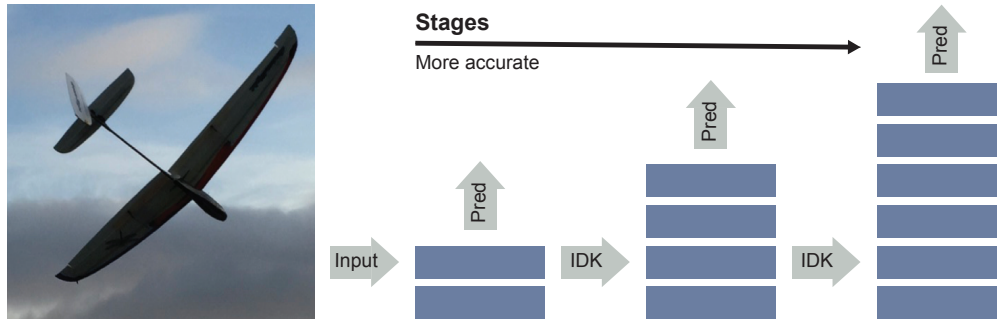


Figure 1: An IDK prediction cascade combines IDK classifiers of increasing accuracy and computational cost such that each will either render a high-accuracy prediction or return IDK passing the input to the next model in the cascade for a more accurate but higher cost prediction

accuracy by composing pre-trained models. IDK prediction cascades (see Fig. 1) are composed of IDK classifiers which are constructed by attaching an augmenting classifier to the existing classifiers, *base models*, enabling the IDK classifiers to predict an auxiliary “I don’t know” (IDK) class besides the original prediction classes. The augmenting classifiers, which are *light-weighted* (the computational cost is negligible compared to the cost of the base models) and *independent* from the base model architectures, measure the uncertainty of the base model predictions. When an IDK classifier predicts the IDK class the subsequent model in the cascade is invoked. The process is repeated until either a model in the cascade predicts a real class or the end of the cascade is reached at which point the last model must render a prediction. Furthermore, we can introduce a human expert as the last model in an IDK cascade to achieve nearly perfect accuracy while minimizing *the cost* of human intervention.

The base models in the model cascades are treated as black-boxes and thus the proposed framework can be applied to any existing model serving systems [8, 9] with little modification. In addition, the proposed IDK cascade framework model naturally fits the edge-cloud scenario where the fast models can be deployed on edge devices (e.g. Nvidia Drive PX2, Jetson TX2, etc.) while the expensive models are stored in the cloud and are only triggered when the fast model is not certain about a prediction.

To build such model cascade, we need to address the following problems: (1) without retraining the base models or obtaining the base model architecture, what is the best measure available to distinguish the easy and hard examples in the workload? (2) to construct the IDK classifiers that can effectively decide the execution path for the given input while not introducing additional computational overhead, what is the proper objective to balance the computational cost and the overall prediction accuracy of the model cascades?

In this work, we propose two search-based methods for constructing IDK classifiers: *cascading by probability* and *cascading by entropy*. Cascade by probability examines the confidence scores of a model directly to estimate uncertainty. Cascade by entropy leverages well-calibrated class conditional probabilities to estimate model uncertainty. Both techniques then search to find the optimal uncertainty threshold at which to predict the IDK class for each model in the cascade. When the uncertainty in a predicted class exceeds this threshold, the model predicts the IDK class instead. While both search-based methods produce reasonable prediction cascades, neither leverages the cascade design when *training* the augmenting classifier.

As a third approach to constructing IDK classifiers we cast the IDK cascade problem in the context of empirical risk minimization with an additional computational cost term and describe how the objective can be easily incorporated into gradient based learning procedures. The empirical risk minimization based approach allows the IDK classifier to trade-off between *cascade accuracy* and *computational cost* when building the prediction cascade.

We apply all three techniques to the image classification task on ImageNet2012 and CIFAR10 to demonstrate we can reduce computation by 37%, resulting in a 1.6x increase in throughput, while maintaining state-of-the-art accuracy. We conduct a detailed study of the impact of adding the computational-cost term to the objective and show that it is critical for training the augmenting classifiers. Compared to the cascades built with cost-oblivious objectives which cannot usually achieve the desired accuracy, the proposed cost-aware objective better serves the goal of model cascading. Furthermore, we demonstrate that in a real autonomous vehicle setting the IDK cascades framework can be applied in conjunction with human experts to achieve 95% accuracy on driving motion prediction task while requiring human intervention less than 30% of the time.

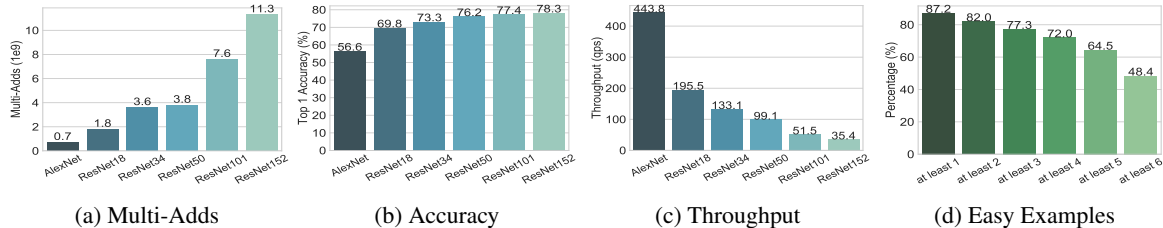


Figure 2: *ImageNet Model Statistics*: (a) Number of Multi-Adds of the top ImageNet models. *flops* is equivalent to multi-adds in this work and will be used in the following sections. (b) The top one prediction accuracy of the top ImageNet models. (c) Throughputs (query per second) of the top ImageNet models with batch size = 1. (d) The fraction of images correctly labeled by at least  $K \in \{1 \dots 6\}$  of the benchmark models

## 2 RELATED WORK

**Compression & Distillation.** Much of the existing work to accelerate predictions from deep neural networks has focused on model compression [3, 2, 1, 10] and distillation [11]. Denton et al. [2] applied low-rank approximations to exploit redundancy in convolution parameters to achieve a factor of two speedup with only 1% reduction in accuracy. Han et al. [10] introduced quantization and Huffman encoding methods to reduce network sizes by orders of magnitude and reduce prediction cost by a factor of 3 to 4. Our work on IDK prediction cascades is complementary to the work on model compression and focuses exclusively on decreasing computation costs. In fact, by coupling model compression with IDK cascades it may be possible to support more aggressive lossy compression techniques. Alternatively, Hinton et al. [11] proposed using soft-targets to transfer knowledge from a costly ensemble to a single model while largely preserving prediction accuracy. Our approach does not require retraining base models and instead focuses on accelerating inference by using more complex models only when necessary. The existing model compression and distillation techniques can be used to construct the fast base models while our framework serves as a bridge to connect models with different levels of complexity and accuracy.

**Cascaded Predictions.** Prediction cascades are a well suited approach to improve prediction performance. Much of the early work on prediction cascades was developed in the context of face detection. While Viola and Jones [5] are credited with introducing the terminology of prediction cascades, prior work by Rowley et al. [4] explored cascading neural networks by combining coarse candidate region detection with high accuracy face detection. More recently, Angelova et al. [6] proposed using deep network cascades and achieved real-time performances on pedestrian detection tasks. Cai et al. [7] also examined cascades for pedestrian detection, proposing a complexity aware term to regularize the cascade objective. While this approach has similarities to the loss function we propose, Cai et al. leverage the cost aware risk to

choose an optimal ordering of cascade elements rather than to train a specific classifier. These papers all focus on using cascades for detection tasks, and only use the earlier models in the cascade to reject negative region or object proposals more cheaply. Positive detection (e.g. object identification) can only be made by the final model in the cascade, which is the only model that can predict the full set of classes in the prediction task. Recently, Huang et al. [12] applied the cascading concept by allowing early exiting within the model. Instead of cascading features of a single model, we aim to cascade the trained models (one may not know the model structure or not be able to retrain) in a practical scenario.

**Uncertainty Classes.** The introduction of an IDK class to capture prediction uncertainty has also been studied under other settings. [13, 14]. Trappenberg [13] introduced an “*I don’t know*” (IDK) class to learn to identify input spaces with high uncertainty. Khani et al. [14] introduced a “*don’t know*” class to enable classifiers to achieve perfect precision when learning semantic mappings. In both cases, the addition of an auxiliary uncertainty class is used to improve prediction accuracy rather than performance. We build on this work by using the IDK class in the construction of cascades to improve performance.

## 3 A MOTIVATING EXAMPLE

In the pursuit of improved accuracy, deep learning models are becoming increasingly expensive to evaluate. To illustrate this trend, in Fig. 2c we plot the throughput for six benchmark models on the ImageNet 2012 datasets. We observe that prediction throughput has decreased by more than an order of magnitude. We expect the trend towards more costly models to continue with improvements in model design and increased adoption of ensemble methods [11]. In contrast, as Fig. 2b shows, the gains in prediction accuracy have increased much more slowly. A result of this trend is that even the cheaper and less advanced models can correctly classify many of the examples.

**Easy Samples.** For many prediction tasks, less accurate models are adequate *most of the time*. For example, a security camera may observe an empty street most of the time and require a more sophisticated model only in the infrequent events that people or objects enter the scene. Even in the standard benchmarks, many of the examples can be correctly classified by older, less advanced models. In Fig. 2d we plot the percentage of images that were correctly classified by an increasing fraction of models. We observe that a large fraction ( $\approx 48\%$ ) of the images are correctly classified by all six of the models, suggesting that these images are perhaps *inherently easier* and may not require the recent substantial increases in model complexity and computational cost.

## 4 IDK PREDICTION CASCADES

We start describing the IDK prediction cascade framework by examining simple two model cascades and then extend these techniques to deeper cascades at the end of this section. We start by formalizing two element cascades for the multiclass prediction problem.

We consider the  $k$  class multiclass prediction problem in which we are given two pre-trained models: (1) a fast but less accurate model  $m^{\text{fast}}$  and (2) an accurate but more costly model  $m^{\text{acc}}$ . In addition, we assume that the fast model estimates the class conditional probability:

$$m^{\text{fast}}(x) = \hat{\mathbf{P}}(\text{class label} \mid x). \quad (1)$$

Many multi-class estimators (e.g., DNNs trained using cross entropy) provide class conditional probabilities. In addition, we are given a dataset  $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$  consisting of  $n$  labeled data points.

To develop IDK prediction cascades we introduce an additional **augmenting classifier**:

$$h_\alpha(m^{\text{fast}}(x)) \rightarrow [0, 1], \quad (2)$$

which evaluates the *distributional output* of  $m^{\text{fast}}(x)$  and returns a number between 0 and 1 encoding how *uncertain* the fast model  $m^{\text{fast}}(x)$  is about a given prediction. The **IDK classifier** is composed of the base model and the augmenting classifier. For simplicity, we will refer to training the augmenting classifier as training the IDK classifier. In this paper we consider several designs for the IDK classifier:

$$h_\alpha^{\text{prb}}(m^{\text{fast}}(x)) = \mathbb{I} \left[ \max_j m^{\text{fast}}(x)_j < \alpha \right] \quad (3)$$

$$h_\alpha^{\text{ent}}(m^{\text{fast}}(x)) = \mathbb{I} \left[ \mathbf{H} \left[ m^{\text{fast}}(x) \right] > \alpha \right] \quad (4)$$

$$h_\alpha^{\text{cst}}(m^{\text{fast}}(x)) = \sigma \left( \alpha_1 f_{\alpha_2} \left( m^{\text{fast}}(x) \right) + \alpha_0 \right), \quad (5)$$

where  $\mathbb{I}$  is the indicator function,  $\mathbf{H}$  is the *entropy* function:

$$\mathbf{H}[m^{\text{fast}}(x)] = - \sum_{j=1}^k m^{\text{fast}}(x)_j \cdot \log m^{\text{fast}}(x)_j, \quad (6)$$

and  $f$  is a feature representation of  $m^{\text{fast}}(x)$ .

While  $f$  can be any featurization of the prediction  $m^{\text{fast}}(x)$ , in this work we focus on the entropy featurization  $f = \mathbf{H}$  as this is a natural measure of uncertainty. When using the entropy featurization, the IDK classifier  $h^{\text{cst}}$  becomes a differentiable approximation of  $h^{\text{ent}}$  enabling direct cost based optimization. In our experimental evaluation, we also evaluate  $f_{\alpha_2}(m^{\text{fast}}(x)) = \mathbf{NN}_{\alpha_2}(m^{\text{fast}}(x))$  which recovers a neural feature encoding of  $x$  and allows us to assess a neural network based IDK classifier in the context of the differentiable cost based optimization.

Given an IDK classifier  $h_\alpha(\cdot)$  we can define a two element IDK prediction cascade as:

$$m^{\text{casc}}(x) = \begin{cases} m^{\text{fast}}(x) & \text{if } h_\alpha(m^{\text{fast}}(x)) \leq 0.5 \\ m^{\text{acc}}(x) & \text{otherwise.} \end{cases} \quad (7)$$

Thus, for a given choice of IDK classifier  $h_\alpha$  we only need to determine the optimal value for parameter  $\alpha$  to ensure maximum accuracy while minimizing the fraction of examples for which the more expensive model is required. In the following, we formalize this objective and describe a set of techniques for choosing the optimal value of  $\alpha$ .

Given the above definition of an IDK prediction cascade we can define two quantities of interest. We define the accuracy  $\mathbf{Acc}(m)$  of a model  $m$  as the zero-one prediction accuracy evaluated on our training data  $\mathcal{D}$ :

$$\mathbf{Acc}(m) = \frac{1}{n} \sum_{i=1}^n \mathbb{I} \left[ y_i = \arg \max_j m(x_i)_j \right]. \quad (8)$$

We define the IDK rate  $\mathbf{IDKRate}(h)$  of an IDK classifier  $h$  as the fraction of training examples that are evaluated by the next model in the cascade:

$$\mathbf{IDKRate}(h) = \frac{1}{n} \sum_{i=1}^n \mathbb{I} \left[ h_\alpha(m^{\text{fast}}(x_i)) > 0.5 \right]. \quad (9)$$

Our goal in designing a prediction cascade is then to maintain the accuracy of the more accurate model while minimizing the IDK rate (i.e., the fraction of examples that require the more costly  $m^{\text{acc}}$  model). We formalize this goal as:

$$\min_{\alpha} \mathbf{IDKRate}(h_\alpha) \quad (10)$$

$$\text{s.t.: } \mathbf{Acc}(m^{\text{casc}}) \geq (1 - \epsilon) \mathbf{Acc}(m^{\text{acc}}). \quad (11)$$

In the following we describe a set of search procedures for achieving this goal for each of the IDK classifier designs.

#### 4.1 BASELINE UNCERTAINTY CASCADES

Similar to [15], as a baseline, we propose using the *confidence scores* (i.e. probability over the predicted class) of  $m^{\text{fast}}(x)$  and follow the IDK classifier design (Eq. 3). The intuition is that if the prediction of  $m^{\text{fast}}$  is insufficiently confident then the more accurate classifier is invoked.

A more rigorous measure of prediction uncertainty is the entropy of the class conditional probability. We therefore propose an entropy based IDK classifier in Eq. 4. The entropy based IDK classifier captures the overall uncertainty as well as the certainty within the dominant class.

Due to the indicator functions neither the *cascade by probability* (Eq. 3) nor the *cascade by entropy* functions (Eq. 4) are differentiable. However, because the parameter  $\alpha$  is a single scalar, we can apply a simple grid to search procedure to find the optimal value for the threshold  $\alpha$ .

#### 4.2 REGULARIZING FOR PREDICTION COST

The uncertainty based cascades described above adopt a relatively simple IDK classifier formulation and rely on grid search to select the optimal parameters. However, by reframing the cascade objective in the context of regularized empirical risk minimization and defining a differentiable regularized loss we can admit more complex IDK classifiers.

In the framework of empirical risk minimization, we define the objective as the sum of the loss  $\mathbf{L}(\cdot, \cdot)$  plus the *computational cost*  $\mathbf{C}(\cdot)$  of invoking the cascaded model:

$$J(\alpha) = \frac{1}{n} \sum_{i=1}^n [\mathbf{L}(y_i, m_\alpha^{\text{casc}}(x_i)) + \lambda \cdot \mathbf{C}(m_\alpha^{\text{casc}}(x_i))] \quad (12)$$

where  $\lambda$  is a hyper parameter which determines the trade-off between the cascade accuracy and the computational cost. Because we are not directly optimizing the fast or accurate models we can adopt the zero-one loss which is compatible with our earlier accuracy goal:

$$\mathbf{L}(y_i, m_\alpha^{\text{casc}}(x_i)) = \left(1 - h_\alpha(m^{\text{fast}}(x))\right) \cdot \mathbb{I}[y_i, m^{\text{fast}}(x_i)] + h_\alpha(m^{\text{fast}}(x)) \cdot \mathbb{I}[y_i, m^{\text{acc}}(x_i)], \quad (13)$$

where  $\mathbb{I}[y_i, m(x_i)]$  is 1 if  $\arg \max_j m(x_i)_j \neq y_i$  and 0 otherwise. The IDK classifier  $h_\alpha(m^{\text{fast}}(x))$  governs which loss is incurred. It is worth noting that alternative loss formulations could be used to further fine-tune the underlying fast and accurate model parameters in the cascaded setting.

The cascaded prediction cost  $\mathbf{C}(\cdot)$  is defined as:

$$\mathbf{C}(m_\alpha^{\text{casc}}(x_i)) = c^{\text{fast}} + h_\alpha(m^{\text{fast}}(x)) \cdot c^{\text{acc}}, \quad (14)$$

where the computational cost of  $m^{\text{fast}}$  and  $m^{\text{acc}}$ , are denoted by  $c^{\text{fast}}$  and  $c^{\text{acc}}$  respectively. In practice, the cost of model  $m^{\text{fast}}$  and  $m^{\text{acc}}$  could be measured in terms of multi-adds, latency, or number of parameters. The formulation of Eq. 14 captures the cascade formulation in which the fast model is always evaluated and the accurate model is evaluated conditioned on the IDK classifier decision.

Combining both prediction loss and computational cost of the IDK cascade, we can now use this *regularized loss* objective function to optimize the IDK prediction cascade with stochastic gradient descent based algorithms. This objective allows us to optimize both prediction precision and overall computational cost in one pass and support more complex parametric IDK classifiers.

#### 4.3 BEYOND TWO ELEMENT CASCADES

We can extend the two element cascade to construct deeper cascades by introducing additional IDK classifiers between each model and then either optimizing the IDK classifier parameters in a stage-wise fashion or by jointly optimizing the IDK classifiers using the extended loss function. More precisely, for an  $N$ -model cascade where  $m^j$  is the  $j$ -th model in the cascade, we define  $N - 1$  IDK classifiers  $h_{\alpha_j}(m^j(x))$ . For non-differentiable IDK classifiers with scalar parameters we can apply the grid search procedure in a stage wise fashion starting with the least accurate model. For more complex differentiable IDK classifiers we can define an extended loss:

$$\mathbf{L}(y_i, m_\alpha^{\text{casc}}(x_i)) = \sum_{j=1}^N \prod_{q=0}^{j-1} p_q (1 - p_j) \mathbb{I}[y_i, m^j(x_i)] \quad (15)$$

where  $p_j = h_{\alpha_j}(m^j(x))$  and  $p_0 = p_N = 1$ . The computational cost function  $\mathbf{C}(\cdot)$  is then generalized as

$$\mathbf{C}(m_\alpha^{\text{casc}}(x_i)) = \sum_{j=1}^N \prod_{q=0}^{j-1} p_q c^j \quad (16)$$

## 5 EXPERIMENTS

In this section, we evaluate the proposed cascading methods in two scenarios: cascading machine learning models with different computation budgets and collaboration between algorithms and human.

To study the prediction accuracy and cost trade-off under each cascade design, we use standard image classification benchmark tasks and models. We evaluate the cascaded models on ImageNet 2012 [16] and CIFAR-10 [17]. We assess whether the proposed IDK cascade approaches can match the state-of-the-art accuracy while significantly reducing the cost of rendering predictions. We also evaluate the robustness of the proposed framework on CIFAR-10.

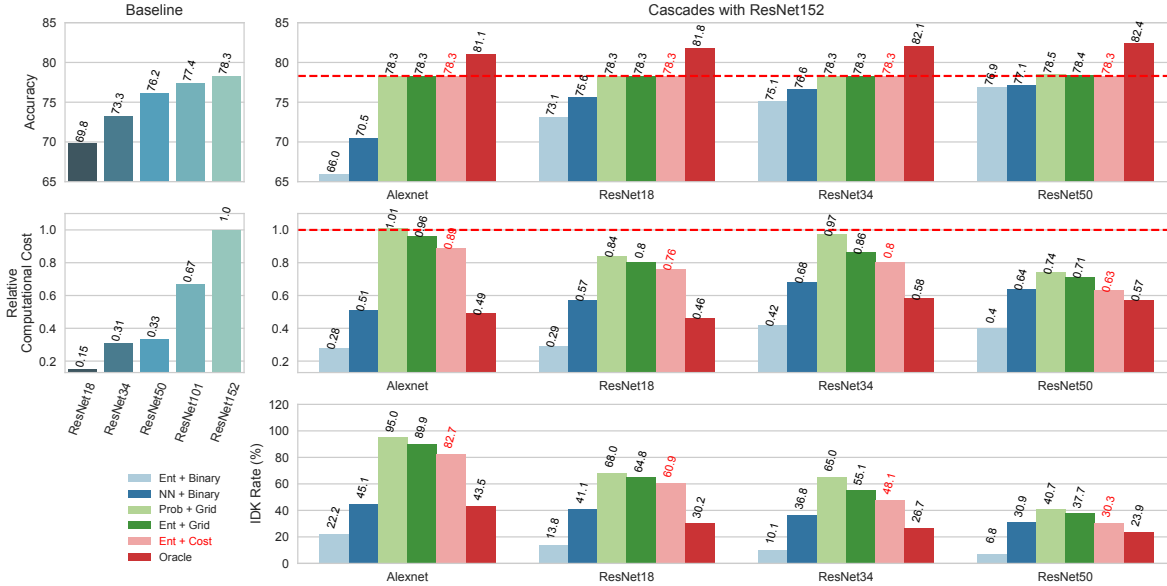


Figure 3: ImageNet Two Model Cascades. **Prob+Grid**: cascade by probability with grid search. **Entropy+Grid**: cascade by entropy with grid search. **NN+Binary**: Neural Network based IDK classifier trained with cost-oblivious cross-entropy loss. **Entropy+Binary**: Entropy based IDK classifier trained with cost-oblivious cross-entropy loss. **Entropy+Cost**: Entropy based IDK classifier with cost-aware objective. **Oracle**: Using ground truth correctness labels as IDK classifier. The comparison of **Prob+Grid**, **Entropy+Grid** and **Entropy+Cost** demonstrates that the proposed cost-aware objective is more effective in constructing model cascades with lower computational costs. The comparison of **NN+Binary**, **Entropy+Binary**, **Entropy+Cost** shows that the vanilla cross-entropy loss commonly used for binary classification leads to a model cascade with lower IDK rate but not achieving the *desired accuracy*

To assess how cascades can be used to augment models with human intervention, we evaluate a motion prediction task, a representative of autonomous vehicle workloads [18]. In this human-in-the-loop prediction task, the human serves as the *accurate* model to further improve the accuracy and safety of autonomous driving. The cascade design is used to determine when the fast model can no longer be trusted and human intervention is required (e.g., by taking over steering).

We evaluate each cascade using a range of different metrics. The **accuracy** and **IDK rate** correspond to the accuracy and IDK rate defined in Eq. 8 and Eq. 9 respectively. In the multi-model cascade setting, we measure **IDK Rate** at each level in the cascade. As a measure of computational cost we compute the **average flops** which is the average floating point arithmetic operations required by the model cascade. Finally, as a relative measure of runtime we compute the **relative computational cost** which is computed as  $FLOP_{casc}/FLOP_{acc}$ .

## 5.1 IMAGE CLASSIFICATION ON IMAGENET

We first demonstrate that the proposed IDK model cascade framework can preserve the accuracy of the expen-

sive models without a loss while reducing the overall computational cost.

### 5.1.1 Experimental Details

On the ImageNet 2012 dataset we study cascades assembled from pre-trained models including AlexNet [19] and residual networks of various depths including ResNet-18, ResNet-34, ResNet-50, and ResNet-152 [20]. Detailed statistics such as top-1 accuracy, FLOPs, etc of the models are shown in Fig. 2. To train the IDK classifiers, we sample 25.6K training images randomly from the ImageNet 2012 training data and report the cascade accuracy on the entire ImageNet 2012 validation data. In grid search for cascade by probability and entropy, we evaluate 100 different settings of  $\alpha$  and select the cascade which has the lowest IDK rate while reaching the desired accuracy. Because 1% reduction in accuracy can translate to a nearly 30% reduction in computational cost on ImageNet (as measured in flops), we set the desired accuracy to be the same as the ResNet-152 (i.e., setting  $\epsilon = 0$  in Eq. 11).

For cascade by entropy via cost-aware objective, we set the hyper-parameter  $\lambda = 0.04$  across different model combinations and use the actual FLOPs number of each

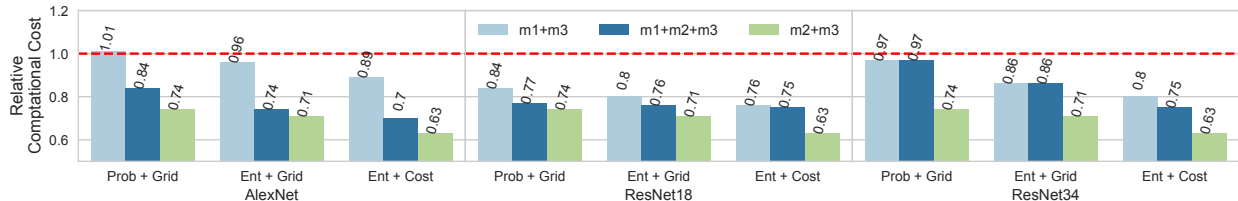


Figure 4: **Three Model Cascade Results.** We consider three element cascades  $m_1 \rightarrow m_2 \rightarrow m_3$  where  $m_2$  and  $m_3$  are chosen to be the optimal two element cascade consisting of ResNet-50 and ResNet-152 respectively and we evaluate AlexNet, ResNet-18, and ResNet-34 as  $m_1$ . We also evaluate each of the three IDK cascade designs and corresponding fitting procedure. In all cases, the accuracy is set to match that of ResNet-152 and we therefore only present the computational costs relative to the ResNet-152 model. In general we find that deeper cascades have diminishing returns due to increased evaluation costs

model as the model cost in the objective. We also compare against a cascade constructed using an oracle IDK classifier as a cascade accuracy upper bound which optimally selects between the fast and accurate models. In addition to proposed cascade designs, we include two more IDK cascades constructed by supervised training an IDK classifier of the form in Eq. 5 using the oracle labels with cost-oblivious objectives. We discuss these alternative baselines in more detail in the next section.

### 5.1.2 Computation Reduction

Detailed results are shown in Fig. 3. We find the best cascade design employs the Entropy features and regularized cost formulation to combine the ResNet-50 and ResNet-152 models. This cascade is able to reduce prediction costs by 37% while achieving the accuracy of the most computationally expensive model. This is also close to the oracle performance, though it assumes a perfect IDK classifier (i.e. the IDK classifier can distinguish the prediction correctness of the fast model with 100% accuracy and only passing the incorrect predictions to the accurate model). In general we find that our regularized cost based formulation outperforms the other baseline techniques.

### 5.1.3 Effectiveness of Cost-Aware Objective

In Fig. 3 we also compare the proposed cost based IDK cascade design with two IDK classifiers following the form of Eq. 5 with cost-oblivious cross entropy loss. The training labels are the correctness of the predictions of the fast model evaluated on the ImageNet2012 dataset. We consider two forms of the feature function  $f$ : entropy based features identical to the cost based cascade and neural network features. The neural network feature function  $f_{\alpha_2}(m^{\text{fast}}(x))$  consists of a 7-layer fully connected network with 1024, 1024, 512, 512, 128, and 64 hidden units, ReLU activation functions, and trained using stochastic gradient descent with momentum and batch

normalization. In general, we find that these sophisticated baselines are unable to accurately predict the success of the fast model and as a consequence are unable to match the accuracy of the cost based cascade formulation. With the cost-aware objective, the IDK cascades can meet the desired accuracy which shows that the proposed objective is more suitable for building model cascades.

### 5.1.4 Three Model Cascades

We also investigate three model cascades and the results are shown in Fig. 4. Compared to the two models ResNet-50 + ResNet-152 cascade, adding a faster model like AlexNet, ResNet-18 or ResNet-34 actually *increases* computational cost, because a reasonable fraction of examples will need to pass through all three models in the cascade. However, the three-model cascade tends to reduce the computational cost relative to a two-model cascade including a less accurate model than ResNet-50. Moreover, adding more accurate models within a cascade consistently improves the overall cascade performance.

Table 1: CIFAR Model Details

Model	% Train Acc	% Test Acc	Flops ( $10^7$ )
VGG19	99.996	93.66	39.8
ResNet18	100.00	95.26	3.7
DLA-48-B-s	99.204	89.06	1.7

## 5.2 Robustness Analysis on CIFAR-10

To further analyze the robustness of the IDK prediction cascades, we conduct a set of experiments on the CIFAR-10 datasets. We consider three models: ResNet-18 [20], VGG19 [21] and a recently proposed compact model DLA-48-B-s [22]. Tab. 1 shows details of the models. As we can see from the table, all models overfit the training data with high accuracy close to 100%. We want to study the robustness of the IDK prediction cascades under such

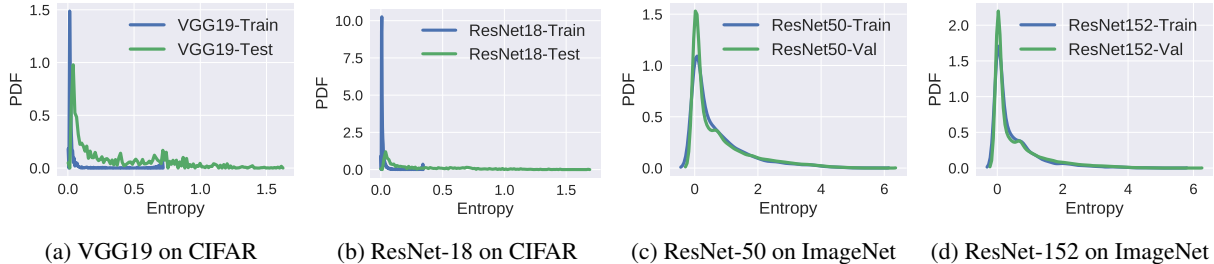


Figure 5: Entropy Distribution. We plot the entropy of the class conditional probability distributions for VGG19 and ResNet-18 on CIFAR-10 and ResNet-50 and ResNet-152 on ImageNet 2012. The VGG model severely overfits the training data and thus can not be used as the base model in the IDK classifier

Table 2: CIFAR Model DLA-48-B-s + ResNet-18 Cascade Results

Type	Desired Acc (%)	Acc (%)	IDK Rate (%)	Avg Flops ( $\times 10^7$ )	Relative Computational Cost
Prob + Grid	95.26	95.23	67.5	4.198	1.135
Entropy + Grid	95.26	95.23	51.7	3.613	0.977
Entropy + Cost	95.26	95.22	<b>48.9</b>	<b>3.509</b>	<b>0.949</b>
Entropy + Grid	95.16	95.16	40.1	3.184	0.861
Entropy + Cost	95.16	95.16	<b>40.0</b>	<b>3.179</b>	<b>0.859</b>
Prob + Grid	95.05	95.07	36.5	3.051	0.824
Entropy + Grid	95.05	95.06	33.7	2.947	0.796
Entropy + Cost	95.05	95.05	<b>32.8</b>	<b>2.915</b>	<b>0.788</b>
Prob + Grid	94.90	94.89	29.9	2.806	0.759
Entropy + Grid	94.90	94.91	30.6	2.832	0.766
Entropy + Cost	94.90	94.91	<b>30.5</b>	<b>2.827</b>	<b>0.764</b>

extreme case. In this experiment, since VGG19 is less accurate and more costly than ResNet-18, we focus on cascades constructed with DLA-48-B-s and ResNet-18.

We evaluate the model cascades with four different cascade accuracy goals shown in Tab. 2. We observe cascade by entropy via the cost-aware objective consistently outperforms grid search methods. Also, by admitting a small 0.03% reduction in accuracy, the IDK rate drops substantially from 48.9% to 30.5%. Compared to the single expensive model, the best model cascade reduces computational costs by 24%.

### 5.2.1 Robustness analysis

The proposed IDK classifiers rely on various measures of uncertainty in the class conditional probability distribution and are therefore sensitive to over confidence often as a result of over-fitting. To assess this effect, we evaluate the entropy distribution of the VGG19 and ResNet-18 models which have been trained to near perfect training accuracy (see Tab. 1). We plot the entropy distribution of these models in Figure 5a and 5b on both training and held-out test data and observe that both models substantially over-estimate their confidence on training data when compared with test data. In contrast, the ResNet-50 and

ResNet-152 models are much better estimators of prediction uncertainty as seen in Figures 5c and 5d. As a consequence, in settings where the fast model is likely to over-fit it is important to use separate held-out data when training the IDK classifier.

### 5.3 DRIVING CONTROL PREDICTION

We evaluate IDK cascades for autonomous driving and demonstrate that we can achieve nearly perfect accuracy with less than 30% human intervention. In this experiment, we apply the IDK model cascade framework on Berkeley DeepDrive Video dataset, a large scale real driving video dataset [18] containing 2.6 million frames in the training video and 384,599 frames for testing. The driving dataset contains 4 discrete motion states: left turn, right turn, forward and stop. The task is to predict the next vehicle motion given previous video frames. Fig. 6 shows some sample frames in the dataset. We use the *Long-term Recurrent Convolutional Networks* (LRCN) [23] model as  $m^{\text{fast}}$  and experiment on a large scale driving video dataset [18]. We consider a human-in-the-loop setting where human serves as the  $m^{\text{acc}}$  with 100% accuracy.

We use the same training setting for the proposed cascade approaches as the image classification task and report the results in Tab. 3. The LRCN model has an accuracy



Figure 6: Sample Frames from the Berkeley DeepDrive Video Dataset

of 84.5%<sup>1</sup> and we set the desired accuracy to 100%, 99% and 95%. We find that with only 28.88% human intervention, the cascade model can achieve 95% accuracy which is about 10% more accurate than the base LRCN model. This experiment demonstrates that the model cascade can be easily applied to real applications which are in high demand of low latency and high accuracy.

Table 3: Driving Model LRCN + Human Expert Cascades

Type	Desired Acc (%)	Acc (%)	IDK rate (%)
Prob + Grid	100.0	99.9	83.91
Ent + Grid	100.0	99.9	83.50
Ent + Cost	100.0	99.9	<b>80.70</b>
Prob + Grid	99.0	99.2	61.72
Ent + Grid	99.0	99.2	61.36
Ent + Cost	99.0	99.1	<b>59.70</b>
Prob + Grid	95.0	95.4	30.08
Ent + Grid	95.0	95.3	30.02
Ent + Cost	95.0	95.1	<b>28.88</b>

## 6 CONCLUSION AND FUTURE WORK

In this paper we revisited the classic idea of prediction cascades to reduce prediction costs. We extended the classic cascade framework focused on binary classifications to multi-class classification setting. We argue that the current deep learning models are “over-thinking” simple inputs in the majority of the real-world applications. Therefore, we aim to learn prediction cascades within the framework of empirical risk minimization and propose a new cost aware loss function, to leverage the accuracy and reduced cost of the IDK cascades.

We focused on build simple cascade with the the pre-trained base models with little training and negligible

<sup>1</sup>A slightly reduced accuracy early version was used.

computation overhead. We tried to answer two questions in this paper: (1) what is a good measure to distinguish the easy and hard examples in the workload without querying much information about the mode itself? We found that the entropy value of the model prediction distribution is a good measure than the vanilla confidence score and can be used as input data to train a light-weighted but effective IDK classifier. (2) How to design the objective function that balances the prediction accuracy and the computation cost? We proposed in this work to use the cost regularized objective which utilizes the actual FLOPs of the base models as the cost measures. Incorporating the cost factor in the objective, we found the model cascade works more effectively than the model cascades with cost-oblivious function.

We also proposed two search based methods *cascade by probability* and *cascade by entropy*, which obtain reasonable performance and require no additional training. We evaluated these techniques on both benchmarks and real-world datasets to show that our approach can successfully identify hard examples in the problem, and substantially reduce the number of invocations of the accurate model with negligible loss in accuracy. We also found that the cost based cascade formulation outperforms uncertainty based techniques.

We believe this work is a first step towards learning to compose models to reduce computational costs. Though not studied in this work, the proposed framework can be easily applied to the existing model serving systems and fit the edge-cloud scenario naturally with little modification. In the future, we would like to explore feature reusing and joint training of the cascade models so that different models can specialize in either easy or hard examples of the given workload.



## Acknowledgements

This research is supported in part by DHS Award HSHQDC-16-3-00083, NSF CISE Expeditions Award CCF-1139158, and gifts from Alibaba, Amazon Web Services, Ant Financial, CapitalOne, Ericsson, GE, Google, Huawei, Intel, IBM, Microsoft, Scotiabank, Splunk and VMware.

## References

- [1] Z. Yang, M. Moczulski, M. Denil, N. de Freitas, A. Smola, L. Song, and Z. Wang, “Deep fried convnets,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1476–1483, 2015.
- [2] E. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, “Exploiting linear structure within convolutional networks for efficient evaluation,” in *Proceedings of the 27th International Conference on Neural Information Processing Systems, NIPS’14*, (Cambridge, MA, USA), pp. 1269–1277, MIT Press, 2014.
- [3] M. Denil, B. Shakibi, L. Dinh, M. Ranzato, and N. D. Freitas, “Predicting parameters in deep learning,” in *Advances in Neural Information Processing Systems 26* (C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, eds.), pp. 2148–2156, 2013.
- [4] H. A. Rowley, S. Baluja, and T. Kanade, “Neural network-based face detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, pp. 23–38, Jan. 1998.
- [5] P. Viola and M. J. Jones, “Robust real-time face detection,” *Int. J. Comput. Vision*, vol. 57, pp. 137–154, May 2004.
- [6] A. Angelova, A. Krizhevsky, V. Vanhoucke, A. S. Ogale, and D. Ferguson, “Real-time pedestrian detection with deep network cascades,” in *BMVC*, pp. 32–1, 2015.
- [7] Z. Cai, M. Saberian, and N. Vasconcelos, “Learning complexity-aware cascades for deep pedestrian detection,” in *ICCV*, 2015.
- [8] D. Crankshaw, X. Wang, G. Zhou, M. J. Franklin, J. E. Gonzalez, and I. Stoica, “Clipper: A low-latency online prediction serving system,” in *NSDI*, pp. 613–627, 2017.
- [9] C. Olston, N. Fiedel, K. Gorovoy, J. Harmsen, L. Lao, F. Li, V. Rajashekhar, S. Ramesh, and J. Soyke, “Tensorflow-serving: Flexible, high-performance ml serving,” *arXiv preprint arXiv:1712.06139*, 2017.
- [10] S. Han, H. Mao, and W. J. Dally, “Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding,” *ICLR*, 2015.
- [11] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” *arXiv preprint arXiv:1503.02531*, 2015.
- [12] G. Huang, D. Chen, T. Li, F. Wu, L. van der Maaten, and K. Q. Weinberger, “Multi-scale dense convolutional networks for efficient prediction,” *arXiv preprint arXiv:1703.09844*, 2017.
- [13] T. P. Trappenberg and A. D. Back, “A classification scheme for applications with ambiguous data,” in *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, vol. 6, pp. 296–301 vol.6, 2000.
- [14] F. Khani, M. C. Rinard, and P. Liang, “Unanimous Prediction for 100% Precision with Application to Learning Semantic Mappings,” *ACL*, 2016.
- [15] D. Kang, J. Emmons, F. Abuzaid, P. Bailis, and M. Zaharia, “Optimizing deep cnn-based queries over video streams at scale,” *arXiv preprint arXiv:1703.02529*, 2017.
- [16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, pp. 1097–1105, 2012.
- [17] A. Krizhevsky, V. Nair, and G. Hinton, “The cifar-10 dataset,” 2014.
- [18] H. Xu, Y. Gao, F. Yu, and T. Darrell, “End-to-end learning of driving models from large-scale video datasets,” *arXiv preprint arXiv:1612.01079*, 2016.
- [19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, pp. 1097–1105, 2012.
- [20] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.
- [21] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

- [22] F. Yu, D. Wang, and T. Darrell, “Deep layer aggregation,” *arXiv preprint arXiv:1707.06484*, 2017.
- [23] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, “Long-term recurrent convolutional networks for visual recognition and description,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2625–2634, 2015.

---

# Learning Fast Optimizers for Contextual Stochastic Integer Programs

---

**Vinod Nair**  
DeepMind  
vinair@google.com

**Krishnamurthy Dvijotham**  
DeepMind  
dvij@google.com

**Iain Dunning**  
DeepMind  
idunning@google.com

**Oriol Vinyals**  
DeepMind  
vinyals@google.com

## Abstract

We present a novel reinforcement learning (RL) approach to learning a fast and highly scalable solver for a two-stage stochastic integer program in the large-scale data setting. Mixed integer programming solvers do not scale to large datasets for this problem class. Additionally, they solve each instance independently, without any knowledge transfer across instances. We address these limitations with a learnable local search solver that jointly learns two policies, one to generate an initial solution and another to iteratively improve it with local moves. The policies use contextual features for a problem instance as input, which enables learning across instances and generalization to new ones. We also propose learning a policy to compute a bound on the objective using dual decomposition. Benchmark results show that on test instances our approach rapidly achieves approximately 30% to 2000% better objective value, which a state of the art integer programming solver (SCIP) requires more than an order of magnitude more running time to match. Our approach also achieves better solution quality on seven out of eight benchmark problems than standard baselines such as Tabu Search and Progressive Hedging.

## 1 INTRODUCTION

Stochastic integer programming (Birge and Louveaux [1997], Shapiro et al. [2009]) arises in various applications that require combinatorial optimization under uncertainty, such as electric grid optimization, finance, and logistics. Many domains involve large-scale data for random variables (e.g., weather, demand, etc.), and require solving a set of related problem instances instead of only one instance (e.g., solve instances periodically, each with different weather and demand forecast distributions).

While powerful solvers exist for deterministic, single-instance mixed integer programs (MIPs), this is not the case for the stochastic, multi-instance setting. Traditional solvers from the optimization literature are not able to scale to high dimensional stochastic MIPs where several thousands of samples are required to represent uncertainty, and do not reuse experience on solving past optimization problems to solve future problems.

In this paper we consider learning an approximate solver with reinforcement learning (RL). We frame optimization as a sequential decision problem where at each step the solver “agent” proposes a solution, and the “environment” evaluates the objective value and any constraint violations to provide reward and observations. Contextual features describing a problem instance are used as an input to the solver so that its actions are adapted to that instance. The parameters of such a *contextual solver* are learned offline on a training set of instances. Once trained, the solver is applied to a new instance from the same distribution. We also learn a *dual policy* that computes a bound on the optimal value to estimate how far the solution computed by the contextual solver is from the global optimum for a given instance.

The advantages of our approach are: a) it can scale to a large set of problem instances, as well as a large set of samples for the random variables in each instance, and b) it can generalize to a new instance to achieve better solution quality and/or time than a solver that treats each instance independently. As the results show, our approach is able to scale to much larger datasets and achieve significant speedups compared to SCIP (Gleixner et al. [2017]), the state-of-the-art open source MIP solver.

**Two-stage stochastic integer programs:** We focus on an important class of stochastic programs called a two-stage stochastic MIP (Birge and Louveaux [1997]). In the first stage, a *planning decision* is optimized under uncertainty, then the values of all random variables are observed, and in the second stage a *recourse decision* is

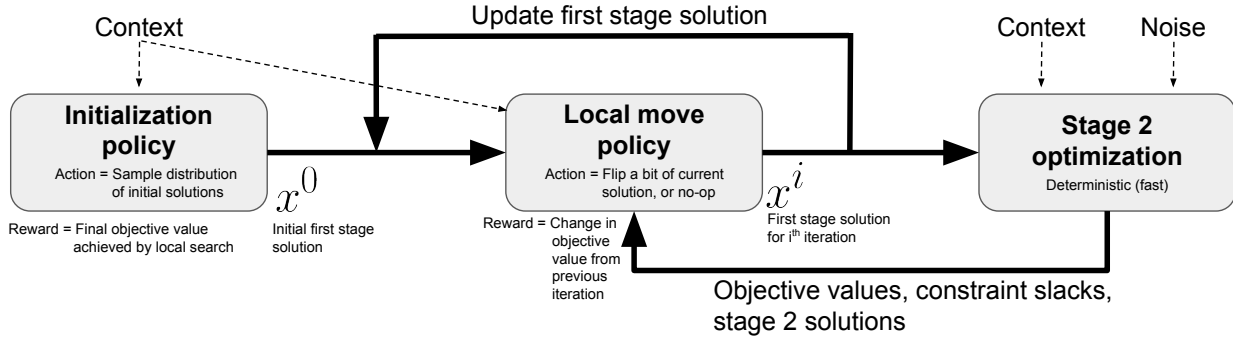


Figure 1: Local search for a two-stage stochastic MIP as an RL problem. The initialization and local move policies are part of the “agent”, while the second stage optimization is part of the “environment.” Given a problem instance specified by the context, the agent’s “action” is to propose a first stage solution at each step of local search. The environment evaluates it on a minibatch of noise samples (denoted by  $\omega$  in the text) by optimizing the second stage problem for each sample using a MIP solver to compute the reward and observations.

deterministically optimized to adapt the plan in response to observations. The uncertainty is exogenous, so it is independent of the first stage solution. This problem structure arises in many applications, such as electric grid day-ahead planning (Zheng et al. [2015]) and logistics (Laporte et al. [1992]).

**Local search solver:** The contextual solver is structured as a local search algorithm over the space of first stage solutions. It consists of two learnable components: a) an *initialization policy* that takes as input the context vector and outputs an initial solution, and b) a *local move policy* that takes the initial solution and applies a sequence of local moves such that the final solution has high objective value. Both of these policies are learned jointly.

Both during training and testing, evaluation of a first stage solution requires solving several second stage optimization problems to approximate the expected objective as a sample average. Since each such problem is a deterministic MIP, they are well-suited for solving with an off-the-shelf MIP solver, so we use SCIP to solve them. This provides another perspective of our work as a hybrid approach that combines a learned contextual solver with a MIP solver in the learning loop to efficiently optimize stochastic two-stage MIPs.

**Contributions:** We develop an approach to learn 1) a contextual local search solver with the initialization and local move policies that computes a solution to the optimization for a given problem instance, and 2) a contextual dual policy that computes an upper bound on the optimal value (achievable by *any algorithm*) for a given problem instance. We benchmark our approach on a set of two-stage stochastic MIPs and show that compared to SCIP it achieves approximately 30% to 2000% better objective value for the same running time and at least an order of magnitude faster for the same objective value. It also

outperforms baseline algorithms such as Tabu Search and Progressive Hedging on seven out of eight problems.

## 2 RELATED WORK

**Learning solvers:** Early examples of learning solvers for combinatorial optimization problems include Zhang and Dietterich [1995], Moll et al. [1999], and Boyan and Moore [2001]. The first two use TD( $\lambda$ ) to learn a value function over the solution space such that applying a local search algorithm with that value function can find a good solution and can generalize to new problem instances. These approaches assume that an initial solution is given (either randomly generated or by another algorithm), whereas our approach learns to generate the initial solution jointly with the local search policy. Boyan and Moore proposed STAGE, an iterative algorithm that alternates between learning a value function over the solution space for one instance and using that function to find a better solution for that instance. This work was primarily focused on the single instance setting.

More recently Vinyals et al. [2015] used supervised learning with pointer networks to approximately solve travelling salesman problems. Subsequent work improved on it by using the same pointer network architecture, but learned with RL (Bello et al. [2016]). Ignoring architectural differences, our initialization policy is similar to Bello et al.’s approach, but we further improve the initial solution using a learned local move policy. As our results show, this is a crucial difference for improving performance. Khalil et al. [2017a] combine graph neural networks with RL to learn a greedy solver for graph combinatorial optimization problems that incrementally extends a partial solution till its complete. Again, our idea of improving the complete solution further with a learned local search policy can be applied here as well.

Additionally, using the dual policy, we can compute a bound on the objective function value even on new problem instances. This allows us to a) assess the quality of the solution produced by the learned solver, and b) to make a fairer comparison to approaches that provide both a solution and a bound since proving the bound can be a significant part of the solver running time.

**Learning to improve solvers:** Several works have focused on applying learning to improve the decisions made by non-learning solvers, e.g., variable selection and node selection decisions in a branch and bound solver. See Lodi and Zarpellon [2017] for a survey. Khalil et al. [2016] treat variable selection as a ranking problem and learn a ranking function. Khalil et al. [2017b] use learning to predict on which nodes of the branch and bound search tree various primal heuristics will succeed. Such approaches can be used as building blocks for an end-to-end solution.

### 3 CONTEXTUAL TWO-STAGE STOCHASTIC INTEGER PROGRAM

A contextual two-stage stochastic integer program is defined as follows (for the case of maximization):

$$\max_x f(x; z) + E_{P(\omega)} \left[ \max_{y \in Y(x, \omega)} g(y; x, \omega, z) \right] \quad (1)$$

where

- $\omega \in \mathbb{R}^d$  is a random variable representing the uncertainty affecting the second stage objective function and constraints.  $\omega$  is independent of the decision variables. We are given a set of I.I.D. samples  $D_\omega = \{\omega_i \sim P(\omega)\}, i = 1, \dots, N_\omega$  with which the expectation in equation 1 is estimated as a sample average.
- $x \in \{0, 1\}^n$  is the first stage decision variable, optimized before observing  $\omega$ . We assume that every  $x$  is feasible.
- $y \in \{0, 1\}^m$  is the second stage decision variable, optimized after observing  $\omega$ , where  $Y(x, \omega) \subseteq \{0, 1\}^m$  is the feasible set defined by second stage constraints. We assume  $Y(x, \omega) \neq \emptyset$  for all  $(x, \omega)$  (i.e., *complete recourse*).
- $z \in \mathbb{R}^c$  is the context feature vector describing an instance of the optimization problem. A set of instances is generated by drawing I.I.D. samples  $D_z = \{z_j \sim P(z)\}, j = 1, \dots, N_z$ .

$f$  and  $g$  represent the first and second stage objective functions, respectively. Unlike LP relaxation based approaches to solving MIPs, our approach can be applied to

nonlinear  $f$ ,  $g$ , and constraints. However, in this work we focus on the linear case to allow a direct comparison to LP relaxation based approaches:

$$\begin{aligned} f(x; z) &= c_1^T(z)x \\ g(y; x, \omega, z) &= c_2^T(x, \omega, z)y \\ Y(x, \omega) &= \{y \in \{0, 1\}^m : B(\omega, z)x + C(\omega, z)y \leq d(\omega, z)\} \end{aligned}$$

**Sample Average Approximation:** The primary approach in the optimization literature for solving equation 1 is the *Sample Average Approximation* (SAA) method (Ahmed and Shapiro [2002]). It replaces the expectation with a sample average and replicates the second stage decision for each sample:

$$\max_x f(x; z) + \frac{1}{N_\omega} \sum_i^{N_\omega} \left[ \max_{y_i \in Y(x, \omega_i)} g(y_i; x, \omega_i, z) \right].$$

This converts the original problem into a deterministic one which can be solved using a deterministic MIP solver (assuming linear  $f$ ,  $g$ , and constraints). The key disadvantage is that the number of second stage decision variables increases by a factor of the number of samples  $N_\omega$ . As a result, the approach does not scale beyond a moderate number of variables and samples. In practice this limitation is overcome by domain experts carefully selecting a small set of samples that they judge will likely affect the solution. By being more scalable, our learning-based approach makes such expert selection unnecessary.

## 4 RL APPROACH

We describe our Reinforcement Learning (RL) approach for learning a solver for the optimization problem defined in section 3.

### 4.1 MAIN IDEA

We formulate our approach as *learning an iterative local search algorithm* over the space of first stage solutions (see figure 1). It has two learnable components which are optimized jointly:

- *Initialization policy:* Given an instance of the optimization problem, generate an initial first stage solution  $x^0$  from which local search starts.
- *Local move policy:* At each iteration of local search, select one of the neighbors of the current first stage solution as the proposal for the next iteration.

The solution  $x^K$  after a fixed number of iterations  $K$  is the solver's output. The objective function value  $r^K$  at

$x^K$  is approximated by:

$$r^K = f(x^K; z) + \frac{1}{N_\omega} \sum_{i=1}^{N_\omega} \left[ \max_{y \in Y(x^K, \omega_i)} g(y; x^K, \omega_i, z) \right] \quad (2)$$

where the expectation is approximated as a sample average. For a given  $x$  and  $\omega$ , the second stage optimization over  $y$  is deterministic. We assume that an efficient solver is available (e.g., SCIP, or a fast heuristic) so that an accurate approximation of the expectation can be computed quickly. This allows us to leverage existing efficient techniques for deterministic optimization to learn a solver for stochastic optimization.

**Limitations:** 1) Our approach relies on offline learning on a dataset of instances, and it is unlikely to be useful in a setting where there is only one instance (or few) to be solved. Our early experiments showed that in the single-instance setting (i.e., treat  $z$  as a constant), our approach’s running times tend to be much higher than baseline algorithms to achieve similar solution quality. 2) The solution found after  $K$  iterations is not guaranteed to be optimal. But if learning is successful, it can be expected to be a good solution, and the bound provided by the dual policy indicates how close to optimal it is.

## 4.2 INITIALIZATION POLICY

The initialization policy is a distribution  $P(x|z)$  over the first stage solution  $x$  given the context feature vector  $z$ . The initial solution is generated by sampling  $x^0 \sim P(x|z)$ . Reward is the expected objective function value achieved by the solution at the end of local search. We use policy gradients to learn  $P(x|z)$ .

**Architecture:** We explored two architectures: 1) a conditional autoregressive generative model, and 2) a simpler, fully connected feedforward neural network with a single hidden layer. Results are better for the former, so we describe it here. Autoregressive generative models provide a flexible neural network parameterization for high dimensional distributions while still allowing tractable exact evaluation of the log probability and its gradient with respect to model parameters. Crucially, they do not make any independence assumptions, thus capturing complicated correlations in the vector  $x$ . Here we consider a conditional version

$$P_{\theta_{init}}(x|z) = \prod_i P_{\theta_{init}}(x_i | x_{<i}, z) \quad (3)$$

where  $x_i$  is the  $i^{th}$  dimension of  $x$ ,  $x_{<i}$  is the set of first  $i - 1$  dimensions of  $x$ , and  $\theta_{init}$  denotes the model’s parameters. Specifically, we use the Neural Autoregressive Density Estimator (NADE) (Larochelle and Murray

[2011]). NADE is not specialized for a particular type of data (e.g., images, time series), which makes our approach application agnostic. It uses weight sharing across the conditional distributions in equation 3 to achieve a compact parameterization. For details see Larochelle and Murray [2011].

**Training:** The loss function for learning  $\theta_{init}$  is

$$\mathcal{L}(\theta_{init}) = -E_{P_{\theta_{init}}(x^0|z)P(r^K|x^0,z)} [r^K]. \quad (4)$$

$\theta_{init}$  is learned with stochastic gradient descent using the policy gradient method of REINFORCE (Williams [1992]). The gradient is given by:

$$\begin{aligned} \nabla_{\theta_{init}} \mathcal{L}(\theta_{init}) = \\ - E [(r^K - b_w(z)) \nabla_{\theta_{init}} (\log P_{\theta_{init}}(x|z))] \end{aligned}$$

where  $b_w(z)$  is a learned baseline function parameterized by  $w$ , and the expectation is with respect to  $P_{\theta_{init}}(x^0|z)P(r^K|x^0,z)$ . The baseline  $b_w(z)$  is a single hidden layer, fully connected feedforward neural network with ReLU hidden units. Its weights  $w$  are learned jointly with the initialization policy by stochastic gradient descent to minimize  $(r^K - b_w(z))^2$ .

## 4.3 LOCAL MOVE POLICY

We learn a policy that makes  $K$  local moves starting from the initial solution  $x^0$  to produce  $x^K$ . At an intermediate step  $k$ , the local move policy takes the first stage solution  $x^k$  and selects one of its neighbors from a Hamming ball of radius 1 as  $x^{k+1}$ . This results in  $n + 1$  possible actions per step: either toggle one of the dimensions of  $x^k$ , or a no-op. The policy outputs a categorical distribution  $\pi_{\theta_{lm}}(a^k|x^k, s^k, x^0, z)$  over the actions, given state  $s^k$  (described below), initial solution  $x^0$ , and context vector  $z$ , parameterized by  $\theta_{lm}$ . The reward at each iteration is the change in objective value from the previous iteration. So the policy is being learned to maximize the total expected increase in objective value relative to the initial solution in an episode. We use the Asynchronous Advantage Actor Critic (A3C) algorithm (Mnih et al. [2016]) for learning.

**Architecture:**  $\pi_{\theta_{lm}}(a^k|x^k, s^k, x^0, z)$  is a two hidden layer, fully connected feedforward neural network with ReLU hidden units. The observations  $o^k$  at step  $k$  consists of:

- *Constraint slacks:* The difference between a second stage constraint’s value and its upper/lower bounds obtained for each sample  $\omega_i$ .
- *Second stage solutions:* The solutions  $y^i$  obtained for each sample  $\omega_i$ .

We define state  $s^k$  to be a fixed size time window of  $J$  observations  $\{o^{k-1}, \dots, o^{k-J}\}$ . The window size is a hyperparameter tuned based on validation set performance. A3C also learns a value function  $V_{\theta_v}(s_k; x_0, z)$  jointly with the policy. Following standard practice, this function shares with the policy all the hidden layers and corresponding parameters, with unshared parameters only in the output layers to compute the value and policy outputs.

**Training:** The reward at step  $k$  of an episode is given by

$$r_{lm}^k = r^{k+1} - r^k,$$

where  $r^k$  is defined by equation 2. A3C defines its loss in terms of the *advantage function*:

$$A(a^k, s^k; x^0, z) = \sum_{l=0}^{L-1} \gamma^l r_{lm}^{k+l} + \gamma^L V_{\theta_v}(s_{k+L}; x_0, z) - V_{\theta_v}(s_k; x_0, z),$$

where the first two terms on the RHS give an estimate of the  $L$ -step return, and  $\gamma$  is the discount factor. The loss is given by

$$\mathcal{L}(\theta_{lm}, \theta_v) = -E_{\pi_{\theta_{lm}}(a^k|x^k, s^k, x^0, z)} [A(a^k, s^k; x^0, z)], \quad (5)$$

with the gradient with respect to  $\theta_{lm}$  given by:

$$\nabla_{\theta_{lm}} \mathcal{L}(\theta_{lm}, \theta_v) = -E [A(a^k, s^k; x^0, z) \nabla_{\theta_{lm}} (\log \pi_{\theta_{lm}}(a^k|x^k, s^k, x^0, z))].$$

The square of the advantage function is an additional loss used to learn the value function parameters  $\theta_v$ . We also apply entropy regularization to encourage exploration by preventing the policy from becoming deterministic. The relative magnitude of these losses' contribution to the total loss are tuned as hyperparameters based on validation set performance.

We use distributed TensorFlow with  $N_w$  parallel workers executing episodes and computing gradients, and one central learner asynchronously applying gradients to update parameters. This setup is replicated for each of the  $N_h$  randomly sampled hyperparameter settings for hyperparameter tuning. We use  $N_w = 20$  and  $N_h = 25$  for each problem in our benchmark.

## 5 BOUNDS VIA LEARNED DUAL POLICIES

An attractive feature of MIP solvers is that they provide an upper bound on the optimal value of a maximization problem, quantifying the objective value gap between a

solution and the global optimum. However, as explained in section 3, directly applying a MIP solver to a two-stage stochastic integer program with a large set of samples is computationally infeasible. We use *dual decomposition* (Carøe and Schultz [1999]) to develop an alternative approach.

The main idea is that if we are allowed to choose a different  $x = x_\omega$  for each sample  $\omega$ , the optimization problem (1) decomposes into independent subproblems for each sample that can be solved in parallel. This constitutes a relaxation of the optimization problem, and hence its optimal value is an upper bound on the original problem. Solving the relaxation only requires a deterministic MIP problem, which can be solved efficiently or bounded using a further LP relaxation.

This bound can be tightened by adding Lagrangian multipliers that encourage the scenario subproblems to agree on a single  $x = x_\omega$ . Since we are interested in large-scale stochastic MIPs and in out-of-sample tests, it is not appropriate to compute an independent dual variable for each scenario  $\omega$ . Instead, we train a neural network that, given features that depend on  $\omega$  (the sample) and  $z$  (the context), predict the dual variables for the problem. Given any such policy, we can obtain a bound on the value of the 2-stage stochastic MIP and further, we can evaluate this bound out of sample (on samples that were unavailable during training) to estimate the ‘‘generalization’’ of the bound.

### 5.1 DUAL DECOMPOSITION OF 2-STAGE STOCHASTIC MIPs

Consider equation 1 in the linear case:

$$\max_{x \in \{0,1\}^n} c_1(z)^T x + \frac{1}{N_\omega} \sum_{\omega \in D_\omega} \left[ \max_{y \in Y(x, \omega, z)} c_2(\omega, z)^T y \right]. \quad (6)$$

We now introduce independent copies  $x_\omega$  for each sample  $\omega$  with constraints to enforce equality of  $x_\omega$ :

$$\begin{aligned} \max_{x, \{x_\omega\}} \frac{1}{N_\omega} \sum_{\omega \in D_\omega} \left[ \max_{y \in Y(x_\omega, \omega, z)} c_1(z)^T x_\omega + c_2(\omega, z)^T y \right], \\ \text{s.t. } x_\omega = x \quad \forall \omega \in \Omega. \end{aligned}$$

We drop the constraints and add a Lagrangian term to obtain a relaxation:

$$\max_{x, \{x_\omega\}} \frac{1}{N_\omega} \sum_{\omega \in D_\omega} \left[ \max_{y \in Y(x_\omega, \omega, z)} c_1^T x_\omega + c_2^T y + \lambda_\omega^T (x - x_\omega) \right], \quad (7)$$

where  $\lambda_\omega$  for each  $\omega$  is a dual variables, and we dropped the dependence of  $c_1, c_2$  on  $\omega, z$  for brevity.

From equation 7 we can derive the following *dual problem* to optimize the upper bound (see supplementary material for details):

$$\min_{\{\lambda_\omega\}} \frac{1}{N_\omega} \sum_{\omega \in D_\omega} h(\omega, z, \lambda_\omega) + \mathbf{1}^T \max \left( \frac{1}{N_\omega} \sum_{\omega \in D_\omega} \lambda_\omega, 0 \right), \quad (8)$$

where

$$h(\omega, z, \lambda_\omega) = \max_{y \in Y(x_\omega, \omega, z)} c_1(z)^T x_\omega + c_2(\omega, z)^T y - \lambda_\omega^T x_\omega.$$

We use (8) to estimate a bound.

## 5.2 LEARNING DUAL POLICIES

There are three issues with applying the above dual decomposition approach to large scale contextual stochastic MIPs:

- *Context ignored:* If we directly apply dual decomposition, even after solving the problem for thousands of context vectors  $z$ , we would start from scratch for a new context. This is wasteful particularly if the solutions for the dual variables are simple functions of  $z$  that can be learned.
- *Generalization:* The problem (8) only implies a bound for the set of scenarios in the training set  $D_\omega$ . However, this does not say anything about how the bound generalizes to unseen samples.
- *Computation:* The problem (8) is expensive to solve if the number of scenarios  $D_\omega$  is large, since one has to solve an optimization problem (to compute  $h$ ) for each scenario  $\omega$  simply to evaluate the objective and compute gradients with respect to  $\lambda_\omega$ .

We address all three issues by learning a *dual policy*  $\lambda_\omega = \lambda_{\theta_d}(\omega, z)$  that maps from the context  $z$  and sample features  $\omega$  to the dual variables  $\lambda_\omega$ , parameterized by  $\theta_d$ . We train the policy to minimize the dual objective averaged over samples  $D_\omega$  and context sample set  $D_z$ :

$$\min_{\theta} \frac{1}{N_z} \sum_{z \in D_z} \frac{1}{N_\omega} \sum_{\omega \in D_\omega} h(\omega, z, \lambda_{\theta_d}(\omega, z)) \quad (9a)$$

$$+ \mathbf{1}^T \max \left( \frac{1}{N_\omega} \sum_{\omega \in D_\omega} \lambda_{\theta_d}(\omega, z), 0 \right). \quad (9b)$$

A major bottleneck in solving this problem is the need to compute  $h$  (which itself involves solving a deterministic

MIP or an LP relaxation of it). Thus, it is desirable to have an algorithm that does not compute  $h$  for each  $z, \omega$  at each iteration.

In our experiments we sample  $z^s, \omega^s$  but also get predictions of the neural network for each  $\omega \in D_\omega$  to compute  $\sigma = \sum_{\omega \in D_\omega} \lambda_\omega(\omega, z^s; \theta)$ . Then, an unbiased estimate of a (sub)-gradient of the objective is given by

$$g^s = \frac{\partial h}{\partial \lambda_\omega}(\omega^s, z^s) \frac{\partial \lambda_\omega(\omega^s, z^s; \theta)}{\partial \theta} + \text{sign}(\sigma) \frac{\partial \lambda_\omega(\omega^s, z^s; \theta)}{\partial \theta} \quad (10)$$

This only requires one backprop through the network and  $N_\omega$  forward passes and computation of  $h$  for a single  $z, \omega$ .

**Architecture:** We use a simple feedforward architecture with a single hidden layer in all our experiments. The architecture has fully connected layers with ReLU activations. We add a skip connection from the context vector  $z$  to the hidden and output layer and a skip connection from  $\omega$  to the output layer. We tune the size of the hidden layer and learning rate as hyperparameters (picked so that the dual bound on a validation set is minimized).

## 6 EVALUATION

### 6.1 BENCHMARK

There is a dearth of standardized, large-scale benchmarks for stochastic programming problems. As a first step, we construct a benchmark using stochastic versions of two standard optimization problems, knapsack and facility location, with large-scale data. We plan to add more problems in the future.

**Knapsack (KS):** The problem is to place a set of items with various sizes and values into a knapsack of specified capacity so as to maximize the total knapsack value without exceeding capacity. For a given problem instance the item sizes are stochastic. In the first stage, the items have to be selected without knowing their precise sizes. After the sizes are revealed, the second stage decision is to remove items selected in the first stage to satisfy the capacity limit, but removal incurs a penalty. Context specifies the values of the items, knapsack capacity, and removal penalty.

**Facility Location (FL):** Adapted from Ntaimo and Sen [2005], the problem is to place facilities at a set of locations with varying costs such that profit is maximized. Customer demand and revenue at the locations are stochastic. There is a linear penalty for not meeting customer demand. In the first stage, locations have to be selected without knowing precise demand and revenue. After demand and revenue are revealed, the second stage



decision is to assign customers to various facilities. Context specifies the location costs, facility capacity, and the linear penalty factor on unsatisfied demand.

**Problem size:** We define “problem size” to be the number of first stage binary decision variables. We use the following sizes:  $\{25, 50, 100, 200\}$ . The number of second stage binary decision variables for KS is the same as in the first stage, while for FL, it is the problem size multiplied by the number of customers (set to 5 in our experiments). In the SAA method the deterministic reformulation multiplies the number of second stage variables by the number of samples. So for problem size 200, even with only 1000 samples, the number of variables already exceeds  $2 \times 10^5$ .

**Data generation:** We generate samples for the context vectors and for the random variables within an instance using a Mixture of Factor Analyzers (Ghahramani and Hinton [1997]). The resulting distributions are structured with dependencies among variables, and the mixture components make them more complex than a Gaussian.

## 6.2 BASELINE SOLVERS

**SCIP** is an open source MIP solver that uses LP relaxation-based branch and bound algorithm. We apply SCIP using the SAA method.

**Tabu Search** (Glover [1986]) is a local search algorithm that makes moves in the space of the first stage decision variable and maintains a tabu list of recently visited solutions to avoid cycles. See supplementary material for more details.

**Progressive Hedging** ([Watson and Woodruff, 2011]) is based on the dual decomposition approach described in section 5.1. It iteratively fixes the relaxation introduced in the dual decomposition by adding penalty terms to promote consistency between  $x_\omega$  and  $x$ . See supplementary material for more details.

SCIP, Tabu Search, and Progressive Hedging do not support any transfer across problem instances as they are originally formulated. We implement a simple way to adapt Tabu Search and Progressive Hedging to the contextual setting. At training time, these optimizers solve as many training instances as possible within the given training budget. At test time, a test instance’s context vector is used for nearest neighbor selection of a training instance. The solution for the test instance is initialized to the final solution of the nearest training instance by L1 distance.

## 6.3 EVALUATION PROTOCOL

Solvers that support contextual optimization are given the same training budget of 500 CPU cores for 12 hours.

GPUs are not used to allow fair comparison to methods that cannot use them. The training budget is used for hyperparameter tuning and parameter learning. For Tabu Search and Progressive Hedging, instead of parameter learning, a set of training instances are solved for nearest neighbor initialization at test time.

We use 1000 contexts and  $10^5$  samples for the random variables in a problem for training, and 100 contexts and 1000 random variable samples each for validation and testing. The context distribution is independent of the distribution over random variables in a problem, so we can sample each separately. The number of variable samples is limited to 1000 only because SCIP becomes too slow and uses excessive memory for some problem instances with more samples. RL approaches can easily scale to much larger number of samples. The solver performance on the validation set is used to select the hyperparameters.

At test time we provide one CPU core per instance for all solvers. Tabu Search and Progressive Hedging both make the same number of second stage solver calls (which dominates the running time) as the local search solver so that all three have (approximately) the same computational budget. SCIP’s computation cannot be easily characterized by the number of second stage solver calls. Instead we set a maximum time limit and a 5% optimality gap, and allow SCIP to run until whichever condition is satisfied first.

# 7 EXPERIMENTS

We present the following results: a) a comparison of the local search solver to SCIP in terms of test solution objective value vs. solver running time, b) a comparison to the objective value achieved by the baselines, and c) results for estimating a bound using the dual policy.

## 7.1 OBJECTIVE VALUE VS. RUNNING TIME

We run SCIP on 100 test problem instances with different running time limits, from 1s to  $10^4$ s. We then compare its objective value to that achieved by the local search solver on the same instances. The running time of the local search solver is fixed to be the time needed to execute one episode per instance and evaluate the solution at the end of the episode.

Figure 2 summarizes the results. Each plot shows the percent difference in SCIP’s objective value with respect to that of the local search solver as a function of the running time ratio of SCIP to the local search solver. The key observation is that SCIP gives significantly worse objective value than the local search solver unless it is given much more running time. For knapsack problems

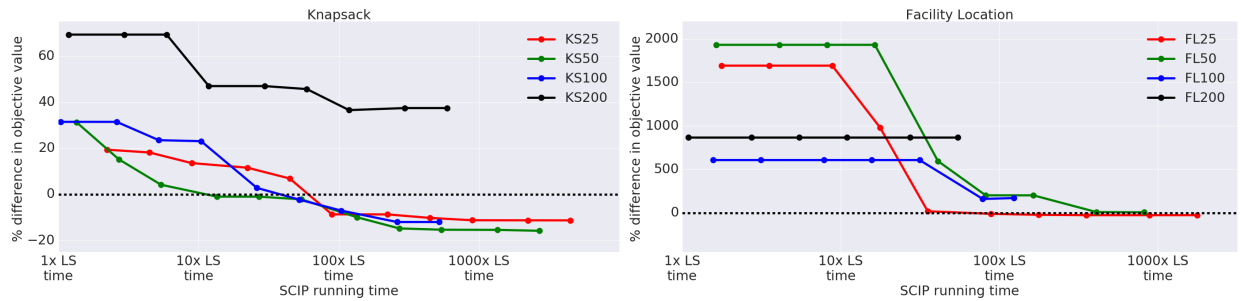


Figure 2: SCIP running time vs. objective value comparison between local search solver and SCIP on test problem instances for the benchmark problems (KS = Knapsack, FL = Facility Location) of sizes  $\{25, 50, 100, 200\}$ . x-axis is SCIP’s average running time as a factor of the local search (LS) solver’s average running time. y-axis is the percent improvement in average objective value given by local search over SCIP. Positive values mean local search is better.

of different sizes, SCIP performs 30 to 70% worse when given approximately the same running time. For problem sizes 25, 50, and 100, SCIP needs more than an order of magnitude more time to catch up with the local search solver’s solution quality. For problem size 200, running SCIP even 1000 times longer still results in more than 35% worse objective value. A similar result is seen in the facility location problem as well, where the objective value gaps are even bigger in favor of the local search solver. For problem sizes 100 and 200, it appears that SCIP needs more than  $10^4$  seconds to improve the objective value significantly, and therefore is not able to catch up with the local search solver’s objective value in that time.

These results show the speedup benefit of learning a contextual solver. Since the solver has already seen at training time problem instances from the same distribution, generalization allows it to quickly find good solutions on unseen problem instances at test time.

## 7.2 OBJECTIVE VALUE COMPARISON

We compare the objective value given by the local search solver to that of Tabu Search, Progressive Hedging, and SCIP. We set the running time of SCIP to be ten times that of the local search solver to make it a stronger baseline. We also evaluate the initialization policy and the local move policy independently to get more insight into the performance of the local search solver. For the former we train only the initialization policy to generate a solution directly, without local search. For the latter we train only a local move policy that is initialized by selecting a solution uniformly randomly.

Table 1 summarizes the results. For each (solver, problem, size) triplet the mean of the objective values over a set of 100 test instances is shown. In all but one case the learned local search solver computes the best solution among all the algorithms tried, and in the remaining case it is within

5% of the best.

Both Tabu Search and Progressive Hedging perform poorly on the larger problem sizes (100, 200), especially for Facility Location. As we have already seen in the previous section, SCIP requires significantly more time to achieve comparable objective values.

A comparison among the different variants of the learned solver shows that the best results are given by learning both the initialization and local move policies jointly. Learning a local move policy to start from a random solution performs the worst among the variants, especially for larger problem sizes (KS200, FL100 and FL200). Learning an initialization policy alone performs better, but uniformly worse than learning both jointly.

The dual policy bounds computed by the approach from section 5.2 also show that on most problems, our learning based approach not only learns to produce good solutions but is also able to prove that these solutions are within 20% of the optimum (5 of 8 problem instances).

## 7.3 INSIGHTS INTO SOLVER BEHAVIOR

Figure 3(a) shows examples of how the objective value varies as a function of solver steps (here for KS100, other problems show similar behavior). The objective value typically improves rapidly for roughly the first one-third of an episode, with smaller gains afterwards. Unlike a greedy algorithm, the solver is not constrained to always improve the objective value. Figure 3(b) shows an example. The highlighted part of an episode shows that the objective value can decrease by a large amount ( $> 10\%$ ) for a significant duration ( $> 10\%$ ) of the episode before it improves again. Approximately 7% of test instances across problem types and sizes show non-greedy behavior. The ability to learn a non-greedy policy explains how the local search solver can outperform greedy methods like Tabu Search.

Solver	KS 25	KS 50	KS 100	KS 200	FL 25	FL 50	FL 100	FL 200
SCIP @10x more runtime	0.5985	0.1681	2.731	1.1273	0.0	0.0	-7.0531	-13.8540
Tabu Search	1.3150	<b>1.4859</b>	-3.0144	-0.0851	0.9781	0.3089	0.0	0.0
Progressive Hedging	1.4666	1.2584	2.9565	1.1983	0.9301	0.0003	0.0014	0.0
RL-based solvers								
Init. policy only	1.2410	1.2977	0.1304	0.9275	0.8213	0.8148	1.1395	1.3337
Local move policy only	1.1427	1.0631	1.2481	0.1846	1.1058	0.7403	-7.3248	-36.8738
Init. policy + local move policy	<b>1.4807</b>	1.4286	<b>3.2920</b>	<b>1.3133</b>	<b>1.1352</b>	<b>1.2416</b>	<b>1.2729</b>	<b>1.8197</b>
Bounds from dual policy								
Dual policy	1.5885	2.4486	3.4289	3.8304	1.2753	1.2903	1.6489	2.4702

Table 1: Comparison of average test set objective values achieved by various solvers across benchmark problems and sizes. KS = Knapsack, FL = Facility Location. The last row shows the upper bounds computed by the dual policy.

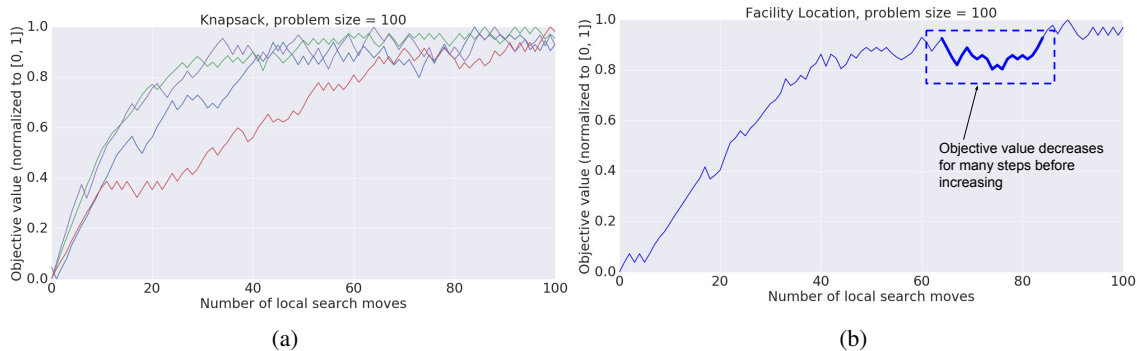


Figure 3: Examples of local search behavior on test instances. (a) Objective value as a function of the number of local search moves for 100-d Knapsack for several test instances. (Other problems show similar behavior.) Objective values are normalized to  $[0,1]$  for display purposes. (b) Example of non-greedy behavior (highlighted) shown by the local solver. Approximately 7% of test instances show this type of behavior.

Non-greediness can also potentially make the solution worse than the initial solution. However, only 0.21% of test instances show a decrease  $> 1\%$  in objective value relative to the initial solution, while 89.8% show an increase  $> 10\%$ . It appears that on some instances the initial solution is already very good, and local search introduces small changes which reduces the objective value slightly.

## 8 CONCLUSIONS

We presented a learned contextual local search solver that jointly learns both an initialization policy and a local move policy. On benchmark problems of two-stage stochastic knapsack and facility location of different sizes, it achieves approximately 30% to 2000% better objective value for the same running time and at least an order of

magnitude faster for the same objective value. It also outperforms baseline algorithms such as Tabu Search and Progressive Hedging on seven out of eight problems. Joint learning is key as learning only one of the policies in isolation results in worse performance. The dual policy is able to compute bounds showing that the local search solver's objective value is within 20% of the optimum for 5 out of 8 problem instances.

Next directions include incorporating a more powerful search procedure such as MCTS into the agent as a policy improvement operator (Silver et al. [2017]).

## References

S. Ahmed and A. Shapiro. The sample average approximation method for stochastic programs with integer recourse. *SIAM Journal of Optimization*, 12:479–502,

- 2002.
- I. Bello, H. Pham, Q. V. Le, M. Norouzi, and S. Bengio. Neural combinatorial optimization with reinforcement learning. *CoRR*, abs/1611.09940, 2016.
- J. R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer-Verlag, 1997.
- J. Boyan and A. W. Moore. Learning evaluation functions to improve optimization by local search. *JMLR*, 1: 77–112, Sept. 2001.
- C. C. Carøe and R. Schultz. Dual decomposition in stochastic integer programming. *Operations Research Letters*, 24(1-2):37–45, 1999.
- Z. Ghahramani and G. E. Hinton. The em algorithm for mixtures of factor analyzers. Technical report, University of Toronto, 1997.
- A. Gleixner, L. Eifler, T. Gally, G. Gamrath, P. Gember, R. L. Gottwald, G. Hendel, C. Hojny, T. Koch, M. Miltenberger, B. Müller, M. E. Pfetsch, C. Puchert, D. Rehfeldt, F. Schlösser, F. Serrano, Y. Shinano, J. M. Viernickel, S. Vigerske, D. Weninger, J. T. Witt, and J. Witzig. The scip optimization suite 5.0. Technical report, 2017.
- F. Glover. Future paths for integer programming and links to artificial intelligence. *Comput. Oper. Res.*, 13(5): 533–549, 1986.
- E. Khalil, H. Dai, Y. Zhang, B. Dilkina, and L. Song. Learning combinatorial optimization algorithms over graphs. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6348–6358. 2017a.
- E. B. Khalil, P. L. Bodic, L. Song, G. Nemhauser, and B. Dilkina. Learning to branch in mixed integer programming. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 724–731, 2016.
- E. B. Khalil, B. Dilkina, G. L. Nemhauser, S. Ahmed, and Y. Shao. Learning to run heuristics in tree search. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI’17*, pages 659–666, 2017b.
- G. Laporte, F. Louveaux, and H. Mercure. The vehicle routing problem with stochastic travel times. *Transportation Science*, 26(3):161–170, 1992.
- H. Larochelle and I. Murray. The neural autoregressive distribution estimator. In *The Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, volume 15 of *JMLR: W&CP*, pages 29–37, 2011.
- A. Lodi and G. Zarpellon. On learning and branching: a survey. *TOP*, 25(2):207–236, 2017.
- V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1928–1937, 2016.
- R. Moll, A. G. Barto, T. J. Perkins, and R. S. Sutton. Learning instance-independent value functions to enhance local search. In M. J. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems 11*, pages 1017–1023. MIT Press, 1999.
- L. Ntaimo and S. Sen. The million-variable march for stochastic combinatorial optimization. 32:385–400, 07 2005.
- A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lectures on Stochastic Programming*. Society for Industrial and Applied Mathematics, 2009.
- D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis. Mastering the game of go without human knowledge. *Nature*, 550, 2017.
- O. Vinyals, M. Fortunato, and N. Jaitly. Pointer networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2692–2700. 2015.
- J.-P. Watson and D. L. Woodruff. Progressive hedging innovations for a class of stochastic mixed-integer resource allocation problems. *Computational Management Science*, 8(4):355–370, Nov 2011.
- R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3-4):229–256, 1992.
- W. Zhang and T. G. Dietterich. A reinforcement learning approach to job-shop scheduling. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2, IJCAI’95*, pages 1114–1120, 1995.
- Q. P. Zheng, J. Wang, and A. L. Liu. Stochastic optimization for unit commitment - a review. *IEEE Transactions on Power Systems*, 30(4):1913–1924, 2015.

---

# Differential Analysis of Directed Networks

---

**Min Ren**

Department of Statistics  
Purdue University  
West Lafayette, IN  
ren80@purdue.edu

**Dabao Zhang**

Department of Statistics  
Purdue University  
West Lafayette, IN  
zhangdb@purdue.edu

## Abstract

We developed a novel statistical method to identify structural differences between networks characterized by structural equation models. We propose to reparameterize the model to separate the differential structures from common structures, and then design an algorithm with calibration and construction stages to identify these differential structures. The calibration stage serves to obtain consistent prediction by building the  $\ell_2$  regularized regression of each endogenous variables against pre-screened exogenous variables, correcting for potential endogeneity issue. The construction stage consistently selects and estimates both common and differential effects by undertaking  $\ell_1$  regularized regression of each endogenous variable against the predicts of other endogenous variables as well as its anchoring exogenous variables. Our method allows easy parallel computation at each stage. Theoretical results are obtained to establish non-asymptotic error bounds of predictions and estimates at both stages, as well as the consistency of identified common and differential effects. Our studies on synthetic data demonstrated that our proposed method performed much better than independently constructing the networks. A real data set is analyzed to illustrate the applicability of our method.

slightly from each other [West et al., 2012], and identifying the subtle difference between them helps design specific drugs. Social networks evolve over times, and monitoring their abrupt changes may serve as surveillance to economic stability or disease epidemics [Pianese et al., 2013, Berkman and Syme, 1979]. However, addressing such practical problems demands differential analysis of large networks, calling for development of efficient statistical method to infer and compare complex structures from high dimensional data. In this paper, we focus on differential analysis of directed acyclic or even cyclic networks which can be described by structural equation models (SEMs).

Many efforts have been made towards construction of a single network via SEM. For example, both Xiong et al. [2004] and Liu et al. [2008] employed a genetic algorithm to search for the best SEM using different information criteria. Most recently, Ni et al. [2017, 2018] employed a hierarchical Bayes approach to construct the SEM based networks. However, these approaches were designed for small or medium scale networks. For large-scale networks that the number of endogenous variables  $p$  exceeds the sample size  $n$ , Cai et al. [2013] proposed a regularization approach to fit a sparse model. Because this method suffers from incapability of parallel computation, it may not be feasible for large networks. Logsdon and Mezey [2010] proposed another penalization approach to fit the model in a node-wise fashion which alleviates the computational burden. Most recently, Lin et al. [2015], Zhu [2018], and Chen et al. [2017] each proposed a two-stage approach to construct SEMs, with different algorithms designed at different stages. As shown by Chen et al. [2017], such a two-stage approach can have superior performance compared to other methods.

To the best of our knowledge, no algorithm has been proposed to conduct differential analysis of directed networks characterized by SEM. While a naive approach would separately construct each individual network and identify common and differential structures, this ap-

## 1 INTRODUCTION

It is of great importance and interest to detect sparse structural differences or differential structures between two cognate networks. For instance, the gene regulatory networks of diseased and healthy individuals may differ

proach fails to take advantage of the commonality as well as sparse differential structures of the paired networks, leading to higher false positive rate or lower power. In this light, we introduce a novel statistical method, specially in the directed network regime, to conduct differential analysis of two networks via appropriate reparameterization of the corresponding models. There are two major features of our method. Firstly, we jointly model the commonality and difference between two networks explicitly. This helps us to gain dramatic performance improvements over the naive construction method. Secondly, benefiting from the flexible framework of SEMs, we are able to conduct differential analysis of directed networks. Most importantly, our method allow for both acyclic and cyclic networks. Compared to the other methods, directionality and allowing for cyclicity are crucial for many network studies, especially in constructing gene regulatory networks. As far as we know, our method is the first work on differential analysis of directed networks that enjoys the two promising features.

The rest of this paper is organized as follows. We first introduce the model and its identifiability condition in Section 2.1 and Section 2.2, respectively. Then, we present our proposed method of **Reparameterization-based Differential analysis of directed Networks**, termed as **RedNet**, in Section 2.3. The theoretical justification of the proposed method is described in Section 2.4. Section 3 includes our studies on synthetic data showing the superior performance of our method, as well as an analysis of the Genotype-Tissue Expression (GTEx) data sets. We conclude our paper with brief discussion in Section 4.

## 2 METHODS

Here we first introduce the model and its identification condition, and then describe our proposed **RedNet** method for identifying common and differential structures between two directed networks, followed with its theoretical justification.

### 2.1 THE MODEL

We consider two networks, each describing the dependencies among a common set of variables or nodes in a unique population. For each node  $i \in \{1, 2, \dots, p\}$  in network  $k \in \{1, 2\}$ , its regulation structure can be represented by the following equation,

$$\underbrace{\mathbf{Y}_i^{(k)}}_{\text{node } i} = \underbrace{\mathbf{Y}_{-i}^{(k)} \boldsymbol{\gamma}_i^{(k)}}_{\text{regulation by others}} + \underbrace{\mathbf{X}^{(k)} \boldsymbol{\phi}_i^{(k)}}_{\text{anchoring regulation}} + \underbrace{\boldsymbol{\epsilon}_i^{(k)}}_{\text{error}}, \quad (1)$$

where  $\mathbf{Y}_i^{(k)}$  is the  $i$ -th column of  $\mathbf{Y}^{(k)}$  and  $\mathbf{Y}_{-i}^{(k)}$  is the submatrix of  $\mathbf{Y}^{(k)}$  by excluding  $\mathbf{Y}_i^{(k)}$ , with  $\mathbf{Y}^{(k)}$  a

$n^{(k)} \times p$  matrix.  $\mathbf{X}^{(k)}$  is a  $n^{(k)} \times q$  matrix with each column standardized to have  $\ell_2$  norm  $\sqrt{n^{(k)}}$ . The vectors  $\boldsymbol{\gamma}_i^{(k)}$  and  $\boldsymbol{\phi}_i^{(k)}$  encode the inter-nodes and anchoring regulatory effects, respectively. The index set of non-zeros of  $\boldsymbol{\phi}_i^{(k)}$  is known and denoted by  $\mathcal{A}_i^{(k)}$ , in other words,  $\mathcal{A}_i^{(k)} = \text{supp}(\boldsymbol{\phi}_i^{(k)})$ . The support set  $\mathcal{A}_i^{(k)}$  indexes the direct causal effects for the  $i$ -th node, and can be pre-specified based on the domain knowledge. However, the size of nonzero effect  $\boldsymbol{\phi}_i^{(k)}$  is unknown and can be estimated. Further property of  $\mathcal{A}_i^{(k)}$  will be discussed in Section 2.2. All elements of the error term are independently distributed following a normal distribution with mean zero and standard deviation  $\sigma_i^{(k)}$ . We assume that the matrix  $\mathbf{X}^{(k)}$  and the error term  $\boldsymbol{\epsilon}_i^{(k)}$  are independent of each other. However  $\mathbf{Y}_{-i}^{(k)}$  and  $\boldsymbol{\epsilon}_i^{(k)}$  may correlate with each other.  $\mathbf{Y}^{(k)}$  and  $\mathbf{X}^{(k)}$  include observed endogenous variables and exogenous variables, respectively.

By combining the  $p$  linear equations in (1), we can rewrite the two sets of linear equations in a systematic fashion as two structural equation models below,

$$\begin{cases} \mathbf{Y}^{(1)} = \mathbf{Y}^{(1)} \boldsymbol{\Gamma}^{(1)} + \mathbf{X}^{(1)} \boldsymbol{\Phi}^{(1)} + \boldsymbol{\mathcal{E}}^{(1)}, \\ \mathbf{Y}^{(2)} = \mathbf{Y}^{(2)} \boldsymbol{\Gamma}^{(2)} + \mathbf{X}^{(2)} \boldsymbol{\Phi}^{(2)} + \boldsymbol{\mathcal{E}}^{(2)}, \end{cases} \quad (2)$$

where each matrix  $\boldsymbol{\Gamma}^{(k)}$  is  $p \times p$  with zero diagonal elements and represents the inter-nodes regulatory effects in the corresponding network. Specifically, excluding the  $i$ -th element (which is zero) from the  $i$ -th column of  $\boldsymbol{\Gamma}^{(k)}$  leads to  $\boldsymbol{\gamma}_i^{(k)}$ . The  $q \times p$  matrix  $\boldsymbol{\Phi}^{(k)}$  contains the anchoring regulatory effects and its  $i$ -th column is  $\boldsymbol{\phi}_i^{(k)}$ . Each error term  $\boldsymbol{\mathcal{E}}^{(k)}$  is  $n^{(k)} \times p$  and has the error term  $\boldsymbol{\epsilon}_i^{(k)}$  as its  $i$ -th column.

Figure 1 gives an illustrative example of networks with three nodes and one anchoring regulation per node for the structural equations in (2). For example, with anchoring regulation on node  $Y_1$ ,  $X_1$  has a direct effect on node  $Y_1$  but indirect effects on node  $Y_2$  and  $Y_3$  via  $Y_1$ .

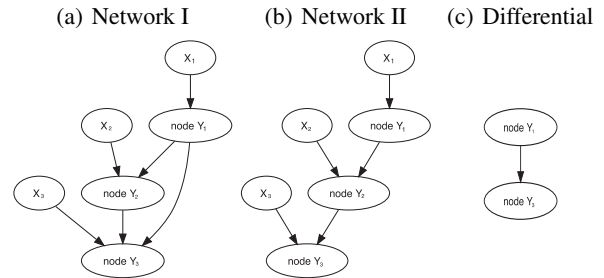


Figure 1: An Illustrative Example of Differential Network Between Two Directed Networks. The error term for each node is not shown for simplicity.

For each network  $k$ , its full model in (2) can be further transformed into the reduced form as follows,

$$\mathbf{Y}^{(k)} = \mathbf{X}^{(k)}\boldsymbol{\pi}^{(k)} + \boldsymbol{\xi}^{(k)}, \quad (3)$$

where the  $q \times p$  matrix  $\boldsymbol{\pi}^{(k)} = \boldsymbol{\Phi}^{(k)}(\mathbf{I} - \boldsymbol{\Gamma}^{(k)})^{-1}$  and the transformed error term  $\boldsymbol{\xi}^{(k)} = \boldsymbol{\mathcal{E}}^{(k)}(\mathbf{I} - \boldsymbol{\Gamma}^{(k)})^{-1}$ . The reduced model (3) reveals variables observed in  $\mathbf{X}^{(k)}$  as instrumental variables which will be used later to correct for the endogeneity issue. Otherwise, directly applying any regularization based regression to equation (1) will result in non-consistent or suboptimal estimation of model parameters [Fan and Liao, 2014, Chen et al., 2017, Lin et al., 2015, Zhu, 2018].

## 2.2 THE MODEL IDENTIFIABILITY

Here we introduce an identifiability assumption which helps to infer an identifiable system (2) from available data. We assume that each endogenous variable is directly regulated by a unique set of exogenous variables as long as it regulates other endogenous variables. That is, any regulatory node needs at least one anchoring exogenous variable to distinguish the corresponding regulatory effects from association. Explicitly let  $\mathcal{M}_{i0}^{(k)}$  denote the index set of endogenous variables which either directly or indirectly regulate the  $i$ -th endogenous variable in the  $k$ -th network. Thus,  $\mathcal{A}_i^{(k)} \subseteq \mathcal{M}_{i0}^{(k)}$ . The model identification condition can be stated in the below.

**Assumption 1.** For any  $i = 1, \dots, p$ ,  $\mathcal{A}_i^{(k)} \neq \emptyset$  if there exists  $j$  such that  $i \in \mathcal{M}_{j0}^{(k)}$ . Furthermore,  $\mathcal{A}_i^{(k)} \cap \mathcal{A}_j^{(k)} = \emptyset$  as long as  $i \neq j$ .

This assumption is slightly less restrictive than the one employed by Chen et al. [2017], and is a sufficient condition for model identifiability as it satisfies the rank condition in Schmidt [1976]. It can be further relaxed to allow nonempty  $\mathcal{A}_i^{(k)} \cap \mathcal{A}_j^{(k)}$  as long as each regulatory node has its own unique anchoring exogenous variables.

The above identifiability assumption not only identifies  $\gamma_i^{(k)}$  in model (1) from  $\boldsymbol{\pi}^{(k)}$  in model (3) but also helps reveal regulatory directionality of the networks. As illustrated in Figure 2, we can not recover the directionality between nodes  $Y_1$  and  $Y_2$  without the extra information provided by the direct causal factors  $X_1$  and  $X_2$  because all four sub-networks consisting of  $Y_1$  and  $Y_2$  (without  $X_1$  and  $X_2$ ) will be Markov equivalent. The known set  $\mathcal{A}_j^{(k)}$  serves as external prior knowledge which helps recover the directionality. In our two-stage construction of the differential network, the additional anchors  $X_1$  and  $X_2$  serve as instrumental variables in the calibration stage, since both  $X_1$  and  $X_2$  are independent of the error terms. The present direct causal effects from  $\mathbf{X}^{(k)}$

together with Assumption 1 differentiates our approach from the classical graphical models [Meinshausen and Bühlmann, 2006, Yuan and Lin, 2007] or the PC algorithm approaches [Spirtes et al., 2000, Kalisch and Bühlmann, 2007], since those methods either cannot recover edge directions or do not allow for cyclic structures due to lack of additional direct causal effects from  $\mathbf{X}^{(k)}$ .

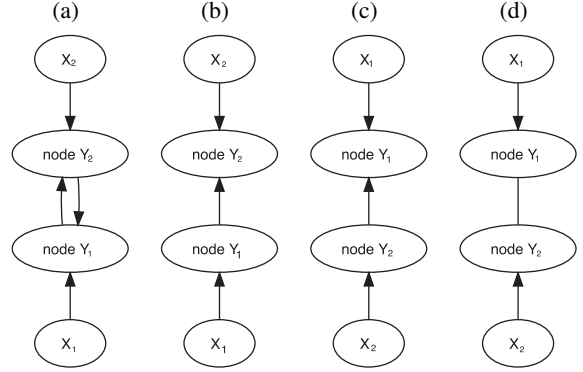


Figure 2: An Illustrative Example of Networks Which Are Not Markov Equivalent. However, without  $X_1$  and  $X_2$ , sub-networks consisting of only node  $Y_1$  and  $Y_2$  will be Markov equivalent.

## 2.3 TWO-STAGE DIFFERENTIAL ANALYSIS OF NETWORKS

Here we intend to develop a regularized version of the two-stage least squares. We first screen for exogenous variables and conduct  $\ell_2$  regularized regression of each endogenous variable against screened exogenous variables to obtain its good prediction which helps address the endogeneity issue in the following stage. At the second stage, we reparametrize the model to explicitly model the common and differential regulatory effects and identify them via the adaptive lasso method.

### 2.3.1 The Calibration Stage

To address the endogeneity issue, we aim for good prediction of each endogenous variable following the reduced model in (3). However, in the high-dimensional setting, the dimension  $q$  of  $\mathbf{X}^{(k)}$  can be much larger than the sample size  $n^{(k)}$ , and any direct prediction with all exogenous variables may not produce consistent prediction. Note that both Lin et al. [2015] and Zhu [2018] proposed to conduct variable selection with lasso or its variants and predict with selected exogenous variables. We here instead propose to first screen for exogenous variables with ISIS [Fan and Lv, 2008], and then apply ridge regression to predict the endogenous variables with screened exogenous variables. While variable screening is more robust

and provides higher coverage of true variables than variable selection, its combination with ridge regression puts less computational burden. Furthermore, as shown by Chen et al. [2017], ridge regression performs well in predicting the endogenous variables.

Let  $\mathcal{M}_i^{(k)}$  denotes the selected index set for  $i$ -th node in  $k$ -th network from the variable screening which reduces the dimension from  $q$  to  $d = |\mathcal{M}_i^{(k)}|$ . The *Sure Independence Screening Property* in Fan and Lv [2008] can be directly applied in our case to guarantee that  $\mathcal{M}_i^{(k)}$  covers the true set  $\mathcal{M}_{i0}^{(k)}$  with a large probability.

**Assumption 2.**  $n^{(1)}$  and  $n^{(2)}$  are at the same order, i.e.,  $n_{\min} = \min(n^{(1)}, n^{(2)}) \asymp n^{(1)} \asymp n^{(2)}$ , and  $p \asymp q$ .

**Theorem 1.** *Assuming Conditions 1-4 in the supplemental materials which restrict positive  $\tilde{\tau}$  and  $\tilde{\kappa}$ , under Assumption 2, there exists some  $\theta \in (0, 1 - 2\tilde{\kappa} - \tilde{\tau})$  such that, when  $d = |\mathcal{M}_i^{(k)}| = O((n_{\min})^{1-\theta})$ , we have, for some constant  $C > 0$ ,*

$$\mathbb{P}(\mathcal{M}_{i0}^{(k)} \subseteq \mathcal{M}_i^{(k)}) = 1 - \mathcal{O}\left(\exp\left\{-\frac{C(n^{(k)})^{1-2\tilde{\kappa}}}{\log(n^{(k)})}\right\}\right).$$

Hereafter we assume that  $\mathcal{M}_i^{(k)}$  successfully covers the true set  $\mathcal{M}_{i0}^{(k)}$  for convenience of stating the following assumptions and theorems. That is, the probability of successful screening is not incorporated into our assumptions or theorems in the below.

For node  $i$  in network  $k$ , let  $\mathbf{X}_{\mathcal{M}_i^{(k)}}^{(k)}$  denotes the submatrix of  $\mathbf{X}^{(k)}$  with prescreened columns which are indexed by  $\mathcal{M}_i^{(k)}$ . With  $\boldsymbol{\pi}_i^{(k)}$  denoting the  $i$ -th column of  $\boldsymbol{\pi}^{(k)}$ , the subvector of  $\boldsymbol{\pi}_i^{(k)}$  indexed by  $\mathcal{M}_i^{(k)}$  will be simply denoted by  $\boldsymbol{\pi}_{\mathcal{M}_i^{(k)}}^{(k)}$  without confusion. Such simplified notations will apply to other vectors and matrices in the rest of this paper.

With  $d$  pre-screened exogenous variables, we can apply ridge regression to the model

$$\mathbf{Y}_i^{(k)} = \mathbf{X}_{\mathcal{M}_i^{(k)}}^{(k)} \boldsymbol{\pi}_{\mathcal{M}_i^{(k)}}^{(k)} + \boldsymbol{\xi}_i^{(k)}, \quad (4)$$

to obtain the estimates  $\hat{\boldsymbol{\pi}}_{\mathcal{M}_i^{(k)}}^{(k)}$  of  $\boldsymbol{\pi}_{\mathcal{M}_i^{(k)}}^{(k)}$ , and predict  $\mathbf{Y}_i^{(k)}$  with  $\hat{\mathbf{Y}}_i^{(k)} = \mathbf{X}_{\mathcal{M}_i^{(k)}}^{(k)} \hat{\boldsymbol{\pi}}_{\mathcal{M}_i^{(k)}}^{(k)}$ .

### 2.3.2 The Construction Stage

With known  $\mathcal{A}_i^{(k)}$ , we can rewrite model (1) as,

$$\mathbf{Y}_i^{(k)} = \mathbf{Y}_{-i}^{(k)} \boldsymbol{\gamma}_i^{(k)} + \mathbf{X}_{\mathcal{A}_i^{(k)}}^{(k)} \boldsymbol{\phi}_{\mathcal{A}_i^{(k)}}^{(k)} + \boldsymbol{\epsilon}_i^{(k)}. \quad (5)$$

Before we use the predicted  $\mathbf{Y}^{(k)}$  to identify both common and differential regulatory effects across the two networks, we first reparametrize the model so as to define differential regulatory effects explicitly,

$$\boldsymbol{\beta}_i^- = \frac{\boldsymbol{\gamma}_i^{(1)} - \boldsymbol{\gamma}_i^{(2)}}{2}, \quad \boldsymbol{\beta}_i^+ = \frac{\boldsymbol{\gamma}_i^{(1)} + \boldsymbol{\gamma}_i^{(2)}}{2}. \quad (6)$$

Here  $\boldsymbol{\beta}_i^-$  represents the **differential regulatory effects** between the two networks. We need compare  $\boldsymbol{\beta}_i^+$  with  $\boldsymbol{\beta}_i^-$  to identify the **common regulatory effects**, that is, effects of all regulations with nonzero values in  $\boldsymbol{\beta}_i^+$  but zero values in  $\boldsymbol{\beta}_i^-$ .

Note that other differential analysis of networks may suggest a different reparametrization to identify common and differential regulatory effects. For example, in a typical case-control study, we may expect few structures in the case network mutated from the control network. While we are interested in identifying differential structures in the case network, we may be also interested in identifying baseline network structures in the control network. Therefore we may reparametrize the model with the regulatory effects in the control network, as well as the differential regulatory effects defined as the difference of regulatory effects between case and control networks. We want to point out that the method described here still applies and we can also derive similar theoretical results as follows.

Following the reparametrization in (6), we can rewrite model (5) as follows,

$$\begin{pmatrix} \mathbf{Y}_i^{(1)} \\ \mathbf{Y}_i^{(2)} \end{pmatrix} = \begin{pmatrix} \mathbf{Y}_{-i}^{(1)} & \mathbf{Y}_{-i}^{(1)} \\ \mathbf{Y}_{-i}^{(2)} & -\mathbf{Y}_{-i}^{(2)} \end{pmatrix} \begin{pmatrix} \boldsymbol{\beta}_i^+ \\ \boldsymbol{\beta}_i^- \end{pmatrix} + \begin{pmatrix} \mathbf{X}_{\mathcal{A}_i^{(1)}}^{(1)} & 0 \\ 0 & \mathbf{X}_{\mathcal{A}_i^{(2)}}^{(2)} \end{pmatrix} \begin{pmatrix} \boldsymbol{\phi}_{\mathcal{A}_i^{(1)}}^{(1)} \\ \boldsymbol{\phi}_{\mathcal{A}_i^{(2)}}^{(2)} \end{pmatrix} + \begin{pmatrix} \boldsymbol{\epsilon}_i^{(1)} \\ \boldsymbol{\epsilon}_i^{(2)} \end{pmatrix}. \quad (7)$$

Denote

$$\mathbf{Y}_i = \begin{pmatrix} \mathbf{Y}_i^{(1)} \\ \mathbf{Y}_i^{(2)} \end{pmatrix}, \quad \mathbf{Z}_{-i} = \begin{pmatrix} \mathbf{Y}_{-i}^{(1)} & \mathbf{Y}_{-i}^{(1)} \\ \mathbf{Y}_{-i}^{(2)} & -\mathbf{Y}_{-i}^{(2)} \end{pmatrix},$$

$$\boldsymbol{\beta}_i = \begin{pmatrix} \boldsymbol{\beta}_i^+ \\ \boldsymbol{\beta}_i^- \end{pmatrix}, \quad \boldsymbol{\epsilon}_i = \begin{pmatrix} \boldsymbol{\epsilon}_i^{(1)} \\ \boldsymbol{\epsilon}_i^{(2)} \end{pmatrix}.$$

Further define the projection matrix for each network,

$$\mathbf{H}_i^{(k)} = I_{n^{(k)}} - \mathbf{X}_{\mathcal{A}_i^{(k)}}^{(k)} \left( \mathbf{X}_{\mathcal{A}_i^{(k)}}^{(k)T} \mathbf{X}_{\mathcal{A}_i^{(k)}}^{(k)} \right)^{-1} \mathbf{X}_{\mathcal{A}_i^{(k)}}^{(k)T}.$$

Applying the projection matrix  $\mathbf{H}_i = \text{diag}\{\mathbf{H}_i^{(1)}, \mathbf{H}_i^{(2)}\}$  to both sides of model (7), we can remove the exogenous variables from the model and obtain,

$$\mathbf{H}_i \mathbf{Y}_i = \mathbf{H}_i \mathbf{Z}_{-i} \boldsymbol{\beta}_i + \mathbf{H}_i \boldsymbol{\epsilon}_i. \quad (8)$$



---

**Algorithm 1** Reparameterization-Based Differential Analysis of Network (ReDNet)

---

**Input:** For  $k \in \{1, 2\}$ ,  $\mathbf{Y}^{(k)}$ ,  $\mathbf{X}^{(k)}$ , index set  $\mathcal{A}_i^{(k)}$  for each  $i \in \{1, 2, \dots, p\}$ . Set  $d = O(n_{\min}^{1-\theta})$ .

**for**  $i \rightarrow 1$  **to**  $p$  **do**

Stage 1.a. Screen for a submatrix  $\mathbf{X}_{\mathcal{M}_i^{(k)}}^{(k)}$  of  $\mathbf{X}^{(k)}$  for  $\mathbf{Y}_i^{(k)}$  versus  $\mathbf{X}^{(k)}$  and set  $\mathbf{X}_{\mathcal{M}_i^{(k)}}^{(k)} = \mathbf{X}^{(k)}$  if  $q \leq n^{(k)}$ .

Stage 1.b. Apply ridge regression to regress  $\mathbf{Y}_i^{(k)}$  against  $\mathbf{X}_{\mathcal{M}_i^{(k)}}^{(k)}$  to obtain prediction  $\hat{\mathbf{Y}}_i^{(k)}$ .

**end for**

**for**  $i \rightarrow 1$  **to**  $p$  **do**

Stage 2. Apply adaptive lasso to regress  $\mathbf{H}_i \mathbf{Y}_i$  against  $\mathbf{H}_i \hat{\mathbf{Z}}_{-i}$  to obtain coefficients estimate  $\hat{\beta}_i$ .

**end for**

**Output:** The common and differential regulatory effects in  $\hat{\beta}_1, \dots, \hat{\beta}_p$ .

---

To address the endogeneity issue, we predict  $\mathbf{Z}_{-i}$  by replacing its component  $\mathbf{Y}_{-i}^{(k)}$  with the predicted value  $\hat{\mathbf{Y}}_{-i}^{(k)}$  from the previous stage, and then regressing  $\mathbf{H}_i \mathbf{Y}_i$  against  $\mathbf{H}_i \hat{\mathbf{Z}}_{-i}$  with the adaptive lasso to consistently estimate  $\beta_i$ . That is, an optimal  $\beta_i$  can be obtained as,

$$\hat{\beta}_i = \arg \min_{\beta_i} \left\{ \frac{1}{n} \|\mathbf{H}_i \mathbf{Y}_i - \mathbf{H}_i \hat{\mathbf{Z}}_{-i} \beta_i\|_2^2 + \lambda_i \omega_i^T |\beta_i|_1 \right\},$$

where  $|\beta_i|_1$  is a vector taking elementwise absolute values of  $\beta_i$ ,  $\omega_i$  is the adaptive weights whose components are inversely proportional to the components of an initial estimator of  $\beta_i$ , and  $\lambda_i$  is the adaptive tuning parameter.

The two-stages algorithm is summarized in Algorithm 1. With the estimator  $\hat{\beta}_i$  from the second stage, we can accordingly obtain estimators  $\hat{\gamma}_i^{(1)} = \hat{\beta}_i^+ + \hat{\beta}_i^-$  and  $\hat{\gamma}_i^{(2)} = \hat{\beta}_i^+ - \hat{\beta}_i^-$ .

## 2.4 THEORETICAL ANALYSIS

As shown in Theorem 1, a screening method like ISIS [Fan and Lv, 2008] can identify  $\mathcal{M}_i^{(k)}$  with size  $d = O(n_{\min}^{1-\theta})$  which covers the true set  $\mathcal{M}_{i0}^{(k)}$  with a sufficiently large probability. For the sake of simplicity and without loss of generality, in the following we assume  $\mathcal{M}_{i0}^{(k)} \subseteq \mathcal{M}_i^{(k)}$ .

We first investigate the consistency of predictions from the first stage. The consistency properties will be characterized by prespecified sequences  $f^{(k)} = o(n^{(k)})$  but  $f^{(k)} \rightarrow \infty$  as  $n^{(k)} \rightarrow \infty$ . We also denote  $f_{\max} = f^{(1)} \vee f^{(2)}$ , i.e.,  $\max\{f^{(1)}, f^{(2)}\}$ .

The following assumption is required for the consistency properties.

**Assumption 3.** For each network  $k$ , the singular values of  $\mathbf{I} - \mathbf{\Gamma}^{(k)}$  are positively bounded from below, and there exist some positive constants  $c_1^{(k)}$  and  $c_2^{(k)}$  such that, for each node  $i$ ,  $\max_{\|\delta\|_2=1} (n^{(k)})^{-1/2} \|\mathbf{X}_{\mathcal{M}_i^{(k)}}^{(k)} \delta\|_2 \leq c_1^{(k)}$  and  $\min_{\|\delta\|_2=1} (n^{(k)})^{-1/2} \|\mathbf{X}_{\mathcal{M}_i^{(k)}}^{(k)} \delta\|_2 \geq c_2^{(k)}$ . Furthermore, the ridge parameter  $\lambda_i^{(k)} = o(n_{\min})$ .

For the ease of exposition, we will omit the subscript  $\mathcal{M}_i^{(k)}$  from  $\mathbf{X}_{\mathcal{M}_i^{(k)}}^{(k)}$  henceforth, and accordingly use  $\pi_i^{(k)}$  and  $\hat{\pi}_i^{(k)}$  which include the zero components of excluded predictors.

Denote  $\mathbf{X} = \text{diag}\{\mathbf{X}^{(1)}, \mathbf{X}^{(2)}\}$ , and

$$\mathbf{Z} = \begin{pmatrix} \mathbf{Y}^{(1)} & \mathbf{Y}^{(1)} \\ \mathbf{Y}^{(2)} & -\mathbf{Y}^{(2)} \end{pmatrix}, \quad \mathbf{\Pi} = \begin{pmatrix} \pi^{(1)} & \pi^{(1)} \\ \pi^{(2)} & -\pi^{(2)} \end{pmatrix}.$$

We use  $\mathbf{\Pi}_j$  to denote the  $j$ -th column of the matrix  $\mathbf{\Pi}$  and  $\pi_j^{(k)}$  to denote the  $j$ -th column of the matrix  $\pi^{(k)}$ . We also use  $\hat{\mathbf{Z}}$  and  $\hat{\mathbf{\Pi}}$  to denote the prediction of  $\mathbf{Z}$  and estimate of  $\mathbf{\Pi}$ , respectively. Note that, with the ridge parameter  $\lambda_i^{(k)}$  for the ridge regression taken on node  $i$  in network  $k$ , we have  $r_i^{(k)} = (\lambda_i^{(k)})^2 \|\pi_i^{(k)}\|_2^2 / n^{(k)}$  and hence define  $r_{\max} = \max_{1 \leq i \leq p} [r_i^{(1)} \vee r_i^{(2)}]$ . Then the estimation and prediction losses at the first stage can be summarized in the following theorem.

**Theorem 2.** Under Assumptions 1-3, for each  $j \in \{1, 2, \dots, 2p\}$ , there will exist some constant  $C_1$  and  $C_2$  such that, with probability at least  $1 - e^{-f^{(1)}} - e^{-f^{(2)}}$ ,

1.  $\|\hat{\mathbf{\Pi}}_j - \mathbf{\Pi}_j\|_2^2 \leq C_1 (d \vee r_{\max} \vee f_{\max}) / n_{\min}$ ;
2.  $\|\mathbf{X}(\hat{\mathbf{\Pi}}_j - \mathbf{\Pi}_j)\|_2^2 \leq C_2 (d \vee r_{\max} \vee f_{\max})$ .

The proof is detailed in the supplemental materials.

Note that these two sets of losses can be controlled by the same upper bounds across the two networks with probability at least  $1 - e^{-f^{(1)} + \log(p)} - e^{-f^{(2)} + \log(p)}$ . Therefore,  $f^{(k)}$  can be selected such that  $f^{(k)} - \log(p) \rightarrow \infty$ , which will provide a probability approaching one to have the network-wide losses approaching zero.

Furthermore, the dimension  $p$  can be divergent up to an exponential order, say  $p = e^{n_{\min}^c}$  for some  $c \in (0, 1)$ . We can set  $f^{(1)} = f^{(2)} = n_{\min}^{(1+c)/2}$  and, apparently,  $f^{(k)} = o(n_{\min})$  but  $f^{(k)} - \log(p) = n_{\min}^{(1+c)/2} - n_{\min}^c \rightarrow \infty$ .

Since the ridge parameter  $\lambda_i^{(k)} = o(n_{\min})$ ,  $r_i^{(k)} = \|\pi_i^{(k)}\|_2^2 \times o(n_{\min})$ . Therefore, when all  $\|\pi_i^{(k)}\|_2$  are uniformly bounded, we have  $r_{\max} = o(n_{\min})$ . Otherwise, the ridge parameter  $\lambda_i^{(k)}$  should be adjusted accordingly

to control both estimation and prediction losses.

Before we characterize the consistency of estimated regulatory effects on the second stage, we first introduce the following concept of restricted eigenvalue which is used to present an assumption.

**Definition 2.1.** *The restricted eigenvalue of a matrix  $\mathbf{A}$  on an index set  $\mathcal{S}$  is defined as*

$$\phi_{re}(\mathbf{A}, \mathcal{S}) = \min_{\|\delta_{\mathcal{S}^c}\|_1 \leq 3\|\delta_{\mathcal{S}}\|_1} \frac{\|\mathbf{A}\delta\|_2}{\sqrt{n}\|\delta_{\mathcal{S}}\|_2}. \quad (9)$$

For the  $i$ -th node, we use  $\mathcal{S}_i$  to denote the non-zero indices of  $\beta_i$ , i.e.,  $\mathcal{S}_i = \text{supp}(\beta_i)$ . Further denote

$$\mathbf{\Pi}_{-i} = \begin{pmatrix} \pi_{-i}^{(1)} & \pi_{-i}^{(1)} \\ \pi_{-i}^{(2)} & -\pi_{-i}^{(2)} \end{pmatrix}.$$

As in Bickel et al. [2009], we impose the following restricted eigenvalue condition on the design matrix in (8).

**Assumption 4.** There exists a constant  $\phi_0 > 0$  such that  $\phi_{re}(\mathbf{H}_i \mathbf{X} \mathbf{\Pi}_{-i}, \mathcal{S}_i) \geq \phi_0$ .

Let  $n = n^{(1)} + n^{(2)}$ ,  $c_{\max} = c_1^{(1)} \vee c_1^{(2)}$ , and  $\mathbf{B} = [\beta_1, \beta_2, \dots, \beta_p]$ . The matrix norms  $\|\cdot\|_1$  and  $\|\cdot\|_{\infty}$  are the maximum of column and row sums of absolute values of the matrix, respectively. For a vector, we define  $\|\cdot\|_{\infty}$  and  $\|\cdot\|_{-\infty}$  to be the maximum and minimum absolute values of its components. Then, we can derive the following loss bounds for the estimation and prediction at the second stage on the basis of Theorem 2.

**Theorem 3.** *Suppose that, for node  $i$ , the adaptive lasso at the second stage takes the tuning parameter*

$$\lambda_i \asymp \|\omega_i\|_{-\infty}^{-1} \|\mathbf{B}\|_1 \|\mathbf{\Pi}\|_1 \sqrt{(d \vee r_{\max} \vee f_{\max}) \log(p) / n_{\min}},$$

and  $\sqrt{(d \vee r_{\max} \vee f_{\max}) / n} + c_{\max} \|\mathbf{\Pi}\|_1 \leq \sqrt{c_{\max}^2 \|\mathbf{\Pi}\|_1^2 + \phi_0^2 / (64C_2 |\mathcal{S}_i|)}$ . Let  $h_n = (\|\mathbf{B}\|_1^2 \wedge 1) \times ((n \|\mathbf{\Pi}\|_1^2 / d) \wedge (d \vee r_{\max} \vee f_{\max})) \log(p)$ . Under Assumptions 1-4, there exist positive constants  $C_3$  and  $C_4$  such that, with probability at least  $1 - 3e^{-C_3 h_n + \log(4pq)} - e^{-f^{(1)} + \log(p)} - e^{-f^{(2)} + \log(p)}$ ,

1. *Estimation Loss:*

$$\|\hat{\beta}_i - \beta_i\|_1 \leq 8C_4 |\mathcal{S}_i| \times \frac{\|\omega_{\mathcal{S}_i}\|_{\infty}^2 \|\mathbf{B}\|_1 \|\mathbf{\Pi}\|_1}{\phi_0^2 \|\omega_i\|_{-\infty}^2} \sqrt{\frac{(d \vee r_{\max} \vee f_{\max}) \log(p)}{n_{\min}}};$$

2. *Prediction Loss:*

$$\frac{1}{n} \|\mathbf{H}_i \hat{\mathbf{Z}}_{-i} (\hat{\beta}_i - \beta_i)\|_2^2 \leq C_4^2 |\mathcal{S}_i| \times \frac{\|\omega_{\mathcal{S}_i}\|_{\infty}^2 \|\mathbf{B}\|_1^2 \|\mathbf{\Pi}\|_1^2}{\phi_0^2 \|\omega_i\|_{-\infty}^2} \frac{(d \vee r_{\max} \vee f_{\max}) \log(p)}{n_{\min}}.$$

The main idea of the proof is to take advantage of the commonly used restricted eigenvalue condition and irrepresentable condition for lasso-type estimator. However, the design matrix in our case includes predicted values instead of the original one, which complicates the proof. We claim that the restricted eigenvalue and irrepresentable condition still hold for the predicted design matrix as long as the estimation and prediction losses are well controlled at the calibration stage. The proof is detailed in the supplemental materials.

The available anchoring regulators as required by Assumption 1 implies that both  $\|\mathbf{B}\|_1 > 0$  and  $\|\mathbf{\Pi}\|_1 > 0$ , so  $h_n / \log(p) \rightarrow \infty$ . That is, these loss bounds hold with a sufficient large probability with properly chosen  $f^{(k)}$ .

The two sets of losses in Theorem 3 can also be controlled across the whole system by the same upper bounds defined by replacing  $|\mathcal{S}_i|$  with  $s_{\max} = \max_i |\mathcal{S}_i|$ , with probability at least  $1 - 3e^{-C_3 h_n + \log(4q) + 2 \log(p)} - e^{-f^{(1)} + 2 \log(p)} - e^{-f^{(2)} + 2 \log(p)}$ . When both  $p$  and  $q$  are divergent up to an exponential order, say  $p \asymp q \asymp e^{n_{\min}^c}$  for some  $c \in (0, 1)$ , we can set  $f^{(1)} = f^{(2)} = n_{\min}^{(1+c)/2}$  to guarantee the bounds at a sufficient large probability. However, the bounds are determined by  $(d \vee r_{\max} \vee f_{\max}) \log(p)$  which is  $o(n_{\min})$  only when  $c < \min(1/3, \theta)$ . Therefore, if  $s_{\max}$  also diverges up to  $n_{\min}^{\tilde{c}}$  with  $\tilde{c} < \min(1/4, \theta/2, 1 - \theta)$ , the losses can be well controlled for  $c < \min((1 - 4\tilde{c})/3, \theta - 2\tilde{c})$ .

Note that, with properly chosen  $f^{(1)}$  and  $f^{(2)}$ , these losses are well controlled at  $o(n_{\min})$ , revealing the fact that we need to have sufficient observations for each network for consistent differential analysis of the two networks.

Let  $W_i = \text{diag}\{\omega_i\}$ . Denote  $\mathcal{I}_i = \frac{1}{n} \mathbf{\Pi}_{-i}^T \mathbf{X}^T \mathbf{H}_i \mathbf{X} \mathbf{\Pi}_{-i}$  and  $\hat{\mathcal{I}}_i = \frac{1}{n} \hat{\mathbf{\Pi}}_{-i}^T \mathbf{X}^T \mathbf{H}_i \mathbf{X} \hat{\mathbf{\Pi}}_{-i}$ . Let  $\mathcal{I}_{i,11}$  be a submatrix of  $\mathcal{I}_i$  with rows and columns both indexed by  $\mathcal{S}_i$ , and  $\mathcal{I}_{i,21}$  be a submatrix of  $\mathcal{I}_i$  with rows and columns indexed by  $\mathcal{S}_i^c$  and  $\mathcal{S}_i$ , respectively.  $\hat{\mathcal{I}}_{i,11}$  and  $\hat{\mathcal{I}}_{i,21}$  are similarly defined from  $\hat{\mathcal{I}}_i$ . We further define the minimal signal strength  $b_i = \max_{j \in \mathcal{S}_i} |\beta_{ij}|$  and  $\psi_i = \|\mathcal{I}_{i,11}^{-1} W_{\mathcal{S}_i}\|_{\infty}$ .

The following assumption, reminiscent of the *adaptive irrepresentable condition* in Huang et al. [2008], helps investigate the selection consistency of regulatory effects.

**Assumption 5.** (Weighted Irrepresentable Condition) There exists a constant  $\tau \in (0, 1)$  such that  $\|W_{\mathcal{S}_i^c}^{-1} \mathcal{I}_{i,21}^{-1} \mathcal{I}_{i,11} W_{\mathcal{S}_i}\|_{\infty} < 1 - \tau$ .

**Theorem 4.** (Variable Selection Consistency) Denote  $\hat{\mathcal{S}} = \{j : \hat{\beta}_{ij} \neq 0, j \neq i\}$ . Suppose that, for node  $i$ ,  $\hat{\mathcal{I}}_{i,11}$  is invertible,  $b_i > \lambda_i \psi_i / (2 - \tau)$ , and  $\sqrt{(d \vee r_{\max} \vee f_{\max}) / n + c_{\max} \|\mathbf{\Pi}\|_1} \leq$

$\sqrt{c_{\max}^2 \|\mathbf{I}\|_1^2 + \min(\phi_0^2/64, \tau(4-\tau)^{-1} \|\omega_i\|_{-\infty}/\psi_i)/(C_2|S_i|)}$ .  
*Under Assumptions 1-5, there exists some constant  $C_5 > 0$  such that  $\hat{S}_i = S_i$  with probability at least  $1 - 3e^{-C_5 h_n + \log(4pq)} - e^{-f^{(1)} + \log(p)} - e^{-f^{(2)} + \log(p)}$ .*

This theorem implies that our proposed method can identify both common and differential regulatory effects between the two networks with a sufficiently large probability. On the other hand, the assumed weighted irrepressible condition means that the true signal should not correlate too much with irrelevant predictors so as to conduct a successful differential analysis. The corresponding proof is detailed in the supplemental materials.

### 3 EXPERIMENTS

#### 3.1 SYNTHETIC DATA EVALUATION

Here we report on experiments with synthetic data to show the superior performance of our method. We compare the method **RedNet** to a naive differential analysis which employs the 2SPLS method proposed by [Chen et al., 2017] to construct each network separately. Note that the 2SPLS method is modified here by applying ISIS to screen exogenous variables before conducting ridge regression to predict endogenous variables, making the naive differential analysis comparable to **RedNet**.

Synthetic data are generated from both acyclic and cyclic networks involving 1000 endogenous variables, with the sample size from 200 to 300. Each network includes a subnetwork of 50 endogenous variables, whose shared and differential structures will be investigated against its pair. On average, each endogenous variable has one regulatory effect in a sparse subnetwork, and three regulatory effects in a dense network. While each pair of subnetworks in comparison share many identical regulatory effects, they also share five regulatory effects but with opposite signs, and each network has five unique regulatory effects (so the total number of differential regulatory effects is 15). The nonzero regulatory effects were independently sampled from a uniform distribution over the range  $[-0.8, -0.3] \cup [0.3, 8]$ . While assuming each node is directly regulated by one exogenous variable, each exogenous variable was sampled from discrete values 0, 1 and 2 with probabilities 0.25, 0.5 and 0.25, respectively. All of the noise terms were independently sampled from the normal distribution  $N(0, 0.1^2)$ . We also conducted differential analysis between two networks with both  $\mathbf{X}^{(1)} \neq \mathbf{X}^{(2)}$  and  $\mathbf{X}^{(1)} = \mathbf{X}^{(2)}$  as in practice the paired networks may or may not share identically valued exogenous variables.

We evaluate the performance in terms of the false discovery rate (FDR), power and Matthews correlation co-

efficient (MCC) [Matthews, 1975]. Let TP, TN, FP and FN denote the numbers of true positives, true negatives, false positives, and false negatives, respectively. MCC is defined as,

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP} + \text{FP})(\text{TP} + \text{FN})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}}.$$

Here we refer nonzero effects as positives and zero effects as negatives. The MCC varies from 0 to 1 with larger values implying better variable selection.

In each differential analysis, the ridge regression employed the generalized cross validation [Golub et al., 1979] to select the ridge parameter, and the adaptive lasso used 10-fold cross-validation to choose its tuning parameter. Following the recommendation by Fan and Lv [2008],  $(n^{(k)})^{0.9}$  variables are screened by ISIS.

For each type of networks, 100 synthetic data sets were generated, and the differential analysis results are summarized in Figure 3. Overall, both **RedNet** and the naive approach maintain high power in identifying both differential and common regulatory effects. However, the naive approach tends to report high FDR, especially over 80% false discoveries of differential regulatory effects. Such a tendency to report false positives by the naive approach results in lower MCC, with dramatic decrease in identifying differential regulatory effects.

While both methods performed stably across networks with  $\mathbf{X}^{(1)} \neq \mathbf{X}^{(2)}$  and  $\mathbf{X}^{(1)} = \mathbf{X}^{(2)}$ , **RedNet** performed better in identifying both common and differential regulatory effects from dense networks than sparse networks in terms of FDR and MCC. However, the naive approach tends to report even higher FDR and so much lower MCC when identifying differential regulatory effects from dense networks, although reporting lower FDR and higher MCC when identifying common regulatory effects from dense networks.

We also calculated the standard errors (SE) of the reported FDR, power, and MCC over 100 synthetic data sets (the results are not shown). They are all small with most at the scale of thousandth and others at the scale of hundredth. Therefore, **RedNet** performed robustly in differential analysis of networks, and the 2SPLS approach by Chen et al. [2017] performed also robustly in constructing single networks.

#### 3.2 THE GENOTYPE-TISSUE EXPRESSION DATA

We performed differential analysis of gene regulatory networks on two sets of genetic genomics data from the Genotype-Tissue Expression (GTEx) project [Carithers et al., 2015], with one collected from human whole blood

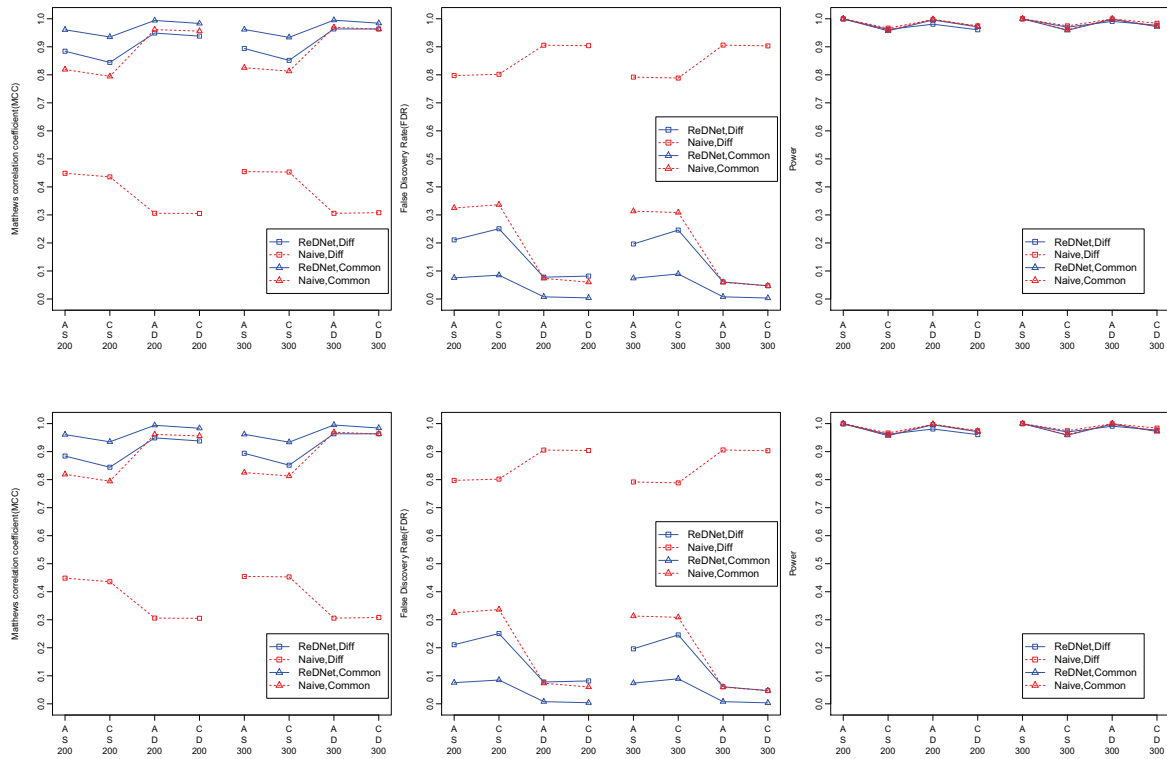


Figure 3: Performance of ReDNet Versus the Naive Approach (i.e. two networks are constructed independently). The results average over 100 synthetic data sets for different types of networks, with letters *A*, *C*, *S*, *D* in the x-axis denoting Acylic, Cyclic, Sparse and Dense networks, respectively. “Diff” and “Common” summarize the performance on differential and common regulatory effects, respectively. The sample size  $n^{(2)} = n^{(2)}$  is either 200 or 300.

(WB) and another one from human muscle skeletal (MS). The WB and MS data included genome-wide genetic and genotypic values from 350 and 367 healthy subjects, respectively. Both data sets were preprocessed following Carithers et al. [2015] and Stegle et al. [2010], resulting in a total of 15,899 genes and 1,083,917 single nucleotide polymorphisms (SNPs) being shared by WB and MS.

Expression quantitative trait loci (eQTL) mapping [Gilad et al., 2008] was conducted and identified 9875 genes with at least one marginally significant cis-eQTL (with  $p$ -value  $< 0.05$ ). For each gene, we further filtered its set of cis-eQTL by controlling the pairwise correlation under 0.9 and keeping up to three cis-eQTL which have the strongest association with the corresponding gene expression. These cis-eQTL serve as anchoring exogenous variables for the genes, and expression levels of different genes are endogenous variables. At completion of preprocessing data, we have 9,875 endogenous variables and 23,920 exogenous variables.

We applied **ReDNet** to infer the differential gene regulation on a set of eighty target genes, which had largest changes on gene-gene correlation between the two tis-

sues. We identified a total of 711 common and 572 differential regulations on the eighty target genes. To evaluate the significance of identified regulations, we bootstrapped 100 data sets, and conducted differential analysis on each bootstrap data set. As summarized in Table 1, 50, 43 and 34 differential regulatory effects were identified in over 70%, 80% and 90% of the bootstrap data sets, respectively.

Table 1: Summary of Regulations Identified in Over 70%, 80%, 90% of the Bootstrap Data Sets by **ReDNet** From GTEx Data. Shown under “Original” are for those identified from the original data.

	Original	70%	80%	90%
Common	711	116	108	93
Differential	572	50	43	34

The top four subnetworks bearing differential regulations on some of the eighty target genes were shown in Figure 4. We also constructed the differential networks using the naive approach (the results are not shown), and reported more regulations which cover the reported ones

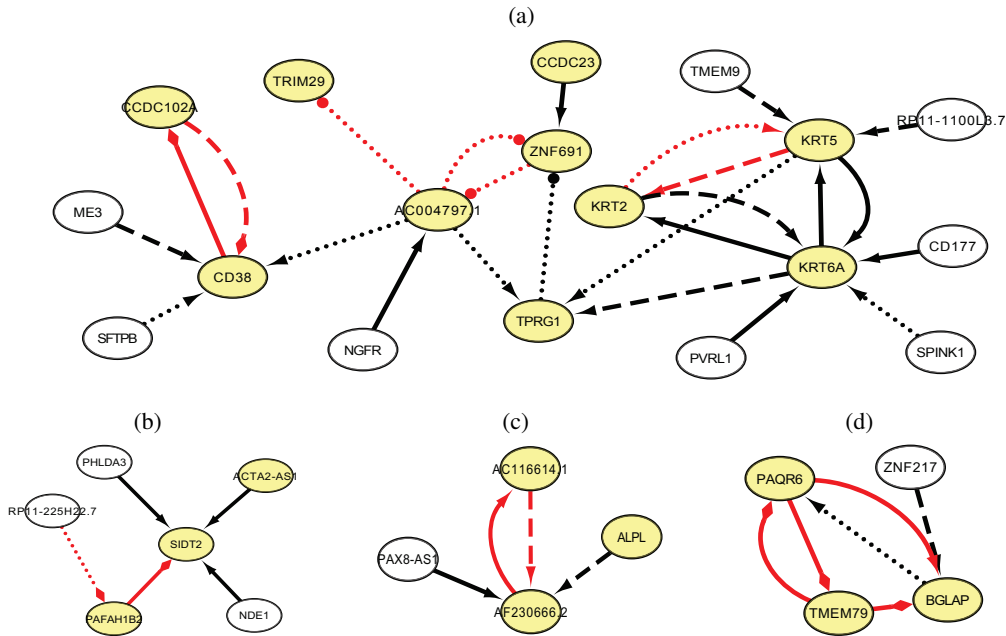


Figure 4: The Top Four Differential Subnetworks of Gene Regulation Identified by **ReDNet** From GTEx Data. The dotted, dashed, and solid lines imply regulations constructed in over 70%, 80%, and 90% of the bootstrap data sets, respectively. Highlighted in yellow are the target genes whose regulatory genes are focused in this study. The differential regulations are in red while common regulations are in black. The arrow head implies up regulation in both networks or no regulation in at most one network; the circle head implies down regulation in the whole blood but up regulation in muscle skeletal; and the diamond head implies up regulation in whole blood but down regulation muscle skeletal.

by **ReDNet**. This concurs with our observation in the synthetic data evaluation that the naive approach tends to report higher false positives, especially for differential regulatory effects.

## 4 CONCLUSION

We have developed a novel two-stage differential analysis method named **ReDNet**. The first stage, i.e., the calibration stage, aims for good prediction of the endogenous variables, and the second stage, i.e., the construction stage, identifies both common and differential network structures in a node-wise fashion. The key idea of **ReDNet** method is to appropriately reparametrize the independent models into a joint model so as to estimate differential and common effects directly. This approach can dramatically reduce the false discovery rate. In the experiments with synthetic data, we demonstrated the effectiveness of our method, which outperformed the naive approach with a large margin. Note that **ReDNet** allows independently conducting all  $\ell_2$  regularized regressions at the same time at the first stage, and all  $\ell_1$  regularized regressions at the same time at the second stage. Therefore, **ReDNet** not only permits parallel computa-

tion but also allows for fast subnetwork construction to avoid potential huge computational demands from differential analysis of large networks.

There are some interesting directions for future research. Firstly, it is worthwhile to explore other re-parametrization approaches such as baseline reparametrization in a case-control study. Secondly, while we only consider differential analysis of two networks, it is possible to generalize our method to compare multiple networks, demanding more complex reparametrization. Finally, applying the proposed method for fully differential analysis of 53 tissues in the GTEx project still provides challenging computational and methodological issues.

## Acknowledgments

The Genotype-Tissue Expression (GTEx) Project was supported by the Common Fund of the Office of the Director of the National Institutes of Health, and by NCI, NHGRI, NHLBI, NIDA, NIMH, and NINDS. The data used for the analysis described in this paper were obtained from dbGaP accession number phs000424.v7.p2 on 08/18/2017.

## References

- Lisa F Berkman and S Leonard Syme. Social networks, host resistance, and mortality: a nine-year follow-up study of alameda county residents. *American Journal of Epidemiology*, 109(2):186–204, 1979.
- Peter J Bickel, Yaacov Ritov, Alexandre B Tsybakov, et al. Simultaneous analysis of lasso and dantzig selector. *The Annals of Statistics*, 37(4):1705–1732, 2009.
- Xiaodong Cai, Juan Andrés Bazerque, and Georgios B Giannakis. Inference of gene regulatory networks with sparse structural equation models exploiting genetic perturbations. *PLoS Computational Biology*, 9(5):e1003068, 2013.
- Latarsha J Carithers, Kristin Ardlie, Mary Barcus, Philip A Branton, Angela Britton, Stephen A Buia, Carolyn C Compton, David S DeLuca, Joanne Peter-Demchok, Ellen T Gelfand, et al. A novel approach to high-quality postmortem tissue procurement: the gtex project. *Biopreservation and Biobanking*, 13(5):311–319, 2015.
- Chen Chen, Min Zhang, and Dabao Zhang. A two-stage penalized least squares method for constructing large systems of structural equations. *arXiv preprint arXiv:1511.00370v2*, 2017.
- Jianqing Fan and Yuan Liao. Endogeneity in high dimensions. *The Annals of Statistics*, 42(3):872, 2014.
- Jianqing Fan and Jinchi Lv. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(5):849–911, 2008.
- Yoav Gilad, Scott A Rifkin, and Jonathan K Pritchard. Revealing the architecture of gene regulation: the promise of eqtl studies. *Trends in Genetics*, 24(8):408–415, 2008.
- Gene H Golub, Michael Heath, and Grace Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.
- Jian Huang, Shuangge Ma, and Cun-Hui Zhang. Adaptive lasso for sparse high-dimensional regression models. *Statistica Sinica*, pages 1603–1618, 2008.
- Markus Kalisch and Peter Bühlmann. Estimating high-dimensional directed acyclic graphs with the pc-algorithm. *Journal of Machine Learning Research*, 8(Mar):613–636, 2007.
- Wei Lin, Rui Feng, and Hongzhe Li. Regularization methods for high-dimensional instrumental variables regression with an application to genetical genomics. *Journal of the American Statistical Association*, 110(509):270–288, 2015.
- Bing Liu, Alberto de La Fuente, and Ina Hoeschele. Gene network inference via structural equation modeling in genetical genomics experiments. *Genetics*, 178(3):1763–1776, 2008.
- Benjamin A Logsdon and Jason Mezey. Gene expression network reconstruction by convex feature selection when incorporating genetic perturbations. *PLoS Computational Biology*, 6(12):e1001014, 2010.
- Brian W Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451, 1975.
- Nicolai Meinshausen and Peter Bühlmann. High-dimensional graphs and variable selection with the lasso. *The Annals of Statistics*, 34:1436–1462, 2006.
- Yang Ni, Yuan Ji, Peter Müller, et al. Reciprocal graphical models for integrative gene regulatory network analysis. *Bayesian Analysis*, 2017. doi:10.1214/17-BA1087.
- Yang Ni, Peter Mller, Yitan Zhu, and Yuan Ji. Heterogeneous reciprocal graphical models. *Biometrics*, 2018. doi: 10.1111/biom.12791.
- Fabio Pianese, Xueli An, Fahim Kawsar, and Hiroki Ishizuka. Discovering and predicting user routines by differential analysis of social network traces. In *2013 IEEE 14th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoW-MoM)*, pages 1–9, June 2013.
- Peter Schmidt. *Econometrics*. New York, Marcel Dekker, 1976.
- Peter Spirtes, Clark N Glymour, and Richard Scheines. *Causation, Prediction, and Search*. MIT Press, 2000.
- Oliver Stegle, Leopold Parts, Richard Durbin, and John Winn. A Bayesian framework to account for complex non-genetic factors in gene expression levels greatly increases power in eqtl studies. *PLoS Computational Biology*, 6(5):e1000770, 2010.
- James West, Ginestra Bianconi, Simone Severini, and Andrew E Teschendorff. Differential network entropy reveals cancer system hallmarks. *Scientific Reports*, 2, 2012.
- Momiao Xiong, Jun Li, and Xiangzhong Fang. Identification of genetic networks. *Genetics*, 166(2):1037–1052, 2004.
- Ming Yuan and Yi Lin. Model selection and estimation in the gaussian graphical model. *Biometrika*, 94(1):19–35, 2007.
- Ying Zhu. Sparse linear models and  $\ell_1$ -regularized 2SLS with high-dimensional endogenous regressors and instruments. *Journal of Econometrics*, 202(2):196–213, 2018.

---

# Sparse-Matrix Belief Propagation

---

**Reid Bixler**

Department of Computer Science  
Virginia Tech  
Blacksburg, VA 24061

**Bert Huang**

Department of Computer Science  
Virginia Tech  
Blacksburg, VA 24061

## Abstract

We propose sparse-matrix belief propagation, which executes loopy belief propagation in pairwise Markov random fields by replacing indexing over graph neighborhoods with sparse-matrix operations. This abstraction allows for seamless integration with optimized sparse linear algebra libraries, including those that perform matrix and tensor operations on modern hardware such as graphical processing units (GPUs). The sparse-matrix abstraction allows the implementation of belief propagation in a high-level language (e.g., Python) that is also able to leverage the power of GPU parallelization. We demonstrate sparse-matrix belief propagation by implementing it in a modern deep learning framework (PyTorch), measuring the resulting massive improvement in running time, and facilitating future integration into deep learning models.

## 1 INTRODUCTION

Belief propagation is a canonical inference algorithm for graphical models such as Markov random fields (MRFs) or Bayesian networks (Pearl, 2014; Wainwright et al., 2008). In graphs with cycles, loopy belief propagation performs approximate inference. Loopy belief propagation passes messages from the variable nodes to their neighbors along the graph structure. These messages are fused to estimate marginal probabilities, also referred to as beliefs. After enough iterations of the algorithm, these beliefs tend to represent a good approximate solution to the actual marginal probabilities. In this paper, we consider pairwise MRFs, which only have unary and pairwise factors.

One drawback of loopy belief propagation is that, though

the algorithm is relatively simple, its implementation requires management of often irregular graph structures. This fact usually results in tedious indexing in software. The algorithm’s message-passing routines can be compiled to be rather efficient, but when implemented in a high-level language, such as those used by data scientists, they can be prohibitively slow. Experts typically resort to writing external software in lower-level, compiled languages such as C++. The implementation of belief propagation (and its variants) as separate, compiled libraries creates a barrier for its integration into high-level data science workflows.

We instead derive loopy belief propagation for pairwise MRFs as a sequence of matrix operations, resulting in sparse-matrix belief propagation. In particular, we use sparse-matrix products to represent the message-passing indexing. The resulting algorithm can then be implemented in a high-level language, and it can be executed using highly optimized sparse and dense matrix operations. Since matrix operations are much more general than loopy belief propagation, they are often built in as primitives in high-level mathematical languages. Moreover, their generality provides access to interfaces that implement matrix operations on modern hardware, such as graphical processing units (GPUs).

In this paper, we describe sparse-matrix belief propagation and analyze its running time, showing that its running time is asymptotically equivalent to loopy belief propagation. We also describe how the abstraction can be used to implement other variations of belief propagation. We then demonstrate its performance on a variety of tests. We compare loopy belief propagation implemented in Python and in C++ against sparse-matrix belief propagation using `scipy.sparse` and PyTorch on CPUs and PyTorch on GPUs. The results illustrate the advantages of the sparse-matrix abstraction, and represent a first step toward full integration of belief propagation into modern machine learning and deep learning workflows.

## 1.1 RELATED WORK

Belief propagation is one of the canonical variational inference methods for probabilistic graphical models (Pearl, 2014; Wainwright et al., 2008). Loopy belief propagation is naturally amenable to fine-grained parallelism, as it involves sending messages across edges in a graph in parallel. The algorithm is classical and well studied, but because it involves tight loops and intricate indexing, it cannot be efficiently implemented in high-level or mathematical programming languages. Instead, practitioners rely on implementations in low-level languages such as C++ (Schmidt, 2007; Andres et al., 2012). Specific variations for parallel computing have been proposed (Schwing et al., 2011), and other variations have been implemented in graph-based parallel-computing frameworks (Low et al., 2014; Malewicz et al., 2010). Specialized implementations have also been created for belief propagation on GPUs (Zheng et al., 2012).

While loopy belief propagation is the canonical message-passing inference algorithm, many variations have been created to address some of its shortcomings. Some variations modify the inference objective to make belief propagation a convex optimization, such as tree-reweighted belief propagation (Wainwright et al., 2003) and convexified belief propagation (Meshi et al., 2009). Other variations compute the most likely variable state rather than marginal probabilities, such as max-product belief propagation (Wainwright et al., 2008) and max-product linear programming (Globerson and Jaakkola, 2008).

Linearized belief propagation approximates the message update formulas with linear operations (Gatterbauer et al., 2015), which, like our approach, can benefit from highly optimized linear algebra libraries and specialized hardware. However, our approach aims to retain the exact non-linear formulas of belief propagation (and variants), while linearized belief propagation is an approximation.

Our approach of using sparse matrices as an abstraction layer for implementing belief propagation relies on sparse matrix operations being implemented in efficient, optimized, compiled libraries. Special algorithms have been developed to parallelize these operations on GPUs, enabling sparse-matrix computations to use the thousands of cores typically available in such hardware (Bell and Garland, 2008). Other libraries for sparse-matrix computation, such as those built into MATLAB (Gilbert et al., 1992) and `scipy.sparse` often seamlessly provide multi-core parallelism. Frameworks for large-scale, distributed matrix computation, such as Apache Spark (Bosagh Zadeh et al., 2016), can also be used as backends for our approach. Finally, one of the more important recent advances in computing hardware, field-programmable gate arrays (FPGAs), also support sparse-

matrix operations (Zhuo and Prasanna, 2005).

By formulating belief propagation as a series of matrix and tensor operations, we make it fit into modern deep learning software frameworks, such as PyTorch (Paszke et al., 2017) and TensorFlow (Abadi et al., 2016). Since these frameworks are designed to easily switch between CPU and GPU computation, we use PyTorch for one of our belief propagation implementations in our experiments. The added advantage is that, once these frameworks fully support back-propagation through sparse matrix products, we will be able to immediately back-propagate through belief propagation. Back-propagating through inference has been shown to allow more robust training of graphical model parameters (Domke, 2013).

## 2 BACKGROUND

In this section, we review belief propagation and some common—but not often documented—simplifications. Define a pairwise Markov random field (MRF) as a factorized probability distribution such that the probability of variable  $x \in \mathbb{X}$  is

$$\Pr(X = \mathbf{x}) = \frac{1}{Z} \exp \left( \sum_{i \in \mathcal{V}} \phi_i(x_i) + \sum_{(i,j) \in \mathcal{E}} \phi(x_i, x_j) \right), \quad (1)$$

where the normalizing constant  $Z$  is

$$Z = \sum_{X \in \mathbb{X}} \exp \left( \sum_{i \in \mathcal{V}} \phi_i(x_i) + \sum_{(i,j) \in \mathcal{E}} \phi(x_i, x_j) \right), \quad (2)$$

and  $\mathbf{x}$  represents the full state vector of all variables,  $x_i \in \mathbb{X}_i$  represents the state of the  $i$ th variable, and  $G = \{\mathcal{V}, \mathcal{E}\}$  is a graph defining the structure of the MRF.

The goal of marginal inference is to compute or approximate the marginal probabilities of the variables

$$\{\Pr(x_1), \dots, \Pr(x_n)\} \quad (3)$$

and of other subsets of variables. In pairwise MRFs, the marginal inference is often limited to the unary marginals and the pairwise marginals along the edges.

### 2.1 LOOPY BELIEF PROPAGATION

Belief propagation is a dynamic programming algorithm for computing marginal inference in tree-structured MRFs that is often applied in practice on non-tree, i.e., loopy, graphs. Loopy belief propagation is therefore a heuristic approximation that works well in many practical scenarios. The algorithm operates by passing messages among variables along the edges of the MRF structure.



The message sent from variable  $x_i$  to variable  $x_j$  is

$$m_{i \rightarrow j}[x_j] = \log \left( \sum_{x_i} \exp(a_{x_i}) \right), \quad (4)$$

where (as shorthand to fit the page)

$$a_{x_i} = \phi_{ij}(x_i, x_j) + \phi_i(x_i) + \sum_{k \in N_i \setminus j} m_{k \rightarrow i}[x_i] - d_{ij}; \quad (5)$$

$N_i$  is the set of neighbors of variable  $i$ , i.e.,  $N_i = \{k | (i, k) \in \mathcal{E}\}$ ; and  $d_{ij}$  is any constant value that is eventually cancelled out by normalization (see Eq. (7)). The message itself is a function of the receiving variable's state, which in a discrete setting can be represented by a vector (hence the bracket notation  $m_{i \rightarrow j}[x_j]$  for indexing by the state of the variable).

The estimated unary marginals, i.e., the *beliefs*, are computed by fusing the incoming messages with the potential function and normalizing:

$$\Pr(x_j) \approx \exp(b_j[x_j]) = \exp \left( \phi_j(x_j) + \sum_{i \in N_j} m_{i \rightarrow j}[x_j] - z_j \right), \quad (6)$$

where  $z$  is a normalizing constant

$$z_j = \log \left( \sum_{x_j} \exp \left( \phi_j(x_j) + \sum_{i \in N_j} m_{i \rightarrow j}[x_j] \right) \right). \quad (7)$$

For convenience and numerical stability, we will consider the log-beliefs

$$b_j[x_j] = \phi_j(x_j) + \sum_{i \in N_j} m_{i \rightarrow j}[x_j] - z_j. \quad (8)$$

We again use bracket notation for indexing into the beliefs because, for discrete variables, the beliefs can most naturally be stored in a lookup table, which can be viewed as a vector.

## 2.2 SIMPLIFICATIONS

The message update in Eq. (4) contains an expression that is nearly identical to the belief definition in Eq. (8). In the message update, the exponent is

$$a_{x_i} = \phi_{ij}(x_i, x_j) + \phi_i(x_i) + \sum_{k \in N_i \setminus j} m_{k \rightarrow i}[x_i] - d_{ij}. \quad (9)$$

The only difference between this expression and the belief update is that the belief uses the constant  $z_j$  to ensure normalization and the message update omits the message

from the receiving variable ( $j$ ). For finite message values, we can use the equivalence

$$\begin{aligned} \phi_i(x_i) + \sum_{k \in N_i \setminus j} m_{k \rightarrow i}[x_i] - d_{ij} &= \\ \phi_i(x_i) + \sum_{k \in N_i} m_{j \rightarrow i}[x_i] - m_{j \rightarrow i}[x_i] - d_{ij} &= \\ b_i[x_i] - m_{j \rightarrow i}[x_i], \end{aligned} \quad (10)$$

where the last equality sets  $d_{ij} = z_i$ . The message and belief updates can then be written as

$$\begin{aligned} b_j[x_j] &= \phi_j(x_j) + \sum_{i \in N_j} m_{i \rightarrow j}[x_j] - z_j, \\ m_{i \rightarrow j}[x_j] &= \\ \log \left( \sum_{x_i} \exp(\phi_{ij}(x_i, x_j) + b_i[x_i] - m_{j \rightarrow i}[x_i]) \right). \end{aligned} \quad (11)$$

These two update equations repeat for all variables and edges, and when implemented in low-level languages optimized by pipelining compilers, yield the fastest computer programs that can run belief propagation. However, the indexing requires reasoning about the graph structure, making any code that implements such updates cumbersome to maintain, prone to errors, and difficult to adapt to new computing environments such as parallel computing settings.

## 3 SPARSE-MATRIX BELIEF PROPAGATION

In this section, we describe sparse-matrix belief propagation and analyze its complexity. Instead of directly implementing the updates, sparse-matrix belief propagation uses an abstraction layer of matrices and tensors. This abstraction allows belief propagation to be written in high-level languages such as Python and MATLAB, delegating the majority of computation into highly optimized linear algebra libraries.

### 3.1 TENSOR REPRESENTATIONS OF THE MRF, BELIEFS, AND MESSAGES

We store a  $c$  by  $n$  belief matrix  $\mathbf{B}$ , where  $n$  is the number of variables and  $c$  is the maximum number of states of any variable. For simplicity, assume all variables have the cardinality  $c$ , i.e.,  $|\mathbb{X}_i| = c$ . This belief matrix is simply the stacked belief vectors, such that  $B_{ij} = b_j[i]$ . Similarly, we rearrange this matrix into an analogously shaped and ordered unary potential function matrix  $\Phi$ , where  $\Phi_{ij} = \phi_j(x_j = i)$ .

We store the pairwise potentials in a three-dimensional tensor  $\Gamma$  of size  $c \times c \times |E|$ . Each  $k$ th slice of the tensor

is a matrix representing the  $k$ th edge potential function as a table, i.e.,  $\Gamma_{ijk} = \phi_{s_k t_k}(i, j)$ .

Consider a view  $E$  of the edge set  $\mathcal{E}$  in which each edge appears in forward and reverse order, i.e.,  $(i, j) \in E$  if and only if  $(j, i) \in E$ . Let the edges also be indexed in an arbitrary but fixed order

$$E = \{(s_1, t_1), (s_2, t_2), \dots, (s_{|E|}, t_{|E|})\}, \quad (12)$$

and define vectors  $\mathbf{s} = [s_1, \dots, s_{|E|}]^\top$  and  $\mathbf{t} = [t_1, \dots, t_{|E|}]^\top$ . These variables encode a fixed ordering for the messages. The particular order does not matter, but to convert message passing into matrix operations, we need the order to be fixed.

In addition to the fixed order, we also define a *message reversal* order vector  $\mathbf{r}$  where

$$E[r_i] = (t, s) \text{ if } E[i] = (s, t). \quad (13)$$

We can represent this reversal vector as a sparse,  $|E| \times |E|$  permutation matrix  $\mathbf{R}$  where  $R_{ij} = 1$  iff  $r_i = j$ .

Define a  $c$  by  $|E|$  message matrix  $\mathbf{M}$  whose  $i$ th column is the  $i$ th message. In other words,  $M_{ij} = m_{s_j \rightarrow t_j}[i]$ , or equivalently,  $\mathbf{M}$  is a horizontal stack of all messages:

$$\mathbf{M} = [m_{s_1 \rightarrow t_1}, m_{s_2 \rightarrow t_2}, \dots, m_{s_{|E|} \rightarrow t_{|E|}}]. \quad (14)$$

Finally, we define a binary *sparse* matrix  $\mathbf{T}$  (for *to*) with  $|E|$  rows and  $n$  columns whose nonzero entries are as follows:

$$T_{ij} = \begin{cases} 1.0 & \text{if } t_i = j \\ 0 & \text{otherwise.} \end{cases} \quad (15)$$

This matrix  $\mathbf{T}$  maps the ordered messages to the variables that receive the messages.

We define an analogous matrix  $\mathbf{F}$  (for *from*), also binary and sparse with  $|E|$  rows and  $n$  columns, whose nonzero entries are as follows:

$$F_{ij} = \begin{cases} 1.0 & \text{if } s_i = j \\ 0 & \text{otherwise.} \end{cases} \quad (16)$$

This matrix  $\mathbf{T}$  maps the ordered messages to the variables that *send* the messages.

### 3.1.1 The logsumexp operation

A common operation that occurs in various steps of computing log-space belief propagation is the logsumexp operation, which is defined as follows:

$$\text{logsumexp}(\mathbf{A}) = \log(\exp(\mathbf{A}) \cdot \mathbf{1}), \quad (17)$$

where the log and exp operations are performed element-wise, and we use the matrix-vector product with the ones

vector ( $\mathbf{1}$ ) as a compact notation for summing across the rows of the exponentiated input matrix.<sup>1</sup> The resulting output is a column vector with the same number of rows as the input matrix  $\mathbf{A}$ .

### 3.1.2 Slice Indexing

To describe the message updates in a compact notation, we use slice indexing, which is common in matrix and tensor software because it can be implemented efficiently in linear time and easy to parallelize. We borrow syntax from `numpy` that is also reminiscent of the famous MATLAB syntax, where

$$\mathbf{A}[:, \mathbf{i}] = [\mathbf{A}[:, i_1], \mathbf{A}[:, i_2], \dots]. \quad (18)$$

This slice indexing allows the reordering, selection, or repetition of the rows or columns of matrices or tensors.

## 3.2 BELIEF PROPAGATION AS TENSOR OPERATIONS

Using these constructed matrices, the belief matrix is updated with the operations

$$\begin{aligned} \tilde{\mathbf{B}} &\leftarrow \Phi + \mathbf{M}^\top \mathbf{T} \\ \mathbf{B} &\leftarrow \tilde{\mathbf{B}} - \mathbf{1} \text{logsumexp}(\tilde{\mathbf{B}}), \end{aligned} \quad (19)$$

where the last operation uses the logsumexp operation to compute the normalizing constants of each belief column vector and multiplies by the ones vector ( $\mathbf{1}$ ) to broadcast it across all rows.<sup>2</sup>

The message matrix  $\mathbf{M}$  is updated with the following formula:

$$\mathbf{M} \leftarrow \text{logsumexp}(\Gamma + \mathbf{B}[:, \mathbf{s}] - \mathbf{M}[:, \mathbf{r}]). \quad (20)$$

This expression uses two forms of shorthand that require further explanation. First, the addition of the tensor  $\Gamma$  and the matrix  $(\mathbf{B}[:, \mathbf{s}] - \mathbf{M}[:, \mathbf{r}])$  requires broadcasting. The tensor  $\Gamma$  is of size  $c \times c \times |E|$ , and the matrix  $(\mathbf{B}[:, \mathbf{s}] - \mathbf{M}[:, \mathbf{r}])$  is of size  $c \times |E|$ . The broadcasting copies the matrix  $c$  times and stacks them as rows to form the same shape as  $\Gamma$ . Second, the logsumexp operation sums across the columns of the summed tensor, outputting

<sup>1</sup>It is especially useful to form this abstraction because this operation is notoriously unstable in real floating-point arithmetic. Numerically stable implementations that adjust the exponent by subtracting a constant and adding the constant back to the output of the logarithm are possible. These stable implementations add a linear-time overhead to the otherwise linear-time operation, so they maintain the same asymptotic running time of the original logsumexp operation.

<sup>2</sup>In `numpy`, this broadcasting is automatically inferred from the size of the matrices being subtracted.

a tensor of shape  $c \times 1 \times |E|$ , which is then squeezed into a matrix of size  $c \times |E|$ .

The message matrix can equivalently be updated with this formula:

$$\mathbf{M} \leftarrow \text{logsumexp}(\mathbf{\Gamma} + \mathbf{B}\mathbf{F}^\top - \mathbf{M}\mathbf{R}). \quad (21)$$

Here  $\mathbf{B}\mathbf{F}^\top - \mathbf{M}\mathbf{R}$  is once again  $c \times |E|$ , and it is equivalent to the slice-notation form above.

Belief propagation is then implemented by iteratively running Eq. (19) and then either of the equivalent Eqs. (20) and (21).

### 3.3 VARIATIONS OF BELIEF PROPAGATION

Many variations of belief propagation can similarly be converted into a sparse-matrix format. We describe some of these variations here.

#### 3.3.1 Tree-Reweighted Belief Propagation

The tree-reweighted variation of belief propagation (TRBP) computes messages corresponding to a convex combination of spanning trees over the input graph. The result is a procedure that optimizes a convex inference objective (Wainwright et al., 2003; Wainwright et al., 2008). The belief and message updates for TRBP are adjusted according to edge-appearance probabilities in a distribution of spanning trees over the MRF graph. These updates can be implemented in matrix form by using a length  $|E|$  vector  $\boldsymbol{\rho}$  containing the appearance probabilities ordered according to edge set  $E$ . The matrix-form updates for the beliefs and messages are then

$$\begin{aligned} \tilde{\mathbf{B}} &\leftarrow \mathbf{\Phi} + (\boldsymbol{\rho} \circ \mathbf{M})^\top \mathbf{T} \\ \mathbf{B} &\leftarrow \tilde{\mathbf{B}} - \mathbf{1} \text{logsumexp} \left( \tilde{\mathbf{B}} \right), \\ \mathbf{M} &\leftarrow \text{logsumexp}(\mathbf{\Gamma}/\boldsymbol{\rho} + \mathbf{B}\mathbf{F}^\top - \mathbf{M}\mathbf{R}), \end{aligned} \quad (22)$$

where element-wise product  $\circ$  and element-wise division  $/$  are applied with appropriate broadcasting.

#### 3.3.2 Convexified Belief Propagation

Another important variation of loopy belief propagation uses counting numbers to adjust the weighting of terms in the factorized entropy approximation. The resulting message update formulas weight each marginal by these counting numbers. Under certain conditions, such as when all counting numbers are non-negative, the inference objective can be shown to be concave, so this method is often referred to as *convexified Bethe belief propagation* (Meshi et al., 2009). We can exactly mimic the message and belief update formulas for convexified belief propagation by instantiating a vector  $\mathbf{c}$  containing the counting

numbers of each edge factor, resulting in the updates

$$\begin{aligned} \tilde{\mathbf{B}} &\leftarrow \mathbf{\Phi} + \mathbf{M}^\top \mathbf{T} \\ \mathbf{B} &\leftarrow \tilde{\mathbf{B}} - \mathbf{1} \text{logsumexp} \left( \tilde{\mathbf{B}} \right), \\ \mathbf{M} &\leftarrow \text{logsumexp}(\mathbf{\Gamma} + (\mathbf{B}\mathbf{F}^\top - \mathbf{M}\mathbf{R})/\mathbf{c}) \circ \mathbf{c}. \end{aligned} \quad (23)$$

The counting numbers for unary factors can be used to compute the inference objective, but they do not appear in the message-passing updates.

#### 3.3.3 Max-Product Belief Propagation

Finally, we illustrate that sparse tensor operations can be used to conduct approximate *maximum a posteriori* (MAP) inference. The max-product belief propagation algorithm (Wainwright et al., 2008) is one method for approximating MAP inference, and it can be implemented with the following updates:

$$\begin{aligned} \mathbf{B} &\leftarrow \text{onehotmax}(\mathbf{\Phi} + \mathbf{M}^\top \mathbf{T}) \\ \mathbf{M} &\leftarrow \text{logsumexp}(\mathbf{\Gamma} + \mathbf{B}\mathbf{F}^\top - \mathbf{M}\mathbf{R}), \end{aligned} \quad (24)$$

where `onehotmax` is a function that returns an indicator vector with 1 for entries that are the maximum of each column and zeros everywhere else, e.g., the “one-hot” encoding. Similar conversions are also possible for variations of max-product, such as max-product linear programming (Globerson and Jaakkola, 2008).

### 3.4 TIME-COMPLEXITY ANALYSIS

To analyze our sparse-matrix formulation of loopy belief propagation, and to show that it requires an asymptotically equivalent running time to normal loopy belief propagation, we first revisit the sizes of all matrices involved in the update equations. The first step is the belief update operation, Eq. (19), which updates the  $c$  by  $n$  belief matrix  $\mathbf{B}$ . The potential matrix  $\mathbf{\Phi}$  is also  $c$  by  $n$ ; the message matrix  $\mathbf{M}^\top$  is  $c$  by  $|E|$ ; and the *sparse* message-recipient indexing matrix  $\mathbf{T}$  is  $|E|$  by  $n$ .

The second step in Eq. (19) normalizes the beliefs. It subtracts from  $\mathbf{B}$  the product of  $\mathbf{1}$ , which is a  $c$  by 1 vector, and  $\text{logsumexp} \left( \tilde{\mathbf{B}} \right)$ , which is 1 by  $n$ . Explicitly writing the numerically stable `logsumexp` operation, the right side of this line can be expanded to

$$\tilde{\mathbf{B}} - \mathbf{1} \log \left( \text{sum} \left( \exp \left( \mathbf{B} - \max \left( \mathbf{B} \right) \right) \right) + \max \left( \mathbf{B} \right) \right). \quad (25)$$

We next examine the message update Eq. (21), which updates  $\mathbf{M}$ . The three-dimensional tensor  $\mathbf{\Gamma}$  is of size  $c \times c \times |E|$ ; the *sparse* message-sender indexing matrix  $\mathbf{F}^\top$  is  $c$  by  $|E|$ ; and the *sparse* reversal permutation matrix  $\mathbf{R}$  is  $|E| \times |E|$ . The message matrix  $\mathbf{M}$  is  $c$  by  $|E|$ .

**CPU computation** These three update steps are the entirety of each belief propagation iteration. From the first line of Eq. (19), the main operation to analyze is the dense-sparse matrix multiplication  $\mathbf{M}^\top \mathbf{T}$ . Considering an  $n \times m$  dense matrix  $A$  and a sparse matrix  $B$  of size  $m \times p$  with  $s$  nonzero entries (i.e.,  $\|B\|_0 = s$ ), the sparse dot product has time complexity  $O(ms)$  in sequential computation, as on a CPU. The time complexity of the sparse dot product depends upon the number of rows  $m$  and the number of sparse elements in the sparse matrix  $B$ . Every other computation in Eq. (19) involves element-wise operations on  $c$  by  $n$  matrices. Thus, Eq. (19) requires  $O(nc + \|\mathbf{T}\|_0 c)$  time. Since the sparse indexing matrix  $\mathbf{T}$  is defined to have a single nonzero entry per column, corresponding to edges, the time complexity of this step is  $O(nc + |E|c)$ .

In the message-update step, Eq. (21), the outer logsumexp operation and the additions involve element-wise operations over  $c \times c \times |E|$  tensors. The matrix multiplications are all dense-sparse dot products, so the total cost of Eq. (21) is  $O(|E|c^2 + \|\mathbf{F}\|_0 c + \|\mathbf{R}\|_0 c)$  (Gilbert et al., 1992). Since both indexing matrices  $\mathbf{F}$  and  $\mathbf{R}$  have one entry per edge, the total time complexity of the message update is  $O(|E|c^2 + |E|c)$ .

The combined computational cost of both steps is  $O(nc + |E|c + |E|c^2)$ . This iteration cost is the same as traditional belief propagation.

**GPU computation** Since the matrix operations in our method are simple, they are easily parallelized on GPUs. Ignoring the overhead of transferring data to GPU memory, we focus on the time complexity of the message passing. First consider the dense-sparse matrix multiplication mentioned previously, with a dense  $n$  by  $m$  matrix  $A$  and sparse  $m$  by  $p$  matrix  $B$  with  $s$  nonzero entries. GPU algorithms for sparse dot products use all available cores to run threads of matrix operations (Bell and Garland, 2008). In this case, each thread can run the multiplication operation of a single column in the sparse matrix  $B$ .

Given  $k$  cores/threads, we assume that there will be two cases: (1) when the number of sparse columns  $m$  is *less than or equal to* the number of cores  $k$  and (2) when the number of sparse columns  $m$  is *more than* the number of cores  $k$ . For either case, let  $s_i$  be the number of nonzero entries in column  $i$ . The time complexity of case (1) is  $O(\max_i s_i)$ , which is the time needed to process whichever column requires the most multiplications. For case (2), the complexity is  $O(\lceil \frac{m}{k} \rceil \max_i s_i)$ , which is the time for each set of cores to process  $k$  columns. In case (2), we are limited by our largest columns, as the rest of our smaller columns will be processed much sooner. Overall, the GPU time complex-

ity of this operation is  $O\left(\max\left(\max_i s_i, \lceil \frac{m}{k} \rceil \max_i s_i\right)\right)$ . In our sparse indexing matrices, each column has at most one element.

For the element-wise dense matrix operations, which have time complexity  $O(nm)$  on the CPU, we can again multi-thread each entry over the number of cores  $k$  in our GPU such that the time complexity is  $O\left(\lceil \frac{nm}{k} \rceil\right)$ .

The running time of the belief propagation steps is then  $O\left(\lceil \frac{n}{k} \rceil c + \lceil \frac{|E|}{k} \rceil c + \lceil \frac{|E|c^2}{k} \rceil\right)$ .

Based on our parallelism analysis, we expect significantly faster running times when running on the GPU, especially in cases where we have a large number of cores  $k$ . While we expect some time loss due to data-transfer overhead to the GPU, this overhead may be negligible when considering the overall time cost for every iteration of the message-passing matrix operations.

Finally, this analysis also applies to other shared-memory, multi-core, multithreaded environments (e.g., on CPUs), since in both CPU and GPU settings, matrix rows can be independently processed.

## 4 EMPIRICAL EVALUATION

In this section, we describe our empirical evaluation of sparse-matrix belief propagation. We measure the running time for belief propagation on a variety of MRFs using different software implementations and hardware, including optimized and compiled code for CPU-based computation and sparse-matrix belief propagation in a high-level language for both CPU- and GPU-based computation.

### 4.1 EXPERIMENTAL SETUP

We generate grid MRFs of varying sizes. We randomly generate potential functions for each MRF such that the log potentials are independently normally distributed with variance 1.0. We use MRFs with different variable cardinalities  $c$  from the set  $\{8, 16, 32, 64\}$ . We run experiments with MRFs structured as square, two-dimensional grids, where the number of rows and columns in the grids are  $\{8, 16, 32, 64, 128, 256, 512\}$ . In other words, the number of variables in models with these grid sizes are, respectively, 64, 256, 1024, 4,096, 16,384, 64,536, and 262,144. We run all implementations of belief propagation until the total absolute change in the messages is less than  $10^{-8}$ .

We run our experiments on different hardware setups. We use two different multi-core CPUs: a 2.4 Ghz Intel i7 with 4 cores and a 4 Ghz Intel i7 with 4 cores.

We also run sparse-matrix belief propagation on various

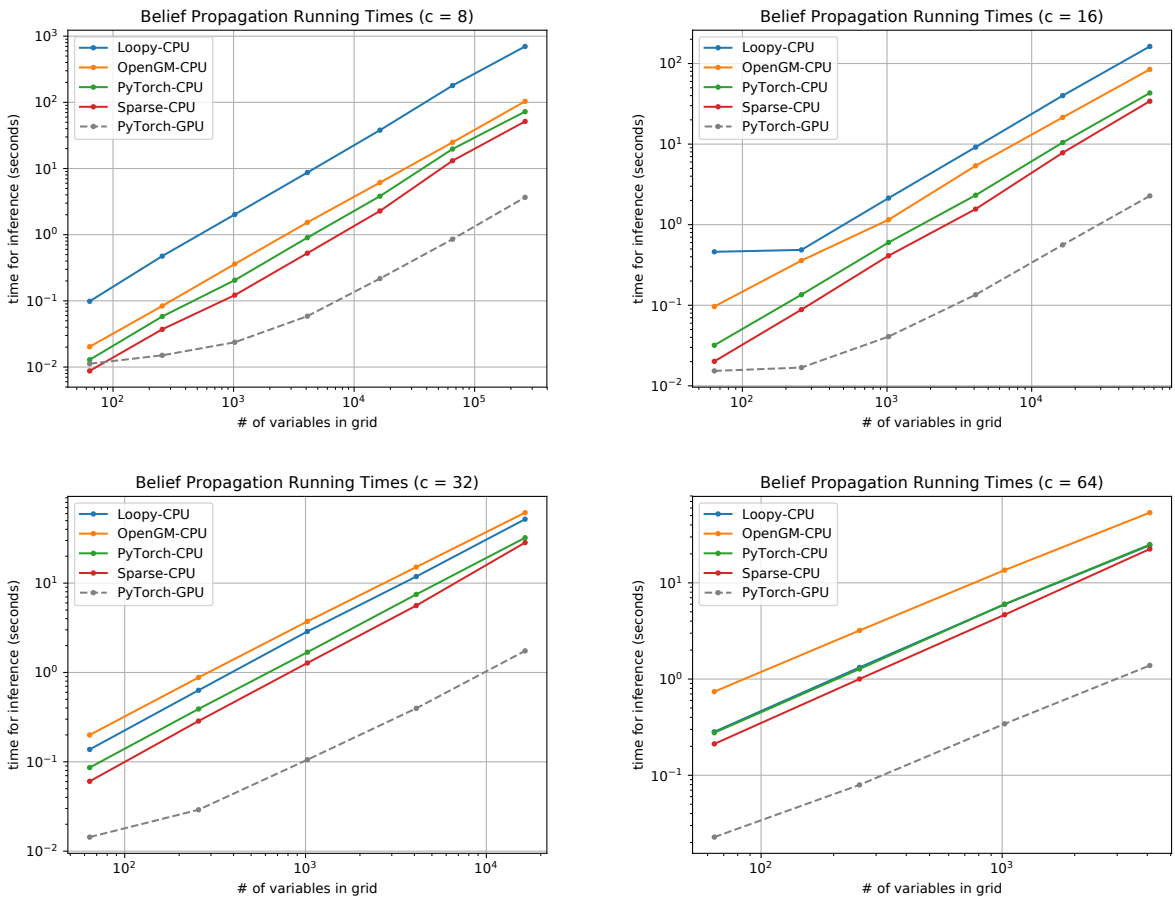


Figure 1: Log-log plots of belief propagation running times for four different implementations on the CPU. Each plot shows the results for different variable cardinalities  $c$ . OpenGM refers to the compiled C++ library, Loopy refers to the direct Python implementation. Sparse uses implements sparse-matrix belief propagation with `scipy.sparse`. And PyTorch implements it with the PyTorch library. We also plot the running time using PyTorch on the least powerful GPU we tested (Nvidia GTX780M) for comparison. The CPU runs plotted here use a 2.4 Ghz Intel i7.

GPUs. We run on an Nvidia GTX 780M (1,536 cores, 4 GB memory), an Nvidia GTX 1080 (2,560 cores, 8 GB), an Nvidia GTX 1080Ti (3,584 cores, 11 GB), and an Nvidia Tesla P40 (3840 cores, 24 GB).

Additional experiments will be available in this paper’s supplemental material.

## 4.2 IMPLEMENTATION DETAILS

We compare four different implementations of belief propagation. First, we use the compiled and optimized C++ implementation of belief propagation in the OpenGM library (Andres et al., 2012). This software represents the low-level implementation. Second, we use a direct implementation of simplified belief propagation (see Section 2.2) in Python and `numpy` using Python loops and dictionaries (hash maps) to manage indexing over graph structure.

Third, we use an implementation of sparse-matrix belief propagation in Python using `scipy.sparse`. Fourth, we use an implementation of sparse-matrix belief propagation in Python using the deep learning library PyTorch, which enables easily switching between CPU and GPU computation.

## 4.3 RESULTS AND DISCUSSION

Considering the results for CPU-based belief propagation in Fig. 1, the sparse-matrix belief propagation is faster than any other CPU-based belief propagation for all MRF sizes and all variable cardinalities. Similarly, the curves show a clear linearity, with a slope suggesting that all the CPU-based belief propagation algorithms increase in time complexity at a linear rate. It is also evident that the PyTorch implementation is consistently slower

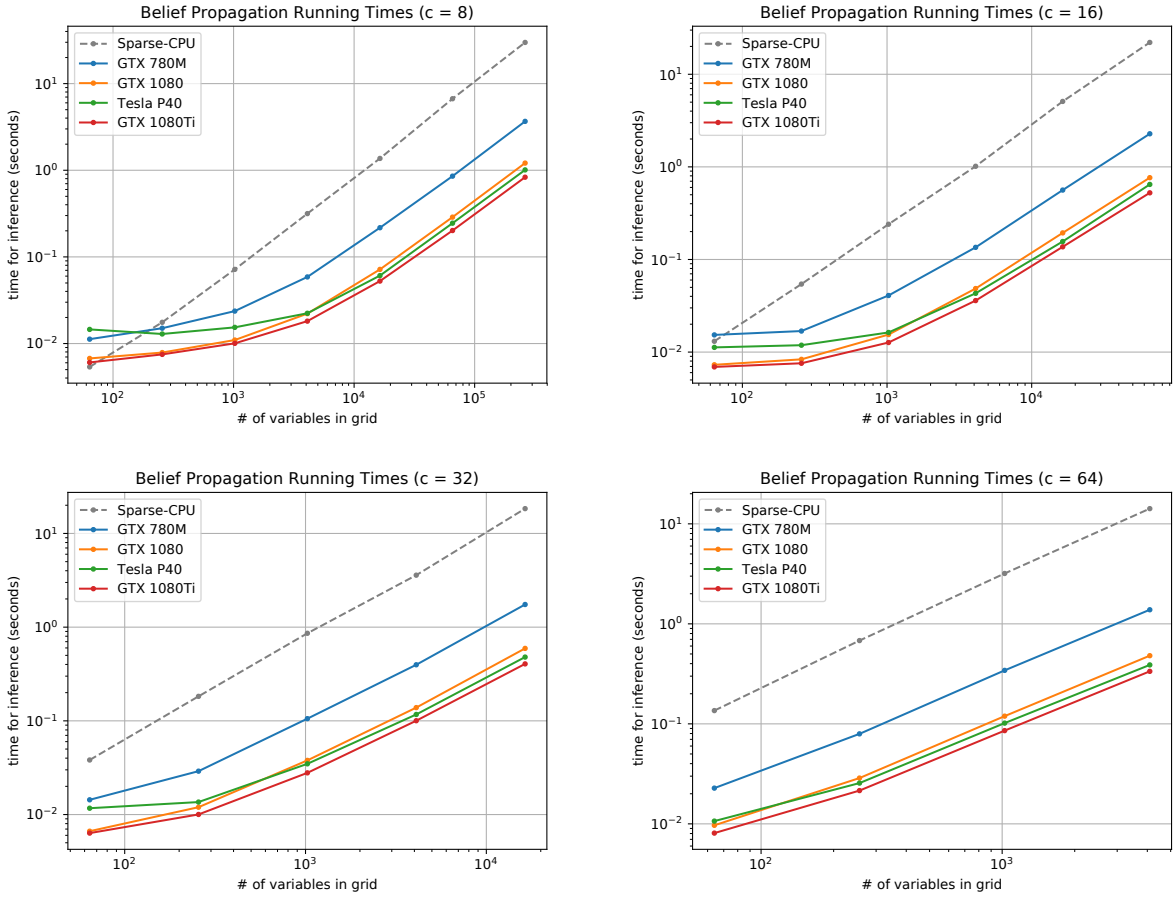


Figure 2: Log-log plots of PyTorch-GPU belief propagation running times for four different GPUs (780M, 1080, Tesla P40, 1080Ti) and the fastest CPU method (Sparse-CPU) with different variable cardinalities  $c$ . The CPU is the 4Ghz Intel i7.

than the `scipy.sparse` implementation, which is to be expected because PyTorch operations incur an additional overhead for their ability to integrate with deep learning procedures (e.g., back-propagation and related tasks).

Notably, we can see that the direct Python loop-based implementation is by far the slowest of these options. However, when the variable cardinality increases to large values, the Python implementation nearly matches the speed of sparse-matrix belief propagation with PyTorch on the CPU. While OpenGM’s belief propagation does offer some benefits compared to Python initially at a lower values of  $c$ , it actually results in the slowest running times at  $c \geq 32$ . We can conclude that despite the compiled and optimized C++ code, the number of variables and states can eventually overshadow any speedups initially seen at lower values.

In Fig. 1, we also include the running times for sparse-matrix belief propagation with PyTorch on the GPU—

shown with the dotted gray line. A direct comparison is not exactly fair, since we are comparing across different computing hardware, though these curves were measured on the same physical computer. The comparison makes clear that the GPU offers significant speed increases over any CPU-based belief propagation, even that of the faster `scipy.sparse` implementation. Interestingly, there is a trend with the GPU runtimes that are sub-linear in the log-log plots, representing the cases where the number of sparse columns is not yet more than the number of cores of the GPU.

Examining the results for GPU-based belief propagation in Fig. 2, a majority of the GPUs are fairly close in running time between the three powerful Nvidia units: the 1080, the 1080Ti, and the P40. The 780M understandably lags behind. As seen previously, until the cores are saturated with operations, there appears to be a pseudo-constant time cost. And once the cores are saturated, the running times grow linearly. This trend is best seen at

$c = 16$  for the first two or three points. We also include the fastest CPU running time, using `scipy.sparse` on an Intel 4 Ghz i7, to illustrate the drastic difference in time between sparse-matrix belief propagation on the CPU and GPU.

These results demonstrate that sparse-matrix belief propagation enables the fastest running times for inference in these grid MRFs on the CPU. And using different software backends (`scipy.sparse` or PyTorch) for the sparse-matrix operations leads to different behavior, with PyTorch incurring some overhead resulting in slower computation. Once ported to the GPU, the speedups are even more drastic, resulting in running times that are many factors faster than those seen on the CPU, easily outweighing the overhead cost of using software backends like PyTorch that support seamless switching from CPU to GPU computation.

## 5 CONCLUSION

We presented sparse-matrix belief propagation, which exactly reformulates loopy belief propagation (and its variants) as a series of matrix and tensor operations. This reformulation creates an abstract interface between belief propagation and a variety of highly optimized libraries for sparse-matrix and tensor operations. We demonstrated how sparse-matrix belief propagation scales as efficiently as low-level, compiled and optimized implementations, yet it can be implemented in high-level mathematical programming languages. We also demonstrated how the abstraction layer allows easy portability to advanced computing hardware by running sparse-matrix belief propagation on GPUs. The immediately resulting parallelization benefits required little effort once the sparse-matrix abstraction was in place. Our software library with these implementations is available at <https://bitbucket.org/berthuang/mrftools/>.

**Open Questions and Next Steps** There are still a number of research directions that we would like to pursue. We mainly focused on analyzing the benefits of sparse-matrix belief propagation on grid-based MRFs, but there are many different structures of MRFs used in important applications (chain models, random graphs, graphs based on structures of real networks). Similarly, applying our approach to real-world examples would help confirm the utility of it in current problems within machine learning. Likewise, we focused on fairly sparse graphs that did not have many non-zero entries per column. It would be interesting to explore the difference between how sparse-matrix belief propagation behaves on the dense and sparse matrices on different hardware and whether fully dense matrices would still result in notable speed improvements,

or if overhead from the sparse-matrix format would become a bottleneck.

Our sparse-matrix formulation of belief propagation is derived for pairwise MRFs, so it remains an open question what modifications are necessary for higher-order MRFs which may have arbitrarily large factors. Benefits similar to those we measured on GPUs can arise from the sparse-matrix abstraction for other computing backends, such as the use of compute-cluster parallelism through libraries such as Apache Spark, or the computation of belief propagation on FPGAs. Finally, the integration of belief propagation into deep learning models is straightforward with the matrix abstraction. Though many popular frameworks do not yet support back-propagation through sparse dot products, such support is forthcoming according to their respective developer communities.

## References

- Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283.
- Andres, B., Beier, T., and Kappes, J. H. (2012). Opengm: A C++ library for discrete graphical models. *CoRR*, abs/1206.0111.
- Bell, N. and Garland, M. (2008). Efficient sparse matrix-vector multiplication on CUDA. Technical report, Nvidia Technical Report NVR-2008-004, Nvidia Corporation.
- Bosagh Zadeh, R., Meng, X., Ulanov, A., Yavuz, B., Pu, L., Venkataraman, S., Sparks, E., Staple, A., and Zaharia, M. (2016). Matrix computations and optimization in Apache Spark. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 31–38. ACM.
- Domke, J. (2013). Learning graphical model parameters with approximate marginal inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(10):2454–2467.
- Gatterbauer, W., Günemann, S., Koutra, D., and Faloutsos, C. (2015). Linearized and single-pass belief propagation. In *Proceedings of the VLDB Endowment*, volume 8, pages 581–592. VLDB Endowment.
- Gilbert, J. R., Moler, C., and Schreiber, R. (1992). Sparse matrices in MATLAB: Design and implementation. *SIAM Journal on Matrix Analysis and Applications*, 13(1):333–356.

- Globerson, A. and Jaakkola, T. S. (2008). Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *Advances in Neural Information Processing Systems*, pages 553–560.
- Low, Y., Gonzalez, J. E., Kyrola, A., Bickson, D., Guestrin, C. E., and Hellerstein, J. (2014). Graphlab: A new framework for parallel machine learning. *arXiv preprint arXiv:1408.2041*.
- Malewicz, G., Austern, M. H., Bik, A. J., Dehnert, J. C., Horn, I., Leiser, N., and Czajkowski, G. (2010). Pregel: a system for large-scale graph processing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 135–146. ACM.
- Meshi, O., Jaimovich, A., Globerson, A., and Friedman, N. (2009). Convexifying the Bethe free energy. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 402–410. AUAI Press.
- Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in PyTorch. In *NIPS Workshop on Autodiff*.
- Pearl, J. (2014). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.
- Schmidt, M. (2007). UGM: A Matlab toolbox for probabilistic undirected graphical models.
- Schwing, A., Hazan, T., Pollefeys, M., and Urtasun, R. (2011). Distributed message passing for large scale graphical models. In *Computer Vision and Pattern Recognition*.
- Wainwright, M. J., Jaakkola, T. S., and Willsky, A. S. (2003). Tree-reweighted belief propagation algorithms and approximate ml estimation by pseudo-moment matching. In *International Conference in Artificial Intelligence and Statistics*.
- Wainwright, M. J., Jordan, M. I., et al. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305.
- Zheng, L., Mengshoel, O., and Chong, J. (2012). Belief propagation by message passing in junction trees: Computing each message faster using GPU parallelization. *arXiv preprint arXiv:1202.3777*.
- Zhuo, L. and Prasanna, V. K. (2005). Sparse matrix-vector multiplication on fpgas. In *Proceedings of the 2005 ACM/SIGDA 13th International Symposium on Field-Programmable Gate Arrays*, pages 63–74. ACM.



---

# Sequential Learning under Probabilistic Constraints

---

**Amirhossein Meisami**  
Adobe, Inc.  
San Jose, CA 95110

**Henry Lam**  
Columbia University  
New York, NY 10027

**Chen Dong**  
Adobe, Inc.  
San Jose, CA 95110

**Abhishek Pani**  
Adobe, Inc.  
San Jose, CA 95110

## Abstract

We provide the first study on online learning problems under stochastic constraints that are “soft”, i.e., need to be satisfied with high probability. These constraints are imposed on all or some stages of the time horizon so that the stage decisions probabilistically satisfy some given safety conditions. The distributions that govern these conditions are learned through the collected observations. Under a Bayesian framework, we introduce a scheme that provides statistical feasibility guarantees through the time horizon, by using posterior Monte Carlo samples to form sampled constraints which leverage the scenario generation approach in chance-constrained programming. We demonstrate how our scheme can be integrated into Thompson sampling and illustrate it with an application in online advertisement.

## 1 INTRODUCTION

Most of the literature in stochastic online learning focuses on performances measured by optimality achievement. Common examples include the minimization of cumulative regret in the multi-arm bandit setting (e.g., Auer et al. (2002); Lai and Robbins (1985)), best arm selection (e.g., Audibert and Bubeck (2010)) and the closely related ranking and selection (e.g., Boesel et al. (2003)) in the simulation literature. In many situations, however, the uncertainty or the stochasticity appears not only in the objective function, but also in the constraints of the problem whose feasibility can be of utmost importance. The focus of this paper is to design sequential methodologies that maintain probabilistic feasibility requirements with rigorous statistical guarantees.

Our study is motivated from a rich set of problems where

“budgets” or “resources” are limited for various operational or commercial reasons, and these constraints are in a sense “soft”, i.e., the capacities placed on these constraints, while preferred to be satisfied, are allowed to be violated with a small probability. Such consideration is common among applications. For example, in online advertisement problems encountered by our co-authors when optimizing spending for large advertisers, the task involves sequentially picking items (e.g., keywords, targets) to maximize revenues, while adhering to a specified marketing budget for a duration. The marketer in general expects to meet the budget goals. However, if occasionally the budget is exceeded the campaign is still acceptable as long as the revenue performance is sustained. Other similar settings include clinical trials, where the costs of competing treatments are substantial and noisy, and over-budget is undesired but sometimes allowable. Whereas past work in stochastic sequential learning has focused on rewards (with hard constraints if needed), this paper provides the first study on a class of problems that not just include the rewards but also stochastic constraints that need to be satisfied with high probability.

Our framework can be viewed as a sequential problem under so-called *probabilistic* or *chance constraints* (Prékopa, 2003), which has been widely used in stochastic programming under limited and uncertain resources (e.g., Shi et al. (2015); Lejeune and Ruszczyński (2007)). A generic representation of a chance constraint is

$$P((x, \xi) \text{ satisfies a given safety condition}) \geq 1 - \alpha \quad (1)$$

where  $x$  is a decision variable and  $\xi$  denotes some randomness distributed under  $P$ . Satisfying the safety condition means that  $(x, \xi)$  lies in a desirable deterministic region, which can be represented by, e.g., a set of inequalities. The given parameter  $\alpha$  is the tolerance level that represents the allowable probabilistic violation.

In the sequential setting,  $x$  would denote a sequence of decisions. The safety condition could include individual

requirements on all or some stages. In many applications of interest,  $P$  needs to be learned as complete distributional knowledge on  $\xi$  is not available. Along the vein of conventional online problems that focus on optimality, at each stage we may observe some components of  $\xi$  so that we can update our belief on  $P$ .

As our main methodological contribution, we analyze an online strategy that provides guarantees on (1) with a high confidence, under a statistical framework that we shall describe. On a high level, it means we can guarantee, with our proposed policy, that

$$\begin{aligned} &P(P((x, \xi) \text{ satisfies a given safety condition}) \\ &\geq 1 - \alpha) \geq 1 - \beta \end{aligned} \quad (2)$$

where the outer probability now refers to the randomness of  $x$  induced by the sequential observations, and  $1 - \beta$  is a confidence level (90% for instance). Our methodology is based on a combination of two ideas. First is a Bayesian extension of the so-called scenario generation or constraint sampling (Calafiore and Campi, 2005; De Farias and Van Roy, 2004) approach in approximating chance-constrained optimization problems. This approach replaces the unknown or difficult chance constraint with a collection of sampled constraints that come from data or from numerical simulation. Viewing such an approach in a Bayesian manner allows it to be blended naturally into popularly used online learning algorithms such as Thompson sampling (Agrawal and Goyal, 2012; Russo and Van Roy, 2016) that also operates via Bayesian updating. Second, by capitalizing results on scenario generation in the static setting, we can derive the precise number of samples required at each stage of the sequential process such that (2) holds throughout the horizon. As far as we know, our formulation and analysis of chance-constraint guarantees in an online setting is new to the literature.

After presenting our theoretical investigation on feasibility guarantees, we illustrate the integration of our scheme into a variant of Thompson sampling in an online advertisement setting. We then numerically demonstrate how this chance-constrained Thompson sampling performs competitively, in achieving feasibility but also maintaining good objective values.

## 2 RELATED WORK

The earliest work in chance constraints dated back to Charnes et al. (1958) and Miller and Wagner (1965). Exact solution techniques for such problems are notoriously difficult due to non-convexity, and are only available in few instances even when  $P$  is known; e.g., Lagoa et al. (2005). Several lines of approximation methodologies

have been proposed. A conventional method is to use so-called safe convex approximation that replaces the chance constraint with more conservative convex constraints (Ben-Tal and Nemirovski, 2000). Rossi et al. (2011, 2015) used policy trees and confidence interval construction to obtain the so-called  $(\alpha, \vartheta)$ -solution. Scenario generation (Calafiore and Campi, 2006; Campi and Garatti, 2008), which we leverage on in this work, uses sampled constraints to populate the feasible region. This approach has several extensions, such as sampling-and-discarding (Campi and Garatti, 2011) and multi-phase schemes (Carè et al., 2014; Calafiore, 2017; Chamanbaz et al., 2016), and relates to sample average approximation (Luedtke et al., 2010). Other data-driven methods include distributionally robust optimization (Calafiore and El Ghaoui, 2006; Zymler et al., 2013) and data-driven robust optimization (Bertsimas et al., 2013).

Our work focuses on chance-constrained problem in an online fashion, under the broad umbrella of sequential decision-making. In the later part of this paper, we demonstrate our proposed strategy in a variant of the stochastic multi-arm bandit problem (Auer et al., 2002) used to address the well-known exploration-exploitation tradeoff. In budgeted bandits, Ding et al. (2013) consider the presence of random costs and an overall budget, where learning and revenue accumulation stops when the budget runs out. Xia et al. (2015) study Thompson sampling for a similar setting; in this work we integrate our strategy into Thompson sampling, especially the one considered in Ferreira et al. (2016) motivated from network revenue management. Other related work include those in the framework of “bandits with knapsacks” (Badanidiyuru et al., 2013; Tran-Thanh et al., 2012; Besbes and Zeevi, 2012) that have been applied in pricing and supply chain management (Wang et al. (2014)) and healthcare (Villar et al. (2015)). The works closest to our online advertisement example are Tran-Thanh et al. (2014) and Amin et al. (2012) that study the problem of item bidding under a budget, but they do not consider probabilistic violation of the constraints that we focus on.

## 3 CHANCE-CONSTRAINED ONLINE LEARNING

Consider a sequence of decision variables  $x_t \in \mathbb{R}^d$ ,  $t = 1, \dots, T$ , and a sequence of random variables  $\xi_t \in \Xi_t$ ,  $t = 1, \dots, T$  assumed independent among the steps  $t$  in a given horizon  $T$ . For convenience, denote  $\xi_{1:t} = (\xi_1, \dots, \xi_t)$  and  $x_{1:t} = (x_1, \dots, x_t)$  as the cumulative randomness and decisions up to  $t$ . Consider a sequence of safety conditions that we write as  $f_t(x_{1:t}, \xi_{1:t}) \in \mathcal{A}_t$ , where each function  $f_t$  maps to some space  $\mathcal{Y}_t$  such that

$\mathcal{A}_t \subset \mathcal{Y}_t$  (for example,  $f_t(x_{1:t}, \xi_{1:t}) \in \mathcal{A}_t$  can be a set of inequalities so that  $\mathcal{Y}_t = \mathbb{R}^m$  for some  $m$  and  $\mathcal{A}_t = \{y \in \mathbb{R}^m : y \leq 0\}$ ).

We are interested in a sequential problem with horizon  $T$ :

$$\begin{aligned} & \max_{x_1, \dots, x_T} h(x_1, \dots, x_T) \\ \text{subject to} & \quad P(f_t(x_{1:t}, \xi_{1:t}) \in \mathcal{A}_t | \mathcal{F}_{t-1}) \geq 1 - \alpha \quad \forall t \in \mathcal{S}, \end{aligned} \quad (3)$$

where the decisions  $x_1, \dots, x_T$  are sequential, i.e.,  $x_{t+1}$  depends on the past observations of  $\xi_{1:t}$  and past decisions  $x_{1:t}$ ,  $\mathcal{S} \subset \{1, \dots, T\}$  is a given set, and  $h(\cdot)$  is the objective function. Note that the function  $f_t$  and the set  $\mathcal{A}_t$  can depend on the time step  $t$ . For convenience, let  $\mathcal{F}_t = \{\xi_{1:t}, x_{1:t}\}$  be the information up to time  $t$ . In each probability  $P$  in (3), the function  $f_t(x_{1:t}, \xi_{1:t})$  can be expressed as  $f_t(x_{1:(t-1)}, x_t, \xi_{1:(t-1)}, \xi_t)$  where  $x_{1:(t-1)}$  and  $\xi_{1:(t-1)}$  belong to the past information  $\mathcal{F}_{t-1}$ .

Consistent with the introduction,  $\alpha$  is a tolerance parameter on the violation of the safety condition. This parameter is assumed constant across  $t$  for convenience, but our analysis can be easily adapted to the case where it varies. Note that  $\mathcal{S}$  determines how many chance constraints need to be maintained throughout the horizon. For example,  $\mathcal{S} = \{1, \dots, T\}$  means there is a budget requirement for each step, and  $\mathcal{S} = \{T\}$  means there is only one overall budget requirement across the whole horizon.

### 3.1 SCENARIO GENERATION FOR STATIC PROBLEMS

We first discuss a well-studied approach to approximate a static version of (3). Suppose  $T = 1$ . In this setting we can simplify notation and write the formulation as

$$\begin{aligned} & \max_x h(x) \\ \text{subject to} & \quad P(f(x, \xi) \in \mathcal{A}) \geq 1 - \alpha \end{aligned} \quad (4)$$

Suppose we can simulate or collect data for  $\xi$  to obtain, say, i.i.d.  $\xi^1, \dots, \xi^N$ . We consider replacing the constraint in (4) by sampled constraints, so that the optimization program becomes

$$\begin{aligned} & \max_x h(x) \\ \text{subject to} & \quad f(x, \xi^n) \in \mathcal{A}, \quad \forall n = 1, \dots, N \end{aligned} \quad (5)$$

We call (5) a sampled program, which serves as a reasonable approximation to (4) when  $N$  is large. However, since  $\xi^n$ 's are randomly generated, the solution obtained from (5) is subject to statistical noise and cannot be guaranteed feasible for (4). The following celebrated result from (Calafiore and Campi, 2006; Campi and Garatti,

2008) gives the sample size needed to guarantee feasibility for (4) with a certain confidence by solving (5).

**Condition 1.** *An optimization program is said to be in class  $\mathcal{R}$  if: 1) It is feasible and the feasible region has a non-empty interior; 2) Its optimal solution exists and is unique.*

**Theorem 1.** *(Adopted from Theorem 2.4, Campi and Garatti (2008)) Suppose for each  $\xi$ ,  $f(x, \xi) \in \mathcal{A}$  is a convex set in  $x \in \mathbb{R}^d$ , and  $h$  is concave. Suppose also that any instance of (5) belongs to  $\mathcal{R}$ . Fix real numbers  $\alpha, \beta \in [0, 1]$ . Then for  $N$  chosen such that*

$$\sum_{i=0}^{d-1} \binom{N}{i} \alpha^i (1 - \alpha)^{N-i} \leq \beta$$

*the optimal solution of the sampled stochastic program (5) is feasible for (4) with probability no smaller than  $1 - \beta$ .*

It is known that this result can be improved, e.g., by using sampling-and-discarding (Campi and Garatti, 2011) and multi-stage or iterative schemes (Carè et al., 2014; Calafiore, 2017; Chamanbaz et al., 2016). In this paper we stick with the requirement in Theorem 1 to illustrate our proposed strategies; improvements can be made accordingly by modifying the use of Theorem 1 to better results available in the literature.

### 3.2 SCENARIO GENERATION UNDER UNKNOWN DISTRIBUTION: A BAYESIAN PERSPECTIVE

The scenario generation approach depicted in Theorem 1 requires direct observations on  $\xi$  or the capacity to obtain Monte Carlo samples for  $\xi$ . In problems with learning, the distribution of  $\xi$  is not fully known, and Theorem 1 does not apply directly. We shall adopt a Bayesian perspective that naturally integrates to many online algorithms (e.g., Thompson sampling). Suppose  $\xi$  follows a parametric distribution  $G|\mu$  with unknown parameter  $\mu$ . After specifying a prior distribution for  $\mu$  and collecting some data historically, we have a posterior distribution for  $\mu$  denoted by  $F$ . We seek to use Monte Carlo sampling to conduct an analog of scenario generation so that a posterior credibility guarantee

$$P_\mu(P_{\xi|\mu}(f(x, \xi) \in \mathcal{A}) \geq 1 - \alpha) \geq 1 - \beta \quad (6)$$

is achieved, where  $P_\mu$  denotes the posterior probability distribution on  $\mu$ , and  $P_{\xi|\mu}$  denotes the distribution of  $\xi$  given a parameter value of  $\mu$ . In other words, we want the chance constraint to hold with a posterior credibility level  $1 - \beta$ . Note that this is a natural Bayesian analog of the frequentist result in Theorem 1. In the setting of

Theorem 1,  $P$  is not known but data are available, so that a  $1 - \beta$  confidence is attained. In our current Bayesian investigation,  $P$  is not known but subject to a posterior belief summarized by the distribution of  $\mu$ , and we want this posterior credibility to be  $1 - \beta$ .

Directly sampling  $\xi$  using any particular value of  $\mu$  does not sufficiently capture the posterior uncertainty. To blend the latter into a scenario generation, we can use a two-level sampling, where in the first level we generate a posterior sample for  $\mu$ , and in the second level we generate  $\xi$  conditional on  $\mu$ . This sampling procedure is described in Algorithm 1.

---

**Algorithm 1** Posterior Constraint Sample Generator (PCSG)

---

1. Repeat  $N$  times:
  - (a) Generate  $\mu^n \sim F$ .
  - (b) Generate  $\xi^n \sim G|\mu^n$ .
2. Impose the constraints  $f(x, \xi^n) \in \mathcal{A}$ ,  $n = 1, \dots, N$  and solve the sampled program

$$\begin{aligned} & \max_x h(x) \\ & \text{subject to } f(x, \xi^n) \in \mathcal{A}, \forall n = 1, \dots, N \end{aligned} \quad (7)$$


---

In PCSG we encounter two different sources of randomness. First is the statistical noise from the uncertainty of  $\mu$ , captured by the posterior credibility level  $1 - \beta$ . The second source is the Monte Carlo error, and we denote by  $1 - \delta$  the confidence level induced from this error. By choosing a suitable sample size  $N$  in terms of  $\alpha, \beta, \delta$ , PCSG turns out to achieve a guarantee below.

**Theorem 2.** *Suppose  $f(x, \xi) \in \mathcal{A}$  is a convex set in  $x \in \mathbb{R}^d$ , and  $h(x)$  is concave. Suppose also that any instance of (7) belongs to  $\mathcal{R}$ . Fix real numbers  $\delta, \alpha, \beta \in [0, 1]$  and choose*

$$\sum_{i=0}^{d-1} \binom{N}{i} (\alpha\beta)^i (1 - \alpha\beta)^{N-i} \leq \delta \quad (8)$$

*Consider a solution  $x$  obtained from the sampled program in PCSG. Then,*

1.  $x$  satisfies (6) with a Monte Carlo confidence  $1 - \delta$ , i.e.,

$$P_{MC}(P_\mu(P_{\xi|\mu}(f(x, \xi) \in \mathcal{A}) \geq 1 - \alpha) \geq 1 - \beta) \geq 1 - \delta \quad (9)$$

*where the outermost  $P_{MC}$  denotes the probability with respect to the  $N$  Monte Carlo samples.*

2.  $x$  satisfies

$$E_{MC}[P_\mu(P_{\xi|\mu}(f(x, \xi) \in \mathcal{A}) \geq 1 - \alpha)] \geq (1 - \beta)(1 - \delta) \quad (10)$$

*where  $E_{MC}[\cdot]$  denotes the expectation with respect to the  $N$  Monte Carlo samples.*

Theorem 2 Part 1 stipulates that choosing  $N$  in (8) achieves chance-constraint feasibility with a Bayesian credibility  $1 - \beta$ , under a Monte Carlo confidence  $1 - \delta$ . Part 2 will be useful in generalizing to the multi-stage setting presented next.

### 3.3 SEQUENTIAL POLICIES

We now move our analysis to the sequential problem depicted in (3). We generalize PCSG, with the posterior update occurring at every step of the horizon and the sample size required at each step modified in order to achieve a chance constraint guarantee over the whole horizon. Let  $N_t$  be the sample size used in step  $t$ , which depends on  $\alpha$  and also the confidence-level parameters  $\beta_t$  and  $\delta_t$ . We denote  $F_0$  as the prior distribution of  $\mu$  and  $F_t$  as the posterior distribution of  $\mu$  at step  $t$ . We denote  $G_t|\mu$  as the distribution of  $\xi_t$  given  $\mu$ . Note that  $\mu$  is a parameter shared among the  $\xi_t$  at different steps so that information can be learned over time. To distinguish the real data from the Monte Carlo samples, we use  $\tilde{\xi}_{1:(t-1)}$  to denote the actual data of  $\xi$  coming from steps 1 to  $t - 1$ .

We have the following procedure:

---

**Algorithm 2** Dynamic PCSG

---

Set  $F_0$  as the prior distribution of  $\mu$ . For  $t = 1, \dots, T$ : While  $t \in \mathcal{S}$ , and given  $F_t$  and the realized  $x_{1:(t-1)}$  and  $\tilde{\xi}_{1:(t-1)}$ :

1. Repeat  $N_t$  times:
  - (a) Generate  $\mu^n \sim F_t$ .
  - (b) Generate  $\xi^n \sim G_t|\mu^n$ .

2. Impose the constraints

$$f_t(x_{1:(t-1)}, x_t, \tilde{\xi}_{1:(t-1)}, \xi^n) \in \mathcal{A}_t, \forall n = 1, \dots, N_t \quad (11)$$

at stage  $t$ .

---

It is understood that in the second step of Dynamic PCSG, the constraints are imposed together with an appropriate objective function (typically the cost-to-go in formulation (3)) to form a stepwise optimization with decision variable  $x_t$ . The following result gives the choice of  $N_t$  and the resulting guarantee:

**Theorem 3.** *Suppose the stepwise safety conditions are all convex sets, the objective function at every step is concave, and  $x_t \in \mathbb{R}^d$ . Suppose also that any instance of the optimization resulted from imposing (11) belongs to  $\mathcal{R}$ . Suppose  $0 \leq \beta_t, \delta_t \leq 1$  are constants such that*

$$\sum_{i=0}^{d-1} \binom{N_t}{i} (\alpha\beta_t)^i (1 - \alpha\beta_t)^{N_t-i} \leq \delta_t \quad (12)$$

and

$$\sum_{t \in \mathcal{S}} (\beta_t + \delta_t - \beta_t \delta_t) \leq \beta\lambda \quad (13)$$

Then the policy obtained from Dynamic PCSG is feasible for (3) under the updated posterior distribution with probability at least  $1 - \beta$ , with overall Monte Carlo confidence  $1 - \lambda$ , i.e.,

$$P_{MC}(P_{\mu_{1:T}}(P_{\xi_t|\mu_t}(f_t(x_t, \xi_t) \in \mathcal{A}_t | \mathcal{F}_{t-1}) \geq 1 - \alpha \forall t \in \mathcal{S}) \geq 1 - \beta) \geq 1 - \lambda \quad (14)$$

where  $P_{\mu_{1:T}}$  denotes the probability with respect to  $\mu_1, \dots, \mu_t$ , where each  $\mu_t \sim F_t$ , the posterior distribution of  $\mu$  at step  $t$ , and  $P_{\xi_t|\mu_t}$  denotes the probability with respect to  $\xi_t$  given a realized parameter of  $\mu_t$ .

Theorem 3 asserts that the round-specific statistical parameters, namely the posterior credibility  $1 - \beta_t$  and the Monte Carlo confidence level  $1 - \delta_t$ , which determine the constraint sample size, can be chosen to satisfy a local condition (12) and a global condition (13) to achieve an overall statistical guarantee.

For convenience we can set  $\beta_t = \delta_t$  and both equal to some constant, say  $\gamma_t$ . This  $\gamma_t$  can be set to be stage-independent or dependent. The following subsection shows two choices of  $\gamma_t$ .

### 3.4 TWO EXPLICIT STRATEGIES

We demonstrate two choices of  $\{N_t\}$  in terms of  $\{\gamma_t\}$ . The first choice is a simple one that requires knowledge of the horizon length  $T$ , by setting  $\gamma_t$  to be a constant. The second choice uses a decaying  $\gamma_t$ , consequently an increasing sample size  $N_t$ , which does not require knowledge of  $T$  a priori. For convenience, we denote  $|\mathcal{S}|$  as the size of the set  $\mathcal{S}$ . For the first strategy, we have:

**Proposition 1.** *Given a time horizon  $T$ , if we let  $\beta_t = \delta_t = \gamma$  for all  $t \in \mathcal{S}$  such that  $\gamma \leq 1 - \sqrt{1 - \beta\lambda/|\mathcal{S}|}$ , then (14) holds.*

The following describes our second strategy that is stage-dependent such that (14) holds without knowing the horizon  $T$  or  $|\mathcal{S}|$  a priori:

**Proposition 2.** *If we let  $\beta_t = \delta_t = \gamma_t$  for all  $t \in \mathcal{S}$  such that  $\gamma_t = (1/\zeta(t)^\rho) \wedge \eta$ , where  $\rho > 1$ ,  $0 < \eta < 1$ , and  $\zeta(t) = \#\{s \in \mathcal{S} : s \leq t\}$  (i.e.,  $\zeta(t)$  is the ‘‘counter’’ of  $t$  in  $\mathcal{S}$ ) such that*

$$2\eta^{1-1/\rho} + \frac{2}{\rho-1} \frac{1}{(1/\eta^{1/\rho} - 1)^{\rho-1}} - \frac{\eta^2}{\eta^{1/\rho} + 1} - \frac{1}{2\rho-1} \frac{1}{(1/\eta^{1/\rho} + 1)^{2\rho-1}} \leq \beta\lambda \quad (15)$$

then (14) holds regardless of  $|\mathcal{S}|$ .

For example, if  $\rho$  is set to be 2, then (15) becomes  $2\sqrt{\eta} + 2\sqrt{\eta}/(1-\sqrt{\eta}) - \eta^2/(\sqrt{\eta}+1) - \eta^{3/2}/(3(1+\sqrt{\eta})^3) \leq \beta\lambda$ .

## 4 INTEGRATION INTO THOMPSON SAMPLING

We illustrate the integration of our strategies with Thompson sampling, which also operates via Bayesian updating, by an example of revenue maximization in online advertising (Pani et al., 2017). The advertiser is interested in maximizing the expected revenue across a portfolios of keywords or biddable ad units while ensuring that the budget constraint is not violated. When the advertiser selects a bid value for a keyword it results in ad clicks, the volume of which is stochastic. The distribution of clicks and the associated revenue is not initially known to the decision-maker and needs to be learned over time. Further, the cost associated with the choice of a bid is also unknown and hence, there is uncertainty regarding how the budget will be affected.

To be more concrete, consider a set of  $K$  bid values  $\{\kappa_1, \dots, \kappa_K\}$ , for  $M$  items labeled  $\{\pi_1, \dots, \pi_M\}$ , over the campaign horizon  $T$ . Bidding value  $j$  on item  $i$  will induce an average revenue  $r_{ij}$  and cost  $c_{ij}$  respectively. These quantities are assumed to follow independent Poisson distributions with initially unknown parameters (the Poisson assumptions come from the click count nature). In each period  $t = 1, \dots, T$ , the advertiser picks a bid value  $j$  from every item, observes the outcome, i.e., the realizations of  $r_{ij}, c_{ij}, i = 1, \dots, M$ . She gains  $\sum_{i=1}^M \sum_{j=1}^K r_{ij} x_{ij}$  and consumes  $\sum_{i=1}^M \sum_{j=1}^K c_{ij} x_{ij}$  from the budget, where  $x_{ij}$  is the allocation portion for bid value  $j$  of item  $i$  (i.e., the fraction of time or the probability in a randomized scheme that is allocated to this particular bid value and item).

The advertiser’s goal is to maximize the total revenue while maintain a budget constraint with high probability. In other words, letting  $r_{ij}(t)$  and  $c_{ij}(t)$  be the realized revenue and cost for a bid at time  $t$ , and  $x_{ij}(t)$  be the corresponding allocation variables, she wants to maximize  $E[\sum_{t=1}^T \sum_{i=1}^M \sum_{j=1}^K r_{ij}(t) x_{ij}(t)]$ . A typical

budget constraint is a bound given by the remaining budget averaged over the remaining horizon. Specifically, let the overall budget be  $B$ . Denoting  $B(t-1) = B - \sum_{u=1}^{t-1} \sum_{i=1}^M \sum_{j=1}^K c_{ij}(u)x_{ij}(u)$  as the remaining budget before epoch  $t \in \mathcal{S}$ , the advertiser wants to keep  $P(\sum_{u=1}^t \sum_{i=1}^M \sum_{j=1}^K c_{ij}(u)x_{ij}(u) \leq B(t-1)/(T-t+1)) \geq 1 - \alpha \forall t \in \mathcal{S}$ . This type of dynamically updated per-round budgets is common in practice and is argued to be more effective than fixed per-round budgets. In the following, we will concentrate on the particular choice described above as the feasibility requirement.

#### 4.1 A BUDGETED ALGORITHM

The setting of this problem resembles a recent study (Ferreira et al., 2016) on a network revenue management problem. They developed a Thompson sampling algorithm to sequentially assign a price vector to items under resource constraints, where each step involves a knapsack optimization problem. Here we present a variant of their algorithm to suit our setting (Algorithm 3). This initial algorithm does not take into account the possibility of constraint violation; the idea is to later illustrate how our Dynamic PCSG strategy can be integrated.

Denote by  $X_{ij}(t-1)$  the allocation units on bid value  $j$  for item  $i$  cumulated in the first  $t-1$  rounds, and denote by  $W_{ij}^r(t-1)$  and  $W_{ij}^c(t-1)$  the total revenue and cost generated by assigning bid value  $j$  to item  $i$  during these periods respectively. In Algorithm 3, the advertiser samples from the joint posterior distributions of  $\theta_{ij}$ , the unknown Poisson rate of the revenue, and  $\mu_{ij}$ , the rate of the cost, corresponding to bid value  $j$  for item  $i$ . We put independent Gamma prior distributions for these parameters and hence the posterior distributions are also independent. The posterior samples of these parameters are then used in a linear program to decide the allocation. This algorithm follows quite intuitively from standard Thompson sampling, in which one generates posterior samples for the unknown parameters, and use them as ‘‘plug-in’’ to solve stage-wise optimization problems.

#### 4.2 CHANCE-CONSTRAINED BUDGETED THOMPSON SAMPLING

We want to ensure  $P(\sum_{u=1}^t \sum_{i=1}^M \sum_{j=1}^K c_{ij}(u)x_{ij}(u) \leq B(t-1)/(T-t+1)) \geq 1 - \alpha \forall t \in \mathcal{S}$  holds with posterior credibility  $1 - \beta$ . To achieve this, we integrate our Dynamic PCSG into Step 2 of Algorithm 3 by restructuring the involved optimization program. Algorithm 4 shows Dynamic PCSG in this particular setting.

The output of this procedure is a set of constraints, which will be used in the linear program of the budgeted

---

**Algorithm 3** Budgeted Thompson Sampling for deterministically constrained problems adopted from Ferreira et al. (2016)

---

Given a total budget  $B(0) = B$ . For  $t = 1, \dots, T$ , do the following:

- 1: For each bid value  $j$  and each item  $i$ , sample  $\theta_{ij}$  from  $\text{Gamma}(W_{ij}^r(t-1) + 1, X_{ij}(t-1) + 1)$  and  $\mu_{ij}$  from  $\text{Gamma}(W_{ij}^c(t-1) + 1, X_{ij}(t-1) + 1)$ .
- 2: Solve the following linear program:

$$\begin{aligned} \max_x \quad & \sum_{j=1}^K \sum_{i=1}^M \theta_{ij} x_{ij} \\ \text{subject to} \quad & \sum_{j=1}^K \sum_{i=1}^M \mu_{ij} x_{ij} \leq \frac{B(t-1)}{T-t+1} \\ & \sum_{j=1}^K x_{ij} \leq 1, \forall i = 1, \dots, M \\ & x_{ij} \geq 0, \forall i = 1, \dots, M, j = 1, \dots, K \end{aligned}$$

to obtain  $(x_{ij}^*(t))_{i=1, \dots, M, j=1, \dots, K}$ .

- 3: The revenue,  $r_{ij}(t)$ , and the cost,  $c_{ij}(t)$ , generated by assigning bid value  $j$  on item  $i$  are revealed. We update  $X_{ij}(t) = X_{ij}(t-1) + x_{ij}^*(t)$ ,  $W_{ij}^r(t) = W_{ij}^r(t-1) + r_{ij}(t)x_{ij}^*(t)$ ,  $W_{ij}^c(t) = W_{ij}^c(t-1) + c_{ij}(t)x_{ij}^*(t)$  and  $B(t) = B(t-1) - \sum_{j=1}^K \sum_{i=1}^M c_{ij}(t)x_{ij}^*(t)$ .
  - 4: If  $B(t) \leq 0$ , the algorithm terminates.
- 

Thompson sampling. Algorithm 5 shows how Dynamic PCSG can be integrated into Algorithm 3. The following is an immediate consequence of Theorem 3:

**Corollary 4.** *Suppose that any instance of the sampled program in (16) belongs to  $\mathcal{R}$ . Suppose  $0 \leq \beta_t, \delta_t \leq 1$  are chosen to satisfy*

$$\sum_{i=0}^{KM-1} \binom{N_t}{i} (\alpha\beta_t)^i (1 - \alpha\beta_t)^{N_t-i} \leq \delta_t$$

and

$$\sum_{t \in \mathcal{S}} (\beta_t + \delta_t - \beta_t \delta_t) \leq \beta \lambda$$

Consider a modification of Algorithm 5 such that at any point of time, if the total budget  $B$  is fully depicted, we refill the shortfall and add extra budget  $B$  (so that the total remaining budget in the next step returns to the full level  $B$ ). Then the sequence of decisions obtained will satisfy

$$\begin{aligned} P_{\mu} \left( P_{\mathbf{c}_t | \mu_t} \left( \sum_{u=1}^t \sum_{i=1}^M \sum_{j=1}^K c_{ij}(u)x_{ij}(u) \leq \frac{B(t-1)}{T-t+1} \right) \right. \\ \left. \geq 1 - \alpha \forall t \in \mathcal{S} \right) \geq 1 - \beta \end{aligned}$$

---

**Algorithm 4** Dynamic PCSG for the Bidding Problem

---

1. Set  $F_{ij}(0), i = 1, \dots, M, j = 1, \dots, K$  as the prior distribution of  $\mu_{ij}$ . For  $t = 1, \dots, T$ : Given  $B(t-1)$  and  $F_{ij}(t-1)$ , the posterior distribution of  $\mu_{ij}$  given  $\mathcal{F}_{t-1}$ ,
  - (a) Repeat  $N_t$  times:
    - i. Generate  $\mu_{ij}^n \sim F_{ij}(t-1)$  independently for each item  $i = 1, \dots, M$  and bid value  $j = 1, \dots, K$ .
    - ii. Generate  $\xi_{ij}^n \sim \text{Poisson}(\mu_{ij}^n)$  for each item  $i = 1, \dots, m$  and bid value  $j = 1, \dots, K$ .
  - (b) Form the constraints

$$\sum_{j=1}^K \sum_{i=1}^M \xi_{ij}^n x_{ij} \leq \frac{B(t-1)}{T-t+1}, \forall n = 1, \dots, N_t$$

---

with Monte Carlo confidence level at least  $1 - \lambda$ .  $P_\mu$  denotes the probability of  $\{\mu_t\}_{t=1, \dots, T}$ , where  $\mu_t$  is the collection  $(\mu_{ij})_{i,j}$  and each element is distributed independently according to the posterior distribution  $F_{ij}(t-1)$ , and  $P_{c_t|\mu_t}$  denotes the probability with respect to the collection  $(c_{ij})_{i,j}$  given the realization of  $(\mu_{ij})_{i,j}$ .

We mention that the “modification of Algorithm 5” introduced in Corollary 4 is only a technicality that takes care of the unusual situation when the entire available budget is prematurely depleted. Since we divide the remaining budget by the remaining horizon (a common practice to set per-round budgets) to form our constraint at each step, the scenario of total budget depletion before the last step rarely happens.

### 4.3 NUMERICAL RESULTS

We examine the empirical performance of our proposed strategy on a synthetic dataset with two items and three bid values ( $M = 2, K = 3$ ) over the time horizon  $T = 100$ . The cost and revenue of each item-bid value pair follow Poisson distributions with parameters taken uniformly from an interval that is calibrated from a real data set owned by a prominent tech firm (blinded for peer-review purpose). We test with five different values of the overall budget  $B = (a \sum_{i=1}^M \bar{\rho}_i) \times T$  where  $a \in [0.5, 0.75, 1, 1.25, 1.5]$  and  $\bar{\rho}_i$  is the average cost of item  $i$  over the  $K$  bid values. The choice of  $B$  roughly matches the scale of the total cost over the time. The per-round budget is defined as the remainder of the overall budget divided by the remaining number of rounds.

To test our chance-constrained Thompson sampling (CCTS) in Algorithm 5, we use three different settings for  $\mathcal{S}$ , i.e.  $\mathcal{S}_1 = \{25, 50, 75\}$ ,  $\mathcal{S}_2 = \{20, 40, 60, 80, 100\}$  and  $\mathcal{S}_3 = \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$ . We enforce  $\alpha = 0.1, \beta = \lambda = 0.3$ , and  $\beta_t = \delta_t = \gamma$  where

---

**Algorithm 5** Chance-constrained Thompson Sampling (CCTS)

---

Initialize  $\alpha, \beta_t, \delta_t \in [0, 1]$  satisfying (13). Given a total budget  $B(0) = B$ . For  $t = 1, \dots, T$ , do the following:

- 1: For each bid value  $j$  and each item  $i$ , sample  $\theta_{ij}$  from  $\text{Gamma}(W_{ij}^r(t-1) + 1, X_{ij}(t-1) + 1)$  and  $\mu_{ij}$  from  $\text{Gamma}(W_{ij}^c(t-1) + 1, X_{ij}(t-1) + 1)$ .
- 2: Run Dynamic PCSG using  $N_t$  samples to get the constraints

$$\sum_{j=1}^K \sum_{i=1}^M \xi_{ij}^n x_{ij} \leq \frac{B(t-1)}{T-t+1}, \forall n = 1, \dots, N_t$$

- 3: Solve the following linear program:

$$\begin{aligned} \max_x \quad & \sum_{j=1}^K \sum_{i=1}^M \theta_{ij} x_{ij} \\ \text{subject to} \quad & \sum_{j=1}^K \sum_{i=1}^M \xi_{ij}^n x_{ij} \leq \frac{B(t-1)}{T-t+1}, \forall n = 1, \dots, N_t \\ & \sum_{j=1}^K x_{ij} \leq 1, \forall i = 1, \dots, M \\ & x_{ij} \geq 0, \forall i = 1, \dots, M, j = 1, \dots, K \end{aligned} \tag{16}$$

to obtain  $(x_{ij}^*(t))_{i=1, \dots, M, j=1, \dots, K}$ .

- 4: The revenue,  $r_{ij}(t)$ , and the cost,  $c_{ij}(t)$ , generated by assigning bid value  $j$  on item  $i$  are revealed. We update  $X_{ij}(t) = X_{ij}(t-1) + x_{ij}^*(t)$ ,  $W_{ij}^r(t) = W_{ij}^r(t-1) + r_{ij}(t)x_{ij}^*(t)$ ,  $W_{ij}^c(t) = W_{ij}^c(t-1) + c_{ij}(t)x_{ij}^*(t)$  and  $B(t) = B(t-1) - \sum_{j=1}^K \sum_{i=1}^M c_{ij}(t)x_{ij}^*(t)$ .
  - 5: If  $B(t) \leq 0$ , the algorithm terminates.
- 

$\gamma$  is taken as the upper bound depicted in Proposition 1. With these configurations, the number of constraints in the involved linear programs are typically in the range of thousands, which gives a run-time of a few minutes in solving the decisions for the whole horizon using our sever machine. Note that, if we consider our online problem a daily problem (common in practice) then this solution time is quite acceptable as we have hours to solve the problem at each stage. Moreover, the number of constraints and hence the run-time can be further reduced by using more recent advances in the constraint sampling literature as depicted at the end of Section 3.1.

For each considered setting, we conduct 500 simulation runs. For each run, we estimate the proportion of violation of the decision using 100 inner repetitions of  $\xi_t$ , at each step  $t \in \mathcal{S}$ . Figure 1 depicts the box-plots showing the distribution of the proportion of violation. For all the tested budget levels and choices of  $\mathcal{S}$ , CCTS was able to maintain the proportion of budget violation well below the 10% tolerance at the relevant steps. This implies, moreover, that the overall violation (i.e., at least

one violation at a step in  $\mathcal{S}$ ) is also below 10%.

Note that the theoretical guarantees studied in the previous sections focus on the feasibility in maintaining the chance constraints. In practice, the objective value performance is also important. To test this, we compare the performance of CCTS, both regarding budget violation and revenue attainment, against the following algorithms: 1) a hypothetical algorithm that assumes the distributions of the costs and revenues are all known and draws Monte Carlo samples from it, otherwise the same as CCTS; 2) the deterministically constrained Thompson sampling (DCTS) in Algorithm 3; 3) the algorithm in Badanidiyuru et al. (2013) that uses reward-to-cost ratios; and 4) Besbes and Zeevi (2012) that uses an initial learning phase (in our experiment we set the learning phase to 50 steps). Figure 2 shows the distributions of the proportion of violations at  $t \in \mathcal{S}_2$  based on 500 simulation runs, for the five described budget levels. We see that only CCTS and the hypothetical algorithm maintains the proportion of violation to well below 10%, whereas the other three algorithms fluctuate around 20-40%, as they do not account for the chance constraint on the per-round budget. On the other hand, Figure 3 shows the average cumulative revenue achieved through the horizon (the bar depicts one standard deviation around the average). DCTS appears the best in terms of cumulative rewards, and Badanidiyuru et al. (2013) and Besbes and Zeevi (2012) perform similarly. CCTS achieves less rewards (around 15%), which can be viewed as the price of maintaining the chance constraint. The hypothetical algorithm performs better than CCTS, not surprisingly given the full distributional knowledge. This behavior persists for  $\mathcal{S}_1$  and  $\mathcal{S}_3$  as well as for several priors we have tested (similar to plots Figures 2 and 3). Thus, in view of achieving overall performances in terms of both controlling violation proportions and attaining cumulative rewards, our CCTS appears to be superior to all the other considered methods.

The above experiments all used the budget violations calculated using the true underlying distribution of the cost. In Table 1, we compare with the calculation based on the evolving posterior distributions, in terms of the average of all the proportions of violations for  $t \in \mathcal{S}_1, \mathcal{S}_2$  and  $\mathcal{S}_3$  at the five described budget levels. The average proportion of violation based on the posterior distribution is consistently lower than the one based on the true distribution for all budget levels and  $\mathcal{S}$ . This is expected since our chance constraint is maintained under the posterior distribution (as Theorem 3 states). However, the proportion of violations is maintained below 10% under the true distribution, thanks to the relatively robust performance of our approach in abiding with the chance constraint.

Table 1: Comparison of the average proportions of violations based on the true distribution and the updated posterior distributions over 500 simulation runs

Budget		$\mathcal{S}_1$	$\mathcal{S}_2$	$\mathcal{S}_3$
$B_1$	True Dist.	0.018	0.019	0.018
	Posterior Dist.	0.011	0.012	0.009
$B_2$	True Dist.	0.016	0.015	0.014
	Posterior Dist.	0.01	0.01	0.008
$B_3$	True Dist.	0.014	0.013	0.013
	Posterior Dist.	0.008	0.009	0.007
$B_4$	True Dist.	0.013	0.011	0.01
	Posterior Dist.	0.009	0.007	0.005
$B_5$	True Dist.	0.008	0.008	0.007
	Posterior Dist.	0.006	0.005	0.004

Alternatively, we also investigated the amount of budget violation at  $t \in \mathcal{S}$ . Figure 4 depicts the distributions of the total amounts of budget violation in  $\mathcal{S}_1, \mathcal{S}_2$  and  $\mathcal{S}_3$  for the five budget levels over 500 simulation runs. Similar to the proportion of budget constraint violations, the average amounts of violation for CCTS and DTS tend to be much lower than the other three algorithms, with at most 25% of those of the other three algorithms in the same setting. This suggests a strong dependence between the proportion and the amount of violation which substantiates the use of CCTS in maintaining over-spending even in the monetary scale.

## 5 CONCLUSION

We studied sequential learning subject to constraints that need to be satisfied with high probability. We investigated a methodology to obtain posterior statistical guarantees for the feasibility of these constraints, by generalizing the constraint sampling approach in chance-constraint programming to a two-level Monte Carlo procedure and analyzing the sample size needed in achieving overall feasibility through the learning horizon. We further incorporated our scheme into Thompson sampling using an online advertisement example, and numerically demonstrated how it led to desirable performances in both feasibility and optimality. As far as we know, this work represents the first methodological investigation of “soft” stochastic constraints in sequential learning. In subsequent work, we will investigate the tightening of the requirements in sampled constraints, via for instance analyzing the correlation among decisions at different stages, and will also study the scalability of this approach to higher-dimensional problems.



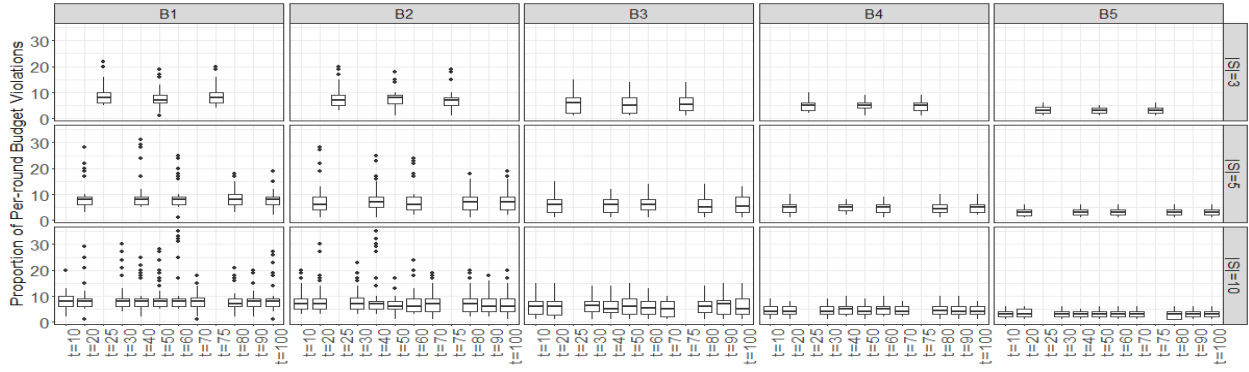


Figure 1: Estimated proportion of violation at each step in  $\mathcal{S}$ , for  $\mathcal{S} = \mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$  and five different budget levels, using 500 simulation runs under CCTS.

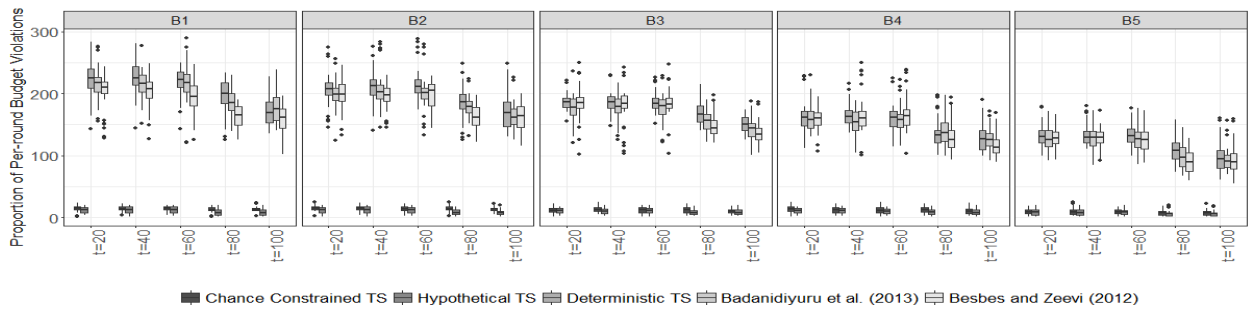


Figure 2: Estimated proportion of violation at each step in  $\mathcal{S}_2$ , with 500 simulation runs for different algorithms

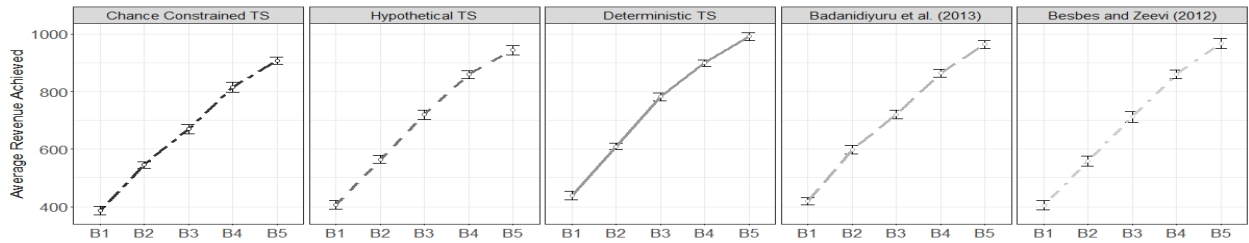


Figure 3: Average cumulative revenue over  $T = 100$  achieved by different algorithms under  $\mathcal{S}_2$

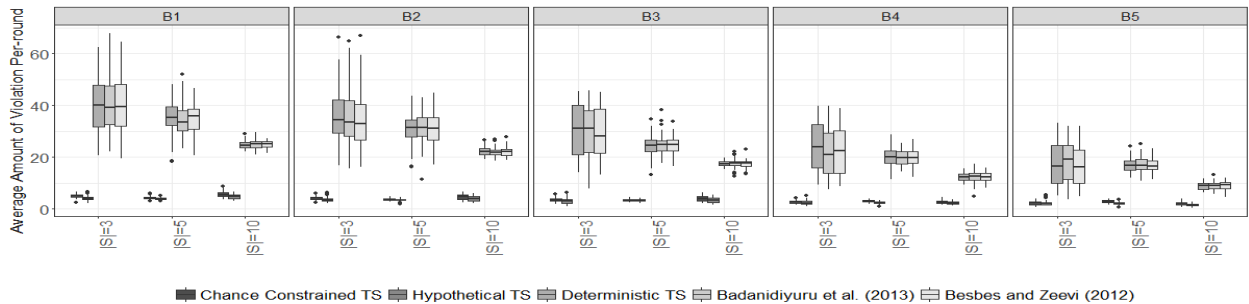


Figure 4: Total Amount of budget violations occurred over  $\mathcal{S}_1, \mathcal{S}_2$  and  $\mathcal{S}_3$  for different algorithms

### Acknowledgements

Support from the Adobe Faculty Research Award is gratefully acknowledged.

### References

Agrawal, S. and Goyal, N. (2012). Analysis of thompson sampling for the multi-armed bandit problem. In

- COLT*, pages 39.1–39.26.
- Amin, K., Kearns, M., Key, P., and Schwaighofer, A. (2012). Budget optimization for sponsored search: Censored learning in mdps. *arXiv preprint arXiv:1210.4847*.
- Audibert, J.-Y. and Bubeck, S. (2010). Best arm identification in multi-armed bandits. In *COLT*, pages 41–53.
- Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256.
- Badanidiyuru, A., Kleinberg, R., and Slivkins, A. (2013). Bandits with knapsacks. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 207–216. IEEE.
- Ben-Tal, A. and Nemirovski, A. (2000). Robust solutions of linear programming problems contaminated with uncertain data. *Mathematical programming*, 88(3):411–424.
- Bertsimas, D., Gupta, V., and Kallus, N. (2013). Data-driven robust optimization. *arXiv preprint arXiv:1401.0212*.
- Besbes, O. and Zeevi, A. (2012). Blind network revenue management. *Operations Research*, 60(6):1537–1550.
- Boesel, J., Nelson, B. L., and Kim, S.-H. (2003). Using ranking and selection to clean up after simulation optimization. *Operations Research*, 51(5):814–825.
- Calafiore, G. and Campi, M. C. (2005). Uncertain convex programs: randomized solutions and confidence levels. *Mathematical Programming*, 102(1):25–46.
- Calafiore, G. C. (2017). Repetitive scenario design. *IEEE Transactions on Automatic Control*, 62(3):1125–1137.
- Calafiore, G. C. and Campi, M. C. (2006). The scenario approach to robust control design. *IEEE Transactions on Automatic Control*, 51(5):742–753.
- Calafiore, G. C. and El Ghaoui, L. (2006). On distributionally robust chance-constrained linear programs. *Journal of Optimization Theory and Applications*, 130(1):1–22.
- Campi, M. C. and Garatti, S. (2008). The exact feasibility of randomized solutions of uncertain convex programs. *SIAM Journal on Optimization*, 19(3):1211–1230.
- Campi, M. C. and Garatti, S. (2011). A sampling-and-discarding approach to chance-constrained optimization: feasibility and optimality. *Journal of Optimization Theory and Applications*, 148(2):257–280.
- Carè, A., Garatti, S., and Campi, M. C. (2014). Fast-fast algorithm for the scenario technique. *Operations Research*, 62(3):662–671.
- Chamanbaz, M., Dabbene, F., Tempo, R., Venkataramanan, V., and Wang, Q.-G. (2016). Sequential randomized algorithms for convex optimization in the presence of uncertainty. *IEEE Transactions on Automatic Control*, 61(9):2565–2571.
- Charnes, A., Cooper, W. W., and Symonds, G. H. (1958). Cost horizons and certainty equivalents: an approach to stochastic programming of heating oil. *Management Science*, 4(3):235–263.
- De Farias, D. P. and Van Roy, B. (2004). On constraint sampling in the linear programming approach to approximate dynamic programming. *Mathematics of Operations Research*, 29(3):462–478.
- Ding, W., Qin, T., Zhang, X.-D., and Liu, T.-Y. (2013). Multi-armed bandit with budget constraint and variable costs. In *AAAI*, pages 232–238.
- Ferreira, K. J., Simchi-Levi, D., and Wang, H. (2016). Online network revenue management using thompson sampling. *Working paper*.
- Lagoa, C. M., Li, X., and Sznaiier, M. (2005). Probabilistically constrained linear programs and risk-adjusted controller design. *SIAM Journal on Optimization*, 15(3):938–951.
- Lai, T. L. and Robbins, H. (1985). Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22.
- Lejeune, M. A. and Ruszczyński, A. (2007). An efficient trajectory method for probabilistic production-inventory-distribution problems. *Operations Research*, 55(2):378–394.
- Luedtke, J., Ahmed, S., and Nemhauser, G. L. (2010). An integer programming approach for linear programs with probabilistic constraints. *Mathematical Programming*, 122(2):247–272.
- Miller, B. L. and Wagner, H. M. (1965). Chance constrained programming with joint constraints. *Operations Research*, 13(6):930–945.
- Pani, A., Raghavan, S., and Sahin, M. (2017). Large-scale advertising portfolio optimization in online marketing.
- Prékopa, A. (2003). Probabilistic programming. *Handbooks in operations research and management science*, 10:267–351.
- Rossi, R., Hnich, B., Tarim, S. A., and Prestwich, S. (2011). Finding  $(\alpha, \vartheta)$ -solutions via sampled scsp. In *IJCAI*, pages 2172–2177.
- Rossi, R., Hnich, B., Tarim, S. A., and Prestwich, S. (2015). Confidence-based reasoning in stochastic constraint programming. *Artificial Intelligence*, 228:129–152.

- Russo, D. and Van Roy, B. (2016). An information-theoretic analysis of thompson sampling. *The Journal of Machine Learning Research*, 17(1):2442–2471.
- Shi, Y., Zhang, J., and Letaief, K. B. (2015). Optimal stochastic coordinated beamforming for wireless cooperative networks with csi uncertainty. *IEEE Transactions on Signal Processing*, 63(4):960–973.
- Tran-Thanh, L., Chapman, A., Rogers, A., and Jennings, N. R. (2012). Knapsack based optimal policies for budget-limited multi-armed bandits. In *AAAI*, pages 1134–1140.
- Tran-Thanh, L., Stavrogiannis, L., Naroditskiy, V., Robu, V., Jennings, N. R., and Key, P. (2014). Efficient regret bounds for online bid optimisation in budget-limited sponsored search auctions. In *UAI*, pages 809–818.
- Villar, S. S., Bowden, J., and Wason, J. (2015). Multi-armed bandit models for the optimal design of clinical trials: benefits and challenges. *Statistical science: a review journal of the Institute of Mathematical Statistics*, 30(2):199.
- Wang, Z., Deng, S., and Ye, Y. (2014). Close the gaps: A learning-while-doing algorithm for single-product revenue management problems. *Operations Research*, 62(2):318–331.
- Xia, Y., Li, H., Qin, T., Yu, N., and Liu, T.-Y. (2015). Thompson sampling for budgeted multi-armed bandits. In *IJCAI*, pages 3960–3966.
- Zymler, S., Kuhn, D., and Rustem, B. (2013). Distributionally robust joint chance constraints with second-order moment information. *Mathematical Programming*, pages 1–32.

---

# Abstraction Sampling in Graphical Models

---

F. Broka, R. Dechter, A. Ihler and K. Kask

University of California, Irvine

Irvine, CA 92697, USA

{fbroka, dechter, ihler, kkask}@ics.uci.edu

## Abstract

We present a new sampling scheme for approximating hard to compute queries over graphical models, such as computing the partition function. The scheme builds upon exact algorithms that traverse a weighted directed state-space graph representing a global function over a graphical model (e.g., probability distribution). With the aid of an abstraction function and randomization, the state space can be compacted (or trimmed) to facilitate tractable computation, yielding a Monte Carlo Estimate that is unbiased. We present the general scheme and analyze its properties analytically and empirically, investigating two specific ideas for picking abstractions - targeting reduction of variance or search space size.

## 1 INTRODUCTION

Imagine that we want to compute a function over a weighted directed graph where the graph is given implicitly, e.g., using a generative state-space search model whose explicit state graph is enormous. We therefore need to resort to approximations such as Monte-Carlo schemes. We focus on weighted search trees over probabilistic graphical models where nodes represent partial variable assignments (or configurations) and arcs are associated with weights describing probabilistic quantities from the model. The task is to compute the partition function - sum cost of all paths in the tree. Some Monte Carlo methods draw independent samples of full configurations (full paths) one variable at a time (e.g., Forward sampling). The key idea of the scheme we propose is that, guided by an abstraction function, each sample (or a “probe”) is a sampled subtree representing *multiple configurations*. We argue that, under some conditions over

the abstraction, such a sampled tree representing  $k$  configurations can be a more accurate estimator than  $k$  independent samples.

Our sampling scheme uses an *abstraction function* that partitions the nodes *at each level in the search tree* into subsets of *abstract states* under the intuition that nodes in the same abstract state root similar subtrees and therefore a single member from each can represent all others. Our *Abstraction Sampling (AS)* scheme brings together ideas from statistics and search to exploit their respective strengths. Search is a systematic process that can explore all configurations in a structured manner, *once*. “It does not leave any stone unturned and does not turn any stone more than once” [Pearl, 1984]. Sampling on the other hand explores only a subset of the paths, that can stochastically cover the whole search space. Abstraction Sampling allows a transition between search and sampling by generating and searching a subtree, and therefore a corresponding subset of configurations, in a coordinated manner that can overcome redundancy and some of the variance associated with random independent samples.

From a search perspective abstraction can be viewed as a license to merge nodes that root similar subtrees, sampling one of the subtrees, thus creating a more compact graph that can be searched more efficiently. From a sampling perspective, an abstract state can be viewed as a form of a “strata” within a stratified sampling scheme [Rubinstein and Kroese, 2007, Rizzo, 2007] where the process of stratified sampling is applied layer by layer. The variance reduction we hope to get rests on similar principles of variance reduction in stratified sampling, as we will elaborate more later.

Abstraction Sampling is inspired by the early work of Knuth and Chen [Knuth, 1975, Chen, 1992] who proposed a method for estimating quantities that can be expressed as aggregates (e.g., sum) of functions defined over the nodes in the graph. Our work extends this work to more general queries over graphical models such as

the partition function and to weighted AND/OR trees.

**Our Contributions.** In this paper we present a new algorithmic framework, Abstraction Sampling, that combines search with stratified importance sampling for answering summation queries (e.g., partition function) over graphical models. Using the notion of abstraction function we can find a cost-effective balance between search and sampling, that is tuned to the problem instance and to time and memory resources, using OR or AND/OR search algorithms. We prove unbiasedness and discuss variance reduction properties, and carry out an extensive empirical evaluation over multiple challenging benchmarks. Our results show that two classes of context-based abstractions (deterministic and randomized) can often significantly improve performance over the baseline of importance sampling, for both OR and AND/OR trees. We also show that our scheme is competitive by comparing it with two state-of-the-art importance sampling schemes.

## 2 BACKGROUND

A graphical model, such as a Bayesian or a Markov network [Pearl, 1988, Darwiche, 2009, Dechter, 2013] can be defined by a 3-tuple  $\mathcal{M} = (\mathbf{X}, \mathbf{D}, \mathbf{F})$ , where  $\mathbf{X} = \{X_i : i \in V\}$  is a set of variables indexed by a set  $V$  and  $\mathbf{D} = \{D_i : i \in D\}$  is the set of finite domains of values for each  $X_i$ . Each function  $\psi_\alpha \in \mathbf{F}$  is defined over a subset of the variables called its scope,  $X_\alpha$ , where  $\alpha \subseteq V$  are the indices of variables in its scope and  $D_\alpha$  denotes the Cartesian product of their domains, so that  $\psi_\alpha : D_\alpha \rightarrow R^{\geq 0}$ . The **primal graph** of a graphical model associates each variable with a node, while arcs connect nodes whose variables appear in the scope of the same local function. A graphical model represents a global function, often a probability distribution, defined by  $Pr(X) \propto \prod_\alpha \psi_\alpha(X_\alpha)$ . An important task is to compute the normalizing constant, also known as the partition function  $Z = \sum_X \prod_\alpha \psi_\alpha(X_\alpha)$ .

**AND/OR search spaces.** A graphical model can be expressed via a weighted state space graph. In a simple OR search space, the states (or nodes) are partial assignments relative to a variable ordering, where each layer corresponds to a new assigned variable. A graphical model can be transformed into a more compact AND/OR search space [Dechter and Mateescu, 2007] by capturing conditional independences, thus facilitating more effective algorithms [Marinescu and Dechter, 2009]. The AND/OR search space is defined relative to a *pseudo tree* of the primal graph.

**DEFINITION 1 (pseudo tree)** A pseudo tree of an undirected graph  $G = (V, E)$  is a directed rooted tree  $\mathcal{T} = (V, E')$  such that every arc of  $G$  not in  $E'$  is a back-arc

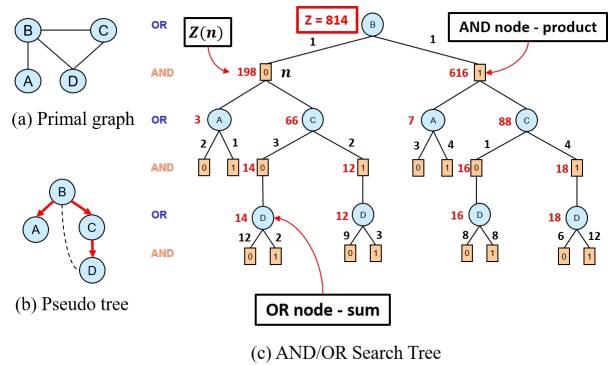


Figure 1: A Simple Graphical Model.

in  $\mathcal{T}$  connecting a node in  $\mathcal{T}$  to one of its ancestors. The arcs in  $E'$  may not all be included in  $E$ .

Given a pseudo tree  $\mathcal{T}$  of a primal graph  $G$ , the *AND/OR search tree*  $T_{\mathcal{T}}$  guided by  $\mathcal{T}$  has alternating levels of OR nodes corresponding to the variables, and AND nodes corresponding to an assignment from its domain with edge costs extracted from the original functions  $\mathbf{F}$  [Dechter and Mateescu, 2007]. Let  $s$  be a node in  $T_{\mathcal{T}}$ . We denote by  $var(s)$  the last variable of the partial value assignment associated with  $s$ . So if  $s$  stands for  $\bar{x}_{1..p} = (x_1, x_2, \dots, x_p)$ , then  $var(s) = X_p$ . Each AND node  $s$  has a cost  $c(s)$  defined to be the product of all factors  $\psi_\alpha$  that are instantiated at  $s$  but not before.

**DEFINITION 2 (solution subtree)** A solution subtree  $\hat{x}_M$  is a subtree of  $T_{\mathcal{T}}$  satisfying: (1) it contains the root of  $T_{\mathcal{T}}$ ; (2) if an OR node is in  $\hat{x}_M$ , exactly one of its AND child nodes is in  $\hat{x}_M$ ; (3) if an AND node is in  $\hat{x}_M$  then all its OR children are in  $\hat{x}_M$ . The product of arc-costs on any full solution tree equals the cost of a full configuration of the model  $\mathcal{M}$ .

**Example 1** Figure 1a is a primal graph of 4 bi-valued variables and 4 binary factors of a graphical model. Figure 1b is a pseudo tree. Figure 1c displays the AND/OR search tree guided by the pseudo tree. A solution subtree is  $(B = 1, A = 0, C = 1, D = 0)$  having a cost of 72.

Each node  $s$  in  $T_{\mathcal{T}}$  can be associated with a *value*,  $Z(s)$  expressing the conditioned partition function rooted at  $s$ . Clearly,  $Z(s)$  can be computed recursively based on the values of the children of  $s$ : OR nodes by summation and AND nodes by multiplication. The value of  $T_{\mathcal{T}}$ , is the value of its root node, which is the partition function of the underlying model,  $\mathcal{M}$ .

The size of the AND/OR search tree,  $T_{\mathcal{T}}$  is exponential in the height of the pseudo tree. It is possible to merge some subtrees using a concept known as *con-*

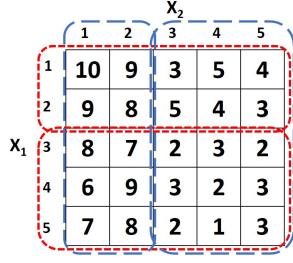


Figure 2: Motivating Example;  $Z=126$ .

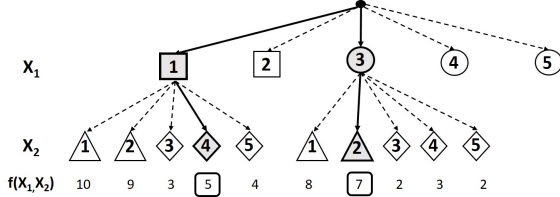


Figure 3: Motivating Example Tree

text (defined later), yielding an AND/OR graph that is exponential in the tree-width of the primal graph [Dechter and Mateescu, 2007]. As noted,  $Z(s)$  can be computed by a depth-first search scheme from leaves to root of the AND/OR tree (See Figure 1 where the values attached to each node). When the pseudo tree is a chain we get back the regular OR tree, where each path corresponds to a full variable configuration.

**Stratified sampling** is a variance reduction technique that can be used with **importance sampling** [Rubinstein and Kroese, 2007] [Liu *et al.*, 2015] [Gogate and Dechter, 2011] to achieve further reduction in variance. Let  $f$  be a non-negative function defined on a sample space  $\mathbf{X}$ . We want to estimate the value  $Z = \sum_{\mathbf{x} \in \mathbf{X}} f(\mathbf{x})$ . In importance sampling, we can estimate  $Z$  by drawing samples from  $\mathbf{X}$  according to a proposal distribution  $q$  and estimating the equivalent expression  $Z = \sum_{\mathbf{x} \in \mathbf{X}} \frac{f(\mathbf{x})}{q(\mathbf{x})} q(\mathbf{x})$ , leading to an importance sampling estimator  $\hat{Z}^I$ . In stratified importance sampling, we first divide the sample space  $\mathbf{X}$  into  $k$  strata of equal area under the distribution  $q$ . Let  $J$  be a random variable with  $k$  values  $\{1, \dots, k\}$  assigning to each configuration in the sample space the index of its strata. For each  $j \in J$ , we compute importance sampling estimators  $\hat{Z}_j^I$  of  $Z_j = \sum_{\mathbf{x} \in \mathbf{X}} \frac{f_j(\mathbf{x})}{q_j(\mathbf{x})} q_j(\mathbf{x})$  from samples drawn from the conditional distributions  $q_j = q(J=j)$  and  $f_j(\mathbf{x})$  be defined as  $f(\mathbf{x})$  if  $J(\mathbf{x}) = j$  and 0 otherwise. The stratified importance sampling estimator is  $\hat{Z}^{SI} = \frac{1}{k} \sum_{j=1}^k \hat{Z}_j^I$ . It can be shown

**THEOREM 1 ([Rizzo, 2007])** Let  $Z_j$  be a uniform ran-

dom variable defined on  $\{1, \dots, k\}$  assigning value  $Z_j$  to  $j$ . Let  $\hat{Z}^{SI}$  be the stratified importance sampling estimator computed using  $m$  samples drawn in each of the  $k$  strata. Let  $\hat{Z}^I$  be the importance sampling estimator computed with  $M = mk$  samples. The variance reduction achieved by moving from  $\hat{Z}^I$  to  $\hat{Z}^{SI}$  is  $\frac{k}{m} \text{Var}(Z_J)$ .

### 3 ABSTRACTION SAMPLING

**A Motivating Example.** Our proposed Abstraction Sampling (AS) algorithm emulates stratified importance sampling on partial configurations (nodes in the search tree) layer by layer (each layer corresponding to a variable). To illustrate the idea consider a small two dimensional function over variables  $X_1$  and  $X_2$  with domains  $D_1 = D_2 = \{1, 2, 3, 4, 5\}$  in Figure 2. Partial configurations of length 1 (corresponding to  $X_1$ ) are partitioned into two abstract states: one where  $X_1 \in \{1, 2\}$  and the other where  $X_1 \in \{3, 4, 5\}$ . Note that this abstraction function tries to put into the same abstract states rows which have roughly the same total mass (sum of entries). At the level of  $X_2$  we have abstract states defined just by variable  $X_2$  - one where  $X_2 \in \{1, 2\}$  and the other by  $X_2 \in \{3, 4, 5\}$ . We assume a uniform proposal distribution; note that the exact answer is  $Z = 126$  (summing all entries in the table).

The sampling process is illustrated in Figure 3. Assume we pick  $X_1 = 1$  and  $X_1 = 3$  as representatives of their respective abstract states, and give them weights 2 and 3, to account for the number of states they represent. All extensions of these 2 assignments to  $X_2$  are generated, with abstract states indicated as diamonds and triangles. There are now 4 candidates for a representative of triangle abstraction with values  $\{10, 9, 8, 7\}$  and corresponding weights  $\{2, 2, 3, 3\}$ , and 6 diamond candidates with values  $\{3, 5, 4, 2, 3, 2\}$  and corresponding weights  $\{2, 2, 2, 3, 3, 3\}$ . We select a representative at random in proportion to their weights. Assume this yields configurations  $(X_1 = 3, X_2 = 2)$  and  $(X_1 = 1, X_2 = 4)$ . We next update the weights of the selected representatives to account for the mass they represent yielding:  $w(X_1 = 3, X_2 = 2) \leftarrow w(X_1 = 3, X_2 = 2)/(3/10) = 3/(3/10) = 10$  and  $w(X_1 = 1, X_2 = 4) \leftarrow w(X_1 = 1, X_2 = 4)/(2/15) = 2/(2/15) = 15$ , yielding an estimate for the partition function  $\hat{Z} = 7 \cdot 10 + 5 \cdot 15 = 145$ .

We also compared the empirical variance of our estimator with a simple importance sampler, using 100000 trials and observed variance reduction from 2337 to 398, an almost 6 fold decrease. While in this example the abstract states were not equal size, the abstractions were selected to yield a large variance between the different abstract states.

---

**Algorithm 1: OR Abstraction Sampling(AS) one probe**


---

**Require:** An implicit OR search tree  $T$  of a model  $\mathcal{M}$ .

$c(s, s')$  is the cost of arc  $(s, s')$  extracted from  $\mathcal{M}$ .  
 $g(s)$  is the product of arc-costs from root to  $s$  and  
 $h(s)$  (heuristic function) is an upper bound on the  
partition function defined on nodes based on  
graphical model  $\mathcal{M}$ . Abstraction  $a$ .

**Ensure:** Sampled tree  $\tilde{T}$ . Estimate  $\hat{Z} = Z(\tilde{T})$ .

- 1: initialize  $\tilde{T} \leftarrow \emptyset, \hat{Z} \leftarrow 0, OPEN \leftarrow \{ \langle s_0, 1 \rangle \}$ ,
  - 2: **while** OPEN is not empty **do**
  - 3:    $\langle s, w(s) \rangle \leftarrow$  remove node in OPEN with  
smallest  $a$
  - 4:   **if**  $s$  is a leaf node in  $T$  **then**
  - 5:      $\hat{Z} \leftarrow \hat{Z} + w(s) \cdot g(s)$
  - 6:   **else**
  - 7:     **for** each child  $s'$  of  $s$  **do**
  - 8:        $w(s') \leftarrow w(s)$
  - 9:       **if**  $a(s') = i$  and OPEN contains a node  
 $\langle s_{\{i\}}, w_{\{i\}} \rangle$  of abstraction  $\{i\}$  **then**
  - 10:          $p \leftarrow \frac{w(s')g(s')h(s')}{w(s')g(s')h(s') + w_{\{i\}}g(s_{\{i\}})h(s_{\{i\}})}$
  - 11:          $w_{\{i\}} \leftarrow \frac{w_{\{i\}}}{(1-p)}$
  - 12:         **with** probability  $p$  **do:**
  - 13:         remove  $s_{\{i\}}$  from OPEN
  - 14:          $OPEN \leftarrow OPEN \cup \{ \langle s', \frac{w(s')}{p} \rangle \}$
  - 15:         add  $s'$  to  $\tilde{T}$
  - 16:     **else**
  - 17:        $OPEN \leftarrow OPEN \cup \{ \langle s', w(s') \rangle \}$
  - 18:  $\tilde{T} \leftarrow \tilde{T} \cup \{ \langle s, w(s) \rangle \}$
- 

### 3.1 THE ALGORITHM

Our proposed Abstraction Sampling algorithm is a Monte Carlo process that generates compact representatives  $\tilde{T}$  of  $T$ , guided by an *abstraction function*. Abstraction Sampling for OR trees (Algorithm 1) builds a subtree  $\tilde{T}$  of  $T$ , level-by-level, in a breadth-first manner. Starting from the root of  $T$ , at each step, it picks a leaf node having the smallest (as explained next) abstract value, and expands it to its child nodes. Each node  $s$  is associated with a weight  $w(s)$  representing the mass the abstract state stands for (root weight is 1). For each newly generated node, the algorithm checks if there already exists a node having the same abstraction. If this is the case, (line 9), it decides with some probability  $p$ , which of the representative nodes to keep and which to discard. The weight of the selected representative is adjusted to account for the discarded one (step 11). Otherwise, it adds the new node to the frontier of nodes called *OPEN* with the weight of its parent node (step 17).

**Layered Abstractions.** Given a weighted directed AND/OR tree  $T$ , an abstraction function,  $a : T \rightarrow I^+$ , where  $I^+$  are integers, partitions the nodes in  $T$ , layer by layer, with the requirements that i) if  $var(s_1) \neq var(s_2) \rightarrow a(s_1) \neq a(s_2)$ , and ii) if  $s' \in ch(s)$  then  $a(s) < a(s')$  (this enforces breadth-first exploration;  $ch(s)$  denotes the children of  $s$  in  $T$ ). Abstraction states are denoted by  $\{i\}$  for an integer  $i$ .

**The Sampling Proposal.** We use  $p$  proportional to  $w(s) \cdot g(s) \cdot h(s)$ , where  $h$  is a heuristic that provides an upper bound on the partition function (of the subproblem rooted at  $s$ ). While the algorithm is unbiased for any sampling probability, the heuristic yields a proposal function whose accuracy can significantly impact its convergence, as in any importance sampling scheme.

**Example 2** Consider the (OR) search tree  $T$  in Fig. 4a. The cost of each solution is obtained by a product on its solution arcs. In Figure 4b we show a probe generated via an abstraction function that puts all nodes that represent a single variable in a single abstract state. In 4c, we see a probe where each domain value of a variable corresponds to a different abstract state, yielding 2 states per variable, and thus 2 nodes per level of the generated tree. The estimate for (b) is  $\hat{Z} = 180$  and for (c)  $\hat{Z} = 156$ .

**Abstraction Sampling for AND/OR trees** requires several modifications. The underlying search tree is an AND/OR tree  $T_{\mathcal{T}}$  along a pseudo-tree  $\mathcal{T}$ , and the abstraction function is defined on AND nodes. The algorithm builds a subtree  $\tilde{T}_{\mathcal{T}}$  of  $T_{\mathcal{T}}$ , level-by-level, breadth first. At each step, it picks a leaf AND node having the smallest abstraction, and expands two levels down - child OR nodes, and their child AND nodes. For newly gen-

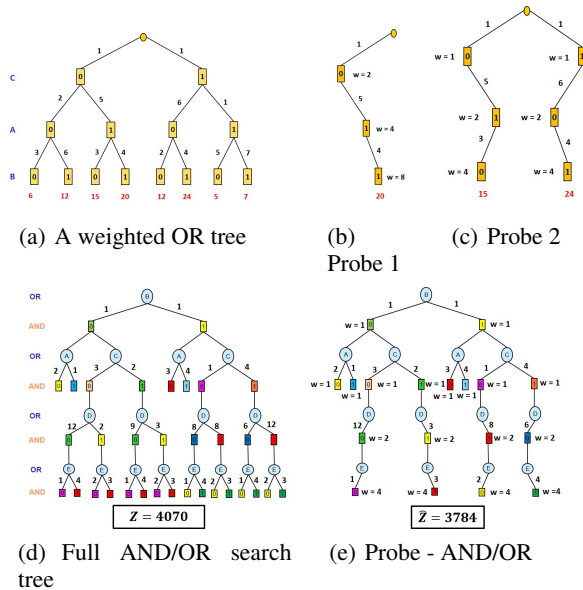


Figure 4: Example of Probes

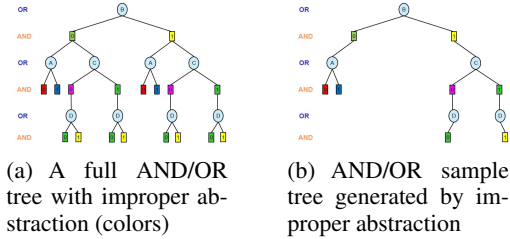


Figure 5: Improper Abstraction for AND/OR

erated AND nodes, the algorithm proceeds as in the OR case to select stochastically the representative node. The estimate is not accumulated during the sampling process. Rather, once a probe  $\hat{T}_{\mathcal{T}}$  is generated, its partition function estimate can be computed in a depth-first manner. Figures 4d and 4e provide an example AND/OR search tree and a possible probe.<sup>1</sup>

### 3.2 PROPER ABSTRACTIONS

To guarantee the validity and unbiasedness of our sampling scheme for general AND/OR trees, we need to ensure that the sampled AND/OR probe would include only full solution subtrees of the underlying AND/OR tree. For example, the sampled subtree in Figure 5b could have been generated from the tree in Figure 5a with the colors as abstraction, yet it contains a solution tree that corresponds just to a partial configurations (e.g.  $(B=0, A=0)$ ) and therefore may lead to unbiased estimates. This situation can be avoided if we require abstractions to be *proper*, i.e. any two AND nodes of the same variable,  $X$ , can have the same abstract state, only if they are descendant of a common AND node in the sampled AND/OR tree.

#### DEFINITION 3 (Branching variable, Proper abstraction)

A variable  $Y$  in a pseudo-tree  $\mathcal{T}$  is a branching variable of  $X$ , if  $Y$  is  $X$ 's closest ancestor with at least 2 child nodes. An abstraction function  $a$  over AND nodes in  $T_{\mathcal{T}}$  is proper if for any two AND nodes  $n_1$  and  $n_2$  having  $\text{var}(n_1) = \text{var}(n_2) = X$ , if  $a(n_1) = a(n_2)$ , then  $n_1$  and  $n_2$  have a common ancestor AND node  $n_3$  s.t.  $\text{var}(n_3) = Y$  where  $Y$  is the branching variable of  $X$ .

Clearly, any OR tree abstraction is proper as there are no branching variables. Abstraction Sampling for AND/OR trees enforces the generation of *proper* probes during the sampling process (for details see AND/OR algorithm in

<sup>1</sup>Note that while the AND/OR-AS algorithm is defined here as a *breadth-first* (BF) traversal of the search space, other formulations are possible. Our AND/OR-AS implementation uses BF traversal on non-branching variables (chains) and *depth-first* (DF) traversal on branching variables, due to superior time/space complexity of DFS algorithms.

supplementary material). For the remainder of the paper we assume that AS is the general algorithm extended to AND/OR spaces that enforces the proper condition.

## 4 PROPERTIES

**Complexity** The *proper* restriction limits compactness of the sampled AND/OR trees. More branchings in the pseudo tree yield more abstract states and consequently larger probes. We can show that:

**THEOREM 2 (size and complexity)** Given a pseudo-tree, the number of states in a probe by AS is  $O(n \cdot m^{b+1})$ , where  $n$  is the number of variables,  $b$  bounds the number of branchings along any path of the pseudo-tree and  $m$  bounds the maximum number of abstract states in the input abstraction function,  $a$ , per variable. Clearly for OR trees,  $b = 0$  yielding size bounded by  $O(nm)$ .

Notice that when we have many branchings in the pseudo tree  $\mathcal{T}$ , the underlying AND/OR tree from which we sample is far more compact than the underlying OR tree.

**Unbiasedness** We can show that our sampling scheme generates an unbiased estimate of the partition function. Detailed proof is in the supplementary materials.

**THEOREM 3 (unbiasedness)** Given a weighted directed AND/OR search tree  $T$  derived from a graphical model, the estimate  $\hat{Z}$  generated by AS is unbiased.

## 5 ON VARIANCE AND ABSTRACTION SELECTION

The proof of unbiasedness works for any sampling distribution  $p$ . The reason for choosing our specific proposal probabilities is to reduce the variance. We can show that

**THEOREM 4 (exact proposal)** If the proposal function  $p$  in AS uses an exact heuristic  $h(n) = Z(n)$ , then  $\hat{Z}$  has zero variance (single probe is exact), for any abstraction.

Theorem 4 addresses the extreme case when the proposal is exact. Next we talk about the other extreme when the abstraction is exact.

**THEOREM 5 (exact abstraction)** When the abstraction function satisfies that  $a(n) = a(n') \Rightarrow Z(n) = Z(n')$  then,  $\hat{Z}$  is exact (i.e.  $\hat{Z} = Z$ ) with one probe, if  $h$  satisfies  $a(n) = a(n') \Rightarrow h(n) = h(n')$ .

Since AS is a type of stratified importance sampling our **sampling perspective**, (e.g., Theorem 1), suggests several variance reduction ways. The first advises to use



abstractions that at each layer partition nodes into equal area abstract states under the proposal. The second is advising towards increasing the variance of the estimators *between abstract states*. The third encourages having more refined abstractions with more abstract states per layer, as long as the condition of equal size abstract states can be maintained. At the same time, our **search perspective** suggests to always unify nodes that root the same subtrees, whenever such information is available. One possibility is to use the notion of *context*.

The **context** of a variable  $X$  in a pseudo-tree  $\mathcal{T}$  identifies a subset  $C(X)$  of its ancestor variables, whose assignment uniquely determines the AND/OR subtree below the node ([Dechter and Mateescu, 2007]). Therefore, two nodes in the search tree having the same context, (namely, the same assignment along their contexts) root identical subtrees. So, if we use abstractions that are context-isomorph and if  $h$  is the mini-bucket heuristic, the two conditions of Theorem 5 hold, yielding:

**Corollary 1** *When the abstraction function is context-isomorph, namely,  $a(n) = a(n') \leftrightarrow C(n) = C(n')$  and if  $h$  is a mini-bucket heuristic, then the partition function estimate,  $\hat{Z}$ , is exact.*

In the following we construct two abstraction function families based on the notion of context: relaxed context-based and randomized context-based abstractions.

**DEFINITION 4 (relaxed context-based abstractions)**

*An abstraction  $a$  at  $X$  is context-based relative to a subset  $S$ ,  $S \subseteq C(X)$ , iff for every  $n_1$  and  $n_2$  having  $var(n_1) = var(n_2) = X$ , we have:  $a(n_1) = a(n_2) \leftrightarrow \pi_S C(n_1) = \pi_S C(n_2)$ . If  $|S| = j$  we say that we use a  $j$ -level context-based abstraction. In particular, 0-level abstractions puts all the nodes of a variable in a single abstract state.*

This family of abstract functions will lead to abstract states in each level having the same number of nodes, which can be viewed as a first approximation to abstract states having the same area under the proposal.

Our second family of abstractions introduces randomness into the actual way the abstraction depends on the context. This can facilitate the generation of many abstractions in an automated manner and potentially lead to tighter estimates. We randomly assign nodes into abstract states based on the value of a random hash function, taking into consideration only the context of a node.

**DEFINITION 5 (randomized context-based abstraction)**

*Let  $k \in \mathbb{I}^+$  and  $d \in \mathbb{I}^+$  be parameters. Let  $N$  be the number of variables in the model and  $K = \{1, 2, \dots, k\}$ . We assume that all domains  $D_i$  of the variables are*

*subsets of the positive integers. We construct an abstraction  $a$ , by first sampling a vector  $\mathbf{c} = (c_1, c_2, \dots, c_N)$  uniformly at random from  $K^N$ . Given a node  $n_j$  at level  $j$  in the search tree, corresponding to the partial assignment  $\bar{x}_j = (x_1, x_2, x_3, \dots, x_j)$ , we compute its hash value  $hash(n_j) = \sum_{i=1}^j c_i x_i I_{C(X_j)}(X_i)$ , where  $I$  is the indicator function. We define its abstraction function value  $a(n_j) = hash(n_j) \bmod d$ . The parameter  $d$  determines the number of abstract states for each layer of the tree.*

## 6 EMPIRICAL EVALUATION

### 6.1 METHODOLOGY

**Implementation and Configuration.** We evaluate our algorithm on 4 benchmark sets using several configurations of the abstraction sampling algorithm. We implemented the algorithm in C++ and ran experiments on a 2.66 GHz processor with 2GB of memory. For our heuristic we use Weighted Mini-Bucket Elimination (WMBE) [Dechter and Rish, 2003, Liu and Ihler, 2011], whose strength is controlled by a parameter called the *i*-bound. Higher *i*-bounds lead to stronger heuristics at the expense of higher computation and memory cost. We use *i*-bound 10 in our experiments. While there is an interplay between the heuristic strength and the abstraction level, we defer such investigation to future work.

**Abstraction Functions.** We use the two types of context-based abstractions introduced in the previous section: relaxed context-based (RelCB) and randomized context-based (RandCB). RelCB is parametrized by its level  $j$ , while RandCB by its parameter  $d$ . We compare the more refined abstractions (higher  $j$  or  $d$ ) to the 0-level abstraction, that combines all nodes in a layer into a single abstract state, corresponding to baseline regular importance sampling. For OR trees, using a fixed variable order, we experiment with abstractions having  $j \in \{4, 8\}$  for RelCB, and  $d \in \{16, 256\}$  for RandCB.

For AND/OR trees we introduce hybrid abstractions that allow a better control of probe size. A hybrid abstraction with parameter  $j.k$  ( $d.k$  for RandCB) performs a  $j$ -level abstraction (parameter  $d$  for RandCB) for nodes having no more than  $k$  branching variables in the path to the root, and 0-level abstraction below it. For RelCB abstraction family, we experiment with a pure 1-level abstraction ( $j = 1$ ) and a hybrid abstraction with parameter 2.5. For RndCB family, we experiment with a pure abstraction with  $d = 2$  and a hybrid abstraction with parameter 4.5. For randomized abstractions, we tested three different random seeds and observed overall similar results; we present results from one seed selected randomly.

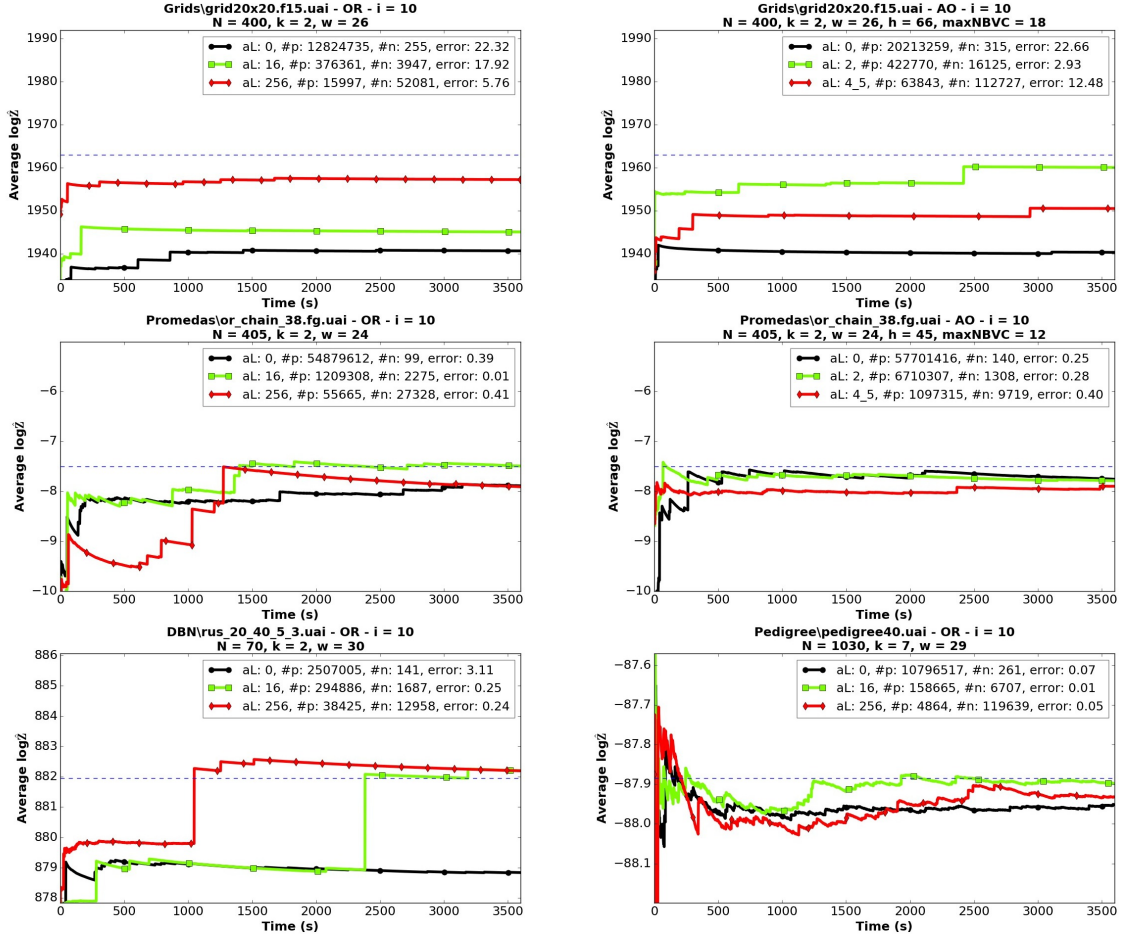


Figure 6: Convergence Plots for Different Abstraction Levels (aL) for Selected Problem Instances.  $\#p$  – number of probes,  $\#n$  – average number of expanded nodes per probe,  $h$  – height of pseudo tree, maxNBVC – max number of branching variables in any path of pseudo tree, error – distance from true value

**Benchmarks.** We tested on instances from 4 benchmarks (BN, DBN, Pedigree, Promedas). We classify instances as small (we do have the exact value of the partition function) or large (we don’t, due to high induced width). We use 130 small and 155 large instances. Summary statistics are in the first column of the results table. Since DBN instances have chain-like pseudo-trees (similar to OR), we test them only on OR trees.

**Performance Measure.** For each instance and configuration (abstraction family, tree type, abstraction level), we ran the AS algorithm for 1 hour and recorded the partition function estimate  $\hat{Z}$  at different times. For small instances (exact  $Z$  known), we compute the log partition function absolute error  $|\log_{10} \hat{Z} - \log_{10} Z|$ . For large instances (exact  $Z$  unknown), we use as a proxy the distance of the log of the estimate from the log of an upper bound of the partition  $Z_{UB}$  (computed using heuristic function)  $|\log_{10} Z_{UB} - \log_{10} \hat{Z}|$ . We present the mean

of these errors aggregated by benchmark in Table 1.

**Comparison with other schemes.** We compare our estimates with two state-of-the-art importance sampling algorithms: Weighted Mini-Bucket Importance Sampling (WMB-IS) [Liu *et al.*, 2015] and IJGP-SampleSearch (IJGP-SS) [Gogate and Dechter, 2011]. We use original implementations from the authors and set the common i-bound parameter to 10. For our algorithms and WMB-IS, we use the same fixed variable order, while IJGP-SS computes its own variable order.

The goal of WMB-IS algorithms [Lou *et al.*, 2017a], [Lou *et al.*, 2017b] is to provide (deterministic/probabilistic) upper/lower bounds of the estimate. For that, they use a “mixture wmb-is” proposal yielding bounded importance weights so that known concentration theorems can be applied. This does not necessarily yield optimal convergence. Our focus is to improve the estimate by aiming to reduce the variance. Our approach

(abstraction sampling) should work with any proposal function. We investigated two proposals, 1) multiplicative one (leading to IS weights that are unbounded), 2) "mixture wmb-is" proposal. We observed improved performance with abstraction sampling in both cases, yet the multiplicative proposal yielded faster convergence in most cases, and it is therefore the one that we report.

## 6.2 RESULTS

The questions addressed by the empirical evaluation are:

- Does Abstraction Sampling result in improved convergence and what insight do we gain on the 2 classes of abstractions?
- What is the impact of tree type (OR vs. AND/OR)?
- Performance of  $AS$  across benchmarks?
- Comparison of  $AS$  with state-of-the-art?

**Individual plots.** Figure 6 has convergence plots for selected problem instances for both OR and AND/OR, using the family of randomized context-based abstractions (RandCB). For each abstraction level we plot the anytime estimate of the log partition function. The dashed line represents the ground truth for the log partition function.

In the first row we show OR and AND/OR plots for the same Grids instance. This is an example where moving to higher level abstractions leads to faster performance (AND/OR error is reduced from 22.66 to 2.93). In the second row (a Promedas instance), the 0-level error is already small, so higher level abstractions have about the same performance. In the third row we present two instances, one from DBN and one from Pedigrees and see that higher level abstractions speed up convergence. In the Pedigree instance, all abstraction levels show similar convergence pattern (the error is small).

**Aggregate table.** Table 1 presents mean errors aggregated over each benchmark for all sampling schemes. This includes our  $AS$  algorithm with both types of trees (OR and AND/OR), and both families of abstractions (RelCB and RandCB) and the two competing schemes. For rows corresponding to our scheme, we present errors for 3 abstraction levels:  $a_0, a_1, a_2$ , where  $a_0$  is 0-level abstraction and independent of the abstraction family (RelCB or RandCB).  $a_1$  and  $a_2$  correspond to higher level abstractions and their definition varies for OR and AND/OR as described earlier. We show errors at 1 min, 20 min, 60 min. In column three we also show the average number of nodes per probe for each abstraction level. For IJGP-SS, we report results only at the 1 hour mark. An "inf" in the table means that the respective algorithm

generates a zero estimate for one or more instances in the benchmark, so the error would be infinity. In parentheses we show the fraction of instances where competing scheme outperforms 0-level abstraction in OR trees.

We see that in both small and large **Grids instances** and for all configurations (OR/AO, RelCB/RandCB), higher level abstractions lead to performance improvement over 0-level abstraction. For grids, randomized (RandCB) abstraction may lead to more significant improvements than RelCB, with AND/OR trees performing even better. For example, after 60 min with OR-RelCB for Grids-small we improve the average error from 4.94 to 3.39, while with OR-RandCB we can improve to 1.41, and AO-RandCB drives down error to 0.84. For both small and large **DBN instances**, only RandCB shows improvement (reducing error from 0.78 to 0.42 for DBN-small, and from 363.93 to 362.88 for DBN-large). For **Pedigree**, the 0-level abstraction is already good, so as expected, higher level abstractions yield little extra benefit. In large **Promedas instances**, RandCB abstraction improved performance in both OR and AND/OR cases, while RelCB was good in the AND/OR case. For small instances, all errors are quite small, still some benefits are gained with the randomized scheme over AND/OR space.

**OR vs. AND/OR.** AND/OR trees usually lead to improved performance over OR. This is expected since AND/OR search spaces are smaller. This is particularly evident for Promedas (small and large) and for large Grids. For example, in Grids-large going from OR-RandCB to AO-RandCB reduces error from 900.01 to 841.84 after 60 min, for  $a_1$  abstraction. For Grids-small the results are mixed on OR vs AND/OR for benchmarks where errors are already small in the 0-level scheme.

**Comparing with state-of-the-art Importance Sampling.** In most benchmarks our 0-level importance sampling baseline is competitive and often outperforms competing sampling algorithms. The point to remember is that the abstraction scheme can be augmented on top of any importance sampling scheme.

**Discussion.** From the analysis of results we gain several valuable insights. **Firstly**, RandCB abstraction consistently demonstrates equal and mostly improved performance compared to the baseline scheme (0-level), while RelCB abstraction is not consistently better. We hypothesize that this difference is due to the ability of RandCB abstractions to generate more uniformly sized abstract states under the proposal. **Secondly**, we observe that in general the AND/OR schemes lead to larger improvements than OR ones. This was not obvious because of the "proper" condition. **Thirdly**, we observe that as expected larger gains are achieved when the baseline importance sampling results in large errors, yet the perfor-

Table 1: Mean Error Aggregated Over Benchmark for a Given Scheme, Time and Abstraction Level ( $a_0, a_1, a_2$ ).  $a_0$  is 0-level abstraction, ( $a_1, a_2$ ) are: OR-RelCB:(4, 8), OR-RandCB:(16, 256), AO-RelCB:(1, 2\_5), AO-RandCB:(2, 4\_5). (#inst,  $\bar{n}$ ,  $\bar{w}$ ,  $\bar{k}$ ,  $|\bar{F}|$ ,  $\bar{s}$ ) are number of instances and averages of number of variables, induced width, max domain size, number of functions, max scope size.

Benchmark #inst, $\bar{n}$ , $\bar{w}$ , $\bar{k}$ , $ \bar{F} $ , $\bar{s}$	Scheme	#nodes per probe $a_0, a_1, a_2$	1 min	20 min	60 min
			$a_0, a_1, a_2$	$a_0, a_1, a_2$	$a_0, a_1, a_2$
DBN-small 60, 70, 30, 2, 16950, 2	OR-RelCB	141, 1963, 22687	1.18, 1.93, 2.58	0.88, 1.86, 1.77	0.78, 1.43, 1.65
	OR-RandCB	141, 1611, 13449	1.18, 1.04, <b>0.81</b>	0.88, 0.71, <b>0.63</b>	0.78, <b>0.42</b> , 0.54
	WMB-IS		9.40	5.69	3.27
	IJGP-SS				1.22
Grids-small 7, 271, 24, 2, 791, 2	OR-RelCB	180, 2774, 42184	6.68, 5.19, 5.07	6.06, 4.71, 4.25	4.94, 4.31, 3.39
	OR-RandCB	180, 2755, 34101	6.68, 5.05, <b>1.97</b>	6.06, 4.10, <b>1.55</b>	4.94, 3.83, 1.41
	AO-RelCB	224, 13388, 91154	5.46, 3.84, 4.70	5.43, 3.68, 3.74	4.83, 2.97, 3.83
	AO-RandCB	224, 9418, 65423	5.46, <b>1.97</b> , 4.27	5.43, 1.72, 3.36	4.83, <b>0.84</b> , 2.77
	WMB-IS		2.94	1.94	1.21
Pedigree-small 22, 917, 26, 5, 917, 4	OR-RelCB	270, 6115, 271925	<b>0.17</b> , 0.19, 0.26	0.17, 0.17, 0.19	0.17, 0.17, <b>0.16</b>
	OR-RandCB	270, 4967, 75980	<b>0.17</b> , 0.20, 0.25	0.17, 0.17, 0.19	0.17, 0.17, 0.19
	AO-RelCB	294, 10286025, 337777	0.18, 0.47, 0.21	<b>0.15</b> , 0.36, 0.17	<b>0.16</b> , 0.20, <b>0.16</b>
	AO-RandCB	294, 1171192, 92627	0.18, 0.24, 0.18	<b>0.15</b> , 0.19, 0.16	<b>0.16</b> , 0.18, <b>0.16</b>
	WMB-IS		inf (1/22)	inf (3/22)	1.06
Promedas-small 41, 666, 26, 2, 674, 3	OR-RelCB	115, 1091, 12801	0.68, 0.77, 1.59	0.33, 0.44, 0.70	0.16, 0.34, 0.47
	OR-RandCB	115, 2174, 28712	0.69, 0.69, 0.62	0.33, 0.28, 0.38	0.16, 0.15, 0.21
	AO-RelCB	110, 825, 5818	0.56, 0.59, 0.66	0.30, 0.34, 0.40	0.15, 0.23, 0.23
	AO-RandCB	110, 753, 6162	0.56, 0.32, <b>0.28</b>	0.30, 0.19, <b>0.15</b>	0.15, <b>0.10</b> , <b>0.10</b>
	WMB-IS		inf (5/41)	1.75	1.15
DBN-large 48, 216, 78, 2, 66116, 2	OR-RelCB	434, 6586, 91881	366.77, 368.29, 369.59	365.32, 366.49, 367.44	363.93, 365.04, 366.20
	OR-RandCB	434, 4858, 71545	366.77, 365.56, <b>365.14</b>	365.32, 364.04, <b>363.53</b>	363.93, 363.14, <b>362.88</b>
	WMB-IS		inf (0/48)	inf (0/48)	inf (0/48)
	IJGP-SS				<b>356.91</b>
Grids-large 19, 3432, 117, 2, 10244, 2	OR-RelCB	2827, 45112, 719763	966.46, 925.86, 927.60	933.64, 900.71, 909.37	928.35, 889.53, 894.59
	OR-RandCB	2827, 45104, 710675	966.46, 945.98, 918.19	933.64, 912.19, 907.30	928.35, 900.01, 894.15
	AO-RelCB	3326, 5485338, 2849697	949.25, 875.81, 910.60	925.85, 863.23, 892.96	918.74, 854.53, 885.18
	AO-RandCB	3326, 3896561, 2826722	949.25, <b>860.66</b> , 885.97	925.85, <b>845.20</b> , 876.74	918.74, <b>841.84</b> , 871.05
	WMB-IS		inf (6/19)	inf (6/19)	inf (7/19)
Promedas-large 88, 962, 48, 2, 974, 3	OR-RelCB	194, 2092, 25156	inf, inf, inf	30.29, inf, inf	29.54, 30.28, 31.89
	OR-RandCB	194, 3586, 54901	inf, inf, 30.24	30.29, inf, 29.27	29.54, 29.26, 28.59
	AO-RelCB	158, 1561, 10840	inf, 30.45, 30.55	30.00, 29.31, 29.32	29.06, 28.67, 28.44
	AO-RandCB	158, 1319, 12082	inf, 29.23, <b>28.97</b>	30.00, 28.47, <b>28.06</b>	29.06, 27.89, <b>27.66</b>
	WMB-IS		inf (1/88)	inf (1/88)	inf (2/88)
IJGP-SS				35.50	

mance does not deteriorate when the baseline errors are small, especially with RandCB abstractions. This suggests using abstraction sampling as a robust enhancement to importance sampling schemes, especially in scenarios when finding a good proposal is difficult or computationally prohibitive. **Fourthly**, the results show that our scheme is competitive with state-of-the-art schemes. **Finally**, our 0-level scheme outperforms WMB-IS although they both rely on WMB-base proposal, but one is multiplicative and the other is mixture. are not identical as we explained.

## 7 CONCLUSION

The paper presents Abstraction Sampling, a scheme that augments search with sampling, exploiting the strength of both. The scheme can be viewed as extending stratified importance sampling to search spaces, allowing the generation of sub-trees representing multiple configurations, rather than a set of independent samples. We proved the scheme's correctness (unbiasedness for both

OR and AND/OR search spaces), analyzed its complexity, discussed convergence properties and provided an extensive empirical evaluation, showing its potential.

The key question is how to design effective abstraction families. In particular, can we devise abstractions yielding equal partitions under the given proposal function, as suggested by theory. Should we aim at domain dependent abstractions? What level of abstraction refinement is cost-effective? Theory suggests that more refined abstractions are superior. We observed that (higher level) abstractions get more effective as the proposals get less effective. We would like to investigate the interaction between these two functions. Finally, since AND/OR search spaces are superior overall, can we overcome the proper restriction to allow more flexible exploration of abstraction functions.

**Acknowledgement.** This work was supported in part by NSF grants IIS-1526842 and IIS-1254071, the US Air Force (Contract FA9453-16-C-0508) and DARPA (Contract W911NF-18-C-0015).

## References

- [Chen, 1992] P.-C. Chen. Heuristic sampling: A method for predicting the performance of tree searching programs. *SIAM Journal on Computing*, 21:295–315, 1992.
- [Darwiche, 2009] A. Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.
- [Dechter and Mateescu, 2007] Rina Dechter and Robert Mateescu. AND/OR search spaces for graphical models. *Artificial Intelligence*, 171(2-3):73–106, 2007.
- [Dechter and Rish, 2003] Rina Dechter and Irina Rish. Mini-buckets: A general scheme for bounded inference. *J. ACM*, 50(2):107–153, 2003.
- [Dechter, 2013] Rina Dechter. *Reasoning with Probabilistic and Deterministic Graphical Models: Exact Algorithms*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2013.
- [Gogate and Dechter, 2011] Vibhav Gogate and Rina Dechter. Samplesearch: Importance sampling in presence of determinism. *Artif. Intell.*, 175(2):694–729, 2011.
- [Knuth, 1975] D.E. Knuth. Estimating the efficiency of backtracking algorithms. *Math. Comput.*, 29:1121–136, 1975.
- [L. H.S. Lelis and Dechter, 2013] L. Otten L. H.S. Lelis and R. Dechter. Predicting the size of depth-first branch and bound search trees. In *Proceedings of IJCAI-2013*, Bejin, China, 2013.
- [Lelis et al., 2014] Levi H. S. Lelis, Lars Otten, and Rina Dechter. Memory-efficient tree size prediction for depth-first search in graphical models. In Barry O’Sullivan, editor, *Principles and Practice of Constraint Programming - 20th International Conference, CP 2014, Lyon, France, September 8-12, 2014. Proceedings*, volume 8656 of *Lecture Notes in Computer Science*, pages 481–496. Springer, 2014.
- [Liu and Ihler, 2011] Qiang Liu and Alexander T. Ihler. Bounding the partition function using holder’s inequality. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 849–856, 2011.
- [Liu et al., 2015] Qiang Liu, John W Fisher III, and Alexander T Ihler. Probabilistic variational bounds for graphical models. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1432–1440, Montreal, Canada, 2015. Curran Associates, Inc.
- [Lou et al., 2017a] Qi Lou, Rina Dechter, and Alexander T. Ihler. Anytime anysace AND/OR search for bounding the partition function. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 860–867, 2017.
- [Lou et al., 2017b] Qi Lou, Rina Dechter, and Alexander T. Ihler. Dynamic importance sampling for anytime bounds of the partition function. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 3199–3207, 2017.
- [Marinescu and Dechter, 2009] R. Marinescu and R. Dechter. AND/OR branch-and-bound search for combinatorial optimization in graphical models. *Artificial Intelligence*, 173(16-17):1457–1491, 2009.
- [Pearl, 1984] J. Pearl. *Heuristics: Intelligent Search Strategies*. Addison-Wesley, 1984.
- [Pearl, 1988] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.
- [Rizzo, 2007] Maria L. Rizzo. *Statistical computing with R*. Chapman & Hall/CRC, 2007.
- [Rubinstein and Kroese, 2007] Reuven Y. Rubinstein and Dirk P. Kroese. *Simulation and the Monte Carlo Method (Wiley Series in Probability and Statistics)*. 2 edition, 2007.

---

# Meta Reinforcement Learning with Latent Variable Gaussian Processes

---

**Steindór Sæmundsson**  
Department of Computing  
Imperial College London  
United Kingdom

**Katja Hofmann**  
Microsoft Research  
Cambridge  
United Kingdom

**Marc Peter Deisenroth**  
Department of Computing  
Imperial College London  
United Kingdom

## Abstract

Learning from small data sets is critical in many practical applications where data collection is time consuming or expensive, e.g., robotics, animal experiments or drug design. Meta learning is one way to increase the data efficiency of learning algorithms by generalizing learned concepts from a set of training tasks to unseen, but related, tasks. Often, this relationship between tasks is hard coded or relies in some other way on human expertise. In this paper, we frame meta learning as a hierarchical latent variable model and infer the relationship between tasks automatically from data. We apply our framework in a model-based reinforcement learning setting and show that our meta-learning model effectively generalizes to novel tasks by identifying how new tasks relate to prior ones from minimal data. This results in up to a 60% reduction in the average interaction time needed to solve tasks compared to strong baselines.

## 1 INTRODUCTION

Reinforcement learning (RL) is a principled mathematical framework for learning optimal controllers from trial and error [28]. However, RL traditionally suffers from data inefficiency, i.e., many trials are needed to learn to solve a specific task. This can be a problem when learners operate in real-world environments where experiments can be time consuming (e.g., where experiments cannot run faster than real time) or expensive. For example, in a robot learning setting, it is impractical to conduct hundreds of thousands of experiments with a single robot because we will have to wait for a long time and the wear and tear on the hardware can cause damage.

There are various ways to address data-efficiency in RL. Model-based RL, where predictive models of the transition function are learned from data, can be used to reduce the number of experiments in the real world. The learned model serves as an emulator of the real world. A challenge with these learned models is the problem of model errors: If we learn a policy based on an incorrect model, the policy is unlikely to succeed on the real task. To mitigate the issue of these model errors it is recommended to use probabilistic models and to take model uncertainty explicitly into account during planning [27, 10]. This approach has been applied successfully to simulated and real-world RL problems [9], where a policy-search approach was used to learn optimal policy parameters. Robustness to model errors and, thereby, increased data efficiency, can be achieved by using model predictive control (MPC) instead of policy search since MPC allows for online updates of the model, whereas policy search would update the model only after a trial [17].

If we are interested in solving a set of related tasks we can use meta learning as an orthogonal approach to increase data efficiency. Generally, the aim of meta learning is to train a model on a set of training tasks and then generalize to new tasks using minimal additional data [12]. The strength of meta learning is to transfer learned knowledge to related situations. For example, we may want to control multiple robot arms with slightly different specifications (e.g., link weights or lengths) or different operating environments (e.g., underwater, in low gravity). Normally, learned controllers deal with a single task. In a robotics context, solutions for multiple related tasks are often desired, e.g., for grasping multiple objects [22] or in robot games, such as robot table tennis [24] or soccer [3]. Much of the literature on meta and transfer learning in RL has focused on multi-task learning, i.e., cases where the system/robot is the same, but the task changes [16, 29, 3, 5, 20, 21, 24, 8, 12]. Although meta learning given multiple or non-stationary dynamics has also been considered in [11, 18, 4, 1].

We adopt a meta learning [26, 32, 12] perspective on the problem of using knowledge from prior tasks for more efficient learning of new ones. We take a probabilistic view and propose to transfer knowledge within a model-based RL setting using a latent variable model. We focus on settings where system specifications differ, but where the task objective is identical. We treat system specifications as a latent variable, and infer these unobserved factors and their effects online. To address the issue of meta learning within the context of data-efficient RL, we propose to learn predictive dynamics models conditioned on the latent variable and to learn controllers using these models. We use Gaussian processes (GPs) [25] to model the dynamics, and MPC for policy learning. To obtain a posterior distribution on the latent variable, we use variational inference. The posterior can be updated online as we observe more and more data, e.g., during the execution of a control strategy. Hence, we systematically combine three orthogonal ideas (probabilistic models, MPC, meta learning) for increased data efficiency in settings where we need to solve different, but related tasks.

## 2 MODEL-BASED RL

We consider stochastic systems of the form

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{c}_t) + \epsilon \quad (1)$$

with state variables  $\mathbf{x} \in \mathbb{R}^D$ , control signals  $\mathbf{c} \in \mathbb{R}^K$  and i.i.d. system noise  $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{E})$ , where  $\mathbf{E} = \text{diag}(\sigma_1^2, \dots, \sigma_D^2)$ . For model-based RL we first aim to learn the unknown transition function  $f$ . In this context, [27, 10] highlighted that probabilistic models of  $f$  are essential for data-efficient learning as they mitigate the effect of model errors. Therefore, we learn the dynamics of the system using a GP.

**GP Dynamics** A GP is a probabilistic, non-parametric model and can be interpreted as a distribution over functions. A GP is defined as an infinite collection of random variables  $\{f_1, f_2, \dots\}$ , any finite number of which are jointly Gaussian distributed [25]. A GP is fully specified by a mean function  $m$  and a covariance function (kernel)  $k$ , which allows us to encode high-level structural assumptions on the underlying function such as smoothness or periodicity. We denote an unknown function  $f$  that is modeled by a GP by  $f \sim GP(m(\cdot), k(\cdot, \cdot))$ . We use the squared exponential (RBF) covariance function

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 \exp\left(-\frac{1}{2}(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{L}^{-1}(\mathbf{x}_i - \mathbf{x}_j)\right) \quad (2)$$

where  $\sigma_f^2$  is the signal variance and  $\mathbf{L}$  is a diagonal matrix of squared length-scales.

**RL with MPC** Our objective is to find a sequence of optimal controls  $\mathbf{c}_0^*, \dots, \mathbf{c}_{H-1}^*$  that minimizes the expected finite-horizon cost

$$J = \mathbb{E} \left[ \sum_{t=1}^H \ell(\mathbf{x}_t) \right], \quad (3)$$

where  $\mathbf{x}_t$  is the state of the system at time  $t$  and  $\ell$  is a known immediate/instantaneous cost function that encodes the task objective. We consider an episodic setting. Initial states  $\mathbf{x}_0$  are sampled from  $p(\mathbf{x}_0) = \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ .

To find the optimal open-loop sequence  $\mathbf{c}_0^*, \dots, \mathbf{c}_{H-1}^*$ , we compute the expected long-term cost  $J$  in (3) using Gaussian approximations  $p(\mathbf{x}_1), \dots, p(\mathbf{x}_H)$  for a given control sequence  $\mathbf{c}_0, \dots, \mathbf{c}_{H-1}$ . The computation of the expected long-term cost is detailed in the supplementary material. Then, we find an open-loop control sequence that minimizes the expected long-term cost and apply the first control signal  $\mathbf{c}_0^*$  to the system, which transitions into the next state. Next we re-plan, i.e., we determine the next open-loop control sequence  $\mathbf{c}_0^*, \dots, \mathbf{c}_{H-1}^*$  from the new state. This iterative MPC approach turns an open-loop controller into a closed-loop controller. Combining MPC with learned GP models for the underlying dynamics increases the robustness to model errors and has shown improved data efficiency in RL [17].

## 3 MODEL-BASED META RL

We assume a setting with a potentially infinite number of dynamical systems that are of the same type but with different specifications (e.g., multiple robotic arms with links of differing lengths and weight). More formally, we assume a distribution over dynamical systems with samples  $f_p \sim p(f)$  indexed by  $p = 1..P$ . Each sample  $f_p$  is a dynamical system of the form (1) with states  $\mathbf{x} \in \mathbb{R}^D$  and control signals  $\mathbf{c} \in \mathbb{R}^K$ . Instead of learning individual predictive models for each dynamical system from scratch, we look to meta learning as an approach to learning new dynamics more data efficiently by leveraging shared structure in the dynamics.

**Meta Learning** Generally, meta learning aims to learn new tasks with minimal data and/or computation using knowledge or inductive biases learned from prior tasks [12]. Here we require our model to accomplish two things simultaneously:

1. **Multi-Task Learning:** Disentangle global and task-specific properties of the different dynamics such that it can solve multiple tasks.
2. **Transfer Learning:** Use global properties to generalize predictive performance to novel dynamics.

We propose to address this meta-learning challenge in a probabilistic way: We model the distribution over systems using a latent embedding  $\mathbf{h}$  and model the dynamics using a global function conditioned on the latent embedding. Each sample  $f_p$  from the distribution is modeled as

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{c}_t, \mathbf{h}_p) + \epsilon, \quad (4)$$

such that the successor state depends on the latent system specification  $\mathbf{h}_p$ . This means, we explicitly model the global properties through a shared function  $f$  and the task-specific variation using a distribution over the latent variables  $p(\mathbf{h}_p)$ . Framing the meta learning problem as a hierarchical Bayesian model means that meta-training becomes inference in a meta-learning model.

**Training and Evaluation** Training corresponds only to the *multi-task learning* aspect of our meta learning approach. We aim to learn the global function  $f$  and the latent embeddings  $\mathbf{h}_p$  given trajectory observations from a set of training systems. For evaluation at test time, we use inference to obtain a distribution over a set of latent variables  $\mathbf{h}_*$  for each test system. Since our objective is to improve data efficiency in an RL setting, we consider two related but distinct measures of performance. One corresponds to the transfer learning aspect of our approach, where we infer only the latent test embeddings without updating the global model  $f$ . We refer to this as the *single-shot performance*. The other measure we use is the additional data required to successfully solve a RL task: *few-shot learning*. In this case, the global model  $f$  is updated with new additional data, thus combining both the multi-task and transfer learning aspects.

The meta RL procedures for training and testing are detailed in algorithms 1 and 2, respectively.

### 3.1 META-LEARNING MODEL

Our meta-learning model is a GP prior on the unknown transition function in (4) with a concatenated state  $\tilde{\mathbf{x}}_t = (\mathbf{x}_t, \mathbf{c}_t, \mathbf{h}_p) \in \mathbb{R}^{D+K+Q}$  as the input to the model. We define  $\mathbf{y}_t = \mathbf{x}_{t+1} - \mathbf{x}_t$  as the targets of the GP and take the mean function to be  $m(\tilde{\mathbf{x}}_t) = \mathbf{0}$ , which encodes that a priori the state does not change [9]. Each dimension of the targets  $\mathbf{y}$  is modeled by an independent GP. We use a Gaussian likelihood

$$p(\mathbf{y}_t | \tilde{\mathbf{x}}_t, \mathbf{f}(\cdot), \boldsymbol{\theta}) = \mathcal{N}(\mathbf{y}_t | \mathbf{f}(\tilde{\mathbf{x}}_t), \mathbf{E}), \quad (5)$$

where  $\boldsymbol{\theta} = \{\mathbf{E}, \mathbf{L}, \sigma_f^2, Q\}$  are the model hyperparameters and  $\mathbf{f}(\cdot) = (f^1(\cdot), \dots, f^D(\cdot))$  denotes a multi-dimensional function. We place a standard-normal prior  $\mathbf{h}_p \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  on the latent variables  $\mathbf{h}_p$ . The full

Initialize dataset  $D$  and model  $M$

▷ Initial random rollouts

```
forall training tasks do
  execute random policy
  add observations to D
end
```

▷ Meta training

```
while training tasks not solved do
  —update—: train M and infer h given D
  forall unsolved training tasks do
    for each step in horizon do
      —plan—: get control sequence using (3)
      —execute—: execute first control in
                sequence
    end
    add observations to D
    check if task solved
  end
end
```

**Algorithm 1:** Model-based Meta RL with MPC (Train)

specification of the model is

$$p(\mathbf{Y}, \mathbf{H}, \mathbf{f}(\cdot) | \mathbf{X}, \mathbf{C}) \quad (6)$$

$$= \prod_{p=1}^P p(\mathbf{h}_p) \prod_{t=1}^{T_p} p(\mathbf{y}_t | \mathbf{x}_t, \mathbf{c}_t, \mathbf{h}_p, \mathbf{f}(\cdot)) p(\mathbf{f}(\cdot))$$

where we denote a collection of vectors in bold uppercase and we have dropped dependence on the hyperparameters for notation purposes. The corresponding graphical model is given in Fig. 1. The figure shows the dependence of individual system observations on the global GPs  $\mathbf{f}(\cdot)$  modeling each dimension of the outputs, the system-specific latent embeddings  $\mathbf{h}_p$  and the observed states and controls.

**Model Properties** Our meta-learning GP (ML-GP) model exhibits three important properties:

1. The latent variable encodes a distribution over plausible systems and is inferred from data
2. Conditioning the GP on the latent variable enables it to disentangle global and task specific variation in the dynamics. Generalization to new dynamics is done by inferring the latent variable of that system.
3. The latent variable is fixed within system trajectories so that inference can be performed online (e.g. while executing a controller).

Fig. 2 illustrates these properties on a toy example.



Given dataset  $D$  and model  $M$  from training  
 ▷ Single shot performance

```

forall test tasks do
  for each step in horizon do
    —plan—: get control sequence using (3)
    —execute—: execute first control in sequence
    —inference—: infer the value of  $h_*$  given
      observations so far
  end
  add observations to  $D$ 
  check if task solved
end
  
```

▷ Meta test

```

while test tasks not solved do
  —update—: train  $M$  and infer  $h$  given  $D$ 
  forall unsolved test tasks do
    for each step in horizon do
      —plan—: get control sequence using (3)
      —execute—: execute first control in
        sequence
    end
    add observations to  $D$ 
    check if task solved
  end
end
  
```

**Algorithm 2:** Model-based Meta RL with MPC (Test)

### 3.2 INFERENCE

To learn the dynamics model we seek to optimize the hyperparameters  $\theta$  w.r.t. the log-marginal likelihood, which involves marginalization of the latent variables in (6). For predictions of the evolution of a system we also need to infer the posterior GP and the posterior distribution of the latent variables  $\mathbf{H} = (h_1, \dots, h_P)$ . We approach this problem with approximate variational inference. We posit a variational distribution that assumes independence between the latent functions of the GP and the latent task variables

$$Q(\mathbf{f}(\cdot), \mathbf{H}) = q(\mathbf{f}(\cdot))q(\mathbf{H}) \quad (7)$$

and minimize the Kullback-Leibler divergence between the approximate and true posterior distributions. Equivalently we can maximize the evidence lower bound

$$\mathcal{L} = \mathbb{E}_{Q(\mathbf{f}(\cdot), \mathbf{H})} \left[ \log \frac{p(\mathbf{Y}, \mathbf{H}, \mathbf{f}(\cdot) | \mathbf{X}, \mathbf{C})}{Q(\mathbf{f}(\cdot), \mathbf{H})} \right], \quad (8)$$

which lower-bounds the log-marginal likelihood [15]. We parameterize our variational distribution such that we can compute the lower bound in (8). We then jointly optimize  $\mathcal{L}$  with respect to the model hyperparameters and the variational parameters.

**Sparse Gaussian Processes** It is important to account for the fact that training a GP on a joint data set of

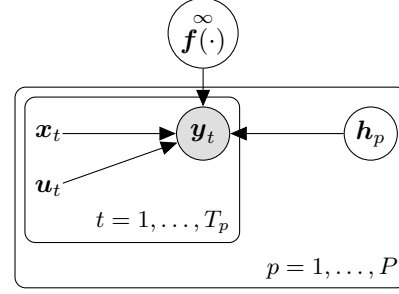


Figure 1: Graphical model for our ML-GP model.

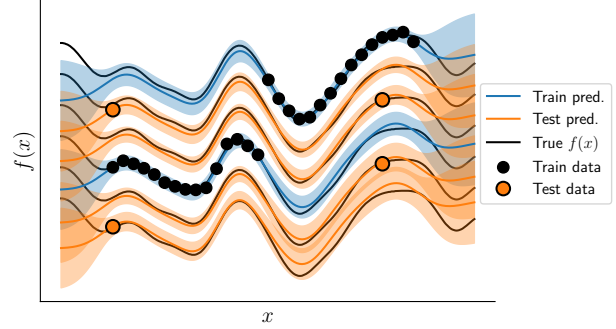


Figure 2: The figure shows six unknown tasks (toy examples) with a shared structure (the same function) and task specific variation (fixed offset). The ML-GP model is able to disentangle the two automatically given the training data (black discs) as demonstrated by the training prediction curves. It also infers a reasonable value for the offset given a single observations from unseen test tasks (orange discs) and can use the global structure to generalize predictive performance on those tasks.

$P$  different systems quickly becomes infeasible due to the  $\mathcal{O}(T^3)$  computational complexity for training and  $\mathcal{O}(T^2)$  for predictions where  $T$  is the total number of observations. To address this we turn to the variational sparse GP approximation [30] and approximate the posterior GP with a variational distribution  $q(\mathbf{f}(\cdot))$  that depends on a small set of  $M \ll T$  inducing points. We introduce a set of  $M$  inducing inputs  $\mathbf{Z} = (z_1, \dots, z_M) \in \mathbb{R}^{M \times (D+K+Q)}$ , which live in the same space as  $\tilde{\mathbf{x}}$ , with corresponding GP function values  $\mathbf{U} = (u_1, \dots, u_M) \in \mathbb{R}^{M \times D}$ . We follow [14] and specify the variational approximation as a combination of the conditional GP prior and a variational distribution over the inducing function values, independent across output dimensions

$$q(f^d(\cdot)) = \int p(f^d(\cdot) | \mathbf{u}^d) q(\mathbf{u}^d) d\mathbf{u}^d. \quad (9)$$

where  $q(\mathbf{u}^d) = \mathcal{N}(\mathbf{u}^d | \mathbf{m}^d, \mathbf{S}^d)$  is a full rank Gaussian. The integral in (9) can be computed in closed form since both terms are Gaussian, resulting in a GP with mean and

covariance functions given by

$$m_q(\cdot) = \mathbf{k}_Z^T(\cdot) \mathbf{K}_{ZZ}^{-1} \mathbf{m}^d \quad (10)$$

$$k_q(\cdot, \cdot) = k(\cdot, \cdot) - \mathbf{k}_Z^T(\cdot) \mathbf{K}_{ZZ}^{-1} (\mathbf{K}_{ZZ} - \mathbf{S}^d) \mathbf{K}_{ZZ}^{-1} \mathbf{k}_Z(\cdot) \quad (11)$$

where  $[\mathbf{k}_Z(\cdot)]_i = k(\cdot, \mathbf{z}_i)$  and  $[\mathbf{K}_{ZZ}]_{ij} = k(\mathbf{z}_i, \mathbf{z}_j)$ . Here, the variational approach has two main benefits: a) it reduces the complexity of training to  $\mathcal{O}(TM^2)$  and predictions to  $\mathcal{O}(TM)$ , b) it enables mini-batch training for further improvement in computational efficiency.

**Latent Variables** For the latent variables  $\mathbf{H}$  we assume a Gaussian variational posterior

$$q(\mathbf{H}) = \prod_{p=1}^P \mathcal{N}(\mathbf{h}_p | \mathbf{n}_p, \mathbf{T}_p) \quad (12)$$

where  $\mathbf{T}_p$  is in general a full rank covariance matrix. We use a diagonal covariance in practice for more efficient computation of the ELBO (8).

**Evidence Lower Bound (ELBO)** The ELBO can be shown to decompose into (see supplementary material)

$$\mathcal{L} = \sum_{p=1}^P \sum_{t=1}^T \mathbb{E}_{q(\mathbf{f}_t | \mathbf{x}_t, \mathbf{c}_t)} [\log p(\mathbf{y}_t | \mathbf{f}_t)] - \text{KL}[q(\mathbf{H}) || p(\mathbf{H})] - \text{KL}[q(\mathbf{U}) || p(\mathbf{U})] \quad (13)$$

where the expectation is taken with respect to

$$q(\mathbf{f}_t | \mathbf{x}_t, \mathbf{c}_t) = \int q(\mathbf{f}_t | \mathbf{x}_t, \mathbf{c}_t, \mathbf{h}_p) q(\mathbf{h}_p) d\mathbf{h}_p. \quad (14)$$

We emphasize that  $q(\mathbf{f}_t | \mathbf{x}_t, \mathbf{c}_t, \mathbf{h}_p) = q(\mathbf{f}(\tilde{\mathbf{x}}_t) | \mathbf{x}_t, \mathbf{c}_t, \mathbf{h}_p)$  is the marginal of the GP evaluated at the inputs  $\tilde{\mathbf{x}}_t$ . The integral in (14) is intractable due to the non-linear dependence on  $\mathbf{h}_p$  in (10) and (11). Given our choice of kernel (RBF) and Gaussian variational distribution  $q(\mathbf{h}_p)$  the first and second moments can be computed in closed form. We could use these terms to compute the log-likelihood term in closed form since the likelihood is Gaussian but in practice this can be prohibitively expensive since it requires the evaluation of a  $TM^2D$  tensor. Instead we avoid computing the moments by approximately integrating out the latent variable using Monte Carlo sampling.

**Training** For the update steps in algorithms 1 and 2 we jointly optimize the GP hyperparameters  $\theta$  and the variational parameters  $\phi = \{\mathbf{Z}, M \{\mathbf{m}^d, \mathbf{S}^d\}_{d=1}^D, \{\mathbf{n}_p, \mathbf{T}_p\}_{p=1}^P\}$  w.r.t. the ELBO. For the inference step in algorithm 2, we optimize only the variational parameters for the latent variables  $\mathbf{h}$ , i.e.  $\phi_{\mathbf{h}} = \{\mathbf{n}_p, \mathbf{T}_p\}_{p=1}^P$ .

In practice, we use a single sample  $\mathbf{h}_p \sim q(\mathbf{h}_p)$  drawn from the variational distribution for each system. We use stochastic mini-batch training, sampling a small number of trajectories and their associated latent variable at a time. Empirically, we found standardizing the input states and controls  $(\mathbf{x}, \mathbf{c})$  and outputs  $(\mathbf{y})$  crucial for successful training of the model. For optimization we used Adam [19] with default hyperparameters:  $\alpha = 1 \times 10^{-2}$ ,  $\beta_1 = 0.9$ ,  $\beta_2 = 0.999$ ,  $\epsilon = 10^{-8}$ .

## 4 EXPERIMENTS

Our experiments focus on evaluating our proposed model in terms of predictive performance, the nature of the latent embeddings and data efficiency. We address the following questions: “Does conditioning the GP on the latent variable allow us to disentangle system specific and global properties of the observations? Does this improve predictive performance in the transfer learning setting?” (Section 4.1). “Is the latent system embedding the model learns a sensible one?” (Section 4.2) “Does the application of our ML-GP in model-based RL lead to data-efficient learning across tasks” (Section 4.3).

As a baseline model we use a sparse GP (SGP) [30] as described in Section 3.2 but without the latent variable that explicitly represents the task. For assessing the model quality (Section 4.1) we additionally evaluate the performance of a standard GP with no sparse approximation. We use the following nonlinear dynamical systems to perform our experiments:

**Cart-pole swing-up** The cart-pole system consists of a cart that moves horizontally on a track with a freely swinging pendulum attached to it. The state of this nonlinear system is the position  $x$  and velocity  $\dot{x}$  of the cart and the angle  $\theta$  and angular velocity  $\dot{\theta}$  of the pendulum. The control signals act as a horizontal force on the cart limited to the range  $c \in [-15, 15]$  N. The mean of the initial state distribution is the state where the pendulum is hanging downward. The task is to learn to swing up and balance the pendulum in the inverted position in the middle of the track.

**Double-pendulum swing-up** The double-pendulum system is a two-link robotic arm with two motors, one in the shoulder and one in the elbow. The state of the system comprises the angles  $\theta_1, \theta_2$  and angular velocities  $\dot{\theta}_1, \dot{\theta}_2$  of the inner and outer pendulums, respectively. The control signals are the torques  $c_{1,2} \in [-4, 4]$  Nm applied to the two motors. The mean of  $p(\mathbf{x}_0)$  is the position where both pendulums are hanging downward. The goal is to find a control strategy that swings the double pendulum up and balances it in the inverted position.

## 4.1 QUALITY OF MODEL LEARNING

In the first set of experiments, we investigate if the latent variable of the ML-GP improves prediction performance on unseen systems compared to the SGP baseline. To assess the effect of the sparse approximation we also include a standard GP baseline (no sparse approximation) in this section. To test the prediction quality, we execute the same fixed control signals<sup>1</sup> on six settings of the cart-pole task to generate one 100-step (10 s) trajectory per training task. The specifications of the training tasks were all combinations  $(m, l)$  of  $m \in \{0.4, 0.6, 0.8\}$ ,  $l \in \{0.5, 0.7\}$  where  $m$  and  $l$  denote the mass and length of the pendulum, respectively. Thus, the total number of data points for our six training tasks is  $T = 600$  amounting to 60 s of interaction time.

For evaluation we use the same sequence of control signals we used for training and compute the one-step prediction quality in terms of root mean squared error (RMSE) and negative log likelihood (NLL) on a set of test tasks. We use 14 held-out test tasks specified as  $m \in \{0.4, 0.6, 0.7, 0.8, 0.9\}$ ,  $l \in \{0.4, 0.5, 0.6, 0.7\}$ , excluding the  $(m, l)$ -combinations of the training tasks.

During evaluation, we observe 10 time steps from an unseen trajectory based on which we infer the latent task  $\mathbf{h}_p$  using variational inference for the ML-GP while leaving the model hyperparameters and other variational parameters fixed. We then predict the next 90 steps using the ML-GP, SGP and GP models. ML-GP also performs online inference of the latent variable after each step. We repeat this experiment with 10 different seeds that determine the initial state, and average the results.

Fig. 3 shows the RMSE and NLL for all 3 models. The ML-GP clearly outperforms both the SGP and GP baselines in terms of both the accuracy of its mean predictions (as evident by the RMSE) as well as capturing the data better under its predictive distribution as seen by the NLL. The NLL accounts for both the mean prediction as well as the uncertainty of the model about the prediction. Both baselines have comparable RMSEs to each other with enough inducing points but generalize poorly on new tasks with overconfident predictions. Fig. 4 illustrates this behavior.

The baselines fail to generalize since they have no observations from the system with this configuration. The ML-GP generalizes from training to new test tasks naturally because it explicitly incorporates the latent variables encoding the system configuration.

<sup>1</sup>the control signals were manually chosen as ones that solved a configuration not included in either the training or test set.

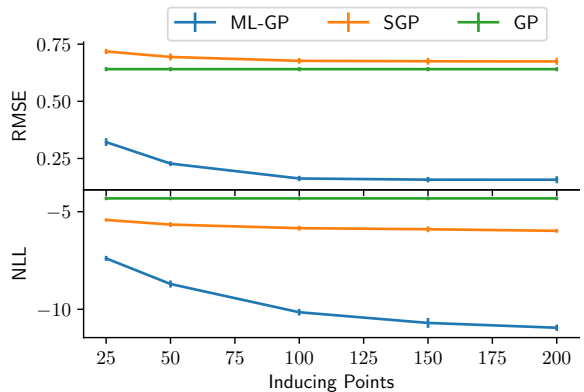


Figure 3: Mean and two standard deviation confidence error-bars of the RMSE and NLL for the ML-GP, SGP and the standard GP model as a function of the number of inducing points. The ML-GP significantly outperforms both baselines.

## 4.2 LATENT EMBEDDING

In order for our model to perform well in meta learning settings, the latent variables  $\mathbf{h}_p$  need to reflect a *sensible* embedding. By sensible we mean it should take on a particular structure: a) locally similar values in the latent space should correspond to similar task specifications and b) moving in latent space should correspond to coherent transitions in task specifications.

Fig. 5 shows an example of an inferred latent embedding of both training and test tasks after the training procedure outlined above. The test-task latent variables are inferred from 10 observations from the held-out systems.

The different colors of the discs denote the four different settings of lengths whereas the colors of the dotted lines connecting the discs denote the five different settings of mass. The figure plots the mean of each  $q(\mathbf{h}_p)$  with two standard deviation error bars in each dimension. The embedding displays an intuitive structure where changes in length or mass are disentangled (denoted by the black arrows) into a length-mass coordinate system with the expected transitive properties, e.g. the lengths are ordered as blue ( $l = 0.4$ ), green ( $l = 0.5$ ), red ( $l = 0.6$ ) and orange ( $l = 0.7$ ). The uncertainty estimates also exhibit qualitatively the intuitive property of being less uncertain about tasks which are similar to (closer to) the training tasks, e.g. comparing the red and blue tasks in fig. 5.

## 4.3 DATA-EFFICIENT RL

Our second set of experiments investigates the performance of the ML-GP model in terms of data efficiency in RL settings. Specifically, we look at whether our meta learning approach is a) at least as efficient at solving a

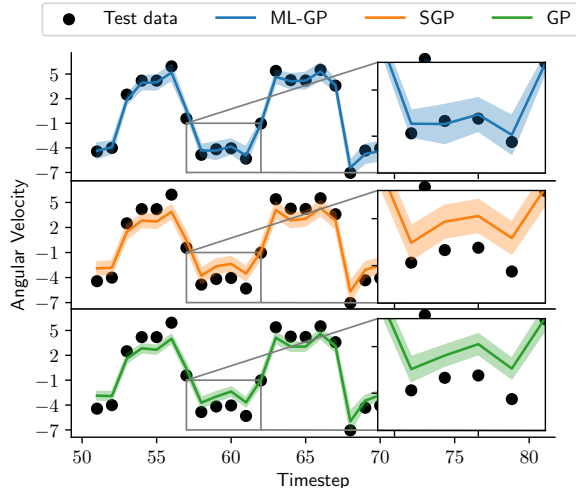


Figure 4: One-step predictions of the angular velocity in cart-pole. The figure shows the true data points (discs) and the predictive distributions with a two standard deviation confidence interval for the ML-GP, SGP and a standard GP. The ML-GP generalizes well to new tasks; both the SGP and GP baselines are overly confident.

set of training tasks, b) more efficient at solving subsequent test tasks, when compared to a non-meta learning baseline and c) whether the ML-GP model improves performance when compared to the SGP model trained with the meta learning approach.

We first learn a model of the dynamics (4), which we then use to learn a policy to control the system. For policy learning we use MPC, minimizing the cost in (3) with a moving horizon to learn an optimal sequence of control signals. We assume we have a set of training systems and evaluate the performance of the models using some held-out test systems with novel configurations (tasks).

We run experiments on both the cart-pole swing-up task and the double-pendulum swing-up task. In both scenarios, we use a sampling frequency of 10 Hz, episodes of 30 steps (3 s) and a planning horizon of 10 steps. For the cart-pole swing-up, solving the task means the pendulum is balanced closer than 8 cm from the goal position for at least the last 10 steps. For the double-pendulum swing-up, it means the outer pendulum is balanced closer than 22 cm for at least the last 10 steps.

At meta-training or test-time, a pass through the training-/test-set means executing the MPC policy learning algorithm on each of the unsolved task in that set. Each execution constitutes a trial for that task. The sets are traversed until all the tasks are solved or all unsolved tasks have executed 15 trials. The training and test procedures are detailed in algorithms 1 and 2 in section 3. All results are averaged over 20 independent random initializations.

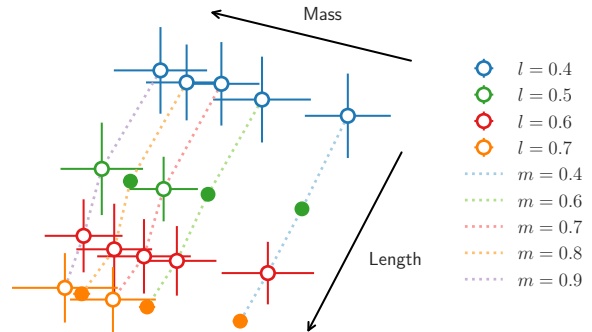


Figure 5: Latent space embedding of cart-pole configurations/tasks. The figure shows the mean (discs) of the inferred latent variables and two standard deviation error bars. Filled discs are training tasks and empty discs are held out test tasks. The colors of the discs represent the length and the colors of the dotted lines between discs represent the mass.

Note that we execute on all (unsolved) tasks before re-training the dynamics model as detailed in section 3. This means that the model is updated with 3 s worth of experience for every task in that pass at a time. On the other hand, the model does not take advantage of additional prior experience until it has completed a pass.

For comparison with the ML-GP model, we use the SGP model trained in two different ways. To establish a lower-bound baseline, we run the model-based RL approach where we train a separate model for each task on both the training and test sets. After each training task we additionally attempt to solve each of the test tasks to evaluate single-shot performance where we report the mean across the training tasks as the single shot success rate. We refer to this baseline as SGP-I which is a sparse variant of the approach in [17] that achieves state-of-the-art in data efficiency. Secondly, we train a single SGP model on all the training tasks simultaneously using the same training approach as we do for ML-GP. We refer to this baseline as SGP-ML.

**Cart-pole swing-up** We train the models on six specifications of the cart-pole dynamics, with  $m \in \{0.4, 0.6, 0.8\}$ ,  $l \in \{0.6, 0.8\}$  and evaluate its performance on a set of four test tasks chosen as  $m \in \{0.7, 0.9\}$ ,  $l = \{0.5, 0.7\}$ . We choose these settings to examine the performance on both interpolation and extrapolation for differing lengths and masses. We choose the squared distance between the tip of the pendulum and goal position (with the pendulum balanced straight in the middle of the track) as the cost. Fig. 6 shows the mean success rate (over initializations and the four test tasks) of ML-GP, SGP-I and SGP-ML against the number of trials executed on the systems. We observe that

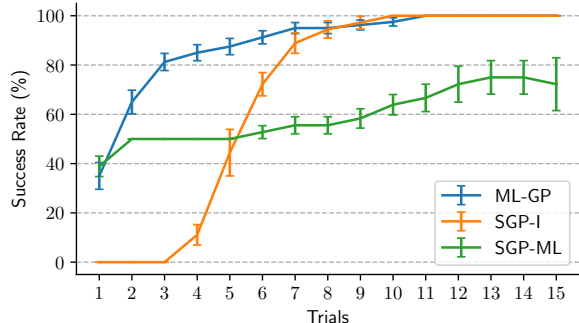


Figure 6: Mean success rate over initializations and the four test tasks for the cart-pole system after training on six tasks. The graph compares ML-GP with SGP-I (trained independently) and SGP-ML (trained on all tasks).

both the ML-GP model and the SGP-ML display generalization to new tasks as evident by the success rate in the first trial (see also Table 1). However, whereas the ML-GP quickly improves with more observations in subsequent trials, the SGP-ML model struggles to solve the remaining tasks. We attribute this failure to the inability of the SGP-ML model to explain variation in the dynamics caused by differences in system specifications.

When comparing with independent training of each system we see that the ML-GP compares favorably, reaching 80% success rate after only three trials and 90% after six trials compared to the SGP-I, which reaches 80% after 7 trials and 90% after 8 trials. We further analyze performance of ML-GP to identify the tasks that were additionally solved between trials 3 and 6. We find that this is due to a consistently challenging system with  $m = 0.9, l = 0.5$ , which requires the learner to extrapolate beyond the range of values seen during training. The mean number of trials required to solve this task is  $4.3 \pm 0.6$ , compared to the task mean of  $2.7 \pm 0.2$  trials. Table 1 shows the mean total time required to solve

Table 1: Mean time spent solving the cart-pole system and the single-shot success rate.

MODEL	TRAIN (s)	TEST (s)	SINGLE SHOT
SGP-I	$16.1 \pm 0.4$	$17.5 \pm 0.4$	$0.08 \pm 0.01$
SGP-ML	$23.7 \pm 1.4$	$20.8 \pm 1.2$	<b><math>0.38 \pm 0.04</math></b>
ML-GP	<b><math>15.1 \pm 0.5</math></b>	<b><math>8.1 \pm 0.6</math></b>	$0.35 \pm 0.05$

the training and test tasks. On average, ML-GP needs less than half the amount of time to solve the test tasks compared to individually training on the tasks (SGP-I). We also see an improvement in the total training time, which suggests that ML-GP derives some transfer benefit during training despite training on the systems on a concurrent trial basis, i.e. we do not update the model

until all systems have executed a given trial. Compared to the SGP-ML, the ML-GP model can maintain an accurate model while learning multiple systems and quickly adapts to new dynamics, whereas the performance of SGP-ML stagnates as reflected in the interaction time on both the training and test systems.

**Double-pendulum swing-up** We repeat the same experimental set-up on the double-pendulum task. We trained on six systems with  $m_1 \in \{0.5, 0.7\}$ ,  $l_1 \in \{0.4, 0.5, 0.7\}$  and evaluate on a set of four test tasks chosen as  $m_1 \in \{0.6, 0.8\}$ ,  $l_1 = \{0.6, 0.8\}$ , where  $m_1, l_1$  are the mass and length of the inner pendulum. The cost is the squared distance between the tip of the outer pendulum and the goal position (with both pendulums standing straight up). Fig. 7 plots the mean success rate against the number of trials executed on the system. Comparing the ML-GP model to the SGP-ML we observe comparable single-shot performance and a qualitatively similar learning curve for the test tasks. However, the ML-GP reaches 90% success rate about four trials before the SGP-ML, around trial nine, i.e. meta learning achieves a significantly higher data efficiency. Compared to independent training of the tasks using SGP-I, the ML-GP leads to significantly less (new) training data needed to solve the tasks. Table 2 reports the mean total

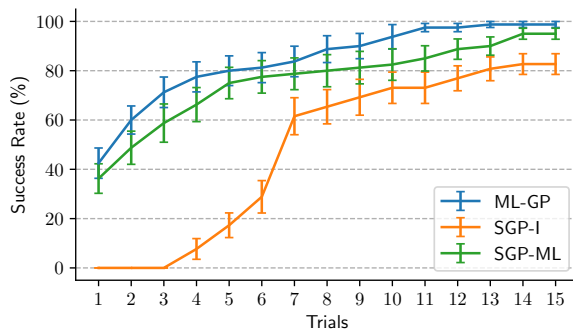


Figure 7: Mean success rate over initializations and the four test tasks for the double pendulum after training on six tasks. The graph compares the ML-GP against the SGP-I (trained independently on each task) and the SGP-ML (trained using the meta learning procedure).

time required to solve the tasks. Compared to the SGP-ML, the performance of the two is similar, although arguably the ML-GP compares favorably in terms of average time needed to solve the test tasks. Compared to the SGP-I, we see improvement during training as well as at test time. The average time needed for ML-GP to solve the test environments is reduced to around 40% to that of the SGP-I.

Table 2: Mean time spent solving the double-pendulum system and the single-shot success rate.

MODEL	TRAIN (s)	TEST (s)	SINGLE SHOT
SGP-I	18.9 ± 0.7	25.9 ± 1.5	0.07 ± 0.01
SGP-ML	17.9 ± 1.3	13.7 ± 2.2	0.36 ± 0.06
ML-GP	<b>16.6 ± 1.1</b>	<b>10.2 ± 1.6</b>	<b>0.43 ± 0.06</b>

## 5 RELATED WORK

Meta learning has long been proposed as a form of learning that would allow systems to systematically build up and re-use knowledge across different but related tasks [26, 32]. MAML is a recent promising model free meta learning approach that learns a set of model parameters that are used to rapidly learn novel tasks [12]. Another interpretation of MAML is formulated in [13], which shares our hierarchical Bayesian formulation of the meta learning problem. However, the model-free setting in which MAML has been applied so far typically require orders of magnitude more training data than the model-based approaches we build up on in the present work.

Our ML-GP model resembles the GP latent variable model (GPLVM), which is typically used in unsupervised settings [23]. In the GPLVM, the GP is used to map a low-dimensional latent embedding to higher-dimensional observations. A Bayesian extension (BG-PLVM) was introduced in [31] where inference over the latent variable is performed using variational inference. To enable minibatch training, and unlike BG-PLVM, we take the approach of [14] and do not marginalize out the inducing variables. The main difference of our model and the GPLVM is that we learn a mapping from both observed and latent inputs to observations.

The combination of observed and latent inputs was investigated in [33] where the authors use Metropolis sampling for inference which does not scale to larger datasets. A similar setup is found in [7] where the model is used for partially observed input data. The work also proposes uses in autoregressive settings similar to ours. Different from us, the distribution over inducing variables is analytically optimized, making minibatch training infeasible.

A related and complimentary line of research are multi-output GPs (MOGPs) [2]. Recently, [6] proposed a latent variable extension to MOGPs (LVMOGP) which is similar to our ML-GP, particularly in their missing data formulation of the model. The crucial difference from our work is that we augment the input space by concatenating the latent variable to the input space while the LVMOGP uses the Kronecker product of two separate kernels applied on the latent and input spaces respectively.

Notably, the two models are equivalent for kernels that naturally decompose as a Kronecker product (e.g. the RBF) but depart from there.

A similar framework to ours is found in [11], called hidden parameter Markov decision processes (HiP-MDP), which parametrizes a family of related dynamics through a low dimensional latent embedding. The HiP-MDP assumes a fixed latent variable within trajectories. Different from us, the authors use an infinite mixture of GP basis functions where the task specific variation is obtained through the weights of the basis functions [11]. This work was extended in [18], replacing the GP basis functions with a Bayesian neural network. This enables non-linear interactions between the latent and addresses scalability. In this work, the interactions between latent and state variables are obtained through the non-linear RBF kernel, and the scalability is addressed through the variational sparse approach.

In [8], an RL setting is considered that is closely related to our meta-learning set-up. The authors use a parametric policy that depends on a known deterministic task variable and augment the policy function to include it as well. In [8], the authors consider the same dynamical system but solve different tasks by augmenting the policy with a task variable. In our work, we look at different settings of the dynamics but the task remains the same. We show how to generalize to the setting where task variables are latent and inferred from interaction data. This dramatically extends applicability in real-world settings.

## 6 CONCLUSION

We proposed a meta learning approach within the context of model-based RL that allows us to transfer knowledge from training configurations of robotic systems to unseen test configurations. The key idea behind our approach is to address the meta learning problem probabilistically using a latent variable model. We use online variational inference to obtain a posterior distribution over the latent variable, which describes the relatedness of tasks. This posterior is then used for long-term predictions of the state evolution and controller learning within a model-based RL setting. We demonstrated that our ML-GP approach is as efficient or better than a non-meta learning baseline when solving multiple tasks at once. The ML-GP further generalizes well to learning models and controllers for unseen tasks giving rise to substantial improvements in data-efficiency on novel tasks.

### Acknowledgements

This work was supported by Microsoft Research through its PhD Scholarship Programme.

## References

- [1] M. Al-Shedivat, T. Bansal, Y. Burda, I. Sutskever, I. Mordatch, and P. Abbeel. Continuous adaptation via meta-learning in nonstationary and competitive environments. In *International Conference on Learning Representations (ICLR)*, 2018.
- [2] M. A. Álvarez, L. Rosasco, and N. D. Lawrence. Kernels for vector-valued functions: A review. *Foundations and Trends in Machine Learning (FTML)*, 4(3):195, 2012.
- [3] S. Barrett, M. Taylor, and P. Stone. Transfer learning for reinforcement learning on a physical robot. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2010.
- [4] A. Cully, J. Clune, D. Tarapore, and J.-B. Mouret. Robots that can adapt like animals. *Nature*, 521:503–507, 2015.
- [5] B. da Silva, G. Konidaris, and A. Barto. Learning parametrized skills. In *International Conference on Machine Learning (ICML)*, 2012.
- [6] Z. Dai, M. A. Álvarez, and N. D. Lawrence. Efficient modeling of latent information in supervised learning using Gaussian processes. In *Neural Information Processing Systems (NIPS)*. 2017.
- [7] A. Damianou and N. D. Lawrence. Semi-described and semi-supervised learning with Gaussian processes. *Uncertainty in Artificial Intelligence (UAI)*, 2015.
- [8] M. P. Deisenroth, P. Englert, J. Peters, and D. Fox. Multi-task policy search for robotics. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.
- [9] M. P. Deisenroth, D. Fox, and C. E. Rasmussen. Gaussian processes for data-efficient learning in robotics and control. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 37(2):408–423, 2015.
- [10] M. P. Deisenroth and C. E. Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *International Conference on Machine Learning (ICML)*, 2011.
- [11] F. Doshi-Velez and G. Konidaris. Hidden parameter Markov decision processes: A semiparametric regression approach for discovering latent task parametrizations. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.
- [12] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning (ICML)*, 2017.
- [13] E. Grant, C. Finn, S. Levine, T. Darrell, and T. Griffiths. Recasting gradient-based meta-learning as hierarchical Bayes. In *International Conference on Learning Representations (ICLR)*, 2018.
- [14] J. Hensman, N. Fusi, and N. D. Lawrence. Gaussian processes for big data. In *Uncertainty in Artificial Intelligence (UAI)*, 2013.
- [15] M. Hoffman, D. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *Journal of Machine Learning Research (JMLR)*, pages 1303–1347, 2013.
- [16] A. Ijspeert, J. Nakanishi, and S. Schaal. Learning attractor landscapes for learning motor primitives. In *Neural Information Processing Systems (NIPS)*, 2002.
- [17] S. Kamthe and M. P. Deisenroth. Data-efficient reinforcement learning with probabilistic model predictive control. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018.
- [18] T. Killian, S. Daulton, G. Konidaris, and F. Doshi-Velez. Robust and efficient transfer learning with hidden parameter Markov decision processes. In *Neural Information Processing Systems (NIPS)*, Long Beach, CA, 2017.
- [19] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.
- [20] J. Kober, E. Otzop, and J. Peters. Reinforcement learning to adjust robot movements to new situations. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2011.
- [21] G. Konidaris, I. Scheidwasser, and A. Barto. Transfer in reinforcement learning via shared features. *Journal of Machine Learning Research (JMLR)*, 13:1333–1371, 2012.
- [22] O. Kroemer, R. Detry, J. Piater, and J. Peters. Combining active learning and reactive control for robot grasping. *Robotics and Autonomous Systems (RAS)*, 58:1105–1116, 2010.
- [23] N. D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *Neural Information Processing Systems (NIPS)*. 2004.

- [24] K. Mülling, J. Kober, O. Kroemer, and J. Peters. Learning to select and generalize striking movements in robot table tennis. *International Journal of Robotics Research (IJRR)*, 2013.
- [25] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.
- [26] T. Schaul and J. Schmidhuber. Metalearning. *Scholarpedia*, 5(6):4650, 2010.
- [27] J. G. Schneider. Exploiting model uncertainty estimates for safe dynamic control learning. In *Neural Information Processing Systems (NIPS)*. 1997.
- [28] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998.
- [29] M. Taylor and P. Stone. Cross-domain transfer for reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2007.
- [30] M. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2009.
- [31] M. Titsias and N. D. Lawrence. Bayesian Gaussian process latent variable model. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.
- [32] R. Vilalta and Y. Drissi. A perspective view and survey of meta-learning. *Artificial Intelligence Review (AI Review)*, 18(2):77–95, 2002.
- [33] C. Wang and R. M. Neal. Gaussian Process Regression with Heteroscedastic or Non-Gaussian Residuals. *ArXiv e-prints*, Dec. 2012.



---

# Non-Parametric Path Analysis in Structural Causal Models

---

Junzhe Zhang and Elias Bareinboim  
Purdue University, USA  
{zhang745, eb}@purdue.edu

## Abstract

One of the fundamental tasks in causal inference is to decompose the observed association between a decision  $X$  and an outcome  $Y$  into its most basic structural mechanisms. In this paper, we introduce counterfactual measures for effects along with a specific mechanism, represented as a path from  $X$  to  $Y$  in an arbitrary structural causal model. We derive a novel non-parametric decomposition formula that expresses the covariance of  $X$  and  $Y$  as a sum over unblocked paths from  $X$  to  $Y$  contained in an arbitrary causal model. This formula allows a fine-grained path analysis without requiring a commitment to any particular parametric form, and can be seen as a generalization of Wright’s decomposition method in linear systems (1923,1932) and Pearl’s non-parametric mediation formula (2001).

## 1 INTRODUCTION

Analyzing the relative strength of different pathways between a decision  $X$  and an outcome  $Y$  is a topic that has interested both scientists and practitioners across disciplines for many decades. Specifically, path analysis allows scientists to explain how Nature’s “black-box” works, and practically, it enables decision analysts to predict how an environment will change under a variety of policies and interventional conditions [Wright, 1923; Baron and Kenny, 1986; Bollen, 1989; Pearl, 2001].

More recently, understanding using causal inference tools how a black-box decision-making system operates has been a target of growing interest in the Artificial Intelligence community, most prominently in the context of Explainability, Transparency, and Fairness [Lu Zhang, 2017; Kusner *et al.*, 2017; Zafar *et al.*, 2017; Kilbertus *et al.*, 2017; Zhang and Bareinboim, 2018a]. For exam-

ple, consider the *standard fairness model* described in Fig. 1(a) that is concerned with the relation between a hiring decision ( $Y$ ) and an applicant’s religious beliefs ( $X$ ), which are *mediated* by the location ( $W$ ), and *confounded* by the education background ( $Z$ ) of the applicant.<sup>1</sup> Directed edges represent functional relations between variables. The relationship between  $X$  and  $Y$  is materialized through four different pathways in the system – the *direct* path  $l_1 : X \rightarrow Y$ , the *indirect* path  $l_2 : X \rightarrow W \rightarrow Y$ , and the *spurious* paths  $l_3 : X \leftarrow Z \rightarrow Y$  and  $l_4 : X \leftarrow Z \rightarrow W \rightarrow Y$ .

Assuming, for simplicity’s sake, that the functional relationships are linear and  $U_{V_i}$  is an independent “error” associated with each variable  $V_i$  (called the linear-standard model), Fig. 1(a) shows the structural coefficients corresponding to each edge – i.e., the value of the variable  $Y$  is decided by the structural function  $Y \leftarrow \alpha_{YX}X + \alpha_{YZ}Z + \alpha_{YW}W + U_Y$ . The celebrated result known as Wright’s method of path coefficients [Wright, 1923, 1934], also known as Wright’s rule, allows one to express the covariance of  $X$  and  $Y$ , denoted by  $\text{Cov}(X, Y)$ , as the sum of the products of the structural coefficients along the paths from  $X$  to  $Y$  in the underlying causal model.<sup>2</sup> In particular,  $\text{Cov}(X, Y)$  is equal to:

$$\underbrace{\alpha_{YX}}_{X \rightarrow Y} + \underbrace{\alpha_{WX}\alpha_{YW}}_{X \rightarrow W \rightarrow Y} + \underbrace{\alpha_{XZ}\alpha_{YZ}}_{X \leftarrow Z \rightarrow Y} + \underbrace{\alpha_{XZ}\alpha_{WZ}\alpha_{YW}}_{X \leftarrow Z \rightarrow W \rightarrow Y}. \quad (1)$$

Using the observational covariance matrix, the decomposition above allows one to answer some compelling questions about the relationship between  $X$  and  $Y$  in the underlying model. For instance, the product  $\alpha_{WX}\alpha_{YW}$  explains how much the indirect discrimination through the location (the path  $l_2$ ) accounts for the observed disparities in the religion composition among hired employees.

The path analysis method gained momentum in the so-

---

<sup>1</sup>This specific setting has been called *standard fairness model* given its generality to representing a variety of decision-making scenarios [Zhang and Bareinboim, 2018a].

<sup>2</sup>For a survey on linear methods, see [Pearl, 2000, Ch. 5].

cial sciences during 1960's, becoming extremely popular in the form of the *mediation formula* in which the total effect of  $X$  on  $Y$  is decomposed into direct and indirect components [Baron and Kenny, 1986; Bollen, 1989; Duncan, 1975; Fox, 1980].<sup>3</sup> The bulk of this literature, however, required a commitment to a particular parametric form, thus falling short of providing a general method for analyzing natural and social phenomena with nonlinearities and interactions [MacKinnon, 2008].

It took a few decades until this problem could be tackled in higher generality. In particular, the advent of non-parametric structural causal models (SCMs) allowed this leap, and a more fine-grained path-analysis with a much broader scope, including models with nonlinearities and arbitrarily complex interactions [Pearl, 2000, Ch. 7]. In particular, Pearl introduced the *causal mediation formula* for arbitrary non-parametric models, which decomposes the total effect  $TE_{x_0, x_1}(Y) = E[Y_{x_1}] - E[Y_{x_0}]$ , the difference between the causal effect of the intervention  $do(x_1)$  and  $do(x_0)$ <sup>4</sup>, into what is now known as the natural direct (*NDE*) and indirect (*NIE*) effects [Pearl, 2001] (see also [Imai *et al.*, 2010, 2011; VanderWeele, 2015]). In the case of the specific linear-standard causal model,

$$TE_{0,1}(Y) = \underbrace{\alpha_{YX}}_{NDE} + \underbrace{\alpha_{WX}\alpha_{YW}}_{NIE}$$

for  $x_0 = 0$  and  $x_1 = 1$  levels. Remarkably, when compared with Eq. 1, *NDE* and *NIE* capture the effects along with the direct and indirect paths, but omits the spurious (non-causal) paths between  $X$  and  $Y$  (in this case,  $l_3, l_4$ ). The mediation formula was recently generalized to account for these spurious paths (more akin to Wright's rules), which appears under the rubric of the *causal explanation formula* [Zhang and Bareinboim, 2018a]. This formula decomposes the total variation  $TV_{x_0, x_1}(Y) = E[Y|x_1] - E[Y|x_0]$  (difference in conditional distributions) into counterfactual measures of the direct (*Ctf-DE*), indirect (*Ctf-IE*), and spurious (*Ctf-SE*) effects. In the linear-standard model, for  $x_0 = 0, x_1 = 1$ ,

$$TV_{0,1}(Y) = \underbrace{\alpha_{YX}}_{Ctf-DE} + \underbrace{\alpha_{WX}\alpha_{YW}}_{Ctf-IE} + \underbrace{\alpha_{XZ}\alpha_{YZ} + \alpha_{XZ}\alpha_{WZ}\alpha_{YW}}_{Ctf-SE}$$

Despite the generality of such results, there are still outstanding challenges when performing path analysis in non-parametric models, i.e.: (1) Estimands are defined relative to specific values assigned to the treatment  $x_1$  and its baseline  $x_0$ , which may be difficult to select in some non-linear settings; (2) Mediators and confounders

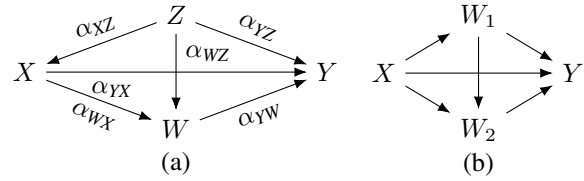


Figure 1: Causal diagrams for (a) the standard fairness model where  $X$  stands for the protected attribute,  $Y$  for the outcome,  $W$  the mediators, and  $Z$  the confounders; (b) the two-mediators setting where causal paths from  $X$  to  $Y$  are mediated by  $W_1, W_2$ .

are collapsed and considered *en bloc*, leading to a coarse decomposition of the relationship between  $X$  and  $Y$  [Pearl, 2001; Vansteelandt and VanderWeele, 2012; Tchetgen and Shpitser, 2012; VanderWeele *et al.*, 2014; Daniel *et al.*, 2015; Zhang and Bareinboim, 2018a]; (3) Path-specific estimands are well-defined [Pearl, 2001; Avin *et al.*, 2005], but not in a way that they sum up to either the total effect (TE) or variation (TV), precluding the comparison of their relative strengths.

This paper aims to circumvent these problems. In particular, we decompose the covariance of a treatment  $X$  and an outcome  $Y$  over effects along different mechanisms between  $X$  and  $Y$ . We define a set of novel counterfactual estimands for measuring the relative strength of a specific mechanism represented as a path from  $X$  to  $Y$  in an arbitrary causal model. These estimands lead to a non-parametric decomposition formula, which expresses the covariance  $\text{Cov}(X, Y)$  as a sum of the unblocked paths from  $X$  to  $Y$  in the causal graph. This formula allows a more fine-grained analysis of the total observed variations of  $Y$  due to  $X$  (both through causal and spurious mechanisms) when compared to the state-of-art methods. More specifically, our contributions are: (1) counterfactual covariance measures for a specific pathway from  $X$  to  $Y$  (causal and spurious) in an arbitrary causal model (Defs. 8, 11-12); (2) non-parametric decomposition formulae of the covariance  $\text{Cov}(X, Y)$  over paths from  $X$  to  $Y$  in the causal model (Thm. 5); (3) the identification formulae estimating the proposed path-specific decomposition from the passively-collected data in the standard model (Thms. 6-7).

## 2 PRELIMINARIES

In this section, we introduce notations used throughout the paper. We will use capital letters to denote variables (e.g.,  $X$ ), and small letters for their values ( $x$ ). The abbreviation  $P(x)$  represents the probabilities  $P(X = x)$ . For arbitrary sets  $A$  and  $B$ , let  $A - B$  denote their differ-

<sup>3</sup>Just to give an idea of this popularity, Baron and Kenny's original paper counts more than 70,000 citations.

<sup>4</sup>By convention [Pearl, 2000], the post-interventional distribution is represented interchangeably by  $P(y_x)$  and  $P(y|do(x))$ . General notation is discussed in the next section.

ence, and let  $|A|$  be the dimension of set  $A$ .  $V_{[i,j]}$  stands for a set  $\{V_i, \dots, V_j\}$  ( $\emptyset$  if  $i > j$ ). We use graphical family abbreviations:  $An(X)_G$ ,  $De(X)_G$ ,  $Non-De(X)_G$ ,  $Pa(X)_G$ ,  $Ch(X)_G$ , which stand for the set of ancestors, descendants, non-descendants, parents and children of  $X$  in  $G$ . We omit the subscript  $G$  when obvious.

The basic semantical framework of our analysis rests on *structural causal models* (SCM) [Pearl, 2000, Ch. 7; Bareinboim and Pearl, 2016]. A SCM  $M$  consists of a set of endogenous variables  $V$  (often observed) and exogenous variables  $U$  (often unobserved). The values of each  $V_i \in V$  are determined by a structural function  $f_i$  taking as argument a combination of the other endogenous and exogenous variables (i.e.,  $V_i \leftarrow f_i(PA_i, U_i)$ ,  $PA_i \subseteq V, U_i \subseteq U$ ). Values of  $U$  are drawn from a distribution  $P(u)$ . A SCM  $M$  is called *Markovian* when the exogenous are mutually independent and each  $U_i \in U$  is associated with only one endogenous  $V_i \in V$ . If  $U_i$  is associated with two or more endogenous variables,  $M$  is called *semi-Markovian*.

Each recursive SCM  $M$  has an associated causal diagram in the form of a directed acyclic graph (DAG)  $G$ , where nodes represent endogenous variables and directed edges represent functional relations (e.g., Figs. 1-2). By convention, the exogenous  $U$  are not explicitly shown in the graph; a dashed-bidirected arrow between  $V_i$  and  $V_j$  indicates the presence of an unobserved confounder (UC)  $U_k$  affecting both  $V_i$  and  $V_j$  (e.g., the path  $V_i \leftarrow U_k \rightarrow V_j$ ).

A path from  $X$  to  $Y$  is a sequence of edges which does not include a particular node more than once. It may go either along or against the direction of the edges. Paths of the form  $X \rightarrow \dots \rightarrow Y$  are *causal* (from  $X$  to  $Y$ ). We use d-separation and blocking interchangeably, following the convention in [Pearl, 2000]. Any unblocked path that is not causal is called *spurious*. The direct link  $X \rightarrow Y$  is the *direct* path and all the other causal paths from  $X$  to  $Y$  are called *indirect*. The set of unblocked paths from  $X$  to  $Y$  given a set  $Z$  in a causal diagram  $G$  is denoted by  $\Pi(X, Y|Z)_G$ ; causal, indirect, and spurious paths are denoted by  $\Pi^c(X, Y|Z)_G$ ,  $\Pi^i(X, Y|Z)_G$ , and  $\Pi^s(X, Y|Z)_G$  ( $G$  will be omitted when obvious). For a causal path  $g$  including nodes  $V_1, V_2$ , we denote  $g(V_1, V_2)$  a subpath of  $g$  from  $V_1$  to  $V_2$ .<sup>5</sup>

An intervention on a set of endogenous variables  $X$  and exogenous variables  $U_i$ , denoted by  $do(x^*, u_i^*)$ , is an operation where values of  $X, U_i$  are set to  $x^*, u_i^*$ , respectively, without regard to how they were ordinarily determined ( $X$  through  $f_X$  and  $U_i$  through  $P(U_i)$ ). Formally, we can rewrite the definition of potential response [Pearl, 2000, Ch. 7.1] to account for operation on  $U_i$ , namely:

<sup>5</sup>Mediators (relative to  $X$  and  $Y$ ) are a set of variables  $W \subseteq De(X) \cap Non-De(Y)$  such that  $|\Pi^s(X, Y|W)| = 0$ .

**Definition 1** (Potential Response). Let  $M$  be a SCM,  $X, Y$  sets of arbitrary variables in  $V$ , and  $U_i$  a set of arbitrary variables in  $U$ . Let  $U_{-i} = U - U_i$ . The potential response of  $Y$  to the intervention  $do(x^*, u_i^*)$  in the situation  $U = u$ , denoted by  $Y_{x^*, u_i^*}(u)$ , is the solution for  $Y$  with  $U_{-i} = u_{-i}, U_i = u_i^*$  in the modified submodel  $M_{x^*}$  where functions  $f_X$  are replaced by constant functions  $X = x^*$ , i.e.,  $Y_{x^*, u_i^*}(u) \triangleq Y_{M_{x^*}}(u_i^*, u_{-i})$ .<sup>6</sup>

$Y_{x^*, u_i^*}(u)$  can be read as the counterfactual sentence “the value that  $Y$  would have obtained in situation  $U_{-i} = u_{-i}$ , had the treatment  $X$  been  $x^*$  and the situation  $U_i$  been  $u_i^*$ .” Averaging  $u$  over the distribution  $P(u)$ , we obtain a counterfactual random variable  $Y_{x^*, u_i^*}$ . If the values of  $x^*, u_i^*$  follow random variables  $X^*, U_i^*$ , we denote the resulting counterfactual  $Y_{X^*, U_i^*}$ .

### 3 A COARSE COVARIANCE DECOMPOSITION

In this section, we introduce counterfactual measures that will allow us to non-parametrically decompose the covariance  $Cov(X, Y)$  in terms of direct, indirect and spurious pathways from  $X$  to  $Y$ . Given space constraints, all proofs are included in [Zhang and Bareinboim, 2018b].

If there exists no spurious path from  $X$  to  $Y$ , then treatment  $X$  is independent of the counterfactual  $Y_{x^*}$ , i.e.,  $(X \perp\!\!\!\perp Y_{x^*})$  [Pearl, 2000, Ch. 11.3.2]. The *spurious covariance* can then be defined as the correlation between the factual variable  $X$  and counterfactual  $Y_{x^*}$ .

**Definition 2** (Spurious Covariance). The spurious covariance between treatment  $X = x^*$  and outcome  $Y$  is:

$$Cov_{x^*}^s(X, Y) = Cov(X, Y_{x^*}). \quad (2)$$

**Property 1.**  $|\Pi^s(X, Y)| = 0 \Rightarrow Cov_{x^*}^s(X, Y) = 0$ .

The *causal covariance* can naturally be defined as the difference between the total and spurious covariance.

**Definition 3** (Causal Covariance). The causal covariance of the treatment  $X = x^*$  and the outcome  $Y$  is:

$$Cov_{x^*}^c(X, Y) = Cov(X, Y - Y_{x^*}). \quad (3)$$

Prop. 2 establishes the correspondence between the causal paths and the causal covariance – if there is no causal path from  $X$  to  $Y$  in the underlying model, the causal covariance equates to zero.

**Property 2.**  $|\Pi^c(X, Y)| = 0 \Rightarrow Cov_{x^*}^c(X, Y) = 0$ .

We consider more detailed measures corresponding to the different causal pathways, and first, the direct path:

<sup>6</sup>An alternative way to see that the replacement operation relative to  $U_i$  is to envision a system where  $U_i$  is observed.

**Definition 4** (Direct Covariance). Given a semi-Markovian model  $M$ , let the set  $W$  be the mediators between  $X$  and  $Y$ . The pure ( $\text{Cov}_{x^*}^{dp}(X, Y)$ ) and total ( $\text{Cov}_{x^*}^{dt}(X, Y)$ ) direct covariance of the treatment  $X = x^*$  on the outcome  $Y$  are defined respectively as

$$\text{Cov}_{x^*}^{dp}(X, Y) = \text{Cov}(X, Y - Y_{x^*, W}), \quad (4)$$

$$\text{Cov}_{x^*}^{dt}(X, Y) = \text{Cov}(X, Y_{W_{x^*}} - Y_{x^*}). \quad (5)$$

By the composition axiom [Pearl, 2000, Ch. 7.3], Eqs. 4 and 5 can be explicitly written as follows<sup>7</sup>:

$$\begin{aligned} \text{Cov}(X, Y - Y_{x^*, W}) &= \text{Cov}(X, Y_{X, W} - Y_{x^*, W}), \\ \text{Cov}(X, Y_{W_{x^*}} - Y_{x^*}) &= \text{Cov}(X, Y_{X, W_{x^*}} - Y_{x^*, W_{x^*}}). \end{aligned}$$

The counterfactual pure direct covariance (Eq. 4) is shown graphically in Fig. 2, where (a) corresponds to the  $Y$ -side, and (b) to the  $Y_{x^*, W}$ -side. Note that from the mediator  $W$  perspective,  $X$  remains at the level that it would naturally have attained, while the “direct” input from  $X$  to  $Y$  varies from its natural level (Fig. 2a) to  $do(x^*)$  (b). The change of the outcome  $Y$  thus measures the effect of the direct path. A similar analysis also applies to the total direct covariance (Eq. 5).

**Property 3.**  $\text{Cov}_{x^*}^{dp}(X, Y) = \text{Cov}_{x^*}^{dt}(X, Y) = 0$  if  $X$  is not a parent of  $Y$  (i.e.,  $X \notin Pa(Y)$ ).

We can turn around the definitions of direct covariance and provide operational estimands for indirect paths.

**Definition 5** (Indirect Covariance). Given a semi-Markovian model  $M$ , let the set  $W$  be the mediators between  $X$  and  $Y$ . The pure ( $\text{Cov}_{x^*}^{ip}(X, Y)$ ) and total ( $\text{Cov}_{x^*}^{it}(X, Y)$ ) indirect covariance of the treatment  $X = x^*$  on the outcome  $Y$  are defined respectively as:

$$\text{Cov}_{x^*}^{ip}(X, Y) = \text{Cov}(X, Y - Y_{W_{x^*}}), \quad (6)$$

$$\text{Cov}_{x^*}^{it}(X, Y) = \text{Cov}(X, Y_{x^*, W} - Y_{x^*}). \quad (7)$$

Eqs. 6 and 7 correspond to the indirect paths, since they capture the covariance of  $X$  and  $Y$ , but only via paths mediated by  $W$ . The first argument of  $Y$  is the same in both halves of the contrast, but this value can either be  $x^*$  (Eq. 7) or at the level that  $X$  would naturally attain without intervention (Eq. 6).

**Property 4.**  $|\Pi^i(X, Y)| = 0 \Rightarrow \text{Cov}_{x^*}^{ip}(X, Y) = \text{Cov}_{x^*}^{it}(X, Y) = 0$ .

Putting these definitions together, we can prove a general non-parametric decomposition of  $\text{Cov}(X, Y)$ :

<sup>7</sup>Consider Eq. 4 as an example. For any  $U = u$ ,  $Y_{X(u), W(u)}(u) = Y_{x^*, w}(u)$  if  $X(u) = x^*$ ,  $W(u) = w$ . By the composition axiom,  $X(u) = x^*$ ,  $W(u) = w$  implies  $Y(u) = Y_{x^*, w}(u)$ , which in turn gives  $Y_{X(u), W(u)}(u) = Y(u)$ . Averaging  $u$  over  $P(u)$ , we obtain  $Y_{X, W} = Y$ .

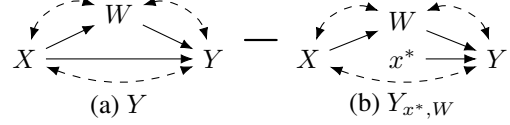


Figure 2: The graphical representation of measuring the pure direct covariance  $\text{Cov}_{x^*}^{dp}(X, Y)$ .

**Theorem 1.**  $\text{Cov}(X, Y)$ ,  $\text{Cov}_{x^*}^s(X, Y)$  and  $\text{Cov}_{x^*}^c(X, Y)$  obey the following non-parametric relationship:

$$\text{Cov}(X, Y) = \text{Cov}_{x^*}^c(X, Y) + \text{Cov}_{x^*}^s(X, Y), \quad (8)$$

where  $\text{Cov}_{x^*}^c(X, Y) = \text{Cov}_{x^*}^{dp}(X, Y) + \text{Cov}_{x^*}^{it}(X, Y) = \text{Cov}_{x^*}^{dt}(X, Y) + \text{Cov}_{x^*}^{ip}(X, Y)$ .

In other words, the covariance between  $X$  and  $Y$  can be partitioned into its corresponding direct, indirect, and spurious components. In particular, Thm. 1 coincides with Eq. 1 in the linear-standard model.

**Corollary 1.** In the linear-standard model, for any  $x^*$ ,  $\text{Cov}_{x^*}^s(X, Y)$ ,  $\text{Cov}_{x^*}^{dp}(X, Y)$ ,  $\text{Cov}_{x^*}^{dt}(X, Y)$ ,  $\text{Cov}_{x^*}^{ip}(X, Y)$  and  $\text{Cov}_{x^*}^{it}(X, Y)$  are equal to:

$$\text{Cov}_{x^*}^s(X, Y) = \alpha_{XZ}\alpha_{YZ} + \alpha_{XZ}\alpha_{WZ}\alpha_{YW},$$

$$\text{Cov}_{x^*}^{dp}(X, Y) = \text{Cov}_{x^*}^{dt}(X, Y) = \alpha_{YX},$$

$$\text{Cov}_{x^*}^{ip}(X, Y) = \text{Cov}_{x^*}^{it}(X, Y) = \alpha_{WX}\alpha_{YW}.$$

Corol. 1 says that the proposed decomposition (Thm. 1) does not depend on the value of  $do(x^*)$  in the linear model of Fig. 1(a), which is not achievable in previous value-specific decompositions [Pearl, 2001; Zhang and Bareinboim, 2018a].<sup>8</sup>

## 4 DECOMPOSING CAUSAL RELATIONS

We now focus on the challenge of decomposing the causal covariance into more elementary components. We use the two-mediators setting (Fig. 1(b)) as example, where  $X$  and  $Y$  are connected through four causal paths: through both  $W_1, W_2$  ( $g_1 : X \rightarrow W_1 \rightarrow W_2 \rightarrow Y$ ), only through  $W_1$  ( $g_2 : X \rightarrow W_1 \rightarrow Y$ ), only through  $W_2$  ( $g_3 : X \rightarrow W_2 \rightarrow Y$ ), and directly ( $g_4 : X \rightarrow Y$ ). Our goal is to decompose the  $\text{Cov}_{x^*}^c(X, Y)$  over the paths  $g_{[1,4]}$ . Our analysis applies to semi-Markovian models, without loss of generality, and the Markovian example (Fig. 1(b)) is used for simplicity of the exposition.

<sup>8</sup>For the nonlinear models, the decomposing terms (e.g.,  $\text{Cov}_{x^*}^s(X, Y)$ ) are still sensitive to the target level  $do(x^*)$ . To circumvent the challenges of picking a specific decision value, one could assign a randomized treatment  $do(x^* \sim P(X))$ , where  $P(X)$  is the distribution over the treatment  $X$  induced by the underlying causal model.

For a node  $S_i \in Pa(Y)$  and a set of causal paths  $\pi$ , the edge  $S_i \rightarrow Y$  defines a funnel operator  $\triangleleft_{S_i \rightarrow Y}$ , which maps from  $\pi$  to the set of paths  $\triangleleft_{S_i \rightarrow Y}(\pi)$  obtained from  $\pi$  by replacing all paths of the form  $X \rightarrow \dots \rightarrow S_i \rightarrow Y$  with  $X \rightarrow \dots \rightarrow S_i$ , and removing all the other paths. As an example, for  $\pi = \{g_1, g_2, g_3\}$ ,  $\triangleleft_{W_2 \rightarrow Y}(\pi) = \{g_1(X, W_2), g_3(X, W_2)\}$ , where  $g_1(X, W_2)$  is the subpath  $X \rightarrow W_1 \rightarrow W_2$  and  $g_3(X, W_2)$  is the subpath  $X \rightarrow W_2$ . We next formalize the notion of path-specific interventions, which isolates the influence of the intervention  $do(x^*)$  passing through a subset  $\pi$  of causal paths from  $X$ , denoted by  $do(\pi[x^*])$  (a similar notion has been introduced by [Pearl, 2001], and then [Avin *et al.*, 2005; Shpitser and Tchetgen, 2016]).

**Definition 6 (Path-Specific Potential Response).** For a SCM  $M$  and an arbitrary variable  $Y \in V$ , let  $\pi$  be a set of causal paths. Let  $X$  be the source variables of paths in  $\pi$ . Further, let  $X_{\pi \rightarrow Y} = \{X_i : \forall X_i \in X, X_i \rightarrow Y \in \pi\}$  and  $S = (Pa(Y)_G \cap V) - X_{\pi \rightarrow Y}$ . The  $\pi$ -specific potential response of  $Y$  to the intervention  $do(\pi[x^*])$  in the situation  $U = u$ , denoted by  $Y_{\pi[x^*]}(u)$ , is defined as:

$$Y_{\pi[x^*]}(u) = \begin{cases} Y_{x^*, S_{\triangleleft_{S_i \rightarrow Y}(\pi)[x^*]}(u)} & \text{if } \pi \neq \emptyset \\ Y(u) & \text{otherwise} \end{cases}$$

where  $S_{\triangleleft_{S_i \rightarrow Y}(\pi)[x^*]}(u)$  is a set of  $\pi$ -specific potential response  $\{S_i \in S : S_i \rightarrow Y \in \pi[x^*]\}$ .<sup>9</sup>

Despite the non-trivial notation, the  $\pi$ -specific counterfactual  $Y_{\pi[x^*]}$  is simply assigning the treatment  $do(x^*)$  exclusively to the causal paths in  $\pi$ , while allowing all the other causal paths to behave naturally. This contrasts with the counterfactual  $Y_{x^*}$ , which can be seen as assigning the treatment  $do(x^*)$  to all causal paths from  $X$  to  $Y$ . For instance, repeatedly applying Def. 6 to  $g_1 : X \rightarrow W_1 \rightarrow W_2 \rightarrow Y$  (see [Zhang and Bareinboim, 2018b, Sec. 2.1]), we obtain the  $g_1$ -specific potential response  $Y_{g_1[x^*]}$  as

$$Y_{g_1[x^*]} = Y_{X, W_1, W_2, X, W_1, x^*} = Y_{W_2, W_1, x^*}.$$

The intervention  $do(g_1[x^*])$  can be visualized more immediately through its graphical representation (Fig. 3(b)) – the treatment  $do(x^*)$  is assigned throughout  $g_1$  while all the other paths are kept at the level that it would have attained “naturally” following  $X$ . The difference of the outcome  $Y$  (induced by  $do(g_1[x^*])$ ) and the unintervened  $Y$  (Fig. 3(a)) measures the relative strength of  $g_1$  itself, which leads to the following definition.

**Definition 7 (Pure Path-Specific Causal Covariance).** For a semi-Markovian model  $M$  and an arbitrary causal path  $g$  from  $X$ , the pure  $g$ -specific causal covariance of the treatment  $X = x^*$  on the outcome  $Y$  is defined as:

$$\text{Cov}_g^c(x^*)(X, Y) = \text{Cov}(X, Y - Y_{g[x^*]}). \quad (9)$$

<sup>9</sup>For a single causal path  $g$ , let  $Y_{g[x^*]}(u) = Y_{\{g\}[x^*]}(u)$ . Averaging  $u$  over  $P(u)$ , we obtain a random variable  $Y_{\pi[x^*]}$ .

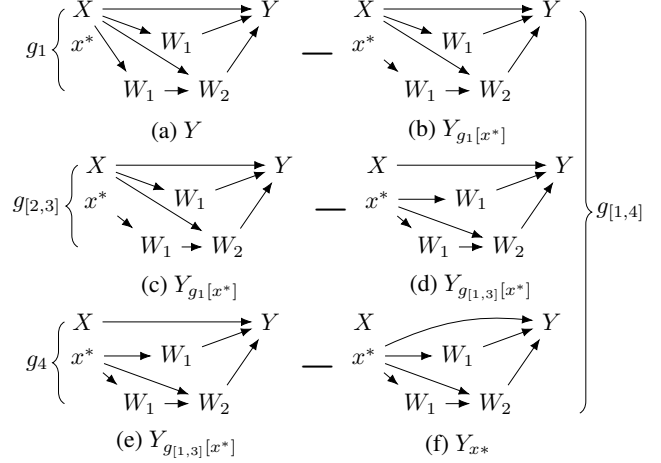


Figure 3: Graphical representations of the causal covariance specific to  $g_1$  (a-b),  $g_{[2,3]}$  (c-d) and  $g_4$  (e-f).

In the previous example, more explicitly, the pure  $g_1$ -specific causal covariance is equal to (Fig. 3(a-b)):

$$\text{Cov}_{g_1[x^*]}^c(X, Y) = \text{Cov}(X, Y - Y_{W_2, W_1, x^*}). \quad (10)$$

For  $U = u$ , the counterfactual  $Y_{\emptyset[x^*]}(u)$  stands for the values of  $Y$  when all causal paths are under the natural regime. Eq. 9 can then be rewritten as:

$$\text{Cov}_g^c(x^*)(X, Y) = \text{Cov}(X, Y_{\emptyset[x^*]} - Y_{g[x^*]}).$$

The pure path-specific causal covariance for  $g$  can be seen as a function of the difference between two path-specific potential response  $Y_{\pi_0[x^*]}$  and  $Y_{\pi_1[x^*]}$  such that  $g \notin \pi_0$  and  $\pi_1 = \pi_0 \cup \{g\}$  (i.e., the different between  $\pi_1$  and  $\pi_0$  is  $g$ ). The difference  $Y_{\pi_1[x^*]} - Y_{\pi_0[x^*]}$ , therefore, measures precisely the effects of  $do(x^*)$  along the target causal path  $g$ . Def. 7 can be generalized to account for the path-specific covariance in terms of path-differences.

**Definition 8 (Path-Specific Causal Covariance).** For a semi-Markovian model  $M$  and an arbitrary causal path  $g$  from  $X$ , let  $\pi$  be a function mapping  $g$  to a set of causal paths  $\pi(g)$  from  $X$  such that  $g \notin \pi(g)$ . The  $g$ -specific causal covariance of the treatment  $X = x^*$  on the outcome  $Y$  is defined as:

$$\text{Cov}_g^c(x^*)(X, Y)_\pi = \text{Cov}(X, Y_{\pi(g)[x^*]} - Y_{\pi(g) \cup \{g\}[x^*]}).$$

The following property establishes the correspondence between a causal path and its path-specific estimand.

**Property 5.**  $g \notin \Pi^c(X, Y) \Rightarrow \text{Cov}_g^c(x^*)(X, Y)_\pi = 0$ .

Prop. 5 follows immediately as a corollary of Lem. 1, which implies that the counterfactuals  $Y_{\pi(g)[x^*]}$  and  $Y_{\pi(g) \cup \{g\}[x^*]}$  define the same variable over  $U$  if  $g$  is not a causal path from  $X$  to  $Y$ .

**Lemma 1.**  $g \notin \Pi^c(X, Y) \Rightarrow Y_{\pi(g)[x^*]}(u) = Y_{\pi(g) \cup \{g\}[x^*]}(u)$ .

Considering again the model in Fig. 1(b), let  $g_{[i,j]} = \{g_k\}_{i \leq k \leq j}$  ( $\emptyset$  if  $i > j$ ). Recall that  $g_4 = \{X \rightarrow Y\}$ , and note that the  $g_4$ -specific causal covariance can be computed using  $\pi(g_4) = g_{[1,3]}$ , which yields:

$$\begin{aligned} \text{Cov}_{g_4[x^*]}^c(X, Y)_\pi &= \text{Cov}(X, Y_{g_{[1,3]}[x^*]} - Y_{g_{[1,4]}[x^*]}) \\ &= \text{Cov}(X, Y_{W_{1,x^*}, W_{2,x^*}} - Y_{x^*}), \end{aligned} \quad (11)$$

which coincides with the direct effect (Eq. 5 with  $W = \{W_1, W_2\}$ ). Fig. 3(e-f) shows a graphical representation of this procedure.

The path-specific quantity given in Def. 8 has another desirable property, namely, the causal covariance  $\text{Cov}_x^c(X, Y)$  can be decomposed as a summation over causal paths from  $X$  to  $Y$ . To witness, first let an order over  $\Pi^c(X, Y)$  be  $\mathcal{L}^c : g_1 < \dots < g_n$ . For a path  $g_i \in \Pi^c(X, Y)$ , the order  $\mathcal{L}^c$  defines a function  $\mathcal{L}_\pi^c$  which maps from  $g_i$  to a set of paths  $\mathcal{L}_\pi^c(g_i)$  that precede  $g_i$  in  $\mathcal{L}^c$ , i.e.,  $\mathcal{L}_\pi^c(g_i) = g_{[1, i-1]}$ . We derive in the sequel a path-specific decomposition formula for the causal covariance relative to an order  $\mathcal{L}^c$ .

**Theorem 2.** *For a semi-Markovian model  $M$ , let  $\mathcal{L}^c$  be an order over  $\Pi^c(X, Y)$ . For any  $x^*$ , the following non-parametric relationship hold:*

$$\text{Cov}_x^c(X, Y) = \sum_{g \in \Pi^c(X, Y)} \text{Cov}_{g[x^*]}^c(X, Y)_{\mathcal{L}_\pi^c}.$$

Thm. 2 can be demonstrated in the model of Fig. 1(a). Let an order  $\mathcal{L}^c$  over  $g_{[1,4]}$  be  $g_i < g_j$  if  $i < j$ . First note that the path-specific causal covariance of  $g_2$  ( $\text{Cov}_{g_2[x^*]}^c(X, Y)_{\mathcal{L}_\pi^c}$ ) and  $g_3$  ( $\text{Cov}_{g_3[x^*]}^c(X, Y)_{\mathcal{L}_\pi^c}$ ) are equal to, respectively,

$$\text{Cov}\left(X, Y_{W_{2W_{1,x^*}}} - Y_{W_{2W_{1,x^*}}, W_{1,x^*}}\right) \quad (12)$$

$$\text{Cov}\left(X, Y_{W_{2W_{1,x^*}}, W_{1,x^*}} - Y_{W_{1,x^*}, W_{2,x^*}}\right) \quad (13)$$

The causal covariance  $\text{Cov}_x^c(X, Y)$  can then be decomposed as the sum of Eqs. 10-13, respectively,  $g_1, g_4, g_2, g_3$ . Fig. 3 describes this decomposition procedure: we measure the difference of the outcome  $Y$  as the intervention  $do(x^*)$  propagates through paths  $g_1, g_2, g_3, g_4$ . The sum of these differences thus equate to the total influence of the intervention  $do(x^*)$  to the outcome  $Y$ , i.e., the causal covariance  $\text{Cov}_{x^*}^c(X, Y)$ .

## 5 DECOMPOSING SPURIOUS RELATIONS

We introduce in the sequel a new strategy to decompose the spurious covariance (Def. 2), which will play a cen-

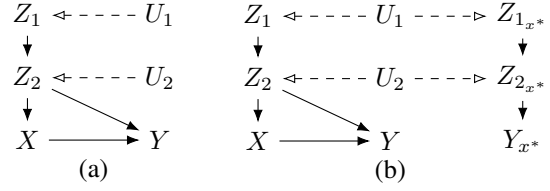


Figure 4: Causal diagrams for (a) the one-confounder setting where  $X$  and  $Y$  are confounded by the variable  $Z_2$ , of which  $Z_1$  is a parent node; (b) the twin network for the model of (a) under  $do(x^*)$ .

tral role in the analysis of the spurious relations relative to the pair  $X, Y$ . The spurious covariance measures the correlation between the observational  $X$  and the counterfactual  $Y_{x^*}$  (Def. 2). We will employ in our analysis the twin network [Balke and Pearl, 1994; Pearl, 2000, Sec. 7.1.4], which is a graphical method to analyzing the relation between observational and counterfactual variables.

Consider the causal model  $M$  in Fig. 4(a), for example, where the exogenous variables  $\{U_1, U_2\}$  are shown explicitly. Its twin network is the union of the model  $M$  (factual) and the submodel  $M_{x^*}$  (counterfactual) under intervention  $do(x^*)$ , which is shown in Fig. 4(b). The factual ( $M$ ) and counterfactual ( $M_{x^*}$ ) worlds share only the exogenous variables (in this case,  $U_1, U_2$ ), which constitute the invariances shared across worlds. In this twin network, the observational  $X$  and the counterfactual  $Y_{x^*}$  are connected through two paths: one through  $U_1$  and the other through  $U_2$ . These paths correspond to two pathways from  $X$  to  $Y$  in the original causal diagram:  $\tau_1 : X \leftarrow Z_2 \leftarrow Z_1 \leftarrow U_1 \rightarrow Z_1 \rightarrow Z_2 \rightarrow Y$ , and  $\tau_2 : X \leftarrow Z_2 \leftarrow U_2 \rightarrow Z_2 \rightarrow Y$ .

Note that when considering the corresponding paths in the original graph (Fig. 4(a)), these paths ( $\tau_1, \tau_2$ ) are not necessarily simple, i.e., they may contain a particular node more than once. Furthermore, each path can be partitioned into a pair of causal paths (say,  $g_l, g_r$ ) from a common source  $U_i \in U$  (e.g.,  $\tau_1$  consists of a pair  $(g_{l_1}, g_{r_1})$ , where  $g_{l_1} : U_1 \rightarrow Z_1 \rightarrow Z_2 \rightarrow X$ , and  $g_{r_1} : U_1 \rightarrow Z_1 \rightarrow Z_2 \rightarrow Y$ ). Indeed, these non-simple paths are referred to as *treks* in the causal inference literature, which usually has been studied in the context of linear models [Spirtes *et al.*, 2001; Sullivant *et al.*, 2010].

**Definition 9 (Trek).** A trek  $\tau$  in  $G$  (from  $X$  to  $Y$ ) is an ordered pair of causal paths  $(g_l, g_r)$  with a common exogenous source  $U_i \in U$  such that  $g_l \in \Pi^c(U_i, X)$  and  $g_r \in \Pi^c(U_i, Y)$ . The common source  $U_i$  is called the top of the trek, denoted  $top(g_l, g_r)$ . A trek is spurious if  $g_r \in \Pi^c(U_i, Y|X)$ , i.e.,  $g_r$  is a causal path from  $U_i$  to  $Y$  that is not intercepted by  $X$ .

We denote the set of treks from  $X$  to  $Y$  in  $G$  by  $\mathcal{T}(X, Y)_G$  and spurious treks by  $\mathcal{T}^s(X, Y)_G$  ( $G$  will be omitted when obvious). We introduce next an estimand for a specific spurious trek. For a spurious trek  $\tau = (g_l, g_r)$  with  $U_i = \text{top}(\tau)$ , first let  $X_{g_l}$  denote the path-specific potential response  $X_{g_l[U_i^l]}$ , where  $U_i^l$  is an i.i.d. draw from the distribution  $P(U_i)$ . Similarly, let  $Y_{x^*, g_r} = Y_{x^*, g_r[U_i^r]}$ <sup>10</sup>, where  $U_i^r \sim P(U_i)$ . Pure trek-specific covariance can then finally be defined.

**Definition 10** (Pure Trek-Specific Spurious Covariance). For a semi-Markovian model  $M$  and a spurious trek  $\tau = (g_l, g_r)$  with  $U_i = \text{top}(g_l, g_r)$ , the pure  $\tau$ -specific covariance of the treatment  $X = x^*$  on the outcome  $Y$  is defined as:

$$\text{Cov}_{\tau[x^*]}^{ts}(X, Y) = \text{Cov}(X - X_{g_l}, Y_{x^*} - Y_{x^*, g_r}).$$

In words, the differences  $X - X_{g_l}$  and  $Y_{x^*} - Y_{x^*, g_r}$  are simply measuring the effects of the causal paths  $g_l$  and  $g_r$  (Lem. 1), while the  $\text{Cov}(\cdot)$  operator is in charge of compounding them. (In the extreme case when  $g_l$  or  $g_r$  are disconnected, the pure  $\tau$ -specific spurious covariance will equate to zero.) For example, the pure  $\tau_1$ -specific spurious covariance  $\text{Cov}_{\tau_1[x^*]}^{ts}(X, Y)$  in Fig. 4(a) is

$$\text{Cov}(X - X_{g_{l_1}}, Y_{x^*} - Y_{x^*, g_{r_1}}). \quad (14)$$

Note that the counterfactuals  $X_{g_{l_1}}$  and  $Y_{x^*, g_{r_1}}$  assign the randomized interventions  $do(U_1^l), do(U_1^r)$  to the paths  $g_{l_1}, g_{r_1}$ , respectively. By Def. 6, Eq. 14 is equal to:

$$\text{Cov}(X - X_{U_1^l}, Y_{x^*} - Y_{x^*, U_1^r}).$$

This quantity can be more easily seen through its graphical representation, see Fig. 5 (top). The main idea is to decompose  $U_1$  into two independent components  $U_1^l, U_1^r$  (Fig. 5b), which is then contrasted with the world in which  $U_1$  is kept intact (a).<sup>11 12</sup> We note that by Def. 6,  $X = X_\emptyset$  and  $Y_{x^*} = Y_{x^*, \emptyset}$ . The pure  $\tau_1$ -specific spurious covariance can be written as:

$$\text{Cov}_{\tau_1[x^*]}^{ts}(X, Y) = \text{Cov}(X_\emptyset - X_{g_{l_1}}, Y_{x^*, \emptyset} - Y_{x^*, g_{r_1}}).$$

More generally, the pure trek-specific spurious covariance for  $\tau = (g_l, g_r)$  measures the covariance of variables  $X_{\pi_l} - X_{\pi_l \cup \{g_l\}}$  and  $Y_{x^*, \pi_r} - Y_{x^*, \pi_r \cup \{g_r\}}$ , where  $\pi_l (\pi_r)$  is an arbitrary set of causal paths from  $U$  that does not contain  $g_l (g_r)$ . This observational will be useful later on, which leads to the trek-specific spurious covariance.

<sup>10</sup> $Y_{x^*, g_r[U_i^r]}$  is the  $g_r$ -specific potential response of  $Y$  to  $do(g_r[U_i^r])$  in the submodel  $M_{x^*}$ .

<sup>11</sup>This operation can be seen as the parallel to the pure path-specific covariance (Def. 7), with the distinct requirement that the replacement operator, used to generate the differences, is not relative to the observed  $X$ , but the corresponding  $U_i$ .

<sup>12</sup>To avoid clutter, Fig. 5 is a projected version of the original twin network focused on the relevant quantities (w.l.g.).

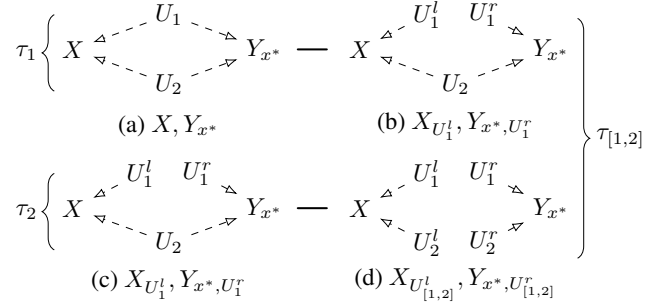


Figure 5: The decomposition procedure of the spurious covariance over the spurious treks  $\tau_1, \tau_2$  (Thm. 3).

**Definition 11** (Trek-Specific Spurious Covariance). For a semi-Markovian model  $M$ , let  $\tau$  be a spurious trek  $(g_l, g_r)$  and  $\pi$  is a function mapping  $\tau$  to a pair  $\pi(\tau) = (\pi_l, \pi_r)$  where  $\pi_l$  and  $\pi_r$  are sets of causal paths from  $U$  such that  $g_l \notin \pi_l$  and  $g_r \notin \pi_r$ . The  $\tau$ -specific spurious covariance of the treatment  $X = x^*$  on the outcome  $Y$ , denoted by  $\text{Cov}_{\tau[x^*]}^{ts}(X, Y)_\pi$ , is defined as

$$\text{Cov}(X_{\pi_l} - X_{\pi_l \cup \{g_l\}}, Y_{x^*, \pi_r} - Y_{x^*, \pi_r \cup \{g_r\}}).$$

The next proposition establishes the relationship between Def. 11 and the corresponding spurious treks. This property can be seen as a necessary condition for any measure of strength for spurious relations.

**Property 6.**  $\tau \notin \mathcal{T}^s(X, Y) \Rightarrow \text{Cov}_{\tau[x^*]}^{ts}(X, Y)_\pi = 0$ .

As an example of Def. 11, the trek  $\tau_2$  in Fig. 4(a) consists of paths  $g_{l_2} : U_2 \rightarrow Z_2 \rightarrow X$  and  $g_{r_2} : U_2 \rightarrow Z_2 \rightarrow Y$ . If we set  $\pi(\tau_2) = (\{g_{l_1}\}, \{g_{r_1}\})$ , the  $\tau_2$ -specific spurious covariance can be measured by  $\text{Cov}_{\tau_2[x^*]}^{ts}(X, Y)_\pi$ , i.e.,

$$\text{Cov}(X_{g_{l_1}} - X_{g_{l_1, [2]}}, Y_{x^*, g_{r_1}} - Y_{x^*, g_{r_1, [2]}}) \quad (15)$$

$$= \text{Cov}(X_{U_1^l} - X_{U_{[1,2]}^l}, Y_{x^*, U_1^r} - Y_{x^*, U_{[1,2]}^r}). \quad (16)$$

Eq. 16 is graphically represented in Fig. 5(c-d), where the effect of the trek  $\tau_2$  is measured. In words, the difference between Fig. 5(c) and (d) is the effect of the causal paths  $g_{l_2}$  and  $g_{r_2}$  when  $U_2$  is kept intact versus when divided into two independent components  $(U_2^l, U_2^r)$ .

Armed with the definition of trek-specific spurious covariance, we can finally study the decomposability of the spurious covariance  $\text{Cov}_{x^*}^s(X, Y)$  (Def. 2). First, let  $U^s \subseteq U$  denote the maximal set of exogenous variables that simultaneously affect variables  $X$  and  $Y_{x^*}$  (common exogenous ancestors), and let an order over  $U^s$  be  $\mathcal{L}_u^s : U_1 < \dots < U_n$ . For each  $U_i \in U^s$ , let  $\mathcal{L}_{l_i}^s$  be an order  $g_{l_1}^i < \dots < g_{l_n}^i$  over the set  $\Pi^c(U_i, X)$ . Similarly, we define  $\mathcal{L}_{r_i}^s$  for  $\Pi^c(U_i, Y|X)$ . The tuple  $\mathcal{L}^s = \langle \mathcal{L}_u^s, \{(\mathcal{L}_{l_i}^s, \mathcal{L}_{r_i}^s)\}_{1 \leq i \leq |U^s|} \rangle$  thus defines an order

for the spurious treks  $\mathcal{T}^s(X, Y)$ . We denote  $\mathcal{L}_\pi^s$  a function which maps from a trek  $\tau$  to sets of paths  $\mathcal{L}_\pi^s(\tau)$  covered by the spurious treks preceding  $\tau$  in  $\mathcal{L}^s$ . Formally, given a spurious trek  $\tau = (g_{l_j}^i, g_{r_k}^i)$ ,  $\mathcal{L}_\pi^s(\tau)$  is equal to

$$(\Pi^c(U_{[1, i-1]}, X) \cup g_{l_{[1, j-1]}}^i, \Pi^c(U_{[1, i-1]}, Y|X) \cup g_{r_{[1, k-1]}}^i).$$

We are now ready to derive the decomposition formula for the spurious covariance  $\text{Cov}_{x^*}^s(X, Y)$ .

**Theorem 3.** *For a semi-Markovian model  $M$ , let  $\mathcal{L}^s = \langle \mathcal{L}_u^s, \{(\mathcal{L}_{l_i}^s, \mathcal{L}_{r_i}^s)\}_{1 \leq i \leq |U^s|} \rangle$  be an order over spurious treks  $\mathcal{T}^s(X, Y)$ . For any  $x^*$ , the following non-parametric relationship hold:*

$$\text{Cov}_{x^*}^s(X, Y) = \sum_{\tau \in \mathcal{T}^s(X, Y)} \text{Cov}_{\tau[x^*]}^{ts}(X, Y)_{\mathcal{L}_\pi^s}$$

For example, in the model of Fig. 4(a),  $U^s = \{U_1, U_2\}$ .  $\tau_1$  ( $\tau_2$ ) is the only spurious trek associated with  $U_1$  ( $U_2$ ). If we consider the order  $\mathcal{L}^s$  such that  $\mathcal{L}_u^s : U_1 < U_2$ , Thm. 3 dictates that  $\text{Cov}_{x^*}^s(X, Y)$  should be decomposed as the sum of Eqs. 14 and 15. Fig. 5 shows the graphical representation of this decomposition procedure: we measure the change of the covariance between  $X$  and  $Y_{x^*}$  as we disconnect the relations going through  $\tau_1$  (associated with  $U_1$ ) and  $\tau_2$  ( $U_2$ ), sequentially. The sum of these changes thus equates to the correlations of  $X$  and  $Y$  along the spurious pathways, i.e., the spurious covariance  $\text{Cov}_{[x^*]}^s(X, Y)$ . (See [Zhang and Bareinboim, 2018b, Sec. 2] for more examples.)

## 6 NON-PARAMETRIC PATH ANALYSIS

In this section, we put the results of the previous sections together and derive a general path-specific decomposition for the covariance of the treatment  $X$  and the outcome  $Y$  without assuming any specific parametric form.

We start by noting that each spurious path from  $X$  to  $Y$  corresponds to a unique set of spurious treks that start on  $X$  and end in  $Y$ . Recall that a spurious path  $l$  can be seen as a pair of causal paths  $(g_l, g_r)$ , where the only node shared among  $g_l$  and  $g_r$  is the common source. For example, the spurious path  $l : X \leftarrow Z_2 \rightarrow Y$  is a pair  $(g_l, g_r)$  such that  $g_l : Z_2 \rightarrow X$  and  $g_r : Z_2 \rightarrow Y$ . We can thus define a rule  $f$  which maps a trek  $\tau \in \mathcal{T}^s(X, Y)$  to a spurious path  $l \in \Pi^s(X, Y)$ . For  $\tau = (g_l, g_r)$ , let  $V_t$  be the most distant recurring node from  $\text{top}(g_l, g_r)$  such that  $V_t$  is the only node shared among subpaths  $g_l(V_t, X)$  and  $g_r(V_t, Y)$ ; the pair  $(g_l(V_t, X), g_r(V_t, Y))$  corresponds to a path  $l$  in  $\Pi^s(X, Y)$ . As an example, the trek  $\tau_1$  in Fig. 4(a) has  $V_t = Z_2$ , which corresponds to the spurious path  $l : X \leftarrow Z_2 \rightarrow Y$ , and similarly,  $f(\tau_1) = l$  as well as  $f(\tau_2) = l$ . Lem. 2 shows that the rule  $f$  forms a valid surjective function.

**Lemma 2.** *For a semi-Markovian model  $M$ , for each spurious trek  $\tau \in \mathcal{T}^s(X, Y)$ , there always exists a unique most distant recurring node  $V_t$ .*

For a spurious path  $l$ , let  $\mathcal{T}^s(l) = f^{-1}(l)$  denote its corresponding treks. Specifically, if  $l \notin \Pi^s(X, Y)$ , then for each  $\tau \in \mathcal{T}^s(l)$ , we must have  $\tau \notin \mathcal{T}^s(X, Y)$ . For instance, if the spurious  $l$  in Fig. 4(a) is disconnected, e.g.,  $Z_2 \not\rightarrow X$ , treks  $\tau_1, \tau_2$  are both disconnected as well. From this observation, we could naturally define the spurious covariance of a path  $l$  as a sum over treks in  $\mathcal{T}^s(l)$ .

**Definition 12** (Path-Specific Spurious Covariance). For a semi-Markovian model  $M$  with an associated causal diagram  $G$ , let  $l$  be an arbitrary spurious path in  $G$ . Let  $\pi$  be a function that maps a trek  $\tau = (g_l, g_r) \in \mathcal{T}^s(l)$  to a pair  $\pi(\tau) = (\pi_l, \pi_r)$ , where  $\pi_l$  and  $\pi_r$  are arbitrary sets of causal paths from  $U$  such that  $g_l \notin \pi_l$  and  $g_r \notin \pi_r$ . The  $l$ -specific spurious covariance of the treatment  $X = x^*$  on the outcome  $Y$  is defined as

$$\text{Cov}_{l[x^*]}^s(X, Y)_\pi = \sum_{\tau \in \mathcal{T}^s(l)} \text{Cov}_{\tau[x^*]}^{ts}(X, Y)_\pi$$

**Property 7.**  $l \notin \Pi^s(X, Y) \Rightarrow \text{Cov}_{l[x^*]}^s(X, Y)_\pi = 0$ .

The surjectivity of the function  $f$  assures that the set  $\{\mathcal{T}^s(l)\}_{l \in \Pi^s(X, Y)}$  forms a partition over the spurious treks  $\mathcal{T}^s(X, Y)$ . From Thm. 3, it follows immediately that the path-specific spurious covariance (Def. 12) has the property that expresses the spurious covariance  $\text{Cov}_{x^*}^s(X, Y)$  as a sum over  $\Pi^s(X, Y)$ .

**Theorem 4.** *For a semi-Markovian model  $M$ , let  $\mathcal{L}^s = \langle \mathcal{L}_u^s, \{(\mathcal{L}_{l_i}^s, \mathcal{L}_{r_i}^s)\}_{1 \leq i \leq |U^s|} \rangle$  be an order over spurious treks  $\mathcal{T}^s(X, Y)$ . For any  $x^*$ , the following non-parametric relationship hold:*

$$\text{Cov}_{x^*}^s(X, Y) = \sum_{l \in \Pi^s(X, Y)} \text{Cov}_{l[x^*]}^s(X, Y)_{\mathcal{L}_\pi^s}$$

As an example, the path  $l : X \leftarrow Z_2 \rightarrow Y$  in Fig. 4(a) corresponds to  $\mathcal{T}^s(l) = \{\tau_1, \tau_2\}$ . For an arbitrary order  $\mathcal{L}^s$ , Thm. 4 is applicable and immediately yields  $\text{Cov}_{x^*}^s(X, Y) = \text{Cov}_{l[x^*]}^s(X, Y)_{\mathcal{L}_\pi^s}$ , which means that the path  $l$  accounts for all the spurious relations between  $X$  and  $Y$ . In other words, the spurious joint variability of  $X$  and  $Y$  is fully explained by the variance of  $Z_2$ , which is a function of the exogenous variables  $U_1$  (through  $\tau_1$ ) and  $U_2$  (through  $\tau_2$ ).

Thms. 1-2 and 4 together lead to a general path-specific decomposition formula, which allows one to non-parametrically decompose the covariance  $\text{Cov}(X, Y)$  over all open paths from  $X$  to  $Y$  in the underlying model.

**Theorem 5** (Path-Specific Decomposition). *For a semi-Markovian model  $M$ , let  $\mathcal{L}^c$  be an order over  $\Pi^c(X, Y)$*



and  $\mathcal{L}^s = \langle \mathcal{L}_u^s, \{(\mathcal{L}_{l_i}^s, \mathcal{L}_{r_i}^s)\}_{1 \leq i \leq |U^s|} \rangle$  be an order over  $\mathcal{T}^s(X, Y)$ . For any  $x^*$ , the following non-parametric relationship hold:

$$\begin{aligned} \text{Cov}(X, Y) &= \sum_{l \in \Pi^c(X, Y)} \text{Cov}_{l[x^*]}^c(X, Y)_{\mathcal{L}_\pi^c} \\ &+ \sum_{l \in \Pi^s(X, Y)} \text{Cov}_{l[x^*]}^s(X, Y)_{\mathcal{L}_\pi^s}. \end{aligned} \quad (17)$$

We illustrate the use of Thm. 5 using the model discussed in Sec. 1 (Fig. 1(a)). Recall that  $X$  and  $Y$  are connected through the causal paths  $l_1, l_2$  and spurious paths  $l_3, l_4$ . Note that  $U^s = \{U_Z\}$  spuriously affects the treatment  $X$  through the path  $g_l = U_Z \rightarrow Z \rightarrow X$ , and the outcome  $Y$  through the paths  $g_{r_1} = U_Z \rightarrow Z \rightarrow Y$  and  $g_{r_2} = U_Z \rightarrow Z \rightarrow W \rightarrow Y$ . Let order  $\mathcal{L}^c$  be  $l_1 < l_2$  and  $\mathcal{L}^s$  be  $g_{r_1} < g_{r_2}$ . For any level  $x^*$ , Thm. 5 equates the covariance  $\text{Cov}(X, Y)$  to the sum of  $\{\text{Cov}_{l_i[x^*]}^c(X, Y)_{\mathcal{L}_\pi^c}\}_{i=1,2}$  and  $\{\text{Cov}_{l_i[x^*]}^s(X, Y)_{\mathcal{L}_\pi^s}\}_{i=3,4}$ , which can be written as

$$\begin{aligned} &\underbrace{\text{Cov}(X, Y - Y_{x^*, W})}_{l_1: X \rightarrow Y} + \underbrace{\text{Cov}(X, Y_{x^*, W} - Y_{x^*})}_{l_2: X \rightarrow W \rightarrow Y} \\ &+ \underbrace{\text{Cov}(X - X_{U_Z^l}, Y_{x^*} - Y_{x^*, W_{x^*} Z_{U_Z^r}})}_{l_3: X \leftarrow Z \rightarrow Y} \\ &+ \underbrace{\text{Cov}(X - X_{U_Z^l}, Y_{x^*, W_{x^*} Z_{U_Z^r}} - Y_{x^*, U_Z^r})}_{l_4: X \leftarrow Z \rightarrow W \rightarrow Y}, \end{aligned} \quad (18)$$

which are all well-defined, computable from the structural causal model [Def. 1; Pearl, 2000, Sec. 7.1].

## 7 IDENTIFYING PATH-SPECIFIC DECOMPOSITION

By and large, identifiability is one of the most studied topics in causal inference. It is acknowledged in the literature that obtaining identifiability may be non-trivial even in the context of less granular measures of causal effects, including quantities without nested counterfactual and following the analysis of Pearl's do-calculus.

In this section, we start the study of identifiability conditions for when the path-specific decomposition formula (Thm. 5) can be estimated from data, when the SCM is not fully known. We'll analyze the causal model discussed in Fig. 1(a) given its generality and potential to encode more complex models. The main assumption encoded in this model is Markovianity, i.e., that all exogenous variables are independent. We show next that identifiability can be obtained under these assumptions.

**Theorem 6.** *The path-specific decomposition of Eq. 18 is identifiable if the distributions  $P(x, y_{x^*})$ ,  $P(x, y_{x^*, W})$  and  $P(x, y_{x^*, W_{x^*}, Z_{U_Z^r}})$  are identifiable. Specifically, in the model of Fig. 1(a),  $P(x, y_{x^*})$ ,  $P(x, y_{x^*, W})$ , and*

*$P(x, y_{x^*, W_{x^*}, Z_{U_Z^r}})$  can be estimated, respectively, from the observational distribution  $P(x, y, z, w)$  as follows:*

$$P(x, y_{x^*}) = \sum_{z, w} P(y|x^*, w, z)P(w|x^*, z)P(x, z)$$

$$P(x, y_{x^*, W}) = \sum_{z, w} P(y|x^*, z, w)P(x, z, w)$$

$$P(x, y_{x^*, W_{x^*}, Z_{U_Z^r}}) = \sum_{z, z', w} P(y|x^*, z, w)P(w|x^*, z')P(x, z')P(z)$$

Note that all the quantities listed in Thm. 6 are expressible in terms of conditional distributions and do not involve any counterfactual (simple nor nested), which are readily estimable from the observational distribution. As an example, the  $l_2$ -specific causal covariance  $\text{Cov}_{l_2[x^*]}^c(X, Y)_{\mathcal{L}_\pi^c}$  in Eq. 18 can be written as  $\text{Cov}(X, Y_{x^*, W}) - \text{Cov}(X, Y_{x^*})$ , which are computed from the counterfactual distributions  $P(x, y_{x^*})$  and  $P(x, y_{x^*, W})$ , respectively. These distributions can be estimated from the observational distribution  $P(x, y, z, w)$  following Thm. 6. Indeed, the path-specific decomposition formula (Thm. 5) is identifiable in the model of Fig. 1(a) regardless of the order  $\mathcal{L}^c$  and  $\mathcal{L}^s$ . (For other decompositions, see [Zhang and Bareinboim, 2018b].)

We further considered the identifiability conditions for the path-specific decomposition formula when the more stringent assumption that the underlying structural functions are linear is imposed.

**Theorem 7.** *Under the assumption of linearity and the assumption of Fig. 1(a), for any arbitrary orders  $\mathcal{L}^c$  and  $\mathcal{L}^s$ , for any  $x$ , the path-specific covariance of  $l_1, l_2, l_3$  and  $l_4$  are equal to:*

$$\begin{aligned} \text{Cov}_{l_1[x^*]}^c(X, Y)_{\mathcal{L}_\pi^c} &= \alpha_{YX}, \quad \text{Cov}_{l_2[x^*]}^c(X, Y)_{\mathcal{L}_\pi^c} = \alpha_{WX}\alpha_{YW} \\ \text{Cov}_{l_3[x^*]}^s(X, Y)_{\mathcal{L}_\pi^s} &= \alpha_{XZ}\alpha_{YZ}, \quad \text{Cov}_{l_4[x^*]}^s(X, Y)_{\mathcal{L}_\pi^s} = \alpha_{XZ}\alpha_{WZ}\alpha_{YW} \end{aligned}$$

*The parameters  $\alpha$  can be estimated from the corresponding (partial) regression coefficients [Pearl, 2000, Ch. 5].*

Clearly, after applying Thm. 7 to Eq. 18, the resulting decomposition coincides with Wright's method of path coefficients in the linear-standard model (Eq. 1).

## 8 CONCLUSIONS

We introduced novel covariance-based counterfactual measures to account for effects along with a specific path from a treatment  $X$  to an outcome  $Y$  (Defs. 8, 11-12). We developed machinery to allow, for the first time, the non-parametric decomposition of the covariance of  $X$  and  $Y$  as a summation over the different pathways in the underlying causal model (Thm. 5). We further provided identification conditions under which the decomposition formula can be estimated from data (Thm. 6-7).

## Acknowledgments

Bareinboim and Zhang are supported in parts by grants from NSF IIS-1704352 and IIS-1750807 (CAREER).

## References

- C. Avin, I. Shpitser, and J. Pearl. Identifiability of path-specific effects. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence IJCAI-05*, pages 357–363, Edinburgh, UK, 2005. Morgan-Kaufmann Publishers.
- A. Balke and J. Pearl. Counterfactual probabilities: Computational methods, bounds, and applications. In R. Lopez de Mantaras and D. Poole, editors, *Uncertainty in Artificial Intelligence 10*, pages 46–54. Morgan Kaufmann, San Mateo, CA, 1994.
- E. Bareinboim and J. Pearl. Causal inference and the data-fusion problem. *Proceedings of the National Academy of Sciences*, 113:7345–7352, 2016.
- Reuben M Baron and David A Kenny. The moderator–mediator variable distinction in social psychological research: Conceptual, strategic, and statistical considerations. *Journal of personality and social psychology*, 51(6):1173, 1986.
- K.A. Bollen. *Structural Equations with Latent Variables*. John Wiley, NY, 1989.
- RM Daniel, BL De Stavola, SN Cousens, and Stijn Vansteelandt. Causal mediation analysis with multiple mediators. *Biometrics*, 71(1):1–14, 2015.
- O.D. Duncan. *Introduction to Structural Equation Models*. Academic Press, New York, 1975.
- J. Fox. Effect analysis in structural equation models. *Sociological Methods and Research*, 9(1):3–28, 1980.
- Kosuke Imai, Luke Keele, and Teppei Yamamoto. Identification, inference and sensitivity analysis for causal mediation effects. *Statist. Sci.*, 25(1):51–71, 02 2010.
- Kosuke Imai, Luke Keele, Dustin Tingley, and Teppei Yamamoto. Unpacking the black box of causality: Learning about causal mechanisms from experimental and observational studies. *American Political Science Review*, 105(4):765–789, 2011.
- Niki Kilbertus, Mateo Rojas Carulla, Giambattista Parascandolo, Moritz Hardt, Dominik Janzing, and Bernhard Schölkopf. Avoiding discrimination through causal reasoning. In *Advances in Neural Information Processing Systems*, pages 656–666, 2017.
- Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. Counterfactual fairness. In *Advances in Neural Information Processing Systems*, pages 4069–4079, 2017.
- Xintao Wu Lu Zhang, Yongkai Wu. A causal framework for discovering and removing direct and indirect discrimination. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 3929–3935, 2017.
- D.P. MacKinnon. *An Introduction to Statistical Mediation Analysis*. Lawrence Erlbaum Associates, New York, 2008.
- J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, New York, 2000.
- J. Pearl. Direct and indirect effects. In *Proc. of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, pages 411–420. Morgan Kaufmann, CA, 2001.
- Ilya Shpitser and Eric Tchetgen Tchetgen. Causal inference with a graphical hierarchy of interventions. *Annals of statistics*, 44(6):2433, 2016.
- Peter Spirtes, Clark N Glymour, and Richard Scheines. *Causation, prediction, and search*. MIT press, 2001.
- Seth Sullivan, Kelli Talaska, and Jan Draisma. Trek separation for gaussian graphical models. *The Annals of Statistics*, pages 1665–1685, 2010.
- Eric J Tchetgen Tchetgen and Ilya Shpitser. Semiparametric theory for causal mediation analysis: efficiency bounds, multiple robustness, and sensitivity analysis. *Annals of statistics*, 40(3):1816, 2012.
- Tyler J VanderWeele, Stijn Vansteelandt, and James M Robins. Effect decomposition in the presence of an exposure-induced mediator-outcome confounder. *Epidemiology*, 25(2):300, 2014.
- Tyler VanderWeele. *Explanation in Causal Inference: Methods for Mediation and Interaction*. Oxford University Press, New York, 2015.
- Stijn Vansteelandt and Tyler J VanderWeele. Natural direct and indirect effects on the exposed: effect decomposition under weaker assumptions. *Biometrics*, 68(4):1019–1027, 2012.
- S. Wright. The theory of path coefficients: A reply to Niles’ criticism. *Genetics*, 8:239–255, 1923.
- Sewall Wright. The method of path coefficients. *The annals of mathematical statistics*, 5(3):161–215, 1934.
- Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proceedings of the 26th International Conference on WWW*, pages 1171–1180, 2017.
- Junzhe Zhang and Elias Bareinboim. Fairness in decision-making – the causal explanation formula. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018a.
- Junzhe Zhang and Elias Bareinboim. Non-parametric path analysis in structural causal models. Technical Report R-34, AI Lab, Purdue University., 2018b.

---

# Stochastic Layer-Wise Precision in Deep Neural Networks

---

**Griffin Lacey\***  
NVIDIA

**Graham W. Taylor**  
University of Guelph  
Vector Institute for Artificial Intelligence  
Canadian Institute for Advanced Research

**Shawki Areibi**  
University of Guelph

## Abstract

Low precision weights, activations, and gradients have been proposed as a way to improve the computational efficiency and memory footprint of deep neural networks. Recently, low precision networks have even shown to be more robust to adversarial attacks. However, typical implementations of low precision DNNs use uniform precision across all layers of the network. In this work, we explore whether a heterogeneous allocation of precision across a network leads to improved performance, and introduce a learning scheme where a DNN stochastically explores multiple precision configurations through learning. This permits a network to learn an optimal precision configuration. We show on convolutional neural networks trained on MNIST and ILSVRC12 that even though these nets learn a uniform or near-uniform allocation strategy respectively, stochastic precision leads to a favourable regularization effect improving generalization.

## 1 INTRODUCTION

Recent advances in deep learning, and convolutional neural networks (CNN) in particular, have led to well-publicized breakthroughs in computer vision (Krizhevsky et al., 2012), speech recognition (Hinton et al., 2012), and natural language processing (NLP) (Bahdanau et al., 2014). Modern CNNs, however, have increasingly large storage and computational requirements (Canziani et al., 2016). This has limited the application scope to data centres that can accommodate clusters of massively parallel hardware accelerators, such as graphics processing units (GPUs). Still, GPU training

of CNNs on large datasets like ImageNet (Deng et al., 2009) can take hours, even on networks with hundreds of GPUs, and achieving linear scaling beyond these sizes is difficult (Goyal et al., 2017). As such, there is a growing interest in investigating more fine-grained optimizations, especially since deployment on embedded devices with limited power, compute, and memory budgets remains an imposing challenge.

Research efforts to reduce model size and speed up inference have shown that training networks with binary or ternary weights and activations (Courbariaux and Bengio, 2016; Rastegari et al., 2016; Li et al., 2016a) can achieve comparable accuracy to full precision networks, while benefiting from reduced memory requirements and improved computational efficiency using bit operations. They may even confer additional robustness to adversarial attacks (Galloway et al., 2018). More recently, the DoReFa-Net model has generalized this finding to include different precision settings for weights vs. activations, and demonstrated how low precision *gradients* can be also employed at training time (Zhou et al., 2016).

These findings suggest that precision in deep learning is not an arbitrary design choice, but rather a dial that controls the trade-off between model complexity and accuracy. However, precision is typically considered at the design level of an entire model, making it difficult to consider as a tunable hyperparameter. We posit that considering precision at a finer granularity, such as a layer or even per-example could grant models more flexibility in which to find optimal configurations, which maximizes accuracy and minimizes computational cost. To remain deterministic about hardware efficiency, we aim to do this for fixed budgets of precision, which have predictable acceleration properties.

In this work we consider learning an optimal precision configuration across the layers of a deep neural network, where the precision assigned to each layer may be different. We propose a stochastic regularization technique

---

\*Work completed while at the University of Guelph

akin to Dropout (Srivastava et al., 2014) where a network explores a different precision configuration per example. This introduces non-differentiable elements in the computational graph which we circumvent using recently proposed gradient estimation techniques.

## 2 RELATED WORK

Recent work related to efficient learning has explored a number of different approaches to reducing the effective parameter count or memory footprint of CNN architectures. Network compression techniques (Han et al., 2015a; Wang and Liang, 2016; Choi et al., 2016; Agustsson et al., 2017) typically compress a pre-trained network while minimizing the degradation of network accuracy. However, these methods are decoupled from learning, and are only suitable for efficient deployment. Network pruning techniques (Wan et al., 2013; Han et al., 2015b; Jin et al., 2016; Li et al., 2016b; Anwar et al., 2017; Molchanov et al., 2016; Tung et al., 2017) take a more iterative approach, often using regularized training and retraining to rank and modify network parameters based on their magnitude. Though coupled with the learning process, iterative pruning techniques tend to contribute to slower learning, and result in sparse connections, which are not hardware efficient.

Our work relates primarily to low precision techniques, which have tended to focus on reducing the precision of weights and activations used for deployment while maintaining dense connectivity. Courbariaux et al. were among the first to explore binary weights and activations (Courbariaux et al., 2015; Courbariaux and Bengio, 2016), demonstrating state-of-the-art results for smaller datasets (MNIST, CIFAR-10, and CVHN). This idea was then extended further with CNNs on larger datasets like ImageNet with binary weights and activations, while approximating convolutions using binary operations (Rastegari et al., 2016). Related work (Kim and Smaragdis, 2016) has validated these results and shown neural networks to be remarkably robust to an even wider class of non-linear projections (Merolla et al., 2016). Ternary quantization strategies (Li et al., 2016a) have been shown to outperform their binary counterparts, moreso when parameters of the quantization module are learned through backpropagation (Zhu et al., 2016). Cai et al. have investigated how to improve the gradient quality of quantization operations (Cai et al., 2017), which is complimentary to our work which relies on these gradients to learn precision. Zhou et al. further explored this idea of variable precision (i.e. heterogeneity across weights and activations) and discussed the general trade-off of precision and accuracy, exploring strategies for training with low precision gradients (Zhou et al., 2016).

Our approach for learning precision closely resembles BitNet (Raghavan et al., 2017), where the optimal precision for each network layer is learned through gradient descent, and network parameter encodings act as regularizers. While BitNet uses the Lagrangian approach of adding constraints on quantization error and precision to the objective function, we allocate bits to layers through sampling from a Gumbel-Softmax distribution constructed over the network layers. This has the advantage of accommodating a defined precision budget, which allows more deterministic hardware constraints, as well as a wider range of quantization encodings through the use of non-integer quantization values early in training. In the allocation of bits on a budget, our work resembles (Wang and Liang, 2016), though we allow more fine-grained control over precision, and prefer a gradient-based approach over clustering techniques for learning optimal precision configurations.

To the best of our knowledge, our work is the first to explore learning precision in deep networks through a continuous-to-discrete annealed quantization strategy. Our contributions are as follows:

- We experimentally confirm a linear relationship between total number of bits and speedup for low precision arithmetic, motivating the use of precision budgets.
- We introduce a gradient-based approach to learning precision through sampling from a Gumbel-Softmax distribution constructed over the network layers, constrained by a precision budget.
- We empirically demonstrate the advantage of our end-to-end training strategy as it improves model performance over simple uniform bit allocations.

## 3 EFFICIENT LOW PRECISION NETWORKS

Low precision learning describes a set of techniques that take network parameters, typically stored at native 32-bit floating point (FP32) precision, and quantize them to a much smaller range of representation, typically 1-bit (binary) or 2-bit (ternary) integer values. While low precision learning could refer to any combination of quantizing weights (W), activations (A), and gradients (G), most relevant work investigates the effects of quantizing weights and activations on model performance. The benefits of quantization are seen in both computational and memory efficiency, though generally speaking, quantization leads to a decrease in model accuracy (Zhou et al., 2016). However, in some cases, the effects of quantization can be lossless or even slightly improve accu-

racy by behaving as a type of noisy regularization (Zhu et al., 2016; Yin et al., 2016). In this work, we adopt the DoReFa-Net model (Zhou et al., 2016) of quantizing all network parameters (W,A,G) albeit at different precision. Table 1 demonstrates this trade-off of precision and accuracy for some common low precision configurations.

The justification for this loss in accuracy is the efficiency gain of storing and computing low precision values. The computational benefits of using binary values are seen from approximating expensive full precision matrix operations with binary operations (Rastegari et al., 2016), as well as reducing memory requirements by packing many low precision values into full precision data types. For other low precision configurations that fall between binary and full precision, a similar formulation is used. The bit dot product equation (Equation 1) shows how both the logical `and` and `bitcount` operations are used to compute the dot product of two low-bitwidth fixed-point integers (bit vectors) (Zhou et al., 2016). Assume  $c_m(x)$  is a placewise bit vector formed from a sequence of  $M$ -bit fixed-point integers  $x = \sum_{m=0}^{M-1} c_m(x)2^m$  and  $c_k(y)$  is a placewise bit vector formed from a sequence of  $K$ -bit fixed-point integers  $y = \sum_{k=0}^{K-1} c_k(y)2^k$ , then

$$x \cdot y = \sum_{m=0}^{M-1} \sum_{k=0}^{K-1} 2^{m+k} \text{bitcount}[\text{and}(c_m(x), c_k(y))],$$

$$c_m(x)_i, c_k(y)_i \in \{0, 1\} \forall i, m, k. \quad (1)$$

Since the computational complexity of the operation is  $\mathcal{O}(MK)$ , the speedup is a function of the total number of bits used to quantize the inputs. Since matrix multiplications are simply sequences of dot products computed over the rows and columns of matrices, this is also true of matrix multiplication operations. As a demonstration, we implement this variable precision bit general matrix multiplication (bit-GEMM) using CUDA, and show the GPU speedup for several configurations in Figure 1.

As seen in Figure 1, there is a correlation between the total number of bits used in each bit-GEMM (shade) and the resulting speedup (point size). As such, from a hardware perspective, it is important to know how many total bits are used for bit-GEMM operations to allow for budgeting computation and memory. It should also be noted that, in our experiments, operations with over 16 total bits of precision were shown to be slower than the full precision equivalent. This is due to the computational complexity of the worst case of  $\mathcal{O}(8 \times 8)$  being slower than the equivalent full precision operation. We therefore focus on operations with 16 or less total bits of precision.

A natural question to follow is then, for a given budget of precision (total number of bits), how do we most

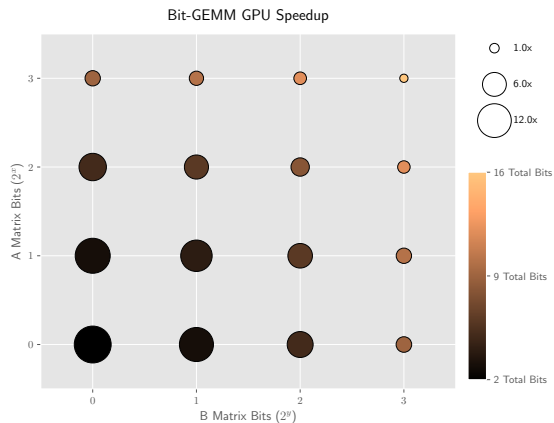


Figure 1: GPU-based bit-GEMM speedup for low precision matrices A and B. Results are compared with a similarly optimized 32-bit GEMM kernel, and run on a NVIDIA Tesla V100 GPU.

efficiently allocate precision to maximize model performance? We seek to answer this question by parametrizing the precision at each layer and learning these additional parameters by gradient descent.

### 3.1 LEARNING PRECISION

The selection of precision for variables in a model can have a significant impact on performance. Consider the DoReFa-Net model — if we decide on a budget of the total number of bits to assign to the weights layer-wise, and train the model under a number of different manual allocations, we obtain the training curves in Figure 2. For each training curve, the number of bits assigned to each layer’s weights are indicated by the integer at the appropriate position (e.g. **444444** indicates 6 quantized layers, all assigned 4 bits).

Varying the number of bits assigned to each layer can cause the error to change by up to several percent, but the best configuration of these bits is unclear without exhaustively testing all possible configurations. This motivates *learning* the most efficient allocation of precision to each layer. However, this is a difficult task for two important reasons:

- unconstrained parameters that control precision will only grow, as higher precision leads to a reduction of loss; and
- quantization involves discrete operations, which are non-differentiable and therefore unsuitable for naïve backpropagation.

The first issue is easily addressed by fixing the total net-

Table 1: DoReFa-Net single-crop top-1 validation error for common weight (W), activation (A), and gradient (G) quantization configurations on the ImageNet Large-Scale Visual Recognition Challenge 2012 (ILSVRC12) dataset. The results are slightly improved over the results originally reported and were reproduced by us based on the public DoReFa-Net (Zhou et al., 2016) codebase.

Model	W	A	G	Top-1 Validation Error
AlexNet (Krizhevsky et al., 2012)	32	32	32	41.4%
BWN (Courbariaux and Bengio, 2016)	1	32	32	44.3%
DoReFa-Net (Zhou et al., 2016)	1	2	6	47.6%
DoReFa-Net (Zhou et al., 2016)	1	2	4	58.4%

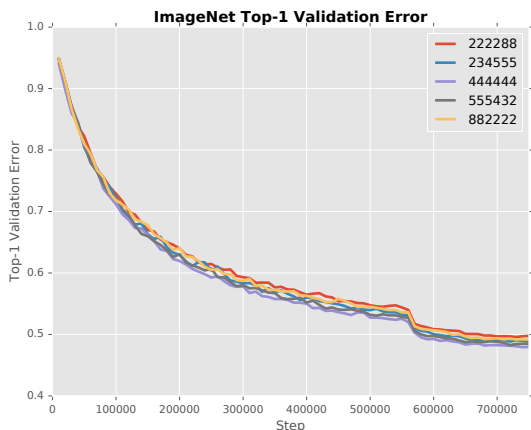


Figure 2: The training error for a variety of DoReFa-Net manual precision allocations is plotted for each weight update. The uniform distribution of bits (i.e. **444444**) leads to the lowest error, while the least uniform configurations (e.g. **222288**, **882222**) lead to the highest error.

work precision (i.e. the sum of the bits of precision at each layer) to a budget  $B$ , similar to the  $B = 24$  configurations seen in Figure 2. The task is then to learn the allocation of precision across layers. The second issue: the non-differentiable nature of quantization operations is an unavoidable problem, as transforming continuous values into discrete values must apply some kind of discrete operator. We avoid this issue by employing a kind of stochastic allocation of precision and rely on recently developed techniques from the deep learning community to backpropagate gradients through discrete stochastic operations.

Intuitively, we can view the precision allocation procedure as sequentially allocating one bit of precision to one of  $L$  layers. Each time we allocate, we draw from a categorical variable with  $L$  outcomes, and allocate that bit to the corresponding layer. This is repeated  $B$  times to match the precision budget. An allocation of  $B$  bits corresponds to a particular precision configuration, and

we sample a new configuration for each input example. The idea of stochastically sampling architectural configurations is akin to Dropout (Srivastava et al., 2014), where each example is processed by a different architecture with tied parameters.

Different from Dropout, which uses a fixed dropout probability, we would like to parametrize the categorical distribution across layers such that we can learn to prefer to allocate precision to certain layers. Learning these parameters by gradient descent requires backprop through an operator that samples from a discrete distribution. To deal with its non-differentiability, we use the Gumbel-Softmax, also known as the Concrete distribution (Jang et al., 2016; Maddison et al., 2016), which, using a temperature parameter, can be smoothly annealed from a uniform continuous distribution on the simplex to a discrete categorical distribution from which we sample precision allocations. This allows us to use a high temperature at the beginning of training to stochastically explore different precision configurations and use a low temperature at the end of training to discretely allocate precision to network layers according to the learned distribution.

Though non-integer bits of precision can be implemented (detailed below), integer bits are more amenable to hardware implementations, which is why we aim to converge toward discrete samples. Table 2 shows examples of sampling from this distribution at different temperatures for a three class distribution. It should be noted that in order to perform unconstrained optimization we parametrize the unnormalized logits (also known as log-odds) instead of the probabilities themselves.

Since the class logits  $\pi_i$  control the probability of allocating a bit of precision to a network layer  $l_i$ , at low temperatures the one-hot samples will allocate bits of precision to the network according to the learned parameters  $\pi_i$ . However, at high temperatures we allocate partial bits to layers. This is possible due to our quantization straight-through estimator (STE), `quantize`, adopted

Table 2: Examples of single samples drawn from a Gumbel-Softmax distribution, parametrized by logits for three classes  $\pi_1, \pi_2, \pi_3$  and a variety of temperatures  $\tau$ . The probability of allocating a bit to layer  $l_i$  is impacted by the logit  $\pi_i$ , where higher values correspond to higher probabilities. The Gumbel-Softmax interpolates between continuous densities on the simplex (at high temperature) and discrete one-hot-encoded categorical distributions (at low temperature).

$\tau$	Class 1 ( $\pi_1 = 1.00$ )	Class 2 ( $\pi_2 = 2.00$ )	Class 3 ( $\pi_3 = -0.50$ )
100.0	0.33	0.33	0.33
10.0	0.33	0.41	0.26
1.00	0.31	0.60	0.09
0.10	0.00	1.00	0.00

from (Zhou et al., 2016):

**Forward:**

$$r_o = \text{quantize}(r_i) = \frac{1}{2^k - 1} \text{round}((2^k - 1)r_i), \quad (2)$$

**Backward:**  $\frac{\partial c}{\partial r_i} = \frac{\partial c}{\partial r_o} \quad (3)$

where  $r_i$  is the real number input,  $r_o$  is the  $k$ -bit output, and  $c$  is the objective function. Since `quantize` produces a  $k$ -bit value on  $[0, 1]$ , quantizing to non-integer values of  $k$  simply produces a more fine-grained range of representation compared to integer values of  $k$ . This is demonstrated in Table 3.

Table 3: The possible output values of the `quantize` operation are shown for a variety of values of  $k$ , clipped between 0 and 1. Non-integer values of  $k$  provide a more fine-grained range of representation between successive integer values (underlined).

$k$	<u>1</u>	1.50	<u>2</u>	2.25	2.50	2.75	<u>3</u>
	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	1.00	0.55	0.33	0.26	0.21	0.17	0.14
		1.00	0.66	0.53	0.42	0.34	0.28
			1.00	0.80	0.64	0.52	0.42
				1.00	0.85	0.69	0.57
					1.00	0.87	0.71
						1.00	0.85
							1.00

Using real values for quantization also provides useful gradients for backpropagation, whereas the small finite set of possible integer values would yield zero gradient almost everywhere.

### 3.2 PRECISION ALLOCATION LAYER

We introduce a new layer type, the precision allocation layer, to implement our precision learning technique. This layer is inserted as a leaf in the computational graph, and is executed on each forward-pass of training. The precision allocation layer is parametrized by the learnable class probabilities  $\pi_i$ , which define the Gumbel-Softmax distribution. Each class probability is associated with a layer  $l_i$ , so the samples assigned to each class are allocated as bits to the appropriate layer. This is illustrated in Figure 3 and stepped through in Example 1.

It should be noted that during the early stages of training before the network performance has converged, allowing the temperature to drop too low results in high-variance gradients while also encouraging largely uneven allocations of bits. This severely hurts the generalization of the network. To deal with this, we empirically observe a temperature where the class probabilities have sufficiently stabilized, and perform hard assignments of bits of precision based on these stabilized class probabilities. To do this, we sample from the Gumbel-Softmax a large number of times and average the results, in order to converge on the expected class values, we fix the layers at these precision values for the remainder of training. We observe that the regularization effects of stochastic bit allocations are most useful during early training, and performing hard assignments greatly improves generalization performance. For all experiments considered, we implement a hard assignment of bits after the temperature drops below 3.0.

## 4 EXPERIMENTS

We evaluate the effects of our precision layer on two common image classification benchmarks, MNIST and ILSVRC12. We consider two separate CNN architectures, a 5-layer network similar to LeNet-5 (Lecun et al., 1998) trained on MNIST, and the AlexNet network (Krizhevsky et al., 2012) trained on ILSVRC12. We use the top-1 single-crop error as a measure of performance, and quantize both the weights and activations in all experiments considered. As in previous works (Zhou et al., 2016), the first layer of AlexNet was not quantized. Initial experiments showed that the effects of learning precision were less beneficial to gradients, so we leave them at full precision in all reported experiments.

Since our motivation is to show the precision layer as an improvement over uniformly quantized low precision models, we compare our results to networks with evenly distributed precision over all layers. We consider three common low precision budgets as powers of

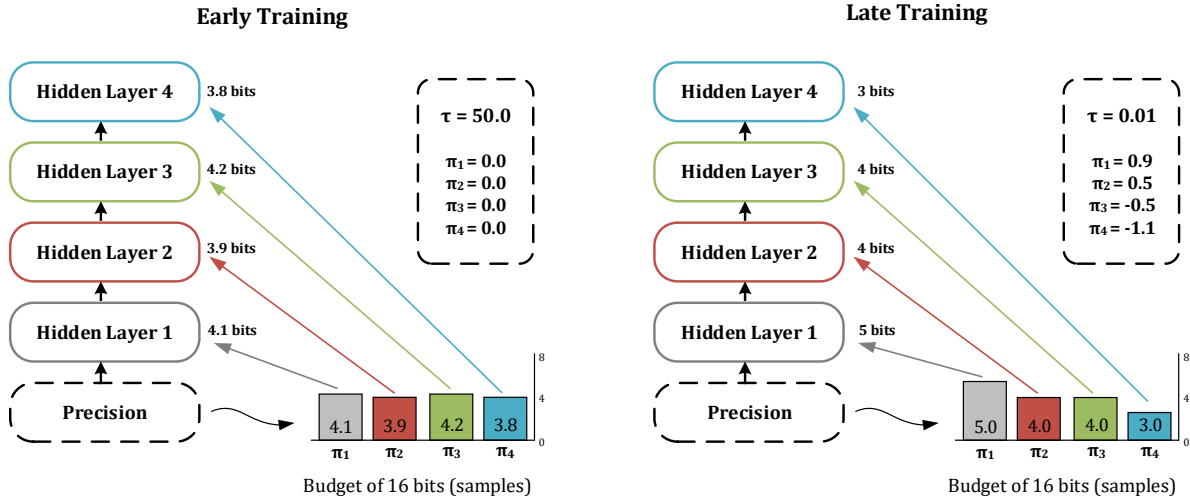


Figure 3: Early in training, the Gumbel-Softmax class probabilities  $\pi_i$  are initialized to 0 while the temperature  $\tau$  is high, generally resulting in uniform allocations of real-valued bits of precision. Later in training, with a low temperature, the learned class probabilities usually result in some layers being allocated more or less discrete bits of precision. In practice, the computational overhead of the precision layer is not noticeable.

**Example 1.** Consider a network with 4 layers, each denoted by  $l_i$ . To learn the precision of these layers, we add a precision layer which constructs a Gumbel-Softmax distribution with 4 classes  $\pi_i$ , where each class is assigned to a layer. Each class probability is initialized to 0.0, and the temperature is initialized to 50.0. For a budget of 16 total bits, two example iterations representative of early training (first iteration of epoch 0) and late training (first iteration of epoch 50) are shown below:

**Epoch = 0**,  $\pi_1 = 0.0$ ,  $\pi_2 = 0.0$ ,  $\pi_3 = 0.0$ ,  $\pi_4 = 0.0$ ,  $\tau = 50.0$

- The precision layer is executed first, which means we sample from the Gumbel-Softmax 16 times because our budget is 16 bits, and accumulate the results of the 16 samples. Each individual sample gives us 4 class outputs which sum to 1 (e.g. [0.25, 0.25, 0.25, 0.25]), so sampling 16 times and accumulating the results for each class means our final results will add to 16. Since the class probabilities are initialized to 0.0, and the temperature is high, the expected samples will be continuous and relatively uniform across all classes.
  - The samples associated with each layer  $l_i$  are:  $l_1 = 4.1$ ,  $l_2 = 3.9$ ,  $l_3 = 4.2$ ,  $l_4 = 3.8$ .
  - We now assign 4.1 bits to layer 1, 3.9 bits to layer 2, 4.2 bits to layer 3, and 3.8 bits to layer 4. These bits are assigned to the appropriate layer by applying the `quantize` operation of Equation 2 to the desired parameters (e.g. weights) with  $k$  as the appropriate bit assignment (e.g.  $k = 4.1$  for  $l_1$ ).
  - The `quantize` operation will transform the layer parameters to one of several discrete positive quantities between 0 and 1 (see Table 3). Though these are fractional bit assignments, the `quantize` operations works the same as if these were integer bit assignments.
  - The iteration then proceeds as normal, with the quantized parameters and class probabilities  $\pi_i$  updated during back-propagation.

**Epoch = 50**,  $\pi_1 = 0.9$ ,  $\pi_2 = 0.5$ ,  $\pi_3 = -0.5$ ,  $\pi_4 = -1.1$ ,  $\tau = 0.01$

- Similar to before, we begin by sampling from the Gumbel-Softmax 16 times because our budget is 16 bits, and accumulate the results. However, the class probabilities have now changed such that  $\pi_1$  corresponds to the most likely class, and  $\pi_4$  the least likely class, so the distribution is no longer uniform. As well, since the temperature is low ( $\tau = 0.01$ ), the samples will now approach discrete.
  - Samples:  $l_1 = 5.0$ ,  $l_2 = 4.0$ ,  $l_3 = 4.0$ ,  $l_4 = 3.0$ .
  - We now assign 5.0 bits to layer 1, 4.0 bits to layer 2, 4.0 bits to layer 3, and 3.0 bits to layer 4, similar to before. Since these bit assignments are integer values, the activity of the `quantize` operation is more intuitive.
  - Similar to before, the `quantize` operation will transform the layer parameters to one of several discrete positive quantities between 0 and 1 (see Table 3).
  - The forward-pass then proceeds as normal, with the quantized parameters and class probabilities  $\pi_i$  updated during back-propagation. Since this is late training, parameter updates will be smaller in magnitude than in early training.



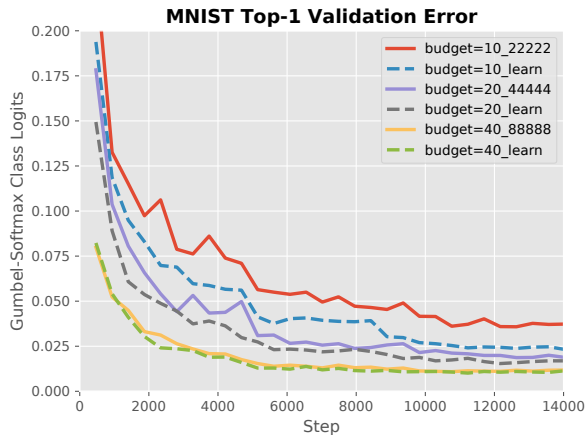


Figure 4: The top-1 errors for the MNIST networks are shown. Both baseline networks with uniform precision allocation (i.e. **budget=10\_22222**) and learned precision networks (i.e. **budget=10\_learn**) are assigned the same total precision budget, indicated by the prefix.

2, which would provide efficient hardware acceleration, where each layer is allocated 2, 4, or 8 bits. Baseline networks with uniform precision allocation are denoted with the allocation for each layer and the budget (e.g. **budget=10\_22222** denotes a baseline network with 2 bits manually allocated to each of 5 layers for a budget of 10 bits), while the networks with learned precision are denoted with the precision budget and *learn* (e.g. **budget=10\_learn** denotes a learned precision network with a budget of 10 total bits, averaging 2 bits per layer).

#### 4.1 MNIST

The training curves for the MNIST-trained models are shown in Figure 4. The learned Gumbel-Softmax class logits are shown in Figure 5.

We observe that the models with learned precision converge faster and reach a lower test error compared to the baseline models across all precision budgets considered, and the relative improvement is more substantial for lower precision budgets. From Figure 5, we observe that the models learn to assign fewer bits to early layers of the network (conv0, conv1) while assigning more bits to later layers of the network (conv2, conv3), as well as preferring smoother (i.e. more uniform) allocations. This result agrees with the empirical observations of (Raghavan et al., 2017). The results on MNIST are summarized in Table 4. Uncertainty is calculated by averaging over 10 runs for each network with different random initializations of the parameters.

#### 4.2 ILSVRC12

The results for ILSVRC12 are summarized in Table 5. Again, models that employ stochastic precision allocation converge faster and ultimately reach a lower test error than their fixed-precision counterparts on the same budget. We observe that networks trained with stochastic precision learn to take bits from early layers and assign these to later layers, similar to the MNIST results. While the class logits for the MNIST network were similar to the ILSVRC12 results, they were not substantial enough to cause changes in bit allocation during our hard assignments. However, the ILSVRC12-trained networks actually make non-uniform hard assignments. This suggests that the precision layer has a larger affect on more complex networks.

### 5 CONCLUSION AND FUTURE WORK

We introduced a precision allocation layer for DNNs, and proposed a stochastic allocation scheme for learning precision on a fixed budget. We have shown that learned precision models outperform uniformly-allocated low precision models. This effect is due to both learning the optimal configuration of precision layer-wise, as well as the regularization effects of stochastically exploring different precision configurations during training. Moreover, the use of precision budgets allow a high level of hardware acceleration determinism which has practical implications.

While the present experiments were focused on accuracy rather than computational efficiency, future work will examine using GPU bit kernels in place of the full precision kernels we used in our experiments. We also intend to investigate stochastic precision in the adversarial setting. This is inspired by Galloway et al. (2018), who report that stochastic quantization at test time yields robustness towards iterative attacks.

Finally, we are interested in a variant of the model where rather than directly parametrizing precision, precision is conditioned on the input. While this reduces hardware acceleration determinism in real-time or memory-constrained settings, it would enable a DNN to dynamically adapt its precision configuration to individual examples.

### References

- E. Agustsson, F. Mentzer, M. Tschannen, L. Cavigelli, R. Timofte, L. Benini, and L. Van Gool. Soft-to-hard vector quantization for end-to-end learned compression of images and neural networks. *arXiv preprint arXiv:1704.00648*, 2017.

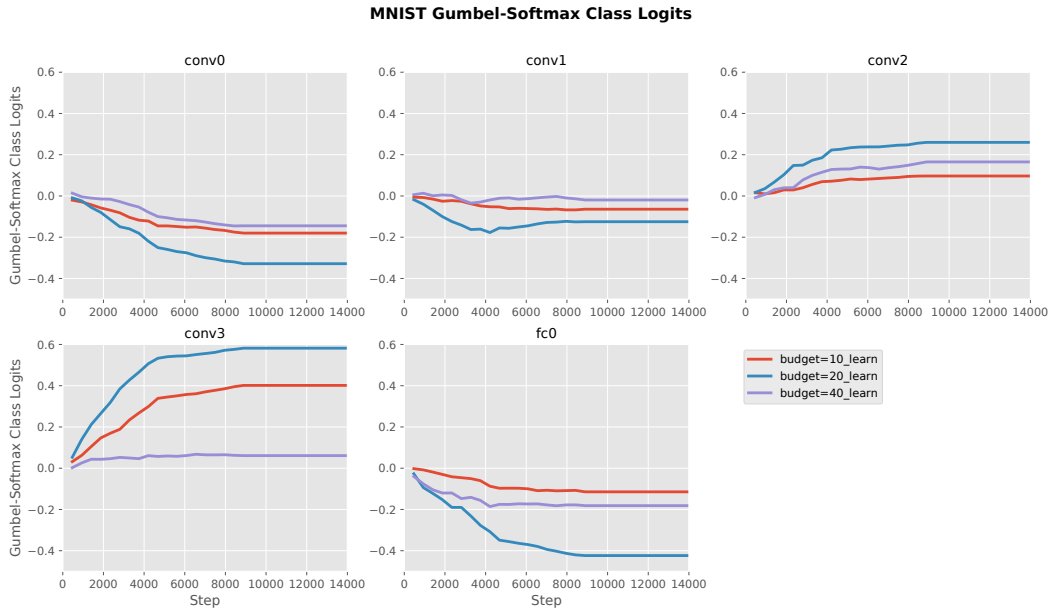


Figure 5: The learned Gumbel-Softmax class logits for different precision budgets.

Table 4: Final training results and top-1 error for the MNIST baseline and learned precision models.

Network	Precision Budget	Val. Error	Final Bit Allocation				
			$l_1$	$l_2$	$l_3$	$l_4$	$l_5$
budget=10_22222	10 Bits	3.73% $\pm$ 0.3	2	2	2	2	2
budget=10_learn	10 Bits	2.32% $\pm$ 0.3	2	2	2	2	2
budget=20_44444	20 Bits	1.88% $\pm$ 0.2	4	4	4	4	4
budget=20_learn	20 Bits	1.69% $\pm$ 0.2	4	4	4	4	4
budget=40_88888	40 Bits	1.18% $\pm$ 0.1	8	8	8	8	8
budget=40_learn	40 Bits	1.14% $\pm$ 0.1	8	8	8	8	8

Table 5: Final training results and top-1-error for the ILSVRC12 baseline and learned precision models.

Network	Precision Budget	Val. Error	Final Bit Allocation						
			$l_1$	$l_2$	$l_3$	$l_4$	$l_5$	$l_6$	$l_7$
budget=14_2222222	14 Bits	49.68% $\pm$ 0.3	2	2	2	2	2	2	2
budget=14_learn	14 Bits	48.54% $\pm$ 0.3	2	2	2	2	2	2	2
budget=28_4444444	28 Bits	47.99% $\pm$ 0.3	4	4	4	4	4	4	4
budget=28_learn	28 Bits	47.69% $\pm$ 0.3	3	3	4	4	5	5	4
budget=56_8888888	56 Bits	47.70% $\pm$ 0.3	8	8	8	8	8	8	8
budget=56_learn	56 Bits	47.46% $\pm$ 0.3	7	7	8	8	9	9	8

- S. Anwar, K. Hwang, and W. Sung. Structured pruning of deep convolutional neural networks. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 13(3):32, 2017.
- D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
- Z. Cai, X. He, J. Sun, and N. Vasconcelos. Deep learning with low precision by half-wave gaussian quantization. *arXiv preprint arXiv:1702.00953*, 2017.
- A. Canziani, A. Paszke, and E. Culurciello. An analysis of deep neural network models for practical applications. *arXiv preprint arXiv:1605.07678*, 2016.
- Y. Choi, M. El-Khamy, and J. Lee. Towards the limit of network quantization. *arXiv preprint arXiv:1612.01543*, 2016.
- M. Courbariaux and Y. Bengio. Binarynet: Training deep neural networks with weights and activations constrained to +1 or -1. *arXiv preprint arXiv:1602.02830*, 2016.
- M. Courbariaux, Y. Bengio, and J. David. Binaryconnect: Training deep neural networks with binary weights during propagations. *arXiv preprint arXiv:1511.00363*, 2015.
- J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009.
- A. Galloway, G. W. Taylor, and M. Moussa. Attacking binarized neural networks. In *International Conference on Learning Representations (ICLR)*, 2018.
- P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.
- S. Han, H. Mao, and W. J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. *arXiv preprint arXiv:1510.00149*, 2015a.
- S. Han, J. Pool, J. Tran, and W. Dally. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems*, pages 1135–1143, 2015b.
- G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.
- E. Jang, S. Gu, and B. Poole. Categorical reparameterization with Gumbel-Softmax. *arXiv preprint arXiv:1611.01144*, 2016.
- X. Jin, X. Yuan, J. Feng, and S. Yan. Training skinny deep neural networks with iterative hard thresholding methods. *arXiv preprint arXiv:1607.05423*, 2016.
- M. Kim and P. Smaragdis. Bitwise neural networks. *arXiv preprint arXiv:1601.06071*, 2016.
- A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105. 2012.
- Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, 1998.
- F. Li, B. Zhang, and B. Liu. Ternary weight networks. *arXiv preprint arXiv:1605.04711*, 2016a.
- H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016b.
- C. J. Maddison, A. Mnih, and Y. W. Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.
- P. Merolla, R. Appuswamy, J. Arthur, S. K. Esser, and D. Modha. Deep neural networks are robust to weight binarization and other non-linear distortions. *arXiv preprint arXiv:1606.01981*, 2016.
- P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz. Pruning convolutional neural networks for resource efficient transfer learning. *arXiv preprint arXiv:1611.06440*, 2016.
- A. Raghavan, M. Amer, S. Chai, and G. W. Taylor. BitNet: Bit-regularized deep neural networks. *arXiv preprint arXiv:1708.04788*, 2017.
- M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. *arXiv preprint arXiv:1603.05279*, 2016.
- N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, 2014.
- F. Tung, S. Muralidharan, and G. Mori. Fine-Pruning: Joint Fine-Tuning and compression of a convolutional network with bayesian optimization. In *British Machine Vision Conference (BMVC)*, 2017.
- L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, and R. Fergus. Regularization of neural networks using dropconnect.

- In *International Conference on Learning Representations (ICLR)*, pages 1058–1066, 2013.
- X. Wang and J. Liang. Scalable compression of deep neural networks. In *Proceedings of the ACM on Multimedia Conference*, pages 511–515, 2016.
- P. Yin, S. Zhang, Y. Qi, and J. Xin. Quantization and training of low Bit-Width convolutional neural networks for object detection. *arXiv preprint arXiv:1612.06052*, 2016.
- S. Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou. DoReFa-Net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016.
- C. Zhu, S. Han, H. Mao, and W. J. Dally. Trained ternary quantization. *arXiv preprint arXiv:1612.01064*, 2016.

---

# Estimation of Personalized Effects Associated With Causal Pathways

---

**Razieh Nabi**

Computer Science Dept.  
Johns Hopkins University  
Baltimore, MD, 21218

**Phyllis Kanki**

Immunology and Infectious Diseases Dept.  
Harvard T. H. Chan School of Public Health  
Boston, MA, 02115

**Ilya Shpitser**

Computer Science Dept.  
Johns Hopkins University  
Baltimore, MD, 21218

## Abstract

The goal of personalized decision making is to map a unit's characteristics to an action tailored to maximize the expected outcome for that unit. Obtaining high-quality mappings of this type is the goal of the dynamic regime literature. In healthcare settings, optimizing policies with respect to a particular causal pathway may be of interest as well. For example, we may wish to maximize the chemical effect of a drug given data from an observational study where the chemical effect of the drug on the outcome is entangled with the indirect effect mediated by differential adherence. In such cases, we may wish to optimize the direct effect of a drug, while keeping the indirect effect to that of some reference treatment. [15] shows how to combine mediation analysis and dynamic treatment regime ideas to define policies associated with causal pathways and counterfactual responses to these policies. In this paper, we derive a variety of methods for learning high quality policies of this type from data, in a causal model corresponding to a longitudinal setting of practical importance. We illustrate our methods via a dataset of HIV patients undergoing therapy, gathered in the Nigerian PEPFAR program.

## 1 INTRODUCTION

There has been growing interest in making personalized decisions in different domains to account for inherent heterogeneity among individuals and optimize individual-level experiences. For instance, personalized medicine aims at systematic use of individual patient history including biological information and biomarkers to

improve patient's health care. Personalized actions can be viewed as realizations of decision rules where available information is mapped to the space of possible decisions.

Making good personalized decisions often involves acting in multiple stages. For instance, multiple successive medical interventions may be required for long-term care of patients with chronic diseases. The goal of personalized medicine is to tailor a sequence of decision rules on treatment, known as dynamic treatment regimes, based on patient characteristics seen so far, to maximize the likelihood of a desirable outcome. A number of algorithms have been developed for finding optimal treatment regimes from either observational data, or data from experiments tailored for providing information on regime quality, such as sequential multiple assignment randomized trials (SMARTs) [3, 6]. These algorithms use methods from causal inference, and aim to predict counterfactual outcomes under policies different from those actually followed in the data [11, 5, 2].

A natural extension of these methods is finding treatment regimes that optimizes a *part* of the effect of the treatment on the outcome. We illustrate the utility of this problem with the following example. Patients infected with human immunodeficiency virus (HIV) are typically put on courses of antiretroviral therapy, as a first line of therapy. Although these sort of medications are effective in combating the disease, the full benefit is not realized since patients often do not fully adhere to the medication regimen. One of the causes of poor adherence to the therapy is toxicity of the medication. Hence, viral failure in a patient receiving treatment may be attributed to either poor drug effectiveness or lack of adherence to an otherwise effective treatment plan. In observational studies of HIV patients, treatments are not randomly assigned, and patients have differential adherence. Because of this, finding a policy that optimizes the *overall effect* of the treatment plan on the outcome entangles two very different causal pathways – the chemical pathway asso-

ciated with the active ingredients in the treatment, and the pathway associated with adherence.

We may, instead, consider the problem of finding a set of policies that optimize only the direct chemical effect of the drug, in the counterfactual situations where the indirect effect mediated by adherence can be kept to that of some reference treatment. Policies of this type may be more directly relevant in precision medicine contexts where adherence varies among patients.

[15] defines counterfactual responses to policies that set treatments only with respect to a particular causal pathway, and gives a general identification algorithm for these responses, as a generalization of similar algorithms for standard dynamic treatment regimes [19], and effects associated with causal pathways in mediation analysis [14]. In this paper, we consider algorithms that can be used to find policies that maximize effects along particular causal pathways from data, in a causal model of most immediate relevance in longitudinal settings.

The paper is organized as follows. We fix our notation, and define counterfactual responses to treatment policies and policies associated with pathways in Section 2. In Section 3, we review existing techniques used in learning policies optimizing the *overall effect* of actions. In Section 4, we fix the causal model corresponding to a typical longitudinal study, prove identification of counterfactual policy responses under this model, and show how techniques for maximizing treatment policies, described in Section 4, may be extended to maximize policies associated with causal pathways. In Section 5, we illustrate the methods we propose via an application involving data on treatment of HIV patients in Nigeria. Our conclusions are in Section 6. The Appendix contains the proofs of all claims, a basic description of statistical inference in semi-parametric models, as context for some of our estimation strategies, visualizations of learned policies, and descriptions of the experiments.

## 2 NOTATIONS AND PRELIMINARIES

Consider a multi-stage decision problem with  $K$  pre-specified decision points, indexed by  $i = 1, \dots, K$ . Let  $Y$  denote the final outcome of interest and  $A_i$  denote the action made at decision point  $i$  with the finite state space of  $\mathfrak{X}_{A_i}$ . The set of all actions is denoted by  $\mathbf{A}$ . Let  $W_0$  denote the available information prior to the first decision, and  $W_i$  denote the information collected between decisions  $i$  and  $i+1$ , ( $Y \equiv W_K$ ). Given  $A_i$ , denote  $\bar{A}_i$  to be all treatments administered from time 1 to  $i$ , similarly for  $W_i$  and  $\bar{W}_i$ . We combine the treatment and covariate history up to treatment decision  $A_i$  into a history vector  $H_i$ . The state space of  $H_i$  is denoted by  $\mathfrak{X}_{H_i}$ .

We are interested in learning policies that map  $H_i$  to values of  $A_i$ , for all  $i$ , that maximize the expected value of the outcome  $Y$ . Doing this from observed data entails considering *counterfactual outcomes*  $Y$  had  $A_i$  been assigned in a different way from what was actually observed. We briefly review graphical causal models, and the potential outcome notation from causal inference, which will be used to define such counterfactuals.

Causal models are sets of distributions defined by restrictions associated with directed acyclic graphs (DAGs). We will use vertices and variables interchangeably – capital letters for a vertex or variable ( $V$ ), bold capital letter for a set ( $\mathbf{V}$ ), lowercase letters for values ( $v$ ), and bold lowercase letters for sets of values ( $\mathbf{v}$ ). By convention, each graph is defined on a vertex set  $\mathbf{V}$ . For a set of values  $\mathbf{a}$  of  $\mathbf{A}$ , and a subset  $\mathbf{A}^\dagger \subseteq \mathbf{A}$ , define  $\mathbf{a}_{\mathbf{A}^\dagger}$  to be a restriction of  $\mathbf{a}$  to elements in  $\mathbf{A}^\dagger$ . For a DAG  $\mathcal{G}$ , and any  $V \in \mathbf{V}$ , we define the parents of  $V \in \mathbf{V}$  to be the set  $\text{pa}_{\mathcal{G}}(V) \equiv \{W \in \mathbf{V} \mid W \rightarrow V\}$ , and the children of  $V \in \mathbf{V}$  to be the set  $\text{ch}_{\mathcal{G}}(V) \equiv \{W \in \mathbf{V} \mid V \rightarrow W\}$ .

Causal models of a DAG  $\mathcal{G}$  consist of distributions defined on counterfactual random variables of the form  $V(\mathbf{a})$  where  $\mathbf{a}$  are values of  $\text{pa}_{\mathcal{G}}(V)$ . These variables represent outcomes of  $V$  had all variables in  $\text{pa}_{\mathcal{G}}(V)$  been set, possibly contrary to fact, to  $\mathbf{a}$ . In this paper we assume Pearl’s functional model for a DAG  $\mathcal{G}$  with vertices  $\mathbf{V}$  which is the set containing any joint distribution over all potential outcome random variables where the sets of variables  $\{\{V(\mathbf{a}_V) \mid \mathbf{a}_V \in \mathfrak{X}_{\text{pa}_{\mathcal{G}}(V)}\} \mid V \in \mathbf{V}\}$  are mutually independent [8]. The *atomic counterfactuals* in the above set model the relationship between  $\text{pa}_{\mathcal{G}}(V)$ , representing direct causes of  $V$ , and  $V$  itself. From these, all other counterfactuals may be defined using *recursive substitution*. For any  $\mathbf{A} \subseteq \mathbf{V} \setminus \{V\}$ ,

$$V(\mathbf{a}) \equiv V(\mathbf{a}_{\text{pa}_{\mathcal{G}}(V) \cap \mathbf{A}}, \{\text{pa}_{\mathcal{G}}(V) \setminus \mathbf{A}\}(\mathbf{a})), \quad (1)$$

where  $\{\text{pa}_{\mathcal{G}}(V) \setminus \mathbf{A}\}(\mathbf{a})$  is taken to mean the (recursively defined) set of counterfactuals associated with variables in  $\text{pa}_{\mathcal{G}}(V)$ , had  $\mathbf{A}$  been set to  $\mathbf{a}$ .

A causal parameter is said to be *identified* in a causal model if it is a function of the observed data distribution  $p(\mathbf{V})$ . In all causal models of a DAG  $\mathcal{G}$  in the literature, all interventional distributions  $p(\{\mathbf{V} \setminus \mathbf{A}\}(\mathbf{a}))$  are identified by the *g-formula*:

$$p(\{\mathbf{V} \setminus \mathbf{A}\}(\mathbf{a})) = \prod_{V \in \mathbf{V} \setminus \mathbf{A}} p(V \mid \text{pa}_{\mathcal{G}}(V)) \Big|_{\mathbf{A}=\mathbf{a}} \quad (2)$$

As an example,  $Y(\mathbf{a})$  in the DAG in Fig. 1 (a), is defined to be  $Y(a, M(a, W), W)$ , and its distribution is identified as  $\sum_{w,m} p(Y|a, m, w)p(m|a, w)p(w)$ . In our sequential decision setting, the relevant counterfactual is  $Y(\bar{a}_K)$ , i.e. the response of  $Y$  had the treatment history  $\bar{A}_K = \bar{a}_K$  been administered, possibly contrary to fact. Comparison of  $Y(\bar{a}_K)$  and  $Y(\bar{a}'_K)$  in expectation,

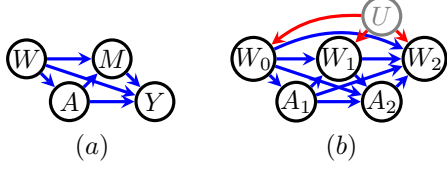


Figure 1: (a) A simple causal DAG, with a single treatment  $A$ , a single outcome  $Y$ , a vector  $W$  of baseline variables, and a single mediator  $M$ . (b) A more complex causal DAG with two treatments  $A_1, A_2$ , an intermediate outcome  $W_1$ , and the final outcome  $W_2$ .

$\mathbb{E}[Y(\bar{a}_K)] - \mathbb{E}[Y(\bar{a}'_K)]$ , where  $\bar{a}_K$  is the treatment history of interest, and  $\bar{a}'_K$  is the reference treatment history, gives the average causal effect of  $\bar{a}_K$  on the outcome  $Y$ .

### Counterfactual Response to Policies

A dynamic treatment regime  $\mathbf{f}_A = \{f_{A_1}, \dots, f_{A_K}\}$  is a sequence of decision rules that forms a treatment plan for patients over time. At the  $i$ th decision point, the  $i$ th rule  $f_{A_i}$  maps the available information  $H_i$  prior to the  $i$ th decision point to a treatment  $a_i$ , i.e.  $f_{A_i} : \mathfrak{X}_{H_i} \mapsto \mathfrak{X}_{A_i}$ . Given a treatment regime  $\mathbf{f}_A$ , we define the counterfactual response  $Y$  had  $\mathbf{A}$  been assigned according to  $\mathbf{f}_A$ , or  $Y(\mathbf{f}_A)$ , as the following generalization of (1)

$$Y(\{f_{A_i}(H_i(\mathbf{f}_A)) | A_i \in \text{pa}_{\mathcal{G}}(Y) \cap \mathbf{A}\}, \{\text{pa}_{\mathcal{G}}(Y) \setminus \mathbf{A}\}(\mathbf{f}_A)). \quad (3)$$

Under a causal model associated with the DAG  $\mathcal{G}$ , the distribution  $p(Y(\mathbf{f}_A))$ , is identified as the following generalization of the g-formula

$$\sum_{V \setminus Y, A_V \in V \setminus \mathbf{A}} \prod p(V | \{f_{A_i}(H_i) | A_i \in \text{pa}_{\mathcal{G}}(V) \cap \mathbf{A}\}, \text{pa}_{\mathcal{G}}(V) \setminus \mathbf{A}) \quad (4)$$

As an example,  $Y(a = f_A(W))$  in Fig. 1 (a) is defined as  $Y(a = f_A(W), M(a = f_A(W), W), W)$ , and its distribution is identified as  $\sum_{w,m} p(Y | a = f_A(w), m, w) p(m | a = f_A(w), w) p(w)$ . We will call policies corresponding to counterfactuals defined in (3) *overall policies*, to distinguish them from policies associated with causal pathways to be defined later.

### Mediation Analysis

An important goal of causal inference is understanding the mechanisms by which the treatment  $A$  influences the outcome  $Y$ . A common framework for mechanism analysis is *mediation analysis* which seeks to decompose the effect of  $A$  on  $Y$  into the *direct effect* and the *indirect effect* mediated by a third variable, or more generally into components associated with particular causal pathways. Consider the graph in Fig. 1 (a): the direct effect corresponds to the effect along the edge  $A \rightarrow Y$ , and indirect effect corresponds to the effect along the path  $A \rightarrow M \rightarrow Y$ , mediated by  $M$ .

Counterfactuals associated with mediation analysis have been defined using a more general type of intervention in a graphical causal model, namely the *edge intervention* [16], which maps a set of directed edges in  $\mathcal{G}$  to values of their source vertices. Edge interventions have a natural interpretation in cases where a treatment variable has multiple components that a) influence the outcome in different ways, b) occur or do not occur together in observed data, and c) may in principle be intervened on separately. For instance, smoking leads to poor health outcomes due to two components: smoke inhalation and exposure to nicotine. A smoker would be exposed to both of these components, while a non-smoker to neither. However, one might imagine exposing someone selectively only to nicotine but not smoke inhalation (via a nicotine patch), or only smoke inhalation but not nicotine (via smoking plant matter not derived from tobacco leaves). These types of hypothetical experiments correspond precisely to edge interventions, and have been used to conceptualize direct and indirect effects [12, 7].

We will write the mapping of a set of edges to values of their source vertices as  $\mathbf{a}_\alpha$  to mean edges in  $\alpha$  are mapped to values in the *multiset*  $\mathbf{a}$  (since multiple edges may share the same source vertex, and be assigned to different values). For a subset  $\beta \subseteq \alpha$  and an assignment  $\mathbf{a}_\alpha$ , denote  $\mathbf{a}_\beta$  to be a restriction of  $\mathbf{a}_\alpha$  to edges in  $\beta$ . For simplicity, in the remainder of the paper we assume that if  $(AW)_{\rightarrow} \in \alpha$ , then for all  $V \in \text{ch}_{\mathcal{G}}(A)$ ,  $(AV)_{\rightarrow} \in \alpha$ , where  $(XY)_{\rightarrow}$  is a directed edge from  $X$  to  $Y$ .

We will write counterfactual responses to edge interventions as  $Y(\mathbf{a}_\alpha)$ . An edge intervention that sets a set of edges  $\alpha$  to values in the multiset  $\mathbf{a}$  is defined via the following generalization of recursive substitution (1):

$$Y(\mathbf{a}_\alpha) \equiv Y(\mathbf{a}_{\{(ZY)_{\rightarrow} \in \alpha\}}, \{\text{pa}_{\mathcal{G}}^{\bar{\alpha}}(Y)\}(\mathbf{a}_\alpha)), \quad (5)$$

where  $\text{pa}_{\mathcal{G}}^{\bar{\alpha}}(Y) \equiv \{W \mid (WY)_{\rightarrow} \notin \alpha\}$ . For example, in the DAG in Fig. 1 (a),  $Y(\mathbf{a}_{\{(AY)_{\rightarrow}, (AM)_{\rightarrow}\}})$  assigning  $(AY)_{\rightarrow}$  to  $a$  and  $(AM)_{\rightarrow}$  to  $a'$  is defined as  $Y(a, M(a', W), W)$ .

Identification of edge interventions in graphical causal models of a DAG corresponds quite closely with identification of regular (node) interventions, as follows. Let  $\mathbf{A}_\alpha \equiv \{A \mid (AB)_{\rightarrow} \in \alpha\}$ . Consider an edge intervention given by the mapping  $\mathbf{a}_\alpha$ . Then, under the functional model of a DAG  $\mathcal{G}$ , the joint distribution of counterfactual responses  $p(\{\mathbf{V} \setminus \mathbf{A}_\alpha\}(\mathbf{a}_\alpha))$  is identified via the following generalization of (2) called the *edge g-formula*:

$$\prod_{V \in \mathbf{V} \setminus \mathbf{A}_\alpha} p(V | \mathbf{a}_{\{(ZV)_{\rightarrow} \in \alpha\}}, \text{pa}_{\mathcal{G}}^{\bar{\alpha}}(V)). \quad (6)$$

For example, in Fig 1 (a),  $p(Y(\mathbf{a}_{\{(AY)_{\rightarrow}, (AM)_{\rightarrow}\}})) = \sum_{W,M} p(Y | a, M, W) p(M | a', W) p(W)$ , which is obtained by marginalizing  $W$  and  $M$  from the edge g-formula.

## Counterfactual Responses To Policies Associated With Pathways

We now define counterfactual responses to policies that operate only with respect to particular outgoing edges from  $\mathbf{A}$ . These counterfactuals generalize those in the previous two sections.

As an example, we can view Fig. 1 (a) as representing a cross-sectional study of HIV patients of the kind described in [4], where  $W$  is a set of baseline characteristics,  $A$  is one of a set of possible antiretroviral treatments,  $M$  is adherence to treatment, and  $Y$  is a binary outcome variable signifying viral failure. In this type of study, we may wish to find  $f_A(W)$  that maximizes the expected outcome  $Y$  had  $A$  been set according to  $f_A(W)$  for the purposes of the direct effect of  $A$  on  $Y$ , and  $A$  were set to some reference level  $a'$  for the purposes of the effect of  $A$  on  $M$ . In other words, we may wish to find  $f_A(W)$  to maximize the counterfactual mean  $\mathbb{E}[Y(f_A(W), M(a', W), W)]$ . This would correspond to finding a treatment policy that maximizes the direct (chemical) effect, if it were possible to keep adherence to a level  $M(a')$  as if a reference (easy to adhere to) treatment  $a$  were given.

We now give a general definition for responses to such path-specific policies. Fix a set of directed edges  $\alpha$ , and define  $\mathbf{A}_\alpha \equiv \{A \mid (AB)_\rightarrow \in \alpha\}$ . As before, we assume if  $(AW)_\rightarrow \in \alpha$ , then for all  $V \in \text{ch}_G(A)$ ,  $(AV)_\rightarrow \in \alpha$ . Define  $f_\alpha \equiv \{f_{A_i}^{(A_i W)_\rightarrow} : \mathfrak{X}_{H_i} \mapsto \mathfrak{X}_{A_i} \mid (A_i W)_\rightarrow \in \alpha\}$  as the set of policies associated with edges in  $\alpha$ . Note that  $f_\alpha$  may contain multiple policies for a given treatment variable  $A$ .

Define  $Y(f_\alpha)$ , the counterfactual response of  $Y$  to the set of path-specific policies  $f_\alpha$ , as the following generalization of (5) and (3):

$$Y(\{f_{A_i}^{(A_i Y)_\rightarrow}(H_i(f_\alpha)) \mid (A_i Y)_\rightarrow \in \alpha\}, \{\text{pa}_G^\alpha(Y)\}(f_\alpha)) \quad (7)$$

To reformulate our earlier example, if  $\tilde{f}_A^{(AM)_\rightarrow}$  assigns  $A$  to a constant value  $a'$ , and  $\tilde{f}_{\{(AY)_\rightarrow, (AM)_\rightarrow\}} \equiv \{f_A^{(AY)_\rightarrow}(W), \tilde{f}_A^{(AM)_\rightarrow}\}$ , then  $Y(\tilde{f}_{\{(AY)_\rightarrow, (AM)_\rightarrow\}}) \equiv Y(f_A^{(AY)_\rightarrow}(W), M(a', W), W)$ .

The joint counterfactual distribution for responses to path-specific policies,  $p(\{V(f_\alpha) \mid V \in \mathbf{V} \setminus \mathbf{A}_\alpha\})$ , is identified under the functional model, and generalizes (6) and (4) as follows:

$$\prod_{V \in \mathbf{V} \setminus \mathbf{A}_\alpha} p(V \mid \{f_{A_i}^{(A_i V)_\rightarrow}(H_i) \mid (A_i V)_\rightarrow \in \alpha\}, \text{pa}_G^\alpha(V)) \quad (8)$$

For example,  $p(Y(f_A(W), M(a', W), W))$  is identified as  $\sum_{W, M} p(Y \mid f_A(W), M, W) p(M \mid a', W) p(W)$  in Fig. 1 (a). We prove a general version of this result in the Appendix. A general identification theory for responses to path-specific policies can be found in [15].

## 3 LEARNING OPTIMAL OVERALL POLICIES

The goal of this paper is developing methods for learning optimal path-specific policies, in cases where responses to such policies are identified. Before discussing optimal path-specific policies, we first review existing approaches to finding optimal overall policies. Optimality may be quantified in a number of ways. The set of optimal policies is commonly defined as  $\mathbf{f}_\mathbf{A}^* \equiv \arg \max_{\mathbf{f}_\mathbf{A}} \mathbb{E}[Y(\mathbf{f}_\mathbf{A})]$ .

We will discuss existing methods for finding optimal policies  $\mathbf{f}_\mathbf{A}^*$  in the context of a causal model implying *positivity and sequential ignorability* [10]. This model is graphically represented, for two time points, in Fig. 1 (b), where  $W_0$  are baseline factors,  $W_1$  and  $W_2$  are intermediate and final outcomes,  $A_1, A_2$  are treatments. Variables  $W_0, W_1, W_2$  may be confounded by a hidden common cause  $U$ . It is well known that in this model,  $\mathbb{E}[Y(\mathbf{f}_\mathbf{A})]$  is identified via (4). We now discuss a number of approaches for computing the optimal set  $\mathbf{f}_\mathbf{A}^*$  given this identifying formula.

### Approaches Based on Backwards Induction

In sequential decision problems, choosing optimal functions  $\mathbf{f}_\mathbf{A}$  may appear to be a difficult search problem over a large set of function combinations. However, it is possible to use ideas from dynamic programming to transform the problem of choosing optimal  $\mathbf{f}_\mathbf{A}$  into a sequential problem where only a single optimal function is chosen at a time.

Multiple modeling approaches are possible here. A simple approach is to model conditional densities of all outcomes  $W_i$  given their past. Assuming all such models were correctly specified, given any particular history  $H_K$  up to the last decision point  $A_K$ , the optimal  $f_{A_K}^*$  is equal to  $\mathbb{I}(\mathbb{E}[W_K(A_K = 1) \mid H_K] > \mathbb{E}[W_K(A_K = 0) \mid H_K])$ , which under our model is equal to  $\mathbb{I}(\mathbb{E}[W_K \mid A_K = 1, H_K] > \mathbb{E}[W_K \mid A_K = 0, H_K])$ . Assuming optimal  $f_{A_{i+1}}^*, \dots, f_{A_K}^*$ , denoted by  $f_{\Delta_{i+1}}^*$ , were already chosen, the optimal  $f_{A_i}^*$  is defined inductively as

$$\mathbb{I} \left[ \mathbb{E}[W_K(A_i = 1, f_{\Delta_{i+1}}^*) \mid H_i] > \mathbb{E}[W_K(A_i = 0, f_{\Delta_{i+1}}^*) \mid H_i] \right].$$

Note that under our model, the counterfactual expectations above are identified via a modification of (4) where the outer summation is only with respect to variables  $W_{i+1}, \dots, W_{K-1}$ . For many kinds of statistical models for densities appearing as terms in (4), evaluating this sum may be challenging. An alternative strategy that avoids repeated summations is modeling the above expectations directly via *Q-functions*  $Q_i$ , which are conditional expectations over *value functions*  $V_{i+1}$ , given the



history. These are defined recursively as follows:

$$\begin{aligned} Q_K(H_K, A_K; \gamma_K) &= \mathbb{E}[W_K | A_K, H_K], \\ V_K(H_K) &= \max_{a_K} Q_K(H_K, a_K; \gamma_K), \end{aligned}$$

and for  $i = K - 1, \dots, 1$ , as

$$\begin{aligned} Q_i(H_i, A_i; \gamma_i) &= \mathbb{E}[V_{i+1}(H_{i+1}) | A_i, H_i], \\ V_i(H_i) &= \max_{a_i} Q_i(H_i, a_i; \gamma_i). \end{aligned}$$

The optimal policy at each stage may be easily derived from Q-functions as  $f_{A_i}^*(H_i) = \arg \max_{a_i} Q_i(H_i, a_i; \gamma_i)$ . Q-functions are recursively defined regression models where outcomes are value functions, and features are histories up to the current decision point. Thus, parameters  $\gamma_i, i = 1, \dots, K$ , of all Q-functions may be learned recursively by maximum likelihood methods applied to regression at stage  $i$ , given that the value function at stage  $i + 1$  was already computed for every row [2, 13].

### Value Search

Consider a restricted class of policies  $\mathcal{F}$  with elements  $\mathbf{f}_A \equiv \{f_{A_i}(H_i); A_i \in \mathbf{A}\}$ . It is often of interest to estimate the optimal policy within the class  $\mathcal{F}$ , even if the class does not contain the true optimal policy. For example,  $\mathcal{F}$  may be the set of clinically interpretable policies. For a sufficiently simple class  $\mathcal{F}$ , we can directly search for the optimal  $\mathbf{f}_A^{*\mathcal{F}} \equiv \arg \max_{\mathbf{f}_A \in \mathcal{F}} \mathbb{E}[Y(\mathbf{f}_A)]$ . This is called policy search or value search.

The expected response to an arbitrary treatment policy  $\beta = \mathbb{E}[Y(\mathbf{f}_A)]$ , for  $\mathbf{f}_A \in \mathcal{F}$  can be estimated in a number of ways. For  $\mathbf{A} = \{A\}$ , a simple estimator for  $\beta$  that uses only the treatment assignment model  $\pi(H; \psi)$  for  $p(A = 1|H)$  is the inverse probability weighting (IPW) estimator:

$$\mathbb{E} \left[ Y C_{f_A} / \pi_{f_A}(H; \hat{\psi}) \right], \quad (9)$$

where  $C_{f_A} \equiv \mathbb{I}(A = f_A(H))$ ,  $\pi_{f_A}(H; \psi) \equiv \pi(H; \psi) f_A(H) + (1 - \pi(H; \psi))(1 - f_A(H))$ , the expectation is evaluated empirically, and  $\hat{\psi}$  is fit by maximum likelihood. This estimator will not in general yield the optimal policy within  $\mathcal{F}$  if  $\pi$  is misspecified. An alternative estimator for  $\beta$  that provides some protection against this is the following:

$$\mathbb{E} \left[ \frac{C_{f_A} Y}{\pi_{f_A}(H; \hat{\psi})} - \frac{C_{f_A} - \pi_{f_A}(H; \hat{\psi})}{\pi_{f_A}(H; \hat{\psi})} \mathbb{E}[Y|H, A = f_A(H); \hat{\gamma}] \right]. \quad (10)$$

The above estimator is *doubly-robust* meaning that it is a consistent estimator if either the propensity score model  $\pi(H; \psi)$  or the regression model  $\mathbb{E}[Y|H, A; \gamma]$  is correctly specified.

Value search methods can in some cases be rephrased as a weighted classification problem in machine learning [20]. In the interest of space, we do not discuss methods based on this observation further here.

### G-Estimation

An alternative method for learning policies is to directly model the counterfactual contrast functions known as *optimal blip-to-zero functions*, or the counterfactual deviations in outcome from a reference treatment value (which we take to be  $A = 0$ ), conditional on history, assuming all future decisions are already optimal. Specifically, for each decision point  $i$ , we posit a *structural nested mean model (SNMM)*  $\gamma_i(H_i, A_i; \psi)$  for the contrast

$$\mathbb{E} \left[ Y(\bar{a}_i, f_{A_{i+1}}^*) | H_i \right] - \mathbb{E} \left[ Y(\bar{a}_{i-1}, a_i = 0, f_{A_{i+1}}^*) | H_i \right].$$

Note that if the true  $\gamma_i(H_i, A_i; \psi)$  were known, the optimal treatment policies are those that maximize the blip function at each stage:  $f_{A_i}^* = \arg \max_{a_i} \gamma_i(H_i, A_i; \psi_i)$ . In order to estimate  $\psi$  using data, let

$$\begin{aligned} U(\psi, \zeta(\psi), \alpha) &= \sum_{i=1}^K \{G_i(\psi) - \mathbb{E}[G_i(\psi) | H_i; \zeta]\} \\ &\quad \times \{d_i(H_i, A_i) - \mathbb{E}[d_i(H_i, A_i) | H_i; \alpha]\}, \quad (11) \end{aligned}$$

where  $d_i(H_i, A_i)$  is any function of  $H_i$  and  $A_i$ , and

$$G_i(\psi) = Y - \gamma_j(H_i, A_i; \psi) + \sum_{k=i+1}^K [\gamma_k(H_k, a_k^*; \psi) - \gamma_k(H_k, a_k; \psi)].$$

Consistent and asymptotically normal (CAN) estimators of  $\psi$  can be obtained using the estimating equations  $\mathbb{E}[U(\psi, \zeta(\psi), \alpha)] = 0$ , as shown in [11]. The estimate obtained in (11) is doubly-robust, meaning that the estimator  $\hat{\psi}$  is consistent if either  $\mathbb{E}[G_i(\psi) | H_i; \zeta]$  or  $p_i(A_i = 1 | H_i; \alpha)$  is correctly specified.

Methods closely related to G-estimation based on *counterfactual regret* were developed in [5]. We do not discuss them here in the interest of space.

## 4 LEARNING OPTIMAL PATH-SPECIFIC POLICIES

We now consider how the methods for optimizing overall policies translate to optimizing path-specific policies. Some of the generalizations we consider are currently only known for single-stage decision problems.

Consider the generalization of Fig. 1 (b) to the longitudinal setting with mediators, shown (for two time points) in Fig. 2 (a). This causal model corresponds to the setting described in detail in [4], representing an observational longitudinal study of HIV patients. Here,  $W_0$  represents the baseline variables of a patient,  $A_1, A_2$  represent treatment assignments, which were chosen based on observed treatment history according to physician's best judgement,  $W_1, W_2$  are intermediate and final outcomes (such as CD4 count or viral failure), and  $M_1, M_2$  are measures of patient adherence to their treatment regimen. We are interested in finding policies  $f_{A_1}(H_1), f_{A_2}(H_2)$  that optimize the effect of  $A_1, A_2$  on  $W_2$  that is either direct or via intermediate outcomes, but not via adherence, and where adherence is kept to that of a reference

treatment  $a'_1, a'_2$ . Specifically, we are interested in choosing  $f_{A_1}, f_{A_2}$  to optimize the counterfactual expectation  $\mathbb{E}[W_2(f_{A_1}, f_{A_2})]$ , which expands via (7) to

$$\mathbb{E} \left[ W_2 \left( \begin{array}{c} W_0, f_{A_1}(H_1), M_1(a'_1), \\ W_1(f_{A_1}(H_1), M_1(a'_1)), f_{A_2}(H_2), \\ M_2(a'_1, a'_2, W_1(f_{A_1}(H_1), M_1(a'_1)), M_1(a'_1)) \end{array} \right) \right). \quad (12)$$

The  $K$  stage version of the causal model in Fig. 2 (a) is given by a complete DAG on variables  $W_0, A_1, M_1, W_1, \dots, A_K, M_K, W_K$ , listed in topological order, with a hidden common cause  $U$  of  $W_0, \dots, W_K$ . Let  $\alpha$  be all directed edges out of  $A_1, \dots, A_K$ . The general version of (12) is the expectation of  $W_K$  taken with respect to the distribution  $p(W_K(f_\alpha))$ , where  $f_\alpha$  sets all edges  $(A_i M_j) \rightarrow$  to  $a'_i$ , and all other edges in  $\alpha$  to a policy  $f_{A_i}(H_i)$ .

Identifiability of  $p(W_K(f_\alpha))$  is given by the following corollary of results in [14], which can be viewed as a generalization of the *collapse of the g-formula* [10] to longitudinal mediation settings.

**Theorem 1** *In the above model, with a positive observed data distribution  $p(W_K(f_\alpha))$  is identified as*

$$\sum_{H_K, M_K} \prod_{i=1}^K \left\{ p(W_i | \bar{M}_{i-1}, \bar{W}_{i-1}, f_{A_i}(H_i)) p(M_i | \bar{a}'_i, \bar{W}_{i-1}, \bar{M}_{i-1}) \right\} p(W_0) \quad (13)$$

As a consequence (12) is identified as

$$\sum_{\bar{W}_1, \bar{M}_2} \mathbb{E} \left[ W_2 | f_{A_1}(H_1), f_{A_2}(H_2), \bar{W}_1, \bar{M}_2 \right] p(M_1 | a'_1, W_0) \times p(W_1 | W_0, f_{A_1}(H_1), M_1) p(M_2 | \bar{W}_1, M_1, a'_1, a'_2) p(W_0).$$

We now discuss a number of strategies for finding optimal path-specific policies identified by (13).

### Parametric Backwards Induction

A simple approach for finding optimal path-specific policies is to combine backwards induction with a maximum likelihood estimator for the conditional densities in the identifying functional (13).

Consider Fig. 2 (a), and let  $a_i$  and  $a'_i$  denote the active and reference levels of  $a_i$ , respectively. Beginning at the last stage  $K = 2$ , and assuming treatment is binary, the optimal decision,  $f_{A_2}^*$ , for a given patient with history  $(w_0, a_1, m_1, w_1)$  sets  $A_2$  to either  $a_2$  or  $a'_2$  to maximize the response  $W_2$ , while keeping  $M_2$  at whatever value it would have attained under a sequence of reference interventions  $(a'_1, a'_2)$ :

$$f_{A_2}^* = \mathbb{I} \left( \mathbb{E}[W_2(a_2, M_2(\bar{a}'_2)) | H_2] > \mathbb{E}[W_2(a'_2, M_2(\bar{a}'_2)) | H_2] \right).$$

The optimal decision at the first stage,  $f_{A_1}^*$ , is given by

$$\mathbb{I} \left\{ \mathbb{E} \left[ W_2 \left( \begin{array}{c} a_1, f_{A_2}^*(H_2), M_1(a'_1), W_1(a_1, M_1(a'_1)), \\ M_2(\bar{a}'_2, M_1(a'_1), W_1(a_1, M_1(a'_1))) \end{array} \right) \middle| H_1 \right] > \mathbb{E} \left[ W_2 \left( \begin{array}{c} a'_1, f_{A_2}^*(H_2), M_1(a'_1), W_1(a'_1, M_1(a'_1)), \\ M_2(\bar{a}'_2, M_1(a'_1), W_1(a'_1, M_1(a'_1))) \end{array} \right) \middle| H_1 \right] \right\}.$$

Under the causal model we described, the above counterfactuals can be estimated using a modification of (13) with no summations over, but instead conditioning on histories  $H_1$  or  $H_2$ . This approach easily generalizes to any number of decision stages. The difficulty here, as before, is the increasing amount of marginalizations that must be performed as the number of stages grows.

### Path-Specific Policies Via Q-Learning

We now describe how to generalize Q-learning to path-specific policies, using the HIV example in Fig. 2 (a) to ground the discussion. Recall that in this example, we wish to set  $A_1$  and  $A_2$  with respect to edges into  $W_1, W_2$  to maximize the outcome, while setting  $A_1$  and  $A_2$  to reference values  $a'_1, a'_2$  for the purposes of edges into  $M_1, M_2$ .

Simply defining Q-functions as expectations over value functions conditional on history does not work in our setting, since mediators behave in a counterfactually different way from either observed variables, or variables we wish to set according to a policy. Moreover, the sequential nature of the problem means the nested counterfactuals needed become quite involved to write down.

An alternative is to define Q-functions not in the observed data distribution, corresponding to Fig. 2 (a), but in a counterfactual distribution where  $A_1, A_2$  behave as observed, *except* for the purposes of edges into  $M_1, M_2$ , in which case they are counterfactually set to  $a'_1, a'_2$ . The graph corresponding to this counterfactual world is shown in Fig. 2 (b), and can be viewed as a generalization of a single world intervention graph [9], where treatment variables are only intervened on for the purposes of certain outgoing edges. Note that descendant variables of  $M_1, M_2$  are marked with a tilde to make clear that these variables are counterfactual and no longer equal to their observed counterparts.

The distribution corresponding to this situation is simply  $p(\mathbf{V}(\alpha_{(A_1 M_1) \rightarrow, (A_1 M_2) \rightarrow, (A_2 M_2) \rightarrow}))$ , where  $\alpha$  sets these edges to  $a'_1, a'_2$ . For  $K$  stages, the distribution is  $p(\mathbf{V}(\alpha_\alpha))$ , where  $\alpha$  are all edges of the form  $(A_i M_j) \rightarrow$ , and  $\alpha$  sets each such edge to the reference value  $a'_i$ . We have the following claim.

**Theorem 2** *In  $K$  stage version of the model in Fig. 2(a),  $p(\mathbf{V}(\alpha_\alpha))$  is identified as*

$$\bar{p}(\tilde{W}_0, \tilde{A}_1, \tilde{M}_1, \tilde{W}_1, \dots, \tilde{W}_K, \tilde{A}_K, \tilde{M}_K) = p(W_0) \prod_{i=1}^K p(W_i | M_i, A_i, H_i) p(A_i | H_i) p(M_i | \bar{a}'_i, H_i \setminus A) \quad (14)$$

We can now define Q-functions as value function expectations on this new distribution, and proceed with backwards induction as before. The only difference between

the previous formulation is how Q-functions parameters are fit. In particular, we must compensate for the fact that  $\tilde{p}$  above is not the observed data distribution. Define

$$\begin{aligned}\tilde{Q}_K(\tilde{H}_K, \tilde{A}_K; \gamma_K) &= \mathbb{E}[\tilde{W}_K | \tilde{A}_K, \tilde{H}_K], \\ \tilde{V}_K(\tilde{H}_K) &= \max_{a_K} \tilde{Q}_K(\tilde{H}_K, a_K; \gamma_K),\end{aligned}\quad (15)$$

and for  $i = K - 1, \dots, 1$ , define

$$\begin{aligned}\tilde{Q}_i(H_i, A_i; \gamma_i) &= \mathbb{E}[\tilde{V}_{i+1}(\tilde{H}_{i+1}) | \tilde{A}_i, \tilde{H}_i], \\ \tilde{V}_i(\tilde{H}_i) &= \max_{a_i} \tilde{Q}_i(\tilde{H}_i, a_i; \gamma_i).\end{aligned}\quad (16)$$

In our example in Fig. 2 (a),  $K = 2$ ,  $H_1 \equiv \{W_0\}$ ,  $\tilde{H}_2 \equiv \{W_0, M_1, W_1\}$ , and  $\mathbb{E}$  denotes expectations with respect to appropriate conditional distributions derived from  $\tilde{p}$ . Q-functions defined in this way can be used to obtain the optimal path-specific policy at each stage.

**Theorem 3** *Given that each  $\tilde{Q}_i, i = 1, \dots, K$  is specified correctly, the optimal treatment at stage  $i$  given  $H_i$  is equal to:  $f_{A_i}^*(H_i) = \arg \max_{a_i} \tilde{Q}_i(H_i, a_i; \gamma_i)$ . Since parameters  $\gamma$  are not generally known, they must be estimated from data. This can be done as follows.*

**Theorem 4** *Assume models in the set  $\{\tilde{Q}_i(\tilde{H}_i, \tilde{A}_i; \gamma_i), p(M_i | A_i, H_i; \phi) | \forall i\}$  are correctly specified. Then the estimation equations*

$$\begin{aligned}\mathbb{E} \left[ \frac{\partial \tilde{Q}_K}{\partial \gamma_K} \{W_K - \tilde{Q}_K(A_K, H_K; \gamma_K)\} w_K(H_K; \hat{\phi}_K) \right] &= 0, \text{ and} \\ \mathbb{E} \left[ \frac{\partial \tilde{Q}_i}{\partial \gamma_i} \{V_{i+1}(H_{i+1}) - \tilde{Q}_i(H_i, A_i; \gamma_i)\} w_i(H_i; \hat{\phi}_i) \right] &= 0,\end{aligned}$$

are consistent for  $\gamma_K$  and  $\gamma_i$ , where

$$w_i(H_i; \hat{\phi}_i) \equiv \frac{p(M_i | \tilde{A}_i = a', H_i; \hat{\phi}_i)}{p(M_i | \tilde{A}_i, H_i; \hat{\phi}_i)} \forall i = 1, \dots, K.$$

## Path-Specific Value Search

For simplicity, we restrict attention to a single-stage decision problem, as shown in Fig. 1 (a), where we are interested in picking a policy that maximizes the counterfactual mean  $\beta = \mathbb{E}[Y(A = f(W), M(a'))]$ .

Consider a restricted class of path-specific policies  $\mathcal{F}$  with elements  $f \equiv \{f(W), \tilde{f}\}$  the latter setting  $A$  to a constant value  $a'$ , regardless of  $W$ . Give any estimation strategy for the counterfactual mean under a path-specific policy, we can implement a direct search for the optimal policies within  $\mathcal{F}$  as before. By analogy with earlier discussion of value search, we give an IPW estimator for  $\beta$  which generalizes (9):

$$\mathbb{E} \left[ \frac{Y \tilde{C}}{\tilde{\pi}(W; \hat{\psi})} \cdot \frac{p(M|A = a', W; \hat{\phi})}{p(M|A = f(W), W; \hat{\phi})} \right], \quad (17)$$

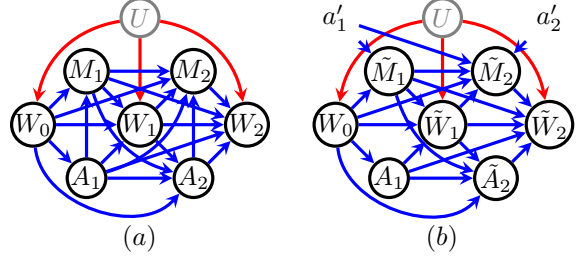


Figure 2: (a) A causal model that generalized Fig. 1 (b) by also considering mediators. (b) A “multiple world intervention graph,” representing the counterfactual situation where adherence levels are kept to a reference treatment level, but the chemical effect of drugs is operating normally.

and an estimator which generalizes (10):

$$\begin{aligned}\mathbb{E} \left[ \frac{\tilde{C}}{\tilde{\pi}(W; \hat{\psi})} \frac{f(M|W, A = a'; \hat{\phi})}{f(M|W, f(W); \hat{\phi})} \{Y - \mathbb{E}[Y|f(W), M, W; \hat{\zeta}]\} \right] &+ \\ \frac{\mathbb{I}(A = a')}{\pi_{a'}(W; \hat{\psi})} \left\{ \mathbb{E}[Y|f(W), M, W; \hat{\zeta}] - \sum_M \mathbb{E}[Y|f(W), M, W; \hat{\zeta}] \right. & \\ \left. p(M|W, A = a'; \hat{\zeta}) \right\} + \sum_M \mathbb{E}[Y|f(W), M, W; \hat{\zeta}] p(M|W, A = a'; \hat{\phi}) &, \quad (18)\end{aligned}$$

where the expectation is evaluated empirically,  $\tilde{C} \equiv \mathbb{I}(A = f(W))$ ,  $\pi_{a'}(W; \psi) = p(A = a' | W; \psi)$ ,  $\tilde{\pi}(W; \hat{\psi}) \equiv \sum_a \pi_a(W; \psi) \mathbb{I}(a = f(W))$ , and  $\hat{\psi}, \hat{\phi}, \hat{\zeta}$  are fit by maximum likelihood. We have the following.

**Theorem 5** *Under regularity assumptions referenced in the Appendix, the estimator in (17) is consistent and asymptotically normal (CAN) if the models in the set  $\{\pi(W; \psi), p(M|W, A; \phi)\}$  are correctly specified, and the estimator in (18) is CAN in the union model, where any two models in the set  $\{\pi(W; \psi), \mathbb{E}[Y|A, M, W; \zeta], p(M|W, A; \phi)\}$  are correctly specified.*

This claim, and the estimators above, are extensions of the results in [17].

## Single-Stage G-Estimation For Path-Specific Policies

Results in [18] generalized optimal blip-to-zero functions to mediation settings, by positing the following SNMM  $\gamma(A, W; \psi)$ :

$$\mathbb{E}[Y(A, M(A = 0)) | W] - \mathbb{E}[Y(A = 0, M(A = 0)) | W]. \quad (19)$$

Note that the policy  $f_A^*$  maximizing  $\mathbb{E}[Y(A = f_A^*(W), M(0))]$  can be directly obtained from  $\gamma$  via  $f_A^*(W) = \arg \max_a \gamma(A, W; \psi)$ . Since  $\psi$  is not generally known, it must be estimated from data. The following is a consistent set of estimating equations for  $\psi$ ,

under assumptions described in [18]:

$$\mathbb{E} \left[ \left( \frac{\mathbb{I}(A=1)p(M|A=0, W)}{p(A=1|W)p(M|A=1, W)} \{Y - \mathbb{E}[Y|A=1, M, W]\} - \right. \right. \\ \left. \left. \frac{\mathbb{I}(A=0)}{p(A=0|W)} \{Y - \mathbb{E}[Y|A=1, M, W] + \gamma(1, W; \psi)\} \right) h(W) \right] = 0 \quad (20)$$

for  $h(W)$  any  $|\psi|$ -dimensional function of  $W$ .

Note that unlike SNMMs associated with overall policies, which can be defined for any number of treatments, SNMMs associated with path-specific policies have only been defined for a single treatment. A longitudinal generalization of these models is left to future work.

## 5 EXPERIMENTS

We now illustrate our methods via a dataset on HIV patients from Nigeria. The data consists of more than 50k treatment-naive HIV-1 infected patients who were enrolled in the Harvard PEPFAR/AIDS Prevention Initiative program prior to Oct 2010. The patients were put on courses of antiretroviral therapy (ART), with five standard first-line regimen, and were followed every six months for at least one year after the ART initiation. Patients stayed on their initial treatment plan unless they were experiencing toxicity within a first-line drug regimen and consequently moved to a second-line regimen. The data has records on demographics and clinical test results such as CD4 counts, viral loads, and toxicity measures at 6-month intervals.

In order to combat chronic diseases such as HIV, patients are required to follow long term use of medications. Full benefits of the medications are realized when patients take their medications as prescribed. Unfortunately, factors such as side effects, caused by drug toxicities, can be developed alongside the course of treatment and lead to poor adherence to the therapy. A primary measure of adherence is provided in the data as average percent adherence that is the total number of days that the patient has drug supply over the total number of days in the time period (6-month intervals).

We restricted our attention to the first year of follow up, including two decision points and picked (log) CD4 count at the end of the first year as the outcome of interest. CD4 count is one of the biomarkers that quantifies how well the immune system is functioning and higher values are more favorable. Drug toxicity and adherence can mediate the effect of treatment on outcome. Since reactions to drug toxicity and adherence behavior vary among patients, we consider the problem of finding policies that optimize the direct chemical effect of the drug where the indirect effect mediated through drug toxicity and adherence are set to some reference levels. We run

an additional experiment where only the effect through adherence is set to a reference level [4]. The latter results are provided in the Appendix for the interest of space.

The causal model can be represented by the DAG in Fig. 2 (a). Treatment decisions at the baseline and the first follow up are respectively denoted by  $A_1$  and  $A_2$ .  $W_1$  is a dichotomized intermediate outcome that indicates whether the CD4 count is above 450 cells/mm<sup>3</sup> at the the end of the same six month period.  $W_2$  is the final outcome and denotes the log CD4 count at the end of the first year after ART initiation.  $W_0$  includes all the baseline factors such as sex, age, and test results prior to any treatment initiation. Toxicity and adherence measures during the first six months and the first year after treatment initiation are collected in  $M_1$  and  $M_2$ , respectively. Drug toxicity indicates any lab toxicities (alanine transaminase  $\geq 120$  UI/L, creatinine  $\geq 260$  mmol/L, hemoglobin  $\leq 8$  g/dL), and adherence indicates whether average percent adherence was no less than 95%.

We first illustrate the results on finding optimal policies in a two-stage decision problem using G-formula and Q-learning methods, and then provide the results for finding optimal policies for the single (first) stage decision problem using value search and G-estimation. In the single stage analysis, we consider the log CD4 count at the end of the sixth month as the outcome of interest. All modeling assumptions are described in the Appendix.

### Methods For The Two-Stage Decision Problem

Optimal overall polices and path policies are estimated as described in Sections 3 and 4. Expected outcomes under optimal policies we learned, along with 95% confidence intervals obtained by bootstrap, are shown in Table. 1. In the interest of space, we do not consider corrections necessary to preserve the validity of these intervals in cases of model discontinuities, but these are straightforward [1]. Both optimal polices have higher expected outcomes than the observed data, using either method. Path-specific policies led to higher expected outcomes compared to overall policies. This could be explained by the phenomenon of countervailing adherence-mediated effect, described in [4]. In other words, in some cases the more chemically effective drugs are also harder to take. In order to visualize optimal policies we learned as clinically interpretable flowcharts, we viewed policy-recommended decisions as class labels, and history as features, and used a multilabel decision tree classifier, as implemented in the `rpart` R package. The results are shown in Fig. 1 and 2 in the Appendix. Since the classifiers are not perfect predictors of the policies, they are to be viewed as interpretable approximations of the true learned policy. Variables involved in the decisions, such

Table 1: Comparison of population log CD4 counts under different policies (under treatment assignments in the observed data, the value is  $5.64 \pm 0.01$  in the 2-stage and  $5.54 \pm 0.01$  in the 1-stage problem). G-formula and Q-learning are used with 2-stage decision points. Value search and G-estimation are used with 1-stage decision point.

	Path Policies	Overall Policies
<b>G-formula</b>	6.89 (5.76, 7.10)	6.79 (5.68, 6.90)
<b>Q-learning</b>	7.34 (6.10, 7.55)	6.89 (5.82, 7.12)
<b>Value search</b>	5.58 (5.54, 5.62)	5.56 (5.55, 5.58)
<b>G-estimation</b>	5.62 (5.50, 5.67)	5.79 (5.59, 6.04)

as age, gender, CD4, virological status, and so on, are clinically reasonable.

### Methods For The Single-Stage Decision Problem

We focus on patients that appear at baseline, and treat the five first-line treatments as a binary decision by grouping the first three (which we denote as group 1) and the last two treatments (which we denote as group 2). We are interested in finding a treatment group assignment at the first decision point that leads to a higher CD4 count at the first follow up, assuming adherence and toxicity are set to a reference point for every patient. The model for this setting is Fig. 1(a), where  $W$  is observed history up to the first decision.

We considered policies of the form  $\mathbb{I}\{CD4_{m00} < \alpha\}$ , where  $CD4_{m00}$  denotes the CD4 count right before the first decision point, to decide what treatment group the patient should be assigned to. The normal range for this variable is 500 to 1500 cells/mm<sup>3</sup>. We searched for an optimal policy in this restricted class by varying  $\alpha$  from 100 cells/mm<sup>3</sup> to 1000 cells/mm<sup>3</sup> by 50 cells/mm<sup>3</sup> increments, and estimated the value for each  $\alpha$  using (18). Under the modeling assumptions described in the Appendix, the optimal threshold between group 1 and group 2 treatments was chosen to be  $\mathbb{I}\{CD4_{m00} < 550 \text{ cells/mm}^3\}$ . In other words, if the CD4 count is less than 550, it is better for the patients to receive any of the treatments in the first treatment group. However, if we pick the optimal policy based on the overall effect of treatment on outcome, then value search leads to policies of the form  $\mathbb{I}\{CD4_{m00} < 250 \text{ cells/mm}^3\}$ . The optimal overall policy decides to switch to group 2 treatments when HIV gets more severe, while the optimal path policy decides on switching when CD4 count is still within a healthy range but at lower values. This implies that if treatment adherence could be kept to that of a reference treatment, initiation of treatments within group 2 could be delayed to lower CD4 values. As before, both policies lead to a higher than observed log CD4 count, with the path policy yielding slightly higher outcomes than the overall policy.

Finally, we learned optimal path policies via G-estimation. We estimated the parameters  $\psi$  for the SNMM using (20). The population log CD4 count under path policies and overall policies learned by G-estimation are given in Table. (1), and a decision tree view of the policies is shown in Fig. 3 in the Appendix. Under our assumptions, the optimal path policy found by G-estimation did not do much better than the outcomes under observed treatments and the overall policies.

## 6 CONCLUSIONS

In this paper, we generalized ideas in mediation analysis and dynamic treatment regimes to consider the problem of estimation of responses to policies associated with particular causal pathways, and methods for learning policies that optimize these responses. Since validating findings in causal inference is difficult, and conclusions are sensitive to specific modeling assumptions made, we developed multiple approaches for learning optimal policies that rely on orthogonal sets of modeling strategies. In particular, we considered strategies based on backwards induction with either plug-in estimation or Q-learning, value search for restricted classes of policies, and G-estimation of structural nested mean models (SNMMs) generalized to mediation.

We illustrated these methods by finding policies for HIV-positive patients that optimize the direct chemical effect of antiretroviral therapy, where the indirect effect mediated through drug toxicity and adherence are set to a reference level. The results in the experiment section suggest that policies that aim to optimize the direct effect of the treatment have better outcome responses than policies that optimize the overall effect of the treatment, and optimal policies decide on clinically relevant variables, such as age, gender, viral load, and CD4 count.

The estimation methods we described are applicable to sequential decision problems, except for G-estimation which is currently only limited to single-stage decision problems. Generalizing the version of G-estimation to longitudinal mediation problems, and deriving semi-parametric estimators for the version of Q-learning we described are areas for future work.

### Acknowledgments

This research was supported in part by the NIH grants R01 AI104459-01A1 and R01 AI127271-01A1. We thank the anonymous reviewers for their insightful comments that greatly improved this manuscript. An earlier version of this manuscript contained simulations by Sourjya Sarkar.

## References

- [1] B. Chakraborty, E. B. Laber, and Y. Zhao. Inference for optimal dynamic treatment regimes using an adaptive m-out-of-n bootstrap scheme. *Biometrics*, 69(3), 2013.
- [2] B. Chakraborty and E. Moodie. *Statistical Methods for Dynamic Treatment Regimes: Reinforcement Learning, Causal Inference, and Personalized Medicine*. New York: Springer-Verlag, 2013.
- [3] J. W. Lavori and R. Dawson. A design for testing clinical strategies: Biased adaptive within-subject randomization. *Journal of the Royal Statistical Society*, 163:29–38, 2000.
- [4] C. Miles, I. Shpitser, P. Kanki, S. Meloni, and E. T. Tchetgen. Quantifying an adherence path-specific effect of antiretroviral therapy in the nigeria pepfar program. *Journal of the American Statistical Society*, 2017.
- [5] S. Murphy. Optimal dynamic treatment regimes. *Journal of the Royal Statistical Society: Series B*, 65:331–366, 2003.
- [6] S. A. Murphy. An experimental design for the development of adaptive treatment strategies. *Statistics in Medicine*, 24:1455–1481, 2005.
- [7] J. Pearl. Direct and indirect effects. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 411–420. Morgan Kaufmann, San Francisco, 2001.
- [8] J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2 edition, 2009.
- [9] T. S. Richardson and J. M. Robins. Single world intervention graphs (SWIGs): A unification of the counterfactual and graphical approaches to causality. *Working Paper 128, Center for Statistics and the Social Sciences, Univ. Washington, Seattle, WA.*, 2013.
- [10] J. M. Robins. A new approach to causal inference in mortality studies with sustained exposure periods – application to control of the healthy worker survivor effect. *Mathematical Modeling*, 7:1393–1512, 1986.
- [11] J. M. Robins. Optimal structural nested models for optimal sequential decisions. In *Proceedings of the second Seattle symposium on biostatistics*, pages 189–326, 2004.
- [12] J. M. Robins and S. Greenland. Identifiability and exchangeability of direct and indirect effects. *Epidemiology*, 3:143–155, 1992.
- [13] P. Schulte, A. Tsiatis, E. Laber, and M. Davidian. Q- and A-learning methods for estimating optimal dynamic treatment regimes. *Statistical Science*, 29:640–661, 2014.
- [14] I. Shpitser. Counterfactual graphical models for longitudinal mediation analysis with unobserved confounding. *Cognitive Science (Rumelhart special issue)*, 37:1011–1035, 2013.
- [15] I. Shpitser and E. Sherman. Identification of personalized effects associated with causal pathways. In *proceedings of the 34th Conference on Uncertainty in Artificial Intelligence*, 2018.
- [16] I. Shpitser and E. T. Tchetgen. Causal inference with a graphical hierarchy of interventions. *Ann. Stat.*, 2015.
- [17] E. J. T. Tchetgen and I. Shpitser. Semiparametric theory for causal mediation analysis: efficiency bounds, multiple robustness, and sensitivity analysis. *Ann. Stat.*, 40(3):1816–1845, 2012.
- [18] E. J. T. Tchetgen and I. Shpitser. Estimation of a semiparametric natural direct effect model incorporating baseline covariates. *Biometrika*, 101(4):849–864, 2014.
- [19] J. Tian and J. Pearl. On the identification of causal effects. Technical Report R-290-L, Department of CS, UCLA, 2002.
- [20] Y. Zhao, D. Zeng, A. J. Rush, and M. R. Kosorok. Estimating individualized treatment rules using outcome weighted learning. *Journal of the American Statistical Association*, 107:1106–1118, 2012.

---

# High-confidence error estimates for learned value functions

---

**Touqir Sajed**

Department of Computing Science  
University of Alberta, Canada  
touqir@ualberta.ca

**Welsey Chung**

Department of Computing Science  
University of Alberta, Canada  
wchung@ualberta.ca

**Martha White**

Department of Computing Science  
University of Alberta, Canada  
whitem@ualberta.ca

## Abstract

Estimating the value function for a fixed policy is a fundamental problem in reinforcement learning. Policy evaluation algorithms—to estimate value functions—continue to be developed, to improve convergence rates, improve stability and handle variability, particularly for off-policy learning. To understand the properties of these algorithms, the experimenter needs high-confidence estimates of the accuracy of the learned value functions. For environments with small, finite state-spaces, like chains, the true value function can be easily computed, to compute accuracy. For large, or continuous state-spaces, however, this is no longer feasible. In this paper, we address the largely open problem of how to obtain these high-confidence estimates, for general state-spaces. We provide a high-confidence bound on an empirical estimate of the value error to the true value error. We use this bound to design an offline sampling algorithm, which stores the required quantities to repeatedly compute value error estimates for any learned value function. We provide experiments investigating the number of samples required by this offline algorithm in simple benchmark reinforcement learning domains, and highlight that there are still many open questions to be solved for this important problem.

## 1 INTRODUCTION

Policy evaluation is a key step in many reinforcement learning systems. Policy evaluation approximates the value of each state—future sum of rewards—given a policy and either a model of the world or a stream of data

produced by an agents choices. In classical policy iteration schemes, the agent continually alternates between improving the policy using the current approximation of the value function, and updating the approximate value function for the new policy. Policy search methods like actor-critic estimate the value function of the current policy to perform gradient updates for the policy.

However, there has been relatively little research into methods for accurately evaluating policy evaluation algorithms when the true values are not available. In most domains where we are interested in performing policy evaluation, it is difficult or impossible to compute the true value function. We may not have access to the transition probabilities or the reward function in every state, making it impossible to obtain the closed form solution of the true value function  $v^*$ . Even if we have access to a full model of the environment, we may not be able to represent the value function if the number of states is too large or the state is continuous. Aside from small finite MDPs like gridworlds and random MDPs, where closed-form solutions can be computed [Geist and Scherrer, 2014, White and White, 2016], we often do not have access to  $v^*$ . In nearly all our well-known benchmark domains, such as Mountain Car, Puddle World, Cart pole, and Acrobot, we must turn to some other method to evaluate learning progress.

One option that has been considered is to estimate the objective minimized by the algorithms. Several papers [Sutton et al., 2008, Du et al., 2017] have compared the performance of the algorithms in terms of their target objective on a batch of samples, using the approximate linear system for the mean-squared projected Bellman error (MSPBE). One estimator, called RUPEE [White, 2015], is designed to incrementally approximate the MSPBE by keeping a running average across data produced during learning. Some terms, such as the feature covariance matrix, can be estimated easily; however, one component of the MSPBE includes the current weights, and is biased by this moving average approach. More problematically,

some algorithms do not converge to the minimum of the MSPBE, such as residual gradient for the mean-squared Bellman error [Baird, 1995] or Emphatic Temporal Difference (ETD) learning [Sutton et al., 2016], which minimize a variant of the MSPBE with a different weighting. This approach, therefore, is limited to comparing algorithms that minimize the same objective.

The more standard approach has been to use rollouts from states to obtain samples of returns. To obtain these rollout estimates, three parameters need to be chosen: the number of states  $m$  from which to rollout, the number of rollouts or trajectories  $t$ , and the length of each rollout. Given these rollouts, the true values can be estimated from each of the  $m$  chosen states, stored offline, and then used for comparison repeatedly during experiments. These evaluation schemes, however, have intuitively chosen parameters, without any guarantees that the distance to the true values, the error, is well-estimated. Early work comparing gradient TD algorithms [Maei et al., 2009] used sampled trajectories—2500 of them—but compared to returns, rather than value estimates. For several empirical studies using benchmark domains, like Mountain Car and Acrobot, there are a variety of choices, including  $t = m = 500$  [Gehring et al., 2016];  $m = 2000$ ,  $t = 300$  and 1000 length rollouts [Pan et al., 2017]; and  $m = 5000$ ,  $t = 5000$  [Le et al., 2017]. For a continuous physical system, [Dann et al., 2014] used as little as 10 rollouts from a state. Otherwise, other papers have mentioned that extensive rollouts are used<sup>1</sup>, but did not describe how [Konidaris et al., 2011, Dabney and Thomas, 2014]. In general, as new policy evaluation algorithms are derived, it is essential to find a solution to this open problem: How can we confidently compare value estimates returned by our algorithms?

In this work, we provide an algorithm that ensures, with high-probability, that the estimated distance has small error in approximating the true distance between the true value function  $v^*$  for an arbitrary estimate  $\hat{v}$ . We focus in the main body of the paper on the clipped mean-absolute percentage value error (CMAPVE) as a representative example of the general strategy. We provide additional results for a variety of other losses in the appendix, to facilitate use for a broader range of error choices. We conclude by demonstrating the rollout parameters chosen for several case studies, highlighting that previous intuitive choices did not effectively direct sampling. We hope for this algorithm to become a standard approach for generating estimates of the true values to facilitate comparison of policy evaluation algorithms by reinforcement learning researchers.

<sup>1</sup>Note that [Boyan and Moore, 1995] used rollouts for a complementary purpose, to train a nonlinear value function, rather than for evaluating policy evaluation algorithms.

## 2 MEASURES OF LEARNING PERFORMANCE

This paper investigates the problem of comparing algorithms that estimate the discounted sum of future rewards *incrementally* for a fixed policy. In this section, we first introduce the policy evaluation problem and motivate a particular measure of learning performance for policy evaluation algorithms. In the following section, we discuss how to estimate this measure.

We model the agent’s interaction with the world as a Markov decision process (MDP), defined by a (potentially uncountable) set of states  $\mathcal{S}$ , a finite set of actions  $\mathcal{A}$ , transitions  $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, \infty)$ , rewards  $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$  and a scalar discount function  $\gamma : \mathcal{S} \rightarrow \mathbb{R}$ . On each time step  $t$ , the agent selects an action according to its *behaviour policy*  $A_t \sim \mu(\cdot|S_t)$ , the environment transitions into a new state  $S_{t+1} \sim P(\cdot|S_t, A_t)$  and the agent receives a scalar reward  $R_{t+1} \stackrel{\text{def}}{=} R(S_t, A_t, S_{t+1})$ . In policy evaluation, the agent’s objective is to estimate the expected return

$$v^*(s) = \mathbb{E}[G_t | S_t = s, A_t \sim \pi] \quad (1)$$

$$\text{for return } G_t = \sum_{i=0}^{\infty} \gamma^i R_{t+i+1}$$

where  $v^*(s)$  is called the *state-value function* for the *target policy*  $\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ . From a stream of data, the agent incrementally approximates this value function,  $\hat{v}$ . For experiments, to report learning curves, we need to measure the accuracy of this estimate every step or at least periodically, such as every 10 steps.

For policy evaluation, when the policy remains fixed, the value error remains the gold standard of evaluation. Ignoring how useful the value function is for policy improvement, our only goal is accuracy with respect to  $v^*$ . Assume some weighting  $d : \mathcal{S} \rightarrow [0, \infty)$ , a probability distribution over states. Given access to  $v^*$ , it is common to estimate the mean squared value error

$$\text{MSVE}(\hat{v}) \stackrel{\text{def}}{=} \mathbb{E}[(\hat{v}(S) - v^*(S))^2] \quad (2)$$

$$= \int_{\mathcal{S}} d(s) (\hat{v}(s) - v^*(s))^2 ds$$

or the mean absolute value error

$$\text{MAVE}(\hat{v}) \stackrel{\text{def}}{=} \mathbb{E}[|\hat{v}(s) - v^*(s)|]. \quad (3)$$

The integral is replaced with a sum if the set of states is finite. Because we consider how to estimate this error for continuous state domains—for which it is more difficult to directly estimate  $v^*$ —we preferentially assume the states are continuous.



These losses, however, have several issues, beyond estimating them. The key issue is that the scale of the returns can be quite different across states. This skews the loss and, as we will see, makes it more difficult to get high-accuracy estimates. Consider a cost-to-goal problem, where the agent receives a reward of  $-1$  per step. From one state the value could be  $-1000$  and for another it could be  $-1$ . For a prediction of  $-990$  and  $-11$  respectively, the absolute value error for both states would be  $-10$ . However, the prediction of  $-990$  for the first state is quite accurate, whereas a prediction of  $-11$  for the second state is highly inaccurate.

One alternative is to estimate a percentage error, or relative error. The mean absolute percentage value error is

$$\text{MAPVE}(\hat{v}) \stackrel{\text{def}}{=} \mathbb{E} \left[ \frac{|\hat{v}(S) - v^*(S)|}{|v^*(S)| + \tau} \right] \quad (4)$$

for some  $\tau \geq 0$ . The term  $\tau$  in the denominator ensures the MAPVE does not become excessively high, if true values of states are zero or near zero. For example, for  $\tau = 1$ , the MAPVE is essentially the MAVE for small  $v^*(s)$ , which reflects that small absolute differences are meaningful for these smaller numbers. For large  $v^*(s)$ , the  $\tau$  has little effect, and the MAPVE becomes a true percentage error, reflecting the fact that we are particularly interested in relative errors for larger  $v^*(s)$ .

Additionally, the MAPVE can be quite large if  $\hat{v}$  is highly inaccurate. When estimating these performance measures, however, it is uninteresting to focus on obtaining high-accuracy estimate of very large MAPVE. Rather, it is sufficient to report that  $\hat{v}$  is highly inaccurate, and focus the estimation of the loss on more accurate  $\hat{v}$ . Towards this goal, we introduce the clipped MAPVE

$$\text{CMAPVE}(\hat{v}) \stackrel{\text{def}}{=} \mathbb{E} \left[ \min \left( c, \frac{|\hat{v}(S) - v^*(S)|}{|v^*(S)| + \tau} \right) \right]$$

for some  $c > 0$ . This  $c$  provides a maximum percentage error. For example, setting  $c = 2$  caps error estimates for approximate values that are worse than 200% inaccurate. Such a level of inaccuracy is already high, and when comparing policy evaluation algorithms, we are much more interested in their percentage error—particularly compared to each other—once we are within a reasonable range around the true values. Note that  $c$  can be chosen to be the maximum value of the loss, and so the following results remain quite general.

Though many losses could be considered, we put forward the CMAPVE as a proposed standard for policy evaluation. The parameters  $\tau$  and  $c$  can both be appropriately chosen by the experimentalist, for a given MDP. These parameters give sufficient flexibility in highlighting differences between algorithms, while still enabling

high-confidence estimates of these errors, which we discuss next. For this reason, we use CMAPVE as the loss in the main body of the text. However, for completeness, we also show how to modify the analysis and algorithms for other losses in the appendix.

### 3 HIGH-CONFIDENCE ESTIMATES OF VALUE ERROR

Our goal now is to approximate the value error, CMAPVE, with high-confidence, for any value function  $\hat{v}$ . Instead of approximating the error directly for each  $\hat{v}$ , the typical approach is to estimate  $v^*$  as accurately as possible, for a large set of states  $s_1, \dots, s_m \sim d$ . Given these high-accuracy estimates  $\bar{v}$ , the true expected error can be approximated from this subset of states for any  $\hat{v}$ .

$$\text{CMAPVE}(\hat{v}) \approx \frac{1}{m} \sum_{i=1}^m \min \left( c, \frac{|\hat{v}(s_i) - \bar{v}(s_i)|}{|\bar{v}(s_i)| + \tau} \right)$$

Since the CMAPVE needs to be computed frequently, for many steps during learning potentially across many algorithms, it is important for this estimate of CMAPVE to be efficiently computable. An important requirement, then, is for the number of states  $m$  to be as small as possible, so that all the  $\bar{v}(s_i)$  can be stored and the summed difference is quick to compute.

One possible approach is to estimate the true value function  $v^*$  using a powerful function approximator, offline. A large batch of data could be gathered, and a learning method used to train  $\bar{v}$ . This large function approximator would not even need to be stored: only  $\bar{v}(s_i)$  would need to be saved once this offline procedure was complete. This approach, however, will be biased by the form of the function approximator, which can favor certain policy evaluation algorithms during evaluation. Further, it is difficult to quantify this bias, particularly in a general way agnostic to the type of function approximator an experimentalist might use for their learning setting.

An alternative strategy is to use many sampled rollouts from this subset of states. This strategy is general—requiring only access to samples from the MDP. A much larger number of interactions can be used with the MDP, to compute  $\bar{v}$ , because this is computed once, offline, to facilitate many further empirical comparisons between algorithms after. For example, one may want to examine the early learning performance of two different policy evaluation algorithms—which may themselves receive only a small number of samples. The cached  $\bar{v}$  then enables computing this early learning performance. However, even offline, there are limits to how many samples can be computed feasibly, particularly for computationally costly simulators [Dann et al., 2014].

Therefore, our goal is the following: how can we *efficiently* compute *high-confidence* estimates of CMAPVE, using a *minimal number of offline rollouts*. The choice of a clipped loss actually enables the number of states  $m$  to remain small (shown in Lemma 2), enabling efficient computation of CMAPVE. In the next section, we address the second point: how to obtain high-confidence estimates, given access to  $\bar{v}$  that approximates  $v^*$ . In the following section, we discuss how to obtain these  $\bar{v}$ .

### 3.1 OVERVIEW

We first provide an overview of the approach, to make it easier to follow the argument. We additionally include a notation table (Table 1), particularly to help discern the various value functions.

Table 1: Table of Notation

$v^*$	true values for policy $\pi$
$\bar{v}^*$	true values for policy $\pi$ , when using truncated rollouts to length $l$
$\bar{v}$	estimated values for policy $\pi$ using $t$ rollouts, when using truncated rollouts to length $l$
$\hat{v}$	estimated values for policy $\pi$ , being evaluated
$d$	distribution over the states $\mathcal{S}$ , $d : \mathcal{S} \rightarrow [0, \infty)$
$m$	number of states $\{s_1, \dots, s_m\}$ , $s_i \sim d$
$\ell_c$	clipped error, $\ell_c(v_1, v_2) = \min\left(c, \frac{ v_1 - v_2 }{ v_2  + \tau}\right)$
$\ell$	true error, $\ell(\hat{v}, v^*) = \mathbb{E}[\ell_c(\hat{v}(S), v^*(S))]$ under $d$
$\hat{\ell}$	approximate error, $\hat{\ell}(\hat{v}, v^*) = \frac{1}{m} \sum_{i=1}^m \ell_c(\hat{v}(s_i), v^*(s_i))$
$R_{\max}$	an upper bound on the maximum absolute value reward, $R_{\max} \geq \sup  R(s, a, s') $
$V_{\max}$	maximum absolute value for the policy $\pi$ for any state, e.g., $V_{\max} = \frac{\max \text{reward} - \min \text{reward}}{1 - \gamma}$
$K$	the number of times the error estimate is queried

First, we consider several value function approximations, for use within the bound, summarized in Table 1. The goal is to determine the accuracy of the estimates of the learned  $\hat{v}$  with respect to the true values  $v^*$ . We estimate true values  $v^*(s_i)$  for  $s_i$  using repeated rollouts from  $s_i$ ; this results in two forms of error. The first is due to truncated rollouts, which for the continuing case would otherwise be infinitely long. The second source of error is due to using an empirical estimate of the true values, by averaging sampled returns. We denote  $\bar{v}^*$  as the true values, for truncated returns, and  $\bar{v}$  as the sample estimate of  $\bar{v}^*$  from  $m$  truncated rollouts.

Second, we consider the approximation in computing the loss: the difference between  $\hat{v}$  and  $v^*$ . We consider the true loss  $\ell(\hat{v}, v^*)$  and the approximate loss  $\hat{\ell}(\hat{v}, v^*)$ , in Table 1. The argument in Theorem 1 revolves around

upper bounding the difference between these two losses, in terms of three terms. These terms are bounded in Lemmas 2, 3 and 4. Lemma 2 bounds the error due to sampling only a subset of  $m$ . Lemma 3 bounds the error from approximating  $v^*$  with truncated rollouts. Lemma 4 bounds the error from dividing by  $|\bar{v}(s_i)| + \tau$  instead of  $|v^*(s_i)| + \tau$ .

Finally, to obtain this general bound, we first assume that we can obtain highly-accurate estimates  $\bar{v}$  of  $\bar{v}^*$ . We state these two assumptions in Assumptions 1 and 2. These estimates could be obtained with a variety of sampling strategies, and so separate it from the main proof. We later develop one algorithm to obtain such estimates, in Section 4.

### 3.2 MAIN RESULT

We will compute rollout values from a set of sampled states  $\{s_i\}_{i=1}^m$ . Each rollout consists of a trajectory simulated, or rolled out, some number of steps. The length of this trajectory can itself be random, depending on if an episode terminates or if the trajectory is estimated to be a sufficiently accurate sample of the full, non-truncated return. We first assume that we have access to such trajectories and rollout estimates and in later sections show how to obtain such trajectories and estimates.

**Assumption 1.** For any  $\epsilon > 0$  and sampled state  $s_i$ , the trajectory lengths  $l_i$  are specified such that,

$$|\bar{v}^*(s_i) - v^*(s_i)| \leq \epsilon (|v^*(s_i)| + \tau) \quad (5)$$

**Assumption 2.** Starting from  $s_i$ , assume you have trajectories of rewards  $\{r_{ijk}\}$  for trajectory index  $j \in \{1, \dots, t\}$  and rollout index  $k \in \{0, \dots, l_{ij} - 1\}$  for a trajectory length  $l_{ij}$  that depends on the trajectory. The approximated rollout values

$$\bar{v}(s_i) \stackrel{\text{def}}{=} \frac{1}{t} \sum_{j=1}^t \sum_{k=0}^{l_{ij}-1} \gamma^k r_{ijk} \quad (6)$$

are an  $(\epsilon, \delta, \tau)$ -approximation to the true expected values, where  $l_{ij}$  is an instance of the random variable  $l_i$

$$\bar{v}^*(s) \stackrel{\text{def}}{=} \mathbb{E} \left[ \sum_{k=0}^{l_i-1} \gamma^k R_k \right] \quad (7)$$

i.e, for  $0 < \epsilon$ , with probability at least  $1 - \delta/2$ , the following holds for all states

$$|\bar{v}^*(s_i) - \bar{v}(s_i)| \leq \epsilon (|\bar{v}^*(s_i)| + \tau) \quad (8)$$

**Theorem 1.** Let  $\{s_1, \dots, s_m\}$  be states sampled according to  $d$ . Assume  $\bar{v}(s_i)$  satisfies Assumption 1 and 2 for

all  $i \in \{1, \dots, m\}$ . Suppose the empirical loss mean estimates are computed  $K$  number of times with different learned value functions  $\hat{v}$  each time. Then the approximation error

$$\hat{\ell}(\hat{v}, \bar{v}) \stackrel{\text{def}}{=} \frac{1}{m} \sum_{i=1}^m \ell_c(\hat{v}(s_i), \bar{v}(s_i)) \quad (9)$$

for all the  $\hat{v}$  satisfies, with probability at least  $1 - \delta$ ,

$$\left| \ell(\hat{v}, v^*) - \hat{\ell}(\hat{v}, \bar{v}) \right| \leq (11) + (12) + (13) \quad (10)$$

**Proof:** We need to bound the errors introduced from having a reduced number of states, a finite set of trajectories to approximate the expected returns for each of those states and truncated rollouts to get estimates of returns. To do so, we first consider the difference under the approximate clipped loss, to the true value function.

$$\left| \ell(\hat{v}, v^*) - \hat{\ell}(\hat{v}, \bar{v}) \right| \leq \left| \ell(\hat{v}, v^*) - \hat{\ell}(\hat{v}, v^*) \right| + \left| \hat{\ell}(\hat{v}, v^*) - \hat{\ell}(\hat{v}, \bar{v}) \right|$$

The first component is bounded in Lemma 2. For the second component, notice that

$$\left| \hat{\ell}(\hat{v}, v^*) - \hat{\ell}(\hat{v}, \bar{v}) \right| \leq \frac{1}{m} \sum_{i=1}^m \left| \ell_c(\hat{v}(s_i), v^*(s_i)) - \ell_c(\hat{v}(s_i), \bar{v}(s_i)) \right|$$

However, these two differences are difficult to compare, because they have different denominators: the first has  $|v^*(s_i)| + \tau$ , whereas the second has  $|\bar{v}(s_i)| + \tau$ . We therefore further separate each component in the sum

$$\begin{aligned} & \left| \ell_c(\hat{v}(s_i), v^*(s_i)) - \ell_c(\hat{v}(s_i), \bar{v}(s_i)) \right| \\ & \leq \left| \ell_c(\hat{v}(s_i), v^*(s_i)) - \min \left( c, \frac{|\hat{v}(s_i) - \bar{v}(s_i)|}{|v^*(s_i)| + \tau} \right) \right| \\ & \quad + \left| \min \left( c, \frac{|\hat{v}(s_i) - \bar{v}(s_i)|}{|v^*(s_i)| + \tau} \right) - \ell_c(\hat{v}(s_i), \bar{v}(s_i)) \right| \end{aligned}$$

The first difference has the same denominator, so

$$\begin{aligned} & \left| \ell_c(\hat{v}(s_i), v^*(s_i)) - \min \left( c, \frac{|\hat{v}(s_i) - \bar{v}(s_i)|}{|v^*(s_i)| + \tau} \right) \right| \\ & = \left| \min \left( c, \frac{|\hat{v}(s_i) - v^*(s_i)|}{|v^*(s_i)| + \tau} \right) - \min \left( c, \frac{|\hat{v}(s_i) - \bar{v}(s_i)|}{|v^*(s_i)| + \tau} \right) \right| \\ & = \frac{1}{|v^*(s_i)| + \tau} \left| \min(c(|v^*(s_i)| + \tau), |\hat{v}(s_i) - v^*(s_i)|) \right. \\ & \quad \left. - \min(c(|v^*(s_i)| + \tau), |\hat{v}(s_i) - \bar{v}(s_i)|) \right| \\ & \leq \frac{1}{|v^*(s_i)| + \tau} \min(c(|v^*(s_i)| + \tau), |v^*(s_i) - \bar{v}(s_i)|) \\ & = \ell_c(\bar{v}(s_i), v^*(s_i)) \end{aligned}$$

The last step follows from the triangle inequality  $\left| |x| - |y| \right| \leq |x - y|$  on the clipped loss (see Lemma 7, in Appendix A, for an explicit proof that the triangle inequality holds for the clipped loss).

Therefore, putting it all together, we have

$$\begin{aligned} & \left| \ell(\hat{v}, v^*) - \hat{\ell}(\hat{v}, \bar{v}) \right| \\ & \leq \left| \ell(\hat{v}, v^*) - \hat{\ell}(\hat{v}, v^*) \right| \\ & \quad + \frac{1}{m} \sum_{i=1}^m \ell_c(\bar{v}(s_i), v^*(s_i)) \\ & \quad + \frac{1}{m} \sum_{i=1}^m \left| \min \left( c, \frac{|\hat{v}(s_i) - \bar{v}(s_i)|}{|v^*(s_i)| + \tau} \right) - \ell_c(\hat{v}(s_i), \bar{v}(s_i)) \right| \end{aligned}$$

where the first, second and third components are bounded in Lemmas 2, 3 and 4 respectively. Finally, due to the application of Hoeffding's bound (Lemma 2) with error probability of atmost  $\delta/2$  and assumption 2 which may not hold with probability atmost  $\delta/2$  and the union bound, we conclude that the final bound holds with probability at least  $1 - \delta$ . ■

**Lemma 2** (Dependence on  $m$ ). *Suppose the empirical loss mean estimates are computed  $K$  number of times. Then with probability at least  $1 - \delta/2$ :*

$$\left| \ell(\hat{v}, v^*) - \hat{\ell}(\hat{v}, v^*) \right| \leq \sqrt{\frac{\log(4K/\delta)c^2}{2m}} \quad (11)$$

**Proof:** Since  $\hat{\ell}(\hat{v}, v^*) = \frac{1}{m} \sum_{i=1}^m \ell_c(\hat{v}(s_i), v^*(s_i))$  is an unbiased estimate of  $\ell(\hat{v}, v^*)$ , we can use Hoeffding's bound for variables bounded between  $[0, c]$ . For any of the  $K$  times, the concentration probability is as follows:

$$\begin{aligned} \Pr \left( \left| \ell(\hat{v}, v^*) - \hat{\ell}(\hat{v}, v^*) \right| \geq t \right) & \leq 2 \exp \left( \frac{-2t^2 m^2}{\sum_{i=1}^m c^2} \right) \\ & = 2 \exp \left( \frac{-2mt^2}{c^2} \right) = \frac{\delta}{2K} \end{aligned}$$

Thus, due to union bound over all the  $K$  times, for all those empirical loss mean estimates, the following holds

$$\Pr \left( \left| \ell(\hat{v}, v^*) - \hat{\ell}(\hat{v}, v^*) \right| \leq t \right) \geq 1 - \delta/2.$$

Rearranging the above, to express  $t$  in terms of  $\delta$ ,

$$2 \exp \left( \frac{-2mt^2}{c^2} \right) = \frac{\delta}{2K} \implies t = \sqrt{\frac{\log(4K/\delta)c^2}{2m}}.$$

Therefore, with probability at least  $1 - \delta$ ,

$$\left| \ell(\hat{v}, v^*) - \hat{\ell}(\hat{v}, v^*) \right| \leq \sqrt{\frac{\log(4K/\delta)c^2}{2m}}. \quad \blacksquare$$

**Lemma 3** (Dependence on Truncated Rollout Errors). *Under Assumption 2 and 1,*

$$\frac{1}{m} \sum_{i=1}^m \ell_c(\bar{v}(s_i), v^*(s_i)) \leq 2\epsilon \quad (12)$$

**Proof:** We can split up this error into sampling error for a finite length rollout and for a finite number of trajectories. We can consider the unclipped error, which is an upper bound on the clipped error.

$$\frac{|v^*(s_i) - \bar{v}(s_i)|}{|v^*(s_i)| + \tau} \leq \frac{|v^*(s_i) - \bar{v}^*(s_i)|}{|v^*(s_i)| + \tau} + \frac{|\bar{v}^*(s_i) - \bar{v}(s_i)|}{|v^*(s_i)| + \tau}$$

These two terms are both bounded by  $\epsilon$ , by assumption. ■

**Lemma 4.** *Under Assumption 2,*

$$\left| \min \left( c, \frac{|\hat{v}(s_i) - \bar{v}(s_i)|}{|v^*(s_i)| + \tau} \right) - \min \left( c, \frac{|\bar{v}(s_i) - \bar{v}(s_i)|}{|\bar{v}(s_i)| + \tau} \right) \right| \leq c(1 - (1 + \epsilon)^{-2}) \quad (13)$$

**Proof:** We need to bound the difference due to the difference in normalizer. To do so, we simply need to find a constant  $\beta > 0$  such that  $\min \left( c, \frac{|\hat{v}(s_i) - \bar{v}(s_i)|}{|v^*(s_i)| + \tau} \right) \geq \beta$

The key is to lower bound  $|v^*(s_i)| + \tau$ , which results in an upper bound on the first term and consequently an upper bound on the difference between the two terms.

$$\begin{aligned} \frac{|\bar{v}(s_i)| + \tau}{|v^*(s_i)| + \tau} &\leq \frac{|\bar{v}(s_i) - \bar{v}^*(s_i)| + |\bar{v}^*(s_i)| + \tau}{|v^*(s_i)| + \tau} \\ &= \frac{|\bar{v}(s_i) - \bar{v}^*(s_i)|}{|v^*(s_i)| + \tau} + \frac{|\bar{v}^*(s_i)| + \tau}{|v^*(s_i)| + \tau} \\ &\leq \frac{\epsilon(|\bar{v}^*(s_i)| + \tau)}{|v^*(s_i)| + \tau} + \frac{|\bar{v}^*(s_i)| + \tau}{|v^*(s_i)| + \tau} \\ &= \frac{(1 + \epsilon)(|\bar{v}^*(s_i)| + \tau)}{|v^*(s_i)| + \tau} \end{aligned}$$

where the second inequality is due to Assumption 2. Now further

$$\begin{aligned} \frac{(|\bar{v}^*(s_i)| + \tau)}{|v^*(s_i)| + \tau} &\leq \frac{|\bar{v}^*(s_i) - v^*(s_i)|}{|v^*(s_i)| + \tau} + \frac{|v^*(s_i)| + \tau}{|v^*(s_i)| + \tau} \\ &\leq \frac{\epsilon(|v^*(s_i)| + \tau)}{|v^*(s_i)| + \tau} + 1 \\ &= \epsilon + 1 \end{aligned}$$

giving

$$\frac{|\bar{v}(s_i)| + \tau}{|v^*(s_i)| + \tau} \leq (1 + \epsilon)^2 \implies |v^*(s_i)| + \tau \geq \frac{|\bar{v}(s_i)| + \tau}{(1 + \epsilon)^2}.$$

So, for  $a = (1 + \epsilon)^2$  and  $b = \frac{|\hat{v}(s_i) - \bar{v}(s_i)|}{|\bar{v}(s_i)| + \tau}$ , the term  $|\min(c, ab) - \min(c, b)|$  upper bounds the difference. Because  $a > 1$  and  $b > 0$ , this term  $|\min(c, ab) - \min(c, b)|$  is maximized when  $ab = c$ , and  $b = c/a$ . In the worst case, therefore,  $|\min(c, ab) - \min(c, b)| \leq c(1 - a^{-1})$  which finishes the proof. ■

### 3.3 SATISFYING THE ASSUMPTIONS

The bounds above relied heavily on accurate sample estimates of  $v^*(s_i)$ . To obtain Assumption 1, we need to rollout trajectories sufficiently far to ensure that truncated sampled returns do not incur too much bias. For problems with discounting, for  $\gamma < 1$ , the returns can be truncated once  $\gamma^l$  becomes sufficiently small, as the remaining terms in the sum for the return have negligible weight. For episodic problems with no discounting, it is likely that trajectories need to be simulated until termination, since rewards beyond the truncation horizon would not be discounted and so could have considerable weight.

We show how to satisfy Assumption 1, for the discounted setting. Note that for the trivial setting of  $R_{\max} = 0$ , it is sufficient to use  $l = 1$ , so we assume  $R_{\max} > 0$ .

**Lemma 5.** *For  $\gamma < 1$  and  $R_{\max} > 0$ , if*

$$l = \left\lceil \frac{\log(\epsilon\tau(1 - \gamma)) - \log(R_{\max})}{\log \gamma} \right\rceil \quad (14)$$

*then  $\bar{v}^*(s)$  satisfies Assumption 1:*

$$|\bar{v}^*(s) - v^*(s)| \leq \epsilon(|v^*(s)| + \tau) \quad (15)$$

**Proof:** The first component can be bounded as

$$\begin{aligned} |\bar{v}^*(s) - v^*(s)| &\leq \left| E \left[ \sum_{k=0}^{\infty} \gamma^k R_k \right] - E \left[ \sum_{k=0}^{l-1} \gamma^k R_k \right] \right| \\ &\leq R_{\max} \left( \frac{1}{1 - \gamma} - \frac{1 - \gamma^l}{1 - \gamma} \right) \\ &= R_{\max} \frac{\gamma^l}{1 - \gamma} \end{aligned}$$

giving

$$\frac{|\bar{v}^*(s) - v^*(s)|}{|v^*(s)| + \tau} \leq R_{\max} \frac{\gamma^l}{\tau(1 - \gamma)}.$$

Setting  $l$  as in (14) ensures  $R_{\max} \frac{\gamma^l}{\tau(1 - \gamma)} \leq \epsilon$ , completing the proof. ■

For Assumption 2, we need a stopping rule for sampling truncated returns that ensures  $\bar{v}$  is within  $\epsilon$  of the true expected value of the truncated returns,  $\bar{v}^*$ . The idea is to continue sampling truncated returns, until the confidence interval around the mean estimate shrinks sufficiently to ensure, with high probability, that the values estimates are within  $\epsilon$  of the true values. Such stopping rules have been designed for non-negative random variables [Domingos and Hulten, 2001, Dagum et al., 2006], and extended to more general random variables [Mnih et al., 2008]. We defer the development of such an algorithm for this setting until the next section.

## 4 THE ROLLOUT ALGORITHM

We can now design a high-confidence algorithm for estimating the accuracy of a value function. Practically, the most important number to reduce is  $m$ , because these values will be stored and used for comparisons on each step. The choice of a clipped loss, however, makes it more manageable to control  $m$ . In this section, we focus more on how much the variability in trajectories, and trajectory length, impact the number of required samples.

The general algorithm framework is given in Algorithm 1. The algorithm is straightforward once given an algorithm to sample rollouts from a given state. The rollout algorithm is where development can be directed, to reduce the required number of samples. This rollout algorithm needs to be designed to satisfy Assumptions 1 and 2. We have already shown how to select trajectory lengths to satisfy Assumption 1. Below, we describe how to select  $m$  and how to satisfy Assumption 2.

---

**Algorithm 1** Offline computation of  $\bar{v}$ , to get high-confidence estimates of value error

---

- 1: ▷ Input  $\epsilon, \delta, \tau$
  - 2: ▷ Compute the values  $\bar{v}$  once offline and store for repeated use
  - 3: Set  $\epsilon_m = \epsilon/2$  and  $\bar{\epsilon} = \epsilon/(2(1+c))$
  - 4:  $m \leftarrow \frac{\log(4K/\delta)c^2}{2\epsilon_m^2}$
  - 5: **for**  $1, \dots, m$  **do**
  - 6:     Sample  $s_i \sim d$
  - 7:      $\bar{v}(s_i) \leftarrow$  Algorithm 2 with  $\bar{\epsilon}, \frac{\delta}{2m}, \tau$
- 

**Specifying the number of sampled states  $m$ .** For the number of required samples for the outer loop in Algorithm 1, we need enough samples to match the bound in Lemma 2.

$$\epsilon_m = \sqrt{\frac{\log(4K/\delta)c^2}{2m}} \implies m = \frac{\log(4K/\delta)c^2}{2\epsilon_m^2} \quad (16)$$

$m$  is chosen as  $\left\lceil \frac{\log(4K/\delta)c^2}{2\epsilon_m^2} \right\rceil \geq \epsilon_m$  and thus we are being slightly conservative regarding the error to ensure correctness with high probability. We opt for a separate choice of  $\epsilon_m$  for this part of the bound, because it is completely separate from the other errors. This number  $\epsilon_m$  could be chosen slightly larger, to reduce the number of required sampled states to compare to, whereas  $\epsilon$  might need to be smaller depending on the choice of  $c$  and  $\tau$ . Separating them explicitly can significantly reduce the  $m$  in the outer loop, both improving time and storage, as well as later comparison time, without impacting the accuracy of the algorithm.

---

**Algorithm 2** High-confidence Monte carlo estimate of the expected return for a state

---

- 1: ▷ Input  $\epsilon, \delta, \tau$ , state  $s$
  - 2: ▷ Output an  $\epsilon, \delta, \tau$ -accurate approx.  $\bar{v}(s_i)$  of  $v^*(s_i)$
  - 3:  $\text{LB} \leftarrow 0, \text{UB} \leftarrow \infty$
  - 4:  $\widehat{\text{LB}} \leftarrow -\infty, \widehat{\text{UB}} \leftarrow \infty$
  - 5:  $\bar{g} \leftarrow 0, M \leftarrow 0$
  - 6:  $j \leftarrow 1, h \leftarrow 0, \beta \leftarrow 1.1, p \leftarrow 1.1, \alpha \leftarrow 1, x \leftarrow 1$
  - 7: **while**  $(1+\epsilon)\text{LB} + 2\epsilon\tau < (1-\epsilon)\text{UB}$  or  $\text{LB} = 0$  **do**
  - 8:      $g \leftarrow$  sample return that satisfies Assumption 1 (e.g., see Algorithm 3 in Appendix D)
  - 9:      $\Delta \leftarrow g - \bar{g}$
  - 10:      $\bar{g} \leftarrow \bar{g} + \frac{\Delta}{j}$
  - 11:      $M \leftarrow M + \Delta(g - \bar{g})$
  - 12:      $\sigma \leftarrow \sqrt{M/j}$
  - 13:     ▷ Compute the confidence interval
  - 14:     **if**  $j \geq \lfloor \beta^h \rfloor$  **then**
  - 15:          $h \leftarrow h + 1$
  - 16:          $\alpha \leftarrow \lfloor \beta^h \rfloor / \lfloor \beta^{h-1} \rfloor$
  - 17:          $x \leftarrow -\alpha \log \frac{\delta(p-1)}{3ph^p}$
  - 18:          $c_j \leftarrow \sigma \sqrt{\frac{2x}{j} + \frac{3V_{\max}x}{j}}$
  - 19:          $\text{LB} \leftarrow \max(\text{LB}, |\bar{g}| - c_j)$
  - 20:          $\text{UB} \leftarrow \min(\text{UB}, |\bar{g}| + c_j)$
  - 21:          $\widehat{\text{LB}} \leftarrow \max(\widehat{\text{LB}}, \bar{g} - c_j)$
  - 22:          $\widehat{\text{UB}} \leftarrow \min(\widehat{\text{UB}}, \bar{g} + c_j)$
  - 23:         **if**  $\frac{\widehat{\text{UB}} - \widehat{\text{LB}}}{2} \leq \epsilon\tau$  **then return**  $\frac{\widehat{\text{UB}} + \widehat{\text{LB}}}{2}$
  - 24:          $j = j + 1$
  - return**  $\frac{\text{sign}(\bar{g})}{2}((1+\epsilon)\text{LB} + (1-\epsilon)\text{UB})$
- 

**Satisfying Assumption 2.** Our goal is to get an  $(\epsilon, \delta, \tau)$ -approximation of  $\bar{v}(s_i)$ , with a feasible number of samples. In many cases, it is difficult to make parametric assumptions about returns in reinforcement learning. A simple strategy is to use a stopping rule for generating returns, based on general concentration inequalities—like Hoeffding’s bound—that make few assumptions about the random variables. If we had a bit more information, however, such as the variance of the returns, we could obtain a tighter bound, using Bernstein’s inequality and so reduce the number of required samples. We cannot know this variance a priori, but fortunately an empirical Bernstein bound has been developed [Mnih et al., 2008]. Using this bound, Mnih et al. [2008] designed EBGStop, which incrementally estimates variance and significantly reduces the number of samples required to get high-confidence estimates.

EBGStop can be used, without modification, given a mechanism to sample truncated returns that satisfy Assumption 1. However, we generalize the algorithm to

allow for our less restrictive condition  $|\bar{v}^*(s_i) - \bar{v}(s_i)| \leq \epsilon(|\bar{v}^*(s_i)| + \tau)$ , as opposed to the original algorithm which ensured  $|\bar{v}^*(s_i) - \bar{v}(s_i)| \leq \epsilon|\bar{v}^*(s_i)|$ . When  $\tau = 0$  in our algorithm, it reduces to the original; since this is a generalization on that algorithm, we continue to call it EBGStop. This modification is important when  $v^*(s_i) = 0$ , since this would require  $\bar{v}^*(s_i) = v^*(s_i)$  when  $\tau = 0$ . For  $\tau > 0$ , once the accuracy is within  $\tau$ , the algorithm can stop. The Algorithm is summarized in Algorithm 2. The proof follows closely to the proof for EBGStop; we include it in Appendix A. Algorithm 2 uses geometric sampling, like EBGStop, to improve sample efficiency. The idea is to avoid checking the stopping condition after every sample. Instead, for some  $\beta > 1$ , the condition is checked after  $\beta^k$  samples; the next check occurs at  $\beta^{k+1}$ . This modification improves sample efficiency from a multiplicative factor of  $\log(\frac{R}{\epsilon|\mu|})$  to  $\log \log(\frac{R}{\epsilon|\mu|})$ , where  $R$  is the range of the random variables and  $\mu$  is the mean.

**Lemma 6.** *Algorithm 2 returns an  $\epsilon, \delta, \tau$ -approximation  $\bar{v}(s_i)$ :*

$$|\bar{v}^*(s_i) - \bar{v}(s_i)| \leq \epsilon(|\bar{v}^*(s_i)| + \tau)$$

**Corollary 1.** *For any  $0 < \epsilon_m$  and  $0 < \bar{\epsilon} < 1$ , Algorithm 1 returns an  $\epsilon, \delta$ -accurate approximation: with probability at least  $1 - \delta$ ,*

$$\left| \ell(\hat{v}, v^*) - \frac{1}{m} \sum_{i=1}^m \ell_c(\hat{v}(s_i), \bar{v}(s_i)) \right| \leq \epsilon$$

where

$$\epsilon = \epsilon_m + 2(1 + c)\bar{\epsilon} \quad (17)$$

Algorithm 1 uses this theorem, for a given desired level of accuracy  $\epsilon$ . To obtain this level of accuracy,  $\epsilon_m = \epsilon/2$  and  $\bar{\epsilon}$  given to Algorithm 2 is set to ensure  $2(1 + c)\bar{\epsilon} = \epsilon$ .

## 5 EXPERIMENTS ON BENCHMARK PROBLEMS

We investigate the required number of samples to get with a level of accuracy, for different probability levels. We report this for two continuous-state benchmark problems—Mountain Car and Puddle World—which have previously been used to compare policy evaluation algorithms. Our goal is to (a) demonstrate how this framework can be used to obtain high-confidence estimates of accuracy and (b) provide some insight into how many samples are needed, even for simple reinforcement learning benchmark domains.

We report the number of returns sampled by Algorithm 2, averaged across several states. The domains, Mountain Car and Puddle World, are as specified in the policy

evaluation experiments by Pan et al. [2017]. For Mountain Car, we use the energy pumping policy, with 60% random action selection for the three actions. For Puddle World, we used a uniform random policy for the four actions. They are both episodic tasks, with a maximum absolute value of  $V_{\max} = 100$  and  $V_{\max} = 8000$  respectively. The variance in Puddle World is particularly high, as it has regions with high-variance, high-magnitude rewards. We sampled  $m = 100$  states uniformly across the state-space, to provide some insight into the variability of the number of returns sampled across the state-space. We tested  $\epsilon \in \{0.01, 0.05, 0.1\}$  and  $\delta \in \{0.01, 0.1\}$ , and set  $\tau = 1.0$ . We focus here on how many returns need to be sampled, rather than the trajectory length, and so do not use  $c$  nor explicitly compute clipped errors  $\ell_c$ .

The results indicate that EBGStop requires a large number of samples, particularly in Puddle World. Figure 1b for Mountain Car and Figure 1a both indicate that decreasing  $\epsilon$  from 0.05 to 0.01, to enable higher-accuracy estimates of value function error, causes an exponential increase in the required number of samples, an increase of  $10^3$  to  $10^4$  for Mountain Car and  $10^5$  to  $10^6$  for Puddle World. An accuracy level of 0.01, which corresponds to difference of 1% for clipped errors, is a typical choice for policy evaluation experiments, yet requires an inordinate number of samples, particularly in Puddle World.

We further investigated lower bounds on the required number of samples. Though EBGStop is a state-of-the-art stopping algorithm, to ensure high-confidence bounds for any distribution with bounded mean and variance, it collects more samples than is actually required. To assess its efficiency gap, we also include an idealistic approach to computing the confidence intervals, using repeated subsamples computed from the simulator. By obtaining many, many estimates of the sample average, using  $t$  samples of the truncated return, we can estimate the actual variability of the sample average. We provide additional details in Appendix D. Such a method to compute the confidence interval is not a viable algorithm to reduce the number of samples generated. Rather, the goal here is to report a lower bound on the number of samples required, for comparison and to motivate the amount the sampling algorithm could be improved. The number of samples generated by EBGStop is typically between 10 to 100 times more than the optimal number of samples, which indicates that there is much room to improve sample efficiency.

## 6 CONCLUSION

In this work, we present the first principled approach to obtain high-confidence error estimates of learned value functions. Our strategy is focused on the setting tackled

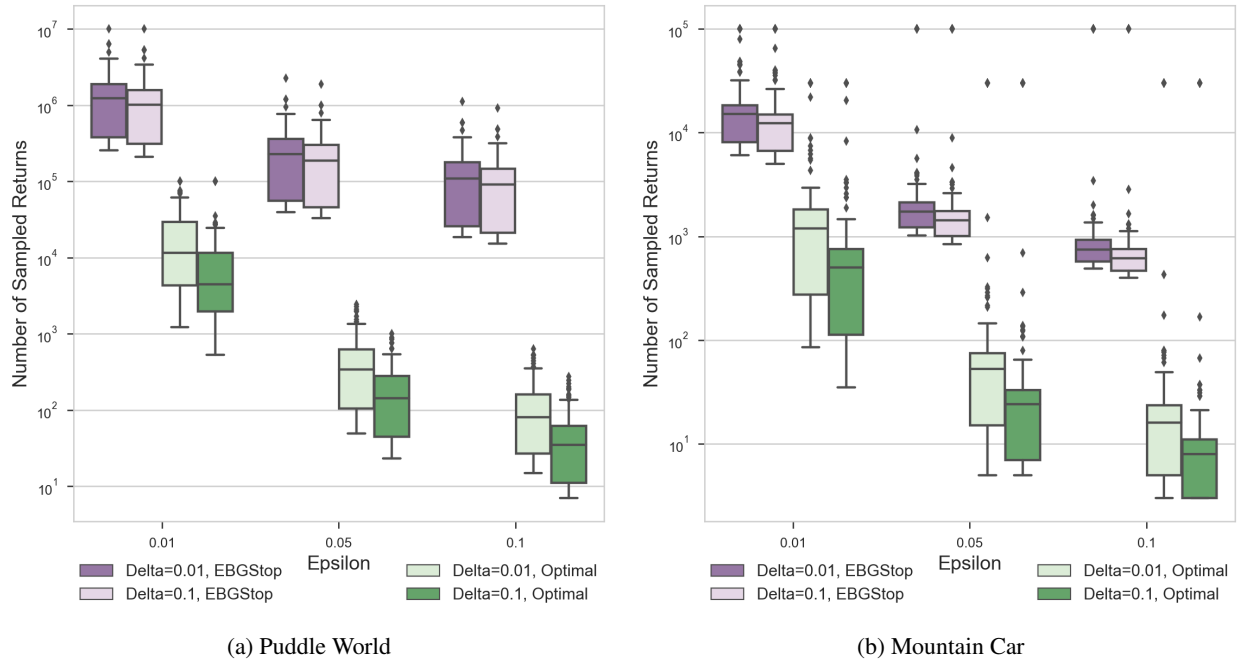


Figure 1: The number of sampled returns to obtain high-confidence estimates  $\bar{v}$  of the true values  $v^*$ . The y-axis is logarithmic, with many more samples used for smaller  $\epsilon$ . The box-plot similarly are logarithmic, and so are actually much larger for smaller  $\epsilon$  than larger  $\epsilon$ . The accuracy  $\epsilon$  has a much larger effect on the number of sampled returns that are required, than the probability  $\delta$ . An additional point of interest is that there are a few states that required significantly more samples for the returns, indicated by the outliers depicted as individual points.

by reinforcement learning empiricists, comparing value function-learning algorithms. In this context, accuracy of value estimates, for multiple algorithms, need to be computed repeatedly, every few steps with increasing data given to the learning algorithms. We provide a general framework for such a setting, where we store estimates of true value functions using samples of truncated returns. The framework for estimating true values for comparison is intentionally generic, to enable any (sample-efficient) stopping algorithm to be used. We propose one solution, which uses empirical Bernstein bounds, to significantly reduce the required number of samples over other concentration inequalities, such as Hoeffding’s bound.

This paper highlights several open challenges. As demonstrated in the experiments, there is a large gap between the actual required number of samples and that provided by the algorithm using an empirical Bernstein stopping-rule. For some simulators, this overestimate could result in a prohibitively large number of samples. Although this is a problem more generally faced by the sampling literature, it is particularly exacerbated in reinforcement learning where the variability across states and returns can be high, with large maximum values. An important avenue, then, is to develop more sample-efficient sampling algorithms to make high-confidence error es-

timates feasible for a broader range of settings in reinforcement learning.

Another open challenge is to address how to sample states  $\{s_1, \dots, s_m\}$ . This paper is agnostic to how these states are obtained. However, it is not always straightforward to sample these from a desired distribution. Some choices are simple, such as randomly selecting these across the state space. For other cases, it is more complicated, such as sampling these from the stationary distribution of the behaviour policy,  $d^\mu$ . The typical strategy is to run  $\mu$  for a burn-in period, so that afterwards it is more likely for states to be sampled from the stationary distribution. The theoretical effectiveness of this strategy, however, is not yet well-understood. There has been work estimating empirical mixing times [Hsu et al., 2015] and some work bounding the number of samples required for burn-in [Paulin, 2015]. Nonetheless, it remains an important open question on how to adapt these results for the general reinforcement learning setting.

One goal of this paper has been to highlight an open problem that has largely been ignored by reinforcement learning empiricists. We hope for this framework to stimulate further work in high-confidence estimates of value function accuracy.

## References

- Matthieu Geist and Bruno Scherrer. Off-policy learning with eligibility traces: a survey. *The Journal of Machine Learning Research*, 2014.
- Adam M White and Martha White. Investigating practical, linear temporal difference learning. In *International Conference on Autonomous Agents and Multiagent Systems*, 2016.
- Adam White. *Developing a predictive approach to knowledge*. PhD thesis, University of Alberta, 2015.
- Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. In *International Conference on Machine Learning*, 1995.
- Richard S Sutton, A R Mahmood, and Martha White. An emphatic approach to the problem of off-policy temporal-difference learning. *The Journal of Machine Learning Research*, 2016.
- R Sutton, C Szepesvári, A Geramifard, and Michael Bowling. Dyna-style planning with linear function approximation and prioritized sweeping. In *Conference on Uncertainty in Artificial Intelligence*, 2008.
- Simon S Du, Jiانشu Chen, Lihong Li, Lin Xiao, and Dengyong Zhou. Stochastic Variance Reduction Methods for Policy Evaluation. In *International Conference on Machine Learning*, 2017.
- HR Maei, C Szepesvári, S Bhatnagar, D Precup, D Silver, and Richard S Sutton. Convergent temporal-difference learning with arbitrary smooth function approximation. In *Advances in Neural Information Processing Systems*, 2009.
- Clement Gehring, Yangchen Pan, and Martha White. Incremental Truncated LSTD. In *International Joint Conference on Artificial Intelligence*, 2016.
- Yangchen Pan, Adam White, and Martha White. Accelerated Gradient Temporal Difference Learning. In *International Conference on Machine Learning*, 2017.
- Lei Le, Raksha Kumaraswamy, and Martha White. Learning Sparse Representations in Reinforcement Learning with Sparse Coding. In *International Joint Conference on Artificial Intelligence*, 2017.
- Christoph Dann, Gerhard Neumann, and Jan Peters. Policy evaluation with temporal differences: a survey and comparison. *The Journal of Machine Learning Research*, 2014.
- J Boyan and A W Moore. Generalization in Reinforcement Learning: Safely Approximating the Value Function. *Advances in Neural Information Processing Systems*, 1995.
- George Konidaris, Scott Niekum, and Philip S Thomas. TDgamma: Re-evaluating Complex Backups in Temporal Difference Learning. In *Advances in Neural Information Processing Systems*, 2011.
- William Dabney and Philip S Thomas. Natural Temporal Difference Learning. In *AAAI Conference on Artificial Intelligence*, 2014.
- Pedro M Domingos and Geoff Hulten. A General Method for Scaling Up Machine Learning Algorithms and its Application to Clustering. In *International Conference on Machine Learning*, 2001.
- Paul Dagum, Richard Karp, Michael Luby, and Sheldon Ross. An Optimal Algorithm for Monte Carlo Estimation. *SIAM Journal on Computing*, 2006.
- Volodymyr Mnih, Csaba Szepesvari, and Jean-Yves Audibert. Empirical Bernstein stopping. In *the 25th international conference*, 2008.
- Daniel Hsu, Aryeh Kontorovich, and Csaba Szepesvari. Mixing Time Estimation in Reversible Markov Chains from a Single Sample Path. In *Advances in Neural Information Processing Systems*, 2015.
- Daniel Paulin. Concentration inequalities for Markov chains by Marton couplings and spectral methods. *Electronic Journal of Probability*, 2015.
- Jean-Yves Audibert, Rémi Munos, and Csaba Szepesvari. Tuning Bandit Algorithms in Stochastic Environments. In *Algorithmic Learning Theory*. 2007.
- BP Welford. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 1962.



---

# Combinatorial Bandits for Incentivizing Agents with Dynamic Preferences

---

Tanner Fiez\*, Shreyas Sekar\*, Liyuan Zheng, and Lillian J. Ratliff  
Electrical Engineering Department, University of Washington

## Abstract

The design of personalized incentives or recommendations to improve user engagement is gaining prominence as digital platform providers continually emerge. We propose a multi-armed bandit framework for matching incentives to users, whose preferences are unknown *a priori* and evolving dynamically in time, in a resource constrained environment. We design an algorithm that combines ideas from three distinct domains: (i) a greedy matching paradigm, (ii) the upper confidence bound algorithm (UCB) for bandits, and (iii) mixing times from the theory of Markov chains. For this algorithm, we provide theoretical bounds on the regret and demonstrate its performance via both synthetic and realistic (matching supply and demand in a bike-sharing platform) examples.

## 1 INTRODUCTION

The theory of *multi-armed bandits* plays a key role in enabling personalization in the digital economy (Scott, 2015). Algorithms from this domain have successfully been deployed in a diverse array of applications including online advertising (Lu et al., 2010; Mehta and Mirrokni, 2011), crowdsourcing (Tran-Thanh et al., 2014), content recommendation (Li et al., 2010), and selecting user-specific incentives (Ghosh and Hummel, 2013; Jain et al., 2014) (e.g., a retailer offering discounts). On the theoretical side, this has been complemented by a litany of *near-optimal regret bounds* for multi-armed bandit settings with rich combinatorial structures and complex agent behavior models (Chen et al., 2016; Gai et al., 2011; Kveton et al., 2015; Sani et al., 2012). At a high

level, the broad appeal of bandit approaches for allocating resources to human agents stems from its focus on balancing *exploration* with *exploitation*, thereby allowing a decision-maker to efficiently identify users' preferences without sacrificing short-term rewards.

Implicit in most of these works is the notion that in large-scale environments, a designer can simultaneously allocate resources to multiple users by running independent bandit instances. In reality, such independent decompositions do not make sense in applications where resources are subject to physical or monetary constraints. In simple terms, matching an agent to a resource immediately constrains the set of resources to which another agent can be matched. Such supply constraints may arise even when dealing with intangible products. For instance, social media platforms (e.g., Quora) seek to maximize user participation by offering incentives in the form of increased recognition—e.g., featured posts or badges (Immorlica et al., 2015). Of course, there are supply constraints on the number of posts or users that can be featured at a given time. As a consequence of these coupling constraints, much of the existing work on multi-armed bandits does not extend naturally to multi-agent economies.

Yet, another important aspect not addressed by the literature concerns human behavior. Specifically, users' preferences over the various resources may be dynamic—i.e. evolve in time as they are repeatedly exposed to the available options. The problem faced by a designer in such a dynamic environment is compounded by the lack of information regarding each user's current state or beliefs, as well as how these beliefs influence their preferences and their evolution in time.

Bearing in mind these limitations, we study a multi-armed bandit problem for matching multiple agents to a finite set of incentives<sup>1</sup>: each incentive belongs to a cate-

---

<sup>1</sup>We use the term incentive broadly to refer to any resource or action available to the agent. That is, incentives are not limited to monetary or financial mechanisms.

\* Authors contributed equally.

gory and global capacity constraints control the number of incentives that can be chosen from each category. In our model, each agent has a *preference profile* or a *type* that determines its rewards for being matched to different incentives. The agent’s type evolves according to a Markov decision process (MDP), and therefore, the rewards vary over time *in a correlated fashion*.

Our work is primarily motivated by the problem faced by a technological platform that seeks to not just maximize user engagement but also to encourage users to make changes in their *status quo* decision-making process by offering incentives. For concreteness, consider a bike-sharing service—an application we explore in our simulations—that seeks to identify optimal incentives for each user from a finite bundle of options—e.g., varying discount levels, free future rides, bulk ride offers, etc. Users’ preferences over the incentives may evolve with time depending on their current type, which in turn depends on their previous experience with the incentives. In addition to their marketing benefits, such incentives can serve as a powerful instrument for *nudging* users to park their bikes at alternative locations—this can lead to spatially balanced supply and consequently, lower rejection rates (Singla et al., 2015).

## 1.1 CONTRIBUTIONS AND ORGANIZATION

Our objective is to design a multi-armed bandit algorithm that repeatedly matches agents to incentives in order to minimize the cumulative *regret* over a finite time horizon. Here, regret is defined as the difference in the reward obtained by a problem specific benchmark strategy and the proposed algorithm (see Definition 1). A preliminary impediment in achieving this goal is the fact that the capacitated matching problem studied in this work is NP-Hard even in the offline case. The major challenge therefore is whether *we can achieve sub-linear (in the length of the horizon) regret in the more general matching environment without any information on the users’ underlying beliefs or how they evolve?*

Following preliminaries (Section 2), we introduce a simple greedy algorithm that provides a  $1/3$ -approximation to the optimal offline matching solution (Section 3). Leveraging this first contribution, the central result in this paper (Section 4) is a new multi-armed bandit algorithm—*MatchGreedy-EpochUCB* (MG-EUCB)—for capacitated matching problems with time-evolving rewards. Our algorithm obtains logarithmic (and hence sub-linear) regret even for this more general bandit problem. The proposed approach combines ideas from three distinct domains: (i) the  $1/3$ -rd approximate greedy matching algorithm, (ii) the traditional UCB algorithm (Auer et al., 2002), and

(iii) mixing times from the theory of Markov chains.

We validate our theoretical results (Section 5) by performing simulations on both synthetic and realistic instances derived using data obtained from a Boston-based bike-sharing service *Hubway* (hub). We compare our algorithm to existing UCB-based approaches and show that the proposed method enjoys favorable convergence rates, computational efficiency on large data sets, and does not get stuck at sub-optimal matching solutions.

## 1.2 BACKGROUND AND RELATED WORK

Two distinct features separate our model from the majority of work on the multi-armed bandit problem: (i) our focus on a capacitated matching problem with finite supply (every user cannot be matched to their optimal incentive), and (ii) the rewards associated with each agent evolve in a correlated fashion but the designer is unaware of each agent’s current state. Our work is closest to (Gai et al., 2011) which considers a matching problem with Markovian rewards. However, in their model the rewards associated with each edge evolve independently of the other edges; as we show via a simple example in Section 2.2, the correlated nature of rewards in our instance can lead to additional challenges and convergence to sub-optimal matchings if we employ a traditional approach as in (Gai et al., 2011).

Our work also bears conceptual similarities to the rich literature on combinatorial bandits (Badanidiyuru et al., 2013; Chen et al., 2016; Kveton et al., 2014, 2015; Wen et al., 2015). However, unlike our work, these papers consider a model where the distribution of the rewards is static in time. For this reason, efficient learning algorithms leveraging oracles to solve generic constrained combinatorial optimization problems developed for the combinatorial semi-bandit setting (Chen et al., 2016; Kveton et al., 2015) face similar limitations in our model as the approach of (Gai et al., 2011). Moreover, the rewards in our problem may not have a linear structure so the approach of (Wen et al., 2015) is not applicable.

The novelty in this work is not the combinatorial aspect but the interplay between combinatorial bandits and the edge rewards evolving according to an MDP. When an arm is selected by an oracle, the reward of every edge in the graph evolves—how it evolves depends on which arm is chosen. If the change occurs in a sub-optimal direction, this can affect future rewards. Indeed, the difficulties in our proofs do not stem from applying an oracle for combinatorial optimization, but from bounding the secondary regret that arises when rewards evolve in a sub-optimal way.

Finally, there is a somewhat parallel body of work

on single-agent reinforcement learning techniques (Azar et al., 2013; Jaksch et al., 2010; Mazumdar et al., 2017; Ratliff et al., 2018) and expert selection where the rewards on the arms evolve in a correlated fashion as in our work. In addition to our focus on multi-agent matchings, we remark that many of these works assume that the designer is aware (at least partially) of the agent’s exact state and thus, can eventually infer the nature of the evolution. Consequently, a major contribution of this work is the extension of UCB-based approaches to solve MDPs with a *fully unobserved state* and rewards associated with each edge that evolve in a correlated fashion.

## 2 PRELIMINARIES

A designer faces the problem of matching  $m$  agents to incentives (more generally jobs, goods, content, etc.) without violating certain capacity constraints. We model this setting by means of a bipartite graph  $(\mathcal{A}, \mathcal{I}, \mathcal{P})$  where  $\mathcal{A}$  is the set of agents,  $\mathcal{I}$  is the set of incentives to which the agents can be matched, and  $\mathcal{P} = \mathcal{A} \times \mathcal{I}$  is the set of all pairings between agents and incentives. We sometimes refer to  $\mathcal{P}$  as the set of arms. In this regard, a matching is a set  $M \subseteq \mathcal{P}$  such that every agent  $a \in \mathcal{A}$  and incentive  $i \in \mathcal{I}$  is present in at most one edge belonging to  $M$ .

Each agent  $a \in \mathcal{A}$  is associated with a type or state  $\theta_a \in \Theta_a$ , which influences the reward received by this agent when matched with some incentive  $i \in \mathcal{I}$ . When agent  $a$  is matched to incentive  $i$ , its type evolves according to a Markov process with transition probability kernel  $P_{a,i} : \Theta_a \times \Theta_a \rightarrow [0, 1]$ . Each pairing or edge of the bipartite graph is associated with some reward that depends on the agent–incentive pair,  $(a, i)$ , as well as the type  $\theta_a$ .

The designer’s policy (algorithm) is to compute a matching repeatedly over a finite time horizon in order to maximize the expected aggregate reward. In this work, we restrict our attention to a specific type of multi-armed bandit algorithm that we refer to as an *epoch mixing policy*. Formally, the execution of such a policy  $\alpha$  is divided into a finite number of time indices  $[n] = \{1, 2, \dots, n\}$ , where  $n$  is the length of the time horizon. In each time index  $k \in [n]$ , the policy selects a matching  $\alpha(k)$  and repeatedly ‘plays’ this matching for  $\tau_k > 0$  iterations within this time index. We refer to the set of iterations within a time index collectively as an *epoch*. That is, within the  $k$ –th epoch, for each edge  $(a, i) \in \alpha(k)$ , agent  $a$  is matched to incentive  $i$  and the agent’s type is allowed to evolve for  $\tau_k$  iterations. In short, an epoch mixing policy proceeds in two time scales—each selection of a matching corresponds to an epoch comprising of  $\tau_k$  iterations for  $k \in [n]$ , and there are a total of  $n$  epochs. It is worth noting that an epoch-based policy was used in the UCB2 algorithm (Auer et al., 2002), albeit with

stationary rewards.

Agents’ types evolve based on the incentives to which they are matched. Suppose that  $\beta_a^{(k)}$  denotes the type distribution on  $\Theta_a$  at epoch  $k$  and  $i \in \mathcal{I}$  is the incentive to which agent  $a$  is matched by  $\alpha$  (i.e.,  $(a, i) \in \alpha(k)$ ). Then,  $\beta_a^{(k+1)}(\theta_a) = \sum_{\theta' \in \Theta_a} P_{a,i}^{\tau_k}(\theta', \theta_a) \beta_a^{(k)}(\theta')$ .

For epoch  $k$ , the rewards are averaged over the  $\tau_k$  iterations in that epoch. Let  $r_{a,i}^\theta$  denote the reward received by agent  $a$  when it is matched to incentive  $i$  given type  $\theta \in \Theta_a$ . We assume that  $r_{a,i}^\theta \in [0, 1]$  and is drawn from a distribution  $\mathcal{T}_r(a, i, \theta)$ . The reward distributions for different edges and states in  $\Theta_a$  are assumed to be independent of each other. Suppose that an algorithm  $\alpha$  selects the edge  $(a, i)$  for  $\tau$  iterations within an epoch. The observed reward at the end of this epoch is taken to be the time-averaged reward over the epoch. Specifically, suppose that the  $k$ –th epoch proceeds for  $\tau_k$  iterations beginning with time  $t_k$ —i.e. one plus the total iterations completed before this—and ending at time  $t_{k+1} - 1 = t_k + \tau_k - 1$ , and that  $\theta_a(t)$  denotes agent  $a$ ’s state at time  $t \in [t_k, t_{k+1} - 1]$ . Then, the time-averaged reward in the epoch is given by  $\mathbf{r}_{a,i}^{\theta_a(t_k)} = \frac{1}{\tau_k} \sum_{t=t_k}^{t_{k+1}-1} r_{a,i}^{\theta_a(t)}$ . We use the state as a superscript to denote dependence of the reward on the agent’s type at the beginning of the epoch. Finally, the total (time-averaged) reward due to a matching  $\alpha(k)$  at the end of an epoch can be written as  $\sum_{(a,i) \in \alpha(k)} \mathbf{r}_{a,i}^{\theta_a(t_k)}$ .

We assume that the Markov chain corresponding to each edge  $(a, i) \in \mathcal{P}$  is *aperiodic* and *irreducible* (Levin et al., 2009). We denote the stationary or steady-state distribution for this edge as  $\pi_{a,i} : \Theta_a \rightarrow [0, 1]$ . Hence, we define the expected reward for edge  $(a, i)$ , given its stationary distribution, to be  $\mu_{a,i} = \mathbb{E} [\sum_{\theta \in \Theta_a} r_{a,i}^\theta \pi_{a,i}(\theta)]$  where the expectation is with respect to the distribution on the reward given  $\theta$ .

### 2.1 CAPACITATED MATCHING

Given  $\mathcal{P} = \mathcal{A} \times \mathcal{I}$ , the designer’s goal at the beginning of each epoch is to select a matching  $M \subseteq \mathcal{P}$ —i.e. a collection of edges—that satisfies some cardinality constraints. We partition the edges in  $\mathcal{P}$  into a mutually exclusive set of classes allowing for edges possessing identical characteristics to be grouped together. In the bike-sharing example, the various classes could denote types of incentives—e.g., edges that match agents to discounts, free-rides, etc. Suppose that  $\mathcal{C} = \{\xi_1, \xi_2, \dots, \xi_q\}$  denotes a partitioning of the edge set such that (i)  $\xi_j \subseteq \mathcal{P}$  for all  $1 \leq j \leq q$ , (ii)  $\bigcup_{j=1}^q \xi_j = \mathcal{P}$ , and (iii)  $\xi_j \cap \xi_{j'} = \emptyset$  for all  $j \neq j'$ . We refer to each  $\xi_j$  as a class and for any given edge  $(a, i) \in \mathcal{P}$ , use  $c(a, i)$  to denote the class that this edge belongs to, i.e.,  $(a, i) \in c(a, i)$  and  $c(a, i) \in \mathcal{C}$ .

Given a capacity vector  $\mathbf{b} = (b_{\xi_1}, \dots, b_{\xi_q})$  indexed on the set of classes, we say that a matching  $M \subseteq \mathcal{P}$  is a feasible solution to the capacitated matching problem if:

- a) for every  $a \in \mathcal{A}$  (resp.,  $i \in \mathcal{I}$ ), the matching must contain at most one edge containing this agent (resp., incentive)
- b) and, the total number of edges from each class  $\xi_j$  contained in the matching cannot be larger than  $b_{\xi_j}$ .

In summary, the *capacitated matching problem* can be formulated as the following integer program:

$$\begin{aligned}
 \max \quad & \sum_{(a,i) \in \mathcal{P}} w(a,i)x(a,i) \\
 \text{s.t.} \quad & \sum_{i \in \mathcal{I}} x(a,i) \leq 1 \quad \forall a \in \mathcal{A} \\
 & \sum_{a \in \mathcal{A}} x(a,i) \leq 1 \quad \forall i \in \mathcal{I} \\
 & \sum_{(a,i) \in \xi_j} x(a,i) \leq b_{\xi_j}, \quad \forall \xi_j \in \mathcal{C} \\
 & x(a,i) \in \{0, 1\}, \quad \forall (a,i) \in \mathcal{P}
 \end{aligned} \tag{P1}$$

We use the notation  $\{\mathcal{P}, \mathcal{C}, \mathbf{b}, (w(a,i))_{(a,i) \in \mathcal{P}}\}$  for a *capacitated matching problem instance*. In (P1),  $w(a,i)$  refers to the weight or the reward to be obtained from the given edge. The term  $x(a,i)$  is an indicator on whether the edge  $(a,i)$  is included in the solution to (P1). Clearly, the goal is to select a maximum weight matching subject to the constraints. In our online bandit problem, the designer’s actual goal in a fixed epoch  $k$  is to maximize the quantity  $\sum_{(a,i) \in \mathcal{P}} \mathbf{r}_{a,i}^{\theta_a(t_k)} x(a,i)$ , i.e.,  $w(a,i) = \mathbf{r}_{a,i}^{\theta_a(t_k)}$ . However, since the reward distributions and the current user type are not known beforehand, our MG-EUCB algorithm (detailed in Section 4.2) approximates this objective by setting the weights to be the average observed reward from the edges in combination with the corresponding confidence bounds.

## 2.2 TECHNICAL CHALLENGES

There are two key obstacles involved in extending traditional bandit approaches to our combinatorial setting with evolving rewards, namely, *cascading sub-optimality* and *correlated convergence*. The first phenomenon occurs when an agent  $a$  is matched to a sub-optimal arm  $i$  (incentive) because its optimal arm  $i^*$  has already been assigned to another agent. Such sub-optimal pairings have the potential to cascade, e.g., when another agent  $a_1$  who is matched to  $i$  in the optimal solution can no longer receive this incentive and so on. Therefore, unlike the classical bandit analysis, the selection of sub-optimal arms cannot be directly mapped to the empirical rewards.

*Correlated Convergence.* As mentioned previously, in our model, the rewards depend on the type or state of an agent, and hence, the reward distribution on any given edge  $(a,i)$  can vary even when the algorithm does not select this edge. As a result, a naïve application of a bandit algorithm can severely under-estimate the expected

reward on each edge and eventually converge to a sub-optimal matching. A concrete example of the poor convergence effect is provided in Example 1. In Section 4.2, we describe how our central bandit algorithm limits the damage due to cascading while simultaneously avoiding the correlated convergence problem.

**Example 1** (Failure of Classical UCB). *Consider a problem instance with two agents  $\mathcal{A} = \{a_1, a_2\}$ , two incentives  $\mathcal{I} = \{i_1, i_2\}$  and identical state space i.e.,  $\Theta_{a_1} = \Theta_{a_2} = \{\theta_1, \theta_2\}$ . The transition matrices and deterministic rewards for the agents for being matched to each incentive are depicted pictorially below: we assume that  $\epsilon > 0$  is a sufficiently small constant.*

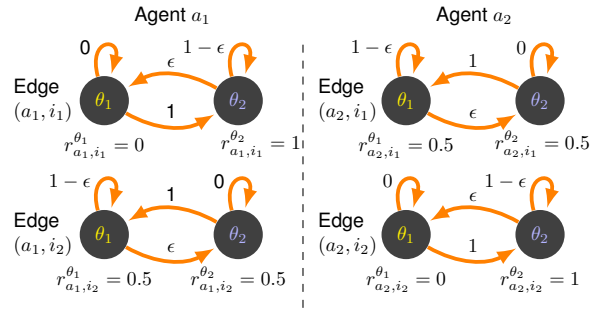


Figure 1: (a) State transition diagram and reward for each edge: note that the state is associated with the agent and not the edge.

Clearly, the optimal strategy is to repeatedly choose the matching  $\{(a_1, i_1), (a_2, i_2)\}$  achieving a reward of (almost) two in each epoch. An implementation of traditional UCB for the matching problem—e.g., the approach in (Chen et al., 2016; Gai et al., 2011; Kveton et al., 2015)—selects a matching based on the empirical rewards and confidence bounds for a total of  $\sum_{k=1}^n \tau_k$  iterations, which are then divided into epochs for convenience. This approach converges to the sub-optimal matching of  $M = \{(a_1, i_2), (a_2, i_1)\}$ . Indeed, every time the algorithm selects this matching, both the agents’ states are reset to  $\theta_1$  and when the algorithm explores the optimum matching, the reward consistently happens to be zero since the agents are in state  $\theta_1$ . Hence, the rewards for the (edges in the) optimum matching are grossly underestimated.

## 3 GREEDY OFFLINE MATCHING

In this section, we consider the capacitated matching problem in the offline case, where the edge weights are provided as input. The techniques developed in this section serve as a base in order to solve the more general online problem in the next section. More specifically, we assume that we are given an arbitrary instance of the capacitated matching problem  $\{\mathcal{P}, \mathcal{C}, \mathbf{b}, (w(a,i))_{(a,i) \in \mathcal{P}}\}$ .

---

**Algorithm 1** Capacitated-Greedy Matching Algorithm

---

```
1: function MG( $(w(a, i))_{(a, i) \in \mathcal{P}}, \mathbf{b}$ )
2:    $G^* \leftarrow \emptyset, E' \leftarrow \mathcal{P}$ 
3:   while  $E' \neq \emptyset$ :
4:     Select  $(a, i) = \arg \max_{(a', i') \in E'} w(a', i')$ 
5:     if  $|G^* \cap c(a, i)| < b_{c(a, i)}$  then
6:        $G^* \leftarrow G^* \cup \{(a, i)\}$ 
7:        $E' \leftarrow E' \setminus \{(a', i') \mid \forall (a', i') : a' = a \text{ or } i' = i\}$ 
8:     else
9:        $E' \leftarrow E' \setminus \{(a, i)\}$ 
10:    return  $G^*$ 
11: end function
```

---

Given this instance, the designer’s objective is to solve (P1). Surprisingly, this problem turns out to be NP-Hard and thus cannot be optimally solved in polynomial time (Garey and Johnson, 1979)—this marks a stark contrast with the classic maximum weighted matching problem, which can be solved efficiently using the Hungarian method (Kuhn, 1955).

In view of these computational difficulties, we develop a simple greedy approach for the capacitated matching problem and formally prove that it results in a one-third approximation to the optimum solution. The greedy method studied in this work comes with a multitude of desirable properties that render it suitable for matching problems arising in large-scale economies. Firstly, the greedy algorithm has a running time of  $O(m^2 \log m)$ , where  $m$  is the number of agents—this near-linear execution time in the number of edges makes it ideal for platforms comprising of a large number of agents. Secondly, since the output of the greedy algorithm depends only on the ordering of the edge weights and is not sensitive to their exact numerical value, learning approaches tend to converge faster to the ‘optimum solution’. This property is validated by our simulations (see Figure 2c). Finally, the performance of the greedy algorithm in practice (e.g., see Figure 2b) appears to be much closer to the optimum solution than the 1/3 approximation guaranteed by Theorem 1 below.

### 3.1 ANALYSIS OF GREEDY ALGORITHM

The greedy matching is outlined in Algorithm 1. Given an instance  $\{\mathcal{P}, \mathcal{C}, \mathbf{b}, (w(a, i))_{(a, i) \in \mathcal{P}}\}$ , Algorithm 1 ‘greedily’ selects the highest weight feasible edge in each iteration—this step is repeated until all available edges that are feasible are added to  $G^*$ . Our main result in this section is that for any given instance of the capacitated matching problem, the matching  $G^*$  returned by Algorithm 1 has a total weight that is at least 1/3-rd that of the maximum weight matching.

**Theorem 1.** *For any given capacitated matching problem instance  $\{\mathcal{P}, \mathcal{C}, \mathbf{b}, (w(a, i))_{(a, i) \in \mathcal{P}}\}$ , let  $G^*$  denote the output of Algorithm 1 and  $M^*$  be any other feasible solution to the optimization problem in (P1) including the optimum matching. Then,  $\sum_{(a, i) \in M^*} w(a, i) \leq 3 \sum_{(a, i) \in G^*} w(a, i)$ .*

The proof is based on a *charging argument* that takes into account the capacity constraints and can be found in Section B.1 of the supplementary material. At a high level, we take each edge belonging to the benchmark  $M^*$  and identify a corresponding edge in  $G^*$  whose weight is larger than that of the benchmark edge. This allows us to charge the weight of the original edge to an edge in  $G^*$ . During the charging process, we ensure that no more than three edges in  $M^*$  are charged to each edge in  $G^*$ . This gives us an approximation factor of three.

### 3.2 PROPERTIES OF GREEDY MATCHINGS

We conclude this section by providing a hierarchical decomposition of the edges in  $\mathcal{P}$  for a fixed instance  $\{\mathcal{P}, \mathcal{C}, \mathbf{b}, (w(a, i))_{(a, i) \in \mathcal{P}}\}$ . In Section 4.1, we will use this property to reconcile the offline version of the problem with the online bandit case. Let  $G^* = \{g_1^*, g_2^*, \dots, g_m^*\}$  denote the matching computed by Algorithm 1 for the given instance such that  $w(g_1^*) \geq w(g_2^*) \geq \dots \geq w(g_m^*)$  without loss of generality<sup>2</sup>. Next, let  $G_j^* = \{g_1^*, g_2^*, \dots, g_j^*\}$  for all  $1 \leq j \leq m$ —i.e. the  $j$  highest-weight edges in the greedy matching.

For each  $1 \leq j \leq m$ , we define the infeasibility set  $H_j^{G^*}$  as the set of edges in  $\mathcal{P}$  that when added to  $G_j^*$  violates the feasibility constraints of (P1). Finally, we use  $L_j^{G^*}$  to denote the marginal infeasibility sets—i.e.  $L_1^{G^*} = H_1^{G^*}$  and

$$L_j^{G^*} = H_j^{G^*} \setminus H_{j-1}^{G^*}, \forall 2 \leq j \leq m. \quad (1)$$

We note that the marginal infeasibility sets denote a mutually exclusive partition of the edge set minus the greedy matching—i.e.,  $\bigcup_{1 \leq j \leq m} L_j^{G^*} = \mathcal{P} \setminus G^*$ . Moreover, since the greedy matching selects its edges in the decreasing order of weight, for any  $g_j^* \in G^*$ , and every  $(a, i) \in L_j^{G^*}$ , we have that  $w(g_j^*) \geq w(a, i)$ .

Armed with our decomposition of the edges in  $\mathcal{P} \setminus G^*$ , we now present a crucial structural lemma. The following lemma identifies sufficient conditions on the *local ordering* of the edge weights for two different instances under which the outputs of the greedy matching for the instances are non-identical.

**Lemma 1.** *Given instances  $\{\mathcal{P}, \mathcal{C}, \mathbf{b}, (w(a, i))_{(a, i) \in \mathcal{P}}\}$  and  $\{\mathcal{P}, \mathcal{C}, \mathbf{b}, (\tilde{w}(a, i))_{(a, i) \in \mathcal{P}}\}$  of the capacitated matching problem, let  $G^* = \{g_1^*, g_2^*, \dots, g_m^*\}$  and  $\tilde{G}$  denote*

<sup>2</sup>If  $g = (a, i)$ , we abuse notation and let  $w(g) = w(a, i)$ .

the output of Algorithm 1 for these instances, respectively. Let  $E_1, E_2$  be conditions described as follows:

$$\begin{aligned} E_1 &= \{\exists j < j' \mid (\tilde{w}(g_j^*) < \tilde{w}(g_{j'}^*)) \wedge (g_j^* \in \tilde{G})\} \\ E_2 &= \{\exists g_j^* \in G^*, (a, i) \in L_j^{G^*} \mid \\ &\quad (\tilde{w}(g_j^*) < \tilde{w}(a, i)) \wedge ((a, i) \in \tilde{G})\}. \end{aligned}$$

If  $G^* \neq \tilde{G}$ , then at least one of  $E_1$  or  $E_2$  must be true.

Lemma 1 is fundamental in the analysis of our MG-EUCB algorithm because it provides a method to map the selection of each sub-optimal edge to a familiar condition comparing empirical rewards to stationary rewards.

## 4 ONLINE MATCHING—BANDIT ALGORITHM

In this section, we propose a multi-armed bandit algorithm for the capacitated matching problem and analyze its regret. For concreteness, we first highlight the information and action sets available to the designer in the online problem. The designer is presented with a *partial instance* of the matching problem without the weights, i.e.,  $\{\mathcal{P}, \mathcal{C}, \mathbf{b}\}$  along with a fixed time horizon of  $n$  epochs but has the ability to set the parameters  $(\tau_1, \tau_2, \dots, \tau_n)$ , where  $\tau_k$  is the number of iterations under epoch  $k$ . The designer’s goal is to design a policy  $\alpha$  that selects a matching  $\alpha(k)$  in the  $k$ -th epoch that is a feasible solution for (P1). At the end of the  $k$ -th epoch, the designer observes the average reward  $\mathbf{r}_{a,i}^{\theta_a(k)}$  for each  $(a, i) \in \alpha(k)$  but *not the agent’s type*. We abuse notation and take  $\theta_a(k)$  to be the agent’s state at the beginning of epoch  $k$ . The designer’s objective is to minimize the regret over the finite horizon.

The expected regret of a policy  $\alpha$  is the difference in the expected aggregate reward of a benchmark matching and that of the matching returned by the policy, summed over  $n$  epochs. Owing to its favorable properties (see Section 3), we use the greedy matching on the stationary state rewards as our benchmark. Measuring the regret with respect to the unknown stationary-distribution is standard with MDPs (e.g., see (Gai et al., 2011; Tekin and Liu, 2010, 2012)). Formally, let  $G^*$  denote the output of Algorithm 1 on the instance  $\{\mathcal{P}, \mathcal{C}, \mathbf{b}, (\mu_{a,i})_{(a,i) \in \mathcal{P}}\}$ —i.e., with the weights  $w(a, i)$  equal the stationary state rewards  $\mu_{a,i}$ .

**Definition 1.** *The expected regret of a policy  $\alpha$  with respect to the greedy matching  $G^*$  is given by*

$$R^\alpha(n) = n \sum_{(a,i) \in G^*} \mu_{a,i} - \sum_{k=1}^n \sum_{(a,i) \in \alpha(k)} \mathbb{E}[\mathbf{r}_{a,i}^{\theta_a(k)}],$$

where the expectation is with respect to the reward and the state of the agents during each epoch.

## 4.1 REGRET DECOMPOSITION

As is usual in this type of analysis, we start by decomposing the regret in terms of the number of selections of each sub-optimal arm (edge). We state some assumptions and define notation before proving our generic regret decomposition theorem. A complete list of the notation used can be found in Section A of the supplementary material.

1. For analytic convenience, we assume that the number of agents and incentives is balanced and therefore,  $|\mathcal{A}| = |\mathcal{I}| = m$ . WLOG, every agent is matched to some incentive in  $G^*$ ; if this is not the case, we can add *dummy incentives* with zero reward.
2. Suppose that  $G^* = \{g_1^*, g_2^*, \dots, g_m^*\}$  such that  $\mu_{g_1^*} \geq \dots \geq \mu_{g_m^*}$  and let  $i^*(a)$  denote the incentive that  $a$  is matched to in  $G^*$ . Let  $L_1^*, \dots, L_m^*$  be the marginal infeasibility sets as defined in (1).
3. Suppose that  $\tau_0 \geq 1$  and  $\tau_k = \tau_0 + \zeta k$  for some non-negative integer  $\zeta$ .

Let  $\mathbb{1}\{\cdot\}$  be the indicator function—e.g.,  $\mathbb{1}\{(a, i) \in \alpha(k)\}$  is one when the edge  $(a, i)$  belongs to the matching  $\alpha(k)$ , and zero otherwise. Define  $T_{a,i}^\alpha(n) = \sum_{k=1}^n \mathbb{1}\{(a, i) \in \alpha(k)\}$  to be the random variable that denotes the number of epochs in which an edge is selected under an algorithm  $\alpha$ . By relating  $\mathbb{E}[T_{a,i}^\alpha(n)]$  to the regret  $R^\alpha(n)$ , we are able to provide bounds on the performance of  $\alpha$ .

By adding and subtracting  $\sum_{(a,i) \in \mathcal{P}} \mathbb{E}[T_{a,i}^\alpha(n)] \mu_{a,i}$  from the equation in Definition 1, we get that

$$\begin{aligned} R^\alpha(n) &= \sum_{(a,i) \in \mathcal{P}} \mathbb{E}[T_{a,i}^\alpha(n)] (\mu_{a,i^*(a)} - \mu_{a,i}) \\ &\quad + \sum_{k=1}^n \sum_{(a,i) \in \mathcal{P}} \mathbb{E}[\mathbb{1}\{(a, i) \in \alpha(k)\} (\mu_{a,i} - \mathbf{r}_{a,i}^{\theta_a(k)})]. \end{aligned}$$

To further simplify the regret, we separate the edges in  $\mathcal{P}$  by introducing the notion of a sub-optimal edge. Formally, for any given  $a \in \mathcal{A}$ , define  $S_a := \{(a, i) \mid \mu_{a,i^*(a)} \geq \mu_{a,i} \forall i \in \mathcal{I}\}$  and  $\mathcal{S} := \bigcup_{a \in \mathcal{A}} S_a$ . Then, the regret bound in the above equation can be simplified by ignoring the contribution of the terms in  $\mathcal{P} \setminus \mathcal{S}$ . That is, since  $\mu_{a,i^*(a)} < \mu_{a,i}$  for all  $(a, i) \in \mathcal{P} \setminus \mathcal{S}$ ,

$$\begin{aligned} R^\alpha(n) &\leq \sum_{(a,i) \in \mathcal{S}} \mathbb{E}[T_{a,i}^\alpha(n)] (\mu_{a,i^*(a)} - \mu_{a,i}) \\ &\quad + \sum_{k=1}^n \sum_{(a,i) \in \mathcal{P}} \mathbb{E}[\mathbb{1}\{(a, i) \in \alpha(k)\} (\mu_{a,i} - \mathbf{r}_{a,i}^{\theta_a(k)})]. \end{aligned} \quad (2)$$

Recall from the definition of the marginal infeasibility sets in (1) that for any given  $(a, i) \in \mathcal{P} \setminus G^*$ , there exists a unique edge  $g_j^* \in G^*$  such that  $(a, i) \in L_j^*$ . Define  $L^{-1}(a, i) := g_j^* \in G^*$  such that  $(a, i) \in L_j^*$ . Now, we can define the reward gap for any given edge as follows:

$$\Delta_{a,i} = \begin{cases} \mu_{a,i^*(a)} - \mu_{a,i}, & \text{if } (a, i) \in \mathcal{S} \\ \mu_{L^{-1}(a,i)} - \mu_{a,i}, & \text{if } (a, i) \in (\mathcal{P} \setminus G^*) \setminus \mathcal{S} \\ \mu_{g_{j-1}^*} - \mu_{g_j^*}, & \text{if } (a, i) = g_j^* \text{ for } j \geq 2 \end{cases}$$

This leads us to our main regret decomposition result which leverages mixing times for Markov chains (Fill, 1991) along with Equation (2) in deriving regret bounds. For an aperiodic, irreducible Markov chain  $P_{a,i}$ , using the notion that it converges to its stationary state under repeated plays of a fixed action, we can prove that for every arm  $(a, i)$ , there exists a constant  $C_{a,i} > 0$  such that  $|\mathbb{E}[\mu_{a,i} - r_{a,i}^{\theta_a^{(k)}}]| \leq C_{a,i}/\tau_k$ —in fact, this result holds for all type distributions  $\beta_a^{(k)}$  of the agent.

**Proposition 1.** *Suppose for each  $(a, i) \in \mathcal{P}$ ,  $P_{a,i}$  is an aperiodic, irreducible Markov chain with corresponding constant  $C_{a,i}$ . Then, for a given algorithm  $\alpha$  where  $\tau_k = \tau_0 + \zeta k$  for some fixed  $\zeta > 0$ , we have that*

$$R^\alpha(n) \leq \sum_{(a,i) \in \mathcal{S}} \mathbb{E}_\alpha [T_{a,i}^\alpha(n)] (\Delta_{a,i} + \frac{C_{a,i}}{\tau_0}) + m \frac{C_{a,i}}{\zeta} (1 + \log(\zeta(n-1)/\tau_0 + 1)).$$

The proof of this proposition is in Section B.2 of the supplementary material.

## 4.2 MG-EUCB ALGORITHM AND ANALYSIS

In the initialization phase, the algorithm computes and plays a sequence of matchings  $M_1, M_2, \dots, M_p$  for a total of  $p$  epochs. The initial matchings ensure that every edge in  $\mathcal{P}$  is selected at least once—the computation of these initial matchings relies on a *greedy covering* algorithm that is described in Section C.1 of the supplementary material. Following this, our algorithm maintains the cumulative empirical reward  $\bar{r}_{a,i}$  for every  $(a, i) \in \mathcal{P}$ . At the beginning of (say) epoch  $k$ , the algorithm computes a greedy matching for the instance  $\{\mathcal{P}, \mathcal{C}, \mathbf{b}, (w(a, i))_{(a,i) \in \mathcal{P}}\}$  where  $w(a, i) = \bar{r}_{a,i}/T_{a,i} + c_{a,i}$ , i.e., the average empirical reward for the edge added to a suitably chosen confidence window. The `INCENT`( $\cdot$ ) function (Algorithm 2, described in the supplementary material since it is a trivial function) plays each edge in the greedy matching for  $\tau_k$  iterations, where  $\tau_k$  increases linearly with  $k$ . This process is repeated for  $n-p$  epochs. Prior to theoretically analyzing MG-EUCB, we return to Example 1 in order to provide intuition for how the algorithm overcomes correlated convergence of rewards.

**Revisiting Example 1:** Why does MG-EUCB work? In Example 1, the algorithm initially estimates the empirical reward of  $(a_1, i_1)$  and  $(a_2, i_2)$  to be zero respectively. However, during the UCB exploration phase, the matching  $M_1 = (a_1, i_1), (a_2, i_2)$  is played again for epoch length  $> 1$  and the state of agent  $a_1$  moves from  $\theta_1$  to  $\theta_2$  during the epoch. Therefore, the algorithm estimates the average reward of each edge within the epoch to be  $\geq 0.5$ , and the empirical reward increases. This continues as the epoch length increases, so that eventually the

---

### Algorithm 2 MatchGreedy-EpochUCB

---

```

1: procedure MG-EUCB( $\zeta, \tau_0, \mathcal{P}$ )
2:    $t_1 \leftarrow 0, \bar{r}_{a,i} \leftarrow 0 \ \& \ T_{a,i} \leftarrow 1 \ \forall (a, i) \in \mathcal{P}$ 
3:    $M_1, M_2, \dots, M_p \subset \mathcal{P}$  s.t.  $(a, i) \in M_j \Leftrightarrow (a, i) \notin M_\ell \ \forall \ell \neq j$   $\triangleright$  see Supplement C.1 for details
4:   INCENT( $\cdot$ )  $\triangleright$  see Alg. 2 in Supplement C
5:   for  $1 \leq n \leq m$   $\triangleright$  play each arm once
6:      $(\bar{r}_{a,i})_{(a,i) \in M_n} \leftarrow \text{INCENT}(M_n, t_n, n, \tau_0, \zeta)$ 
7:      $t_{n+1} \leftarrow t_n + \tau_0 + \zeta n$ 
8:   end for
9:   while  $n > m$ 
10:     $M_G = \text{MG}((\bar{r}_{a,i}/T_{a,i} + c_{a,i}^{T_{a,i}}(n))_{(a,i) \in \mathcal{P}})$ 
11:     $(r_{a,i}(t_n))_{(a,i) \in M_G} \leftarrow \text{INCENT}(M_G, t_n, n, \tau_0, \zeta)$ 
12:     $\bar{r}_{a,i} \leftarrow \bar{r}_{a,i} + \frac{1}{\tau_0 + \zeta n} r_{a,i}(t_n) \ \forall (a, i) \in M_G$ 
13:     $T_{a,i} \leftarrow T_{a,i} + 1 \ \forall (a, i) \in M_G$ 
14:     $t_{n+1} \leftarrow t_n + \tau_0 + \zeta n; n \leftarrow n + 1$ 
15:  end while
16: end procedure

```

---

empirical reward for  $(a_1, i_1)$  exceeds that of  $(a_1, i_2)$  and the algorithm correctly identifies the optimal matching as we move from exploration to exploitation.

In order to characterize the regret of the MG-EUCB algorithm, Proposition 1 implies that it is sufficient to bound the expected number of epochs in which our algorithm selects each sub-optimal edge. The following theorem presents an upper bound on this quantity.

**Theorem 2.** *Consider a finite set of  $m$  agents  $\mathcal{A}$  and incentives  $\mathcal{I}$  with corresponding aperiodic, irreducible Markov chains  $P_{a,i}$  for each  $(a, i) \in \mathcal{P}$ . Let  $\alpha$  be the MG-EUCB algorithm with mixing time sequence  $\{\tau_k\}$  where  $\tau_k = \tau_0 + \zeta k$ ,  $\tau_0 > 0$ , and  $\zeta > 0$ . Then for every  $(a, i) \in \mathcal{S}$ ,*

$$\mathbb{E}_\alpha [T_{a,i}(n)] \leq \frac{4m^2}{\Delta_{a^*,i^*}^2} \left( \frac{\rho_{a^*,i^*}}{\sqrt{\tau_0}} + \sqrt{6 \log n + 4 \log m} \right)^2 + 2(1 + \log(n))$$

where  $(a^*, i^*) = \operatorname{argmax}_{(a_1, i_1) \in \mathcal{P} \setminus g_1^*} \left[ \frac{4}{\Delta_{a_1, i_1}^2} \left( \frac{\rho_{a_1, i_1}}{\sqrt{\tau_0}} + \sqrt{6 \log n + 4 \log m} \right)^2 \right]$ , and  $\rho_{a,i}$  is a constant specific to edge  $(a, i)$ .

The full proof of the theorem is provided can be found in the supplementary material.

**Proof** (sketch.) There are three key ingredients to the proof: (i) linearly increasing epoch lengths, (ii) overcoming cascading errors, and (iii) application of the Azuma-Hoeffding concentration inequality.

By increasing the epoch length linearly, MG-EUCB ensures that as the algorithm converges to the optimal

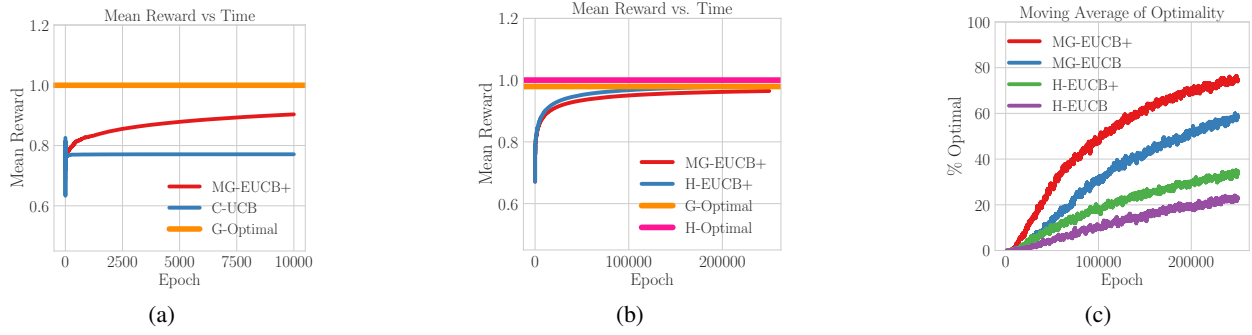


Figure 2: Synthetic Experiments: Comparison of MG-EUCB(+) and H-EUCB(+) to their respective offline solutions (G- and H-optimal, respectively) and to C-UCB (classical UCB). We use the following set up: (i)  $|\mathcal{A}| = |\mathcal{I}| = |\Theta_a| = 10$  (see Supplement D for more extensive experiments) (ii) each state transition matrix  $P_{a,i}$  associated with an arm  $(a, i) \in \mathcal{P}$  was selected uniformly at random within the class of aperiodic and irreducible stochastic matrices; (iii) the reward for each arm, state pair  $r_{a,i}^\theta$  is drawn i.i.d. from a distribution  $\mathcal{T}_r(a, i, \theta)$  belonging to either a Bernoulli, Uniform, or Beta distribution; (iv)  $\tau_0 = 50$  and  $\zeta = 1$ .

matching, it also plays each arm for a longer duration within an epoch. This helps the algorithm to progressively discard sub-optimal arms without selecting them too many times when the epoch length is still small. At the same time, the epoch length is long enough to allow for sufficient mixing and separation between multiple near-optimal matchings. If we fix the epoch length as a constant, the resulting regret bounds are considerably worse because the agent states may never converge to the steady-state distributions.

To address cascading errors, we provide a useful characterization. For a given  $(a, i)$ , suppose that  $u_{a,i}^k(t)$  refers to the average empirical reward obtained from edge  $(a, i)$  up to epoch  $t-1$  plus the upper confidence bound parameter, given that edge  $(a, i)$  has been selected for exactly  $k$  times in epochs 1 to  $t-1$ . For any given epoch  $k$  where the algorithm selects a sub-optimal matching, i.e.,  $\alpha(k) \neq G^*$ , we can apply Lemma 1 and get that at least one of the following conditions must be true:

1.  $\mathbb{1}\{\exists j < j' \mid (u_{g_j^*}^k(t) > u_{g_{j'}^*}^k(t)) \wedge (g_{j'}^* \in \alpha(t))\}$
2.  $\mathbb{1}\{\exists j, (a, i) \in L_j^* \mid (u_{g_j^*}^k(t) < u_{a,i}^k(t)) \vee ((a, i) \in \alpha(k))\} = 1$

This is a particularly useful characterization because it maps the selection of each sub-optimal edge to a familiar condition that compares the empirical rewards to the stationary rewards. Therefore, once each arm is selected for  $O(\log(n))$  epochs, the empirical rewards approach the ‘true’ rewards and our algorithm discards sub-optimal edges. Mathematically, this can be written as

$$\begin{aligned} \mathbb{E}_\alpha[T_{a',i'}(n)] &= 1 + \sum_{t=p+1}^n \mathbb{1}\{(a', i') \in \alpha(t)\} \\ &\leq \ell m^2 + \sum_{j=1}^m \sum_{(a,i) \in L_j^+} \sum_{t=p+1}^n \sum_{s=1}^{t-1} \sum_{k=\ell}^{t-1} ( \\ &\mathbb{1}\{u_{g_j^*}^s(t) \leq u_{a,i}^k(t)\}), \end{aligned}$$

where  $\ell$  is some carefully chosen constant,  $L_j^+ = L_j^* \cup \{g_{j+1}^*\}$  and  $L_m^+ = L_m^*$ .

With this characterization, for each  $s$ , we find an upper bound on the probability of the event  $\{u_{g_j^*}^s(t) \leq u_{a,i}^k(t)\}$ . However, this is a non-trivial task since the reward obtained in any given epoch is *not independent* of the previous actions. Specifically, the underlying Markov process that generates the rewards is common across the edges connected to any given agent in the sense, that the initial distribution for each Markov chain that results from pulling an edge is the distribution at the end of the preceding pull. Therefore, we employ Azuma-Hoeffding (Azuma, 1967; Hoeffding, 1963), a concentration inequality that does not require independence in the arm-based observed rewards. Moreover, unlike the classical UCB analysis, the empirical reward can differ from the expected stationary reward due to the distributions  $\mathcal{T}_r(a, i, \theta)$  and  $\beta_{a,i}^k \neq \pi_{a,i}$ . To account for this additional error term, we use bounds on the convergence rates of Markov chains to guide the choice of the confidence parameter  $c_{a,i}^k(t)$  in Algorithm 2. Applying the Azuma-Hoeffding inequality, we can show that with high probability, the difference between the empirical reward and the stationary reward of edge  $(a, i)$  is no larger than  $c_{a,i}^k(t)$ . ■

As a direct consequence of Proposition 1 and Theorem 2, we get that for a fixed instance, the regret only increases logarithmically with  $n$ .

## 5 EXPERIMENTS

In this section, we present a set of illustrative experiments with our algorithm (MG-EUCB) on synthetic and



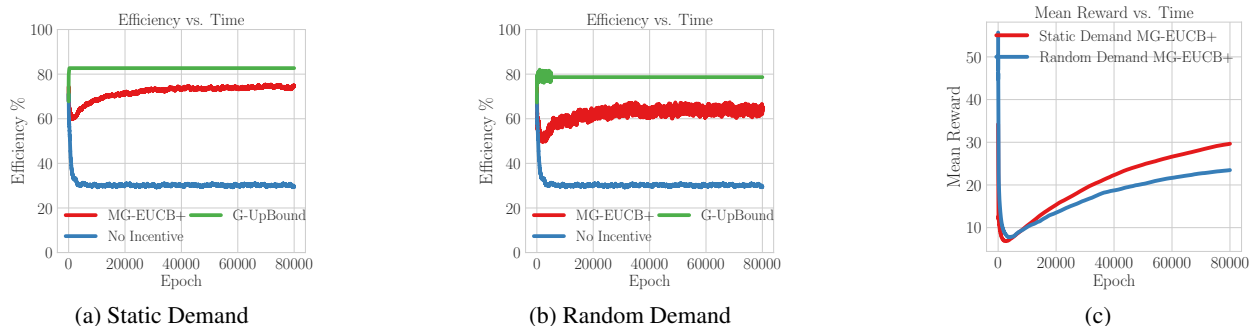


Figure 3: Bike-share Experiments: Figures 3a and 3b compare the efficiency (percentage of demand satisfied) of the bike-share system with two demand models under incentive matchings selected by MG-EUCB+ with upper and lower bounds given by the system performance when the incentives are computed via the benchmark greedy matching that uses the state information and when no incentives are offered respectively. In Figure 3c we plot the mean reward of the MG-EUCB+ algorithm with static and random demand which gives the expected number of agents who accept an incentive within each epoch.

real data. We observe much faster convergence with the greedy matching as compared to the Hungarian algorithm. Moreover, as is typical in the bandit literature (e.g., (Auer et al., 2002)), we show that a *tuned* version of our algorithm (MG-EUCB+), in which we reduce the coefficient on the  $\log(n)$  term in the UCB ‘confidence parameter’ from six to three, further improves the convergence of our algorithm. Finally we show that our algorithm can be effectively used as an incentive design scheme to improve the performance of a bike-share system.

## 5.1 SYNTHETIC EXPERIMENTS

We first highlight the failure of classical UCB approaches (C-UCB)—e.g., as in (Gai et al., 2011)—for problems with correlated reward evolution. In Figure 2a, we demonstrate that C-UCB converges almost immediately to a suboptimal solution, while this is not the case for our algorithm (MG-EUCB+). In Figure 2b, we compare MG-EUCB and MG-EUCB+ with a variant of Algorithm 2 that uses the Hungarian method (H-EUCB) for matchings. While H-EUCB does have a ‘marginally’ higher mean reward, Figure 2c reveals that the MG-EUCB and MG-EUCB+ algorithms converge much faster to the optimum solution of the greedy matching than the Hungarian alternatives.

## 5.2 BIKE-SHARE EXPERIMENTS

In this problem, we seek to incentivize participants in a bike-sharing system; our goal is to alter their intended destination in order to balance the spatial supply of available bikes appropriately and meet future user demand. We use data from the Boston-based bike-sharing service Hubway (hub) to construct the example. Formally, we

consider matching each agent  $a$  to an incentive  $i = s'_a$ , meaning the algorithm proposes that agent  $a$  travel to station  $s'_a$  as opposed to its intended destination  $s_a$  (potentially, for some monetary benefit). The agent’s state  $\theta_a$  controls the probability of accepting the incentive by means of a distance threshold parameter and a parameter of a Bernoulli distribution, both of which are drawn uniformly at random. More details on the data and problem setup can be found in Section D of the supplementary material.

Our bike-share simulations presented in Figure 3 show approximately a 40% improvement in system performance when compared to an environment without incentives and convergence towards an upper bound on system performance. Moreover, our algorithm achieves this significant performance increase while on average matching less than 1% of users in the system to an incentive.

## 6 Conclusion

We combine ideas from greedy matching, the UCB multi-armed bandit strategy, and the theory of Markov chain mixing times to propose a bandit algorithm for matching incentives to users, whose preferences are unknown a priori and evolving dynamically in time, in a resource constrained environment. For this algorithm, we derive logarithmic gap-dependent regret bounds despite the additional technical challenges of cascading sub-optimality and correlated convergence. Finally, we demonstrate the empirical performance via examples.

## Acknowledgments

This work is supported by NSF Awards CNS-1736582 and CNS-1656689. T. Fiez was also supported in part by an NDSEG Fellowship.

## References

- Hubway: Metro-boston's bikeshare program. [available online: <https://thehubway.com>].
- P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2):235–256, May 2002. doi: 10.1023/A:1013689704352.
- M. G. Azar, A. Lazaric, and E. Brunskill. Regret bounds for reinforcement learning with policy advice. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 97–112, 2013.
- K. Azuma. Weighted sums of certain dependent random variables. *Tohoku Math. J.*, 19(3):357–367, 1967. doi: 10.2748/tmj/1178243286.
- A. Badanidiyuru, R. Kleinberg, and A. Slivkins. Bandits with knapsacks. In *Proc. 54th Annual IEEE Symp. Foundations of Computer Science*, pages 207–216, 2013.
- W. Chen, Y. Wang, Y. Yuan, and Q. Wang. Combinatorial multi-armed bandit and its extension to probabilistically triggered arms. *J. Machine Learning Research*, 17:50:1–50:33, 2016. URL <http://jmlr.org/papers/v17/14-298.html>.
- J. Fill. Eigenvalue bounds on convergence to stationarity for nonreversible markov chains, with an application to the exclusion process. *Ann. Appl. Probab.*, 1(1):62–87, 1991.
- G. Folland. *Real Analysis*. Wiley, 2nd edition, 2007.
- Y. Gai, B. Krishnamachari, and M. Liu. On the combinatorial multi-armed bandit problem with markovian rewards. In *Proc. Global Communications Conf.*, pages 1–6, 2011. doi: 10.1109/GLOCOM.2011.6134244.
- M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979. ISBN 0-7167-1044-7.
- A. Ghosh and P. Hummel. Learning and incentives in user-generated content: multi-armed bandits with endogenous arms. In *Proc. of ITCS 2013*, pages 233–246, 2013.
- W. Hoeffding. Probability inequalities for sums of bounded random variables. *J. American Statistical Association*, 58(301):13–30, 1963. doi: 10.2307/2282952.
- Nicole Immorlica, Gregory Stoddard, and Vasilis Syrgkanis. Social status and badge design. In *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*, pages 473–483, 2015.
- S. Jain, B. Narayanaswamy, and Y. Narahari. A multiarmed bandit incentive mechanism for crowdsourcing demand response in smart grids. In *Proc. of AAAI 2014*, pages 721–727, 2014.
- T. Jaksch, R. Ortner, and P. Auer. Near-optimal Regret Bounds for Reinforcement Learning. *J. Machine Learning Research*, 11:1563–1600, 2010.
- H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics*, 2(1-2):83–97, 1955.
- B. Kveton, Z. Wen, A. Ashkan, H. Eydgahi, and B. Eriksson. Matroid bandits: Fast combinatorial optimization with learning. In *Proc. of UAI 2014*, pages 420–429, 2014.
- Branislav Kveton, Zheng Wen, Azin Ashkan, and Csaba Szepesvari. Tight regret bounds for stochastic combinatorial semi-bandits. In *Artificial Intelligence and Statistics*, pages 535–543, 2015.
- D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society, 2009.
- Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proc. 19th Intern. Conf. World Wide Web*, pages 661–670, 2010.
- T. Lu, D. Pál, and M. Pál. Contextual multi-armed bandits. In *Proc. of AISTATS 2010*, pages 485–492, 2010.
- E. Mazumdar, R. Dong, V. Rúbies Royo, C. Tomlin, and S. S. Sastry. A Multi-Armed Bandit Approach for Online Expert Selection in Markov Decision Processes. *arxiv:1707.05714*, 2017.
- A. Mehta and V. Mirrokni. Online ad serving: Theory and practice, 2011.
- L. J. Ratliff, S. Sekar, L. Zheng, and T. Fiez. Incentives in the dark: Multi-armed bandits for evolving users with unknown type. *arxiv*, 2018.
- Amir Sani, Alessandro Lazaric, and Rémi Munos. Risk-aversion in multi-armed bandits. In *Proc. of NIPS 2012*, pages 3284–3292, 2012.
- S. L. Scott. Multi-armed bandit experiments in the online service economy. *Applied Stochastic Models in Business and Industry*, 31(1):37–45, 2015.
- A. Singla, M. Santoni, G. Bartók, P. Mukerji, M. Meenen, and Andreas Krause. Incentivizing users for balancing bike sharing systems. In *Proc. of AAAI 2015*, pages 723–729, 2015.
- Cem Tekin and Mingyan Liu. Online algorithms for the multi-armed bandit problem with markovian rewards.

In *Communication, Control, and Computing (Allerton)*, 2010 48th Annual Allerton Conference on, pages 1675–1682. IEEE, 2010.

Cem Tekin and Mingyan Liu. Online Learning of Rested and Restless Bandits. *IEEE Transactions on Information Theory*, 58(8):5588–5611, 2012.

L. Tran-Thanh, S. Stein, A. Rogers, and N. R. Jennings. Efficient crowdsourcing of unknown experts using bounded multi-armed bandits. *Artif. Intell.*, 214: 89–111, 2014.

Zheng Wen, Branislav Kveton, and Azin Ashkan. Efficient learning in large-scale combinatorial semi-bandits. In *International Conference on Machine Learning*, pages 1113–1122, 2015.

---

# Sparse Multi-Prototype Classification

---

**Vikas K. Garg**  
CSAIL, MIT  
vgarg@csail.mit.edu

**Lin Xiao**  
Microsoft Research  
lin.xiao@microsoft.com

**Ofer Dekel**  
Microsoft Research  
oferd@microsoft.com

## Abstract

We introduce a new class of sparse multi-prototype classifiers, designed to combine the computational advantages of sparse predictors with the non-linear power of prototype-based classification techniques. This combination makes sparse multi-prototype models especially well-suited for resource constrained computational platforms, such as the IoT devices. We cast our supervised learning problem as a convex-concave saddle point problem and design a provably-fast algorithm to solve it. We complement our theoretical analysis with an empirical study that demonstrates the merits of our methodology.

## 1 INTRODUCTION

As modern machine-learned models become more accurate, they also tend to grow bigger and become more expensive to compute. Deep neural networks, massive decision tree ensembles, and other contemporary machine learning predictors can have hundreds of millions of parameters, resulting in a large memory footprint and a high computational cost. These models become especially prohibitive when the goal is to deploy them on resource impoverished platforms, such as wearable computers or IoT devices (Kumar *et al.*, 2017). Similarly, their high cost makes it difficult to build systems that need to keep track of many different models, such as those that maintain a separate model per user. Unsurprisingly, these issues have fostered a renewed interest in learning models that strike a better balance between accuracy and cost.

Efforts to develop machine learning techniques that

produce more compact models can be broadly bifurcated into two schools of thought. The first approach is to train a large and accurate model topology, and subsequently compress it using an approximation method. Some of these approximation techniques include pruning (Han *et al.*, 2016; Nan *et al.*, 2016; Luo *et al.*, 2017), low-rank matrix approximation (Sainath *et al.*, 2013; Nakkiran *et al.*, 2015), hashing (Chen *et al.*, 2015), and parameter quantization or binarization (Hubara *et al.*, 2016; Han *et al.*, 2016). Another popular technique is to use a large model to generate training data for a smaller model (Bucila *et al.*, 2006).

The second approach incorporates compression more intimately into the training objective. For example, the well-known Lasso (Tibshirani, 1996) and Elastic-Net (Zou & Hastie, 2005) algorithms use a sparsity-inducing regularization term to control the sparsity of a linear model. The resulting sparse predictor relies only on a small subset of the available features, and is therefore economical to store and make predictions. The weakness of these approaches is that linear predictors are typically not expressive enough to achieve state-of-the-art accuracy. Another common idea is to define models that are specified by a small number of prototypes, for example, by learning a Support Vector Machine (SVM) with a small support set (Dekel & Singer, 2007; Dekel *et al.*, 2008), or by finding a compressed set of reference points for a Nearest Neighbor model (Kusner *et al.*, 2014; Zhong *et al.*, 2017; Gupta *et al.*, 2017). A main drawback of these techniques is that they typically require solving highly non-convex optimization problems, which makes it difficult to guarantee their convergence and optimality. Another shortcoming of many of these approaches is that the prototypes that they learn are typically dense.

In this paper, we subscribe to the second approach mentioned above, and address the cost-accuracy

tradeoff by designing the training objective appropriately. Specifically, we introduce a class of models that we call *Sparse Multi-Prototype* (SMP) classifiers. SMP classifiers attempt to combine the sparsity benefits of linear models with the non-linear power of multi-prototype methods. Namely, each class is associated with a small set of prototypes, and each of those prototypes is sparse. But for their sparsity, SMP classifiers are reminiscent of multi-class SVMs (Weston & Watkins, 1999; Crammer & Singer, 2001), and their multi-prototype extensions (Aiolli & Sperduti, 2005).

We formulate the training procedure for SMP classifiers as a convex optimization problem. Specifically, we cast the SMP training problem as a convex-concave saddle point optimization problem and show that this formulation admits fast convergence via a primal-dual proximal point algorithm due to Chambolle and Pock (Chambolle & Pock, 2011, 2016; He *et al.*, 2017; Zhang & Xiao, 2015; Yu *et al.*, 2017). On one hand, our formulation induces sparsity by incorporating a regularization term, similar to the  $\ell_1$  term used in Lasso and Elastic-Net. On the other hand, it controls the number of prototypes using another regularization term, similar to the one used to derive convex formulations of clustering (Hocking *et al.*, 2011) and regression (Feng *et al.*, 2012). Our optimization formulation and algorithm are different from the ones used in these papers.

The rest of the paper is organized as follows. We set up the problem in Section 2. We then show, in Section 3, how our problem can be posed as a saddle point problem that admits fast and provable convergence via the Chambolle-Pock procedure. We present the results of our experiments in Section 4.

## 2 PROBLEM FORMULATION

Let  $\mathcal{Y}$  be a finite set of labels. Suppose that we are given a set of training examples  $\{(x_i, y_i)\}_{i=1}^m$ , where each  $x_i \in \mathbb{R}^n$  and  $y_i \in \mathcal{Y}$ . Without loss of generality, we assume an ordering of the training examples: examples from first class precede those in second, examples from second precede those in third, and so on.

Our goal is to learn a classifier  $c: \mathbb{R}^n \mapsto \mathcal{Y}$ . Assume (without loss of generality) that  $c$  is defined by a set of scoring functions  $\{\phi_y\}_{y \in \mathcal{Y}}$ , where the value  $\phi_y(x)$  is interpreted as the score of predicting the label  $y$  for the instance  $x$ . Using these scoring functions,

our classifier takes the form

$$c(x) = \arg \max_{y \in \mathcal{Y}} \phi_y(x).$$

The classifier  $c(x)$  correctly classifies the example  $(x, y)$  if and only if

$$\phi_y(x) - \max_{y' \neq y} \phi_{y'}(x) > 0. \quad (1)$$

We use this property to define the empirical loss

$$\sum_{i=1}^m \ell \left( \phi_{y_i}(x_i) - \max_{y' \neq y_i} \phi_{y'}(x_i) \right),$$

where  $\ell$  is a convex monotonically non-increasing loss function that upper bounds the error indicator function (for instance,  $\ell$  could be hinge-loss or log-loss). Clearly, this loss is an upper-bound on the number of multiclass classification mistakes.

If we use a linear score function for each class, i.e.,

$$\phi_y(x) = w_y \cdot x, \quad y \in \mathcal{Y},$$

where each  $w_y \in \mathbb{R}^n$  is called a *class prototype*, and  $\cdot$  denotes the inner product of two vectors, then we obtain the multi-class support vector machine (Weston & Watkins, 1999; Crammer & Singer, 2001).

In this paper, we allow multiple prototypes for each class (Aiolli & Sperduti, 2005). Suppose we have in total  $N$  prototypes  $w_1, \dots, w_N \in \mathbb{R}^n$ , and let  $J_y$  be the set of the prototype indices associated with class  $y$  for each  $y \in \mathcal{Y}$ . We let the score function for class label  $y$  be

$$\phi_y(x) = \max_{j \in J_y} w_j \cdot x.$$

Since  $\ell$  is monotonically non-increasing, we have

$$\begin{aligned} & \ell \left( \phi_{y_i}(x_i) - \max_{y' \neq y_i} \phi_{y'}(x_i) \right) \\ &= \ell \left( \max_{j \in J_{y_i}} w_j \cdot x_i - \max_{j \notin J_{y_i}} w_j \cdot x_i \right) \\ &\leq \ell \left( w_{j(i)} \cdot x_i - \max_{j \notin J_{y_i}} w_j \cdot x_i \right) \\ &= \max_{j \notin J_{y_i}} \ell \left( (w_{j(i)} - w_j) \cdot x_i \right), \end{aligned}$$

where  $j(i) \in J_{y_i}$  is any fixed assignment of prototype to the example  $(x_i, y_i)$ . The last expression above is a convex function in all the prototypes  $w_1, \dots, w_N$ , and so is the average loss function

$$\frac{1}{m} \sum_{i=1}^m \max_{j \notin J_{y_i}} \ell \left( (w_{j(i)} - w_j) \cdot x_i \right). \quad (2)$$

Note that  $j(i)$  needs to be fixed before we optimize over the prototypes, but is not required to maximize  $w_j \cdot x_i$  over  $j \in J_{y_i}$ . This relaxation helps us to obtain a convex upper bound on the loss in the general case.

Setting  $N = |\mathcal{Y}|$  and  $|J_y| = 1$  recovers the loss for the multi-class SVM, and we have  $j(i) = \arg \max_{j \in J_{y_i}} w_j \cdot x_i$ . In the other extreme case, we can let  $N = m$  and associate each training example  $(x_i, y_i)$  with a prototype  $w_i$ . In this case, we can have  $\phi_{y_i}(x_i) = w_i \cdot x_i = \arg \max_{j \in J_{y_i}} w_j \cdot x_i$ . However, this approach requires excessive amount of storage and computation, and also may cause significant overfitting.

In practice, we can cluster the training examples in each class into  $p$  groups, where  $p$  is much smaller than the number of examples in the class. Then we can have  $p$  prototypes for each class  $y \in \mathcal{Y}$ , and associate the examples in each cluster within the class with a common prototype:  $j(i) = j(i')$  if  $y_i = y_{i'}$ , and  $x_i$  and  $x_{i'}$  belong to the same cluster.

## 2.1 SMOOTHING THE LOSS

In order to leverage the fast algorithms designed for smooth convex optimization, we focus on smooth loss functions. In particular, we use the log-loss

$$\ell(\alpha) = \log(1 + \exp(-\alpha)).$$

Although this is a smooth function, the average loss function defined in (2) is non-smooth, due to the max operators in the sum. We can make the loss function smooth using the usual trick of softmax. Specifically, we can replace the function  $u(z) = \max_j \ell(z_j)$  with

$$\tilde{u}(z) = \log \left( 1 + \sum_j \exp(-z_j) \right). \quad (3)$$

As a result, the smoothed loss function is

$$f(W) = \frac{1}{m} \sum_{i=1}^m \log \left( 1 + \sum_{j \notin J_{y_i}} \exp((w_j - w_{j(i)}) \cdot x_i) \right), \quad (4)$$

where  $W \in \mathbb{R}^{N \times n}$  is a matrix formed by stacking the vectors  $w_1^T, \dots, w_N^T$  as its rows.

## 2.2 ENFORCING GROUP SPARSITY

Instead of relying on a separate clustering stage to reduce the number of prototypes, we can use a more principled approach based on convex optimization. Suppose we start with a large number of prototypes, for example, by having a separate prototype for each

training example. While minimizing the average loss function, we may add the regularization term

$$\sum_{y \in \mathcal{Y}} \sum_{\substack{j > i \\ i, j \in J_y}} \|w_i - w_j\|_\infty, \quad (5)$$

which encourages some of the prototypes in each class to merge, forming a smaller set of distinct prototypes.

We introduce some notations to simplify our presentation. Let  $W_y \in \mathbb{R}^{|J_y| \times n}$  be the matrix formed by stacking the set of prototypes  $\{w_j^T : j \in J_y\}$  as its rows. For each class  $y \in \mathcal{Y}$ , we form a  $b_y \times |J_y|$  matrix  $B_y$ , where  $b_y = \binom{|J_y|}{2}$ . Specifically, each row of  $B_y$  corresponds to a pair  $(i, j)$  such that  $i < j$  and  $i, j \in J_y$ , with value 1 at index  $i$ , -1 at index  $j$ , and 0 elsewhere. Then the penalty function in (5) can be written as

$$\sum_{y \in \mathcal{Y}} \|B_y W_y\|_{\infty, 1},$$

where the matrix norm  $\|\cdot\|_{\infty, 1}$  is defined as

$$\|U\|_{\infty, 1} = \sum_i \|U_{i, \cdot}\|_\infty = \sum_i \max_{r \in [n]} |U_{i, r}|.$$

Therefore, the regularized loss can be written as

$$f(W) + \lambda \sum_{y \in \mathcal{Y}} \|B_y W_y\|_{\infty, 1},$$

where  $\lambda > 0$  is a regularization parameter and  $f$  is the smoothed average loss defined in (4).

## 2.3 IMPOSING PROTOTYPE SPARSITY

In addition to the group sparsity aimed at having fewer prototypes, we can also induce sparsity in each prototype by adding the following regularization:

$$h_\eta(W_y) \triangleq \|W_y\|_{1, 1} + \frac{\eta}{2} \|W_y\|_F^2,$$

where  $\|\cdot\|_F$  denotes the matrix Frobenius norm and

$$\|U\|_{1, 1} = \sum_i \|U_{i, \cdot}\|_1 = \sum_i \sum_{r \in [n]} |U_{i, r}|.$$

In other words,  $h_\eta$  is an elastic-net type of regularization, where  $\eta$  is a parameter to trade off between the  $\ell_1$  and  $\ell_2$  regularizations.

In summary, we would like to solve the following sparse multi-prototype (SMP) classification problem:

$$\min_W \left\{ f(W) + \lambda \sum_{y \in \mathcal{Y}} \|B_y W_y\|_{\infty, 1} + \mu \sum_{y \in \mathcal{Y}} h_\eta(W_y) \right\}, \quad (6)$$

---

**Algorithm 1** The Chambolle-Pock (CP) Algorithm

---

**input:** parameters  $\tau, \sigma$ , and initial point  $(w^0, v^0)$   
Set  $\bar{w}^0 = w^0$   
**for**  $t = 0, 1, 2, \dots$  **do**  
     $v^{t+1} = \text{prox}_{\sigma g}(v^t + \sigma K \bar{w}^t)$   
     $w^{t+1} = \text{prox}_{\tau h}(w^t - \tau(\nabla f(w^t) + K^T v^{t+1}))$   
     $\bar{w}^{t+1} = 2w^{t+1} - w^t$

---

where  $\lambda, \mu > 0$  are regularization hyperparameters. This is a convex optimization problem. However, due to the complex structure of the regularization terms, it is not clear how to solve this minimization problem directly in an efficient manner (e.g., how to compute the proximal mapping of the group sparsity regularization). In the next section, we tackle this problem using a primal-dual first-order algorithm.

### 3 PRIMAL-DUAL ALGORITHM

Chambolle & Pock (2011, 2016) developed a class of primal-dual first-order algorithms for solving the following form of convex-concave saddle-point problems with bilinear coupling:

$$\min_{w \in \mathbb{R}^d} \max_{v \in \mathbb{R}^{d'}} f(w) + h(w) + \langle Kw, v \rangle - g^*(v), \quad (7)$$

where  $f$  is convex and differentiable, and both  $h$  and  $g^*$  are convex but may be non-differentiable. In particular,  $g^*$  can be considered as the conjugate function of some convex function  $g$ . Here  $K$  is a bilinear coupling matrix of dimension  $d' \times d$ . In addition, it is assumed that the proximal mappings of  $h$  and  $g^*$ ,

$$\begin{aligned} \text{prox}_h(w) &= \arg \min_{u \in \mathbb{R}^d} \left\{ f(u) + \frac{1}{2} \|u - w\|_2^2 \right\}, \\ \text{prox}_{g^*}(v) &= \arg \min_{z \in \mathbb{R}^{d'}} \left\{ g^*(z) + \frac{1}{2} \|z - v\|_2^2 \right\}, \end{aligned}$$

can be computed efficiently. Intuitively, the proximal map  $\text{prox}_h(w)$  looks for a point  $u$  that has a low cost  $f(u)$  and is not too far from  $w$ .

Algorithm 1 shows the CP algorithm (Chambolle & Pock, 2016) for solving the convex-concave saddle-point problem (7). Suppose  $\nabla f$  is Lipschitz continuous with Lipschitz constant  $L_f$ , i.e.,

$$\|\nabla f(u) - \nabla f(w)\|_2 \leq L_f \|u - w\|_2, \quad \forall u, w \in \mathbb{R}^d,$$

and the spectral norm of  $K$  is bounded by  $L$ , i.e.,  $\|K\| \leq L$ . Chambolle & Pock (2016) showed that this algorithm enjoys an  $O(1/t)$  convergence rate (the reduction of optimization error after  $t$  iterations) provided that the step size parameters  $\sigma$  and

$\tau$  satisfy the condition

$$\left( \frac{1}{\tau} - L_f \right) \frac{1}{\sigma} \geq L^2. \quad (8)$$

In the rest of this section, we show how to transform the SMP problem (6) into the form of (7), and how to compute the relevant proximal mappings as well as choose the step sizes.

#### 3.1 SADDLE-POINT FORMULATION

Let  $g(U) = \|U\|_{\infty,1}$ , and let  $\langle U, V \rangle$  denote the inner product between the two matrices, i.e.,  $\langle U, V \rangle = \text{Tr}(U^T V)$ . The conjugate function of  $g$  is defined as

$$g^*(V) = \max_U \{ \langle U, V \rangle - g(U) \} = \begin{cases} 0 & \text{if } \|V\|_{1,\infty} \leq 1 \\ +\infty & \text{otherwise,} \end{cases}$$

where  $\|V\|_{1,\infty} = \max_i \|V_i, \cdot\|_1 = \max_i \sum_j |V_{i,j}|$  is the dual norm of  $\|\cdot\|_{\infty,1}$ . We replace the group sparsity regularizations  $g_y(B_y W_y) = \|B_y W_y\|_{\infty,1}$  in (6) by

$$\max_{V_y} \{ \langle B_y W_y, V_y \rangle - g_y^*(V_y) \},$$

which yields the convex-concave saddle-point problem

$$\begin{aligned} \min_W \max_V \left\{ f(W) + \mu \sum_{y \in \mathcal{Y}} h_\eta(W_y) \right. \\ \left. + \lambda \sum_{y \in \mathcal{Y}} \left( \langle B_y W_y, V_y \rangle - g_y^*(V_y) \right) \right\}. \quad (9) \end{aligned}$$

Here the subscript  $y$  in  $g_y$  and  $g_y^*$  indicates that their arguments may have different dimensions; more specifically,  $V_y \in \mathbb{R}^{b_y \times n}$  with  $b_y = \binom{|J_y|}{2}$ . With some delicate vectorization of the matrix variables, we can put the formulation from (9) in the exact form of (7).

Without loss of generality, let the multi-class labels be  $\{1, 2, \dots, |\mathcal{Y}|\}$ . Denote by  $\text{vec}(A)$  the column vector formed by stacking the columns of matrix  $A$  on top of one another. By a slight abuse of notation, we define

$$\text{vec}(W) \triangleq [\text{vec}(W_1^\top)^\top \text{vec}(W_2^\top)^\top \dots \text{vec}(W_{|\mathcal{Y}|}^\top)^\top]^\top.$$

Note that  $\text{vec}(W) \in \mathbb{R}^{Nn}$ , where  $N$  is the total number of prototypes. Likewise, we form  $\text{vec}(V) \in \mathbb{R}^{bn}$ , where  $b \triangleq \sum_{y \in \mathcal{Y}} b_y$ , by concatenating the vectorizations of  $\{V_y\}$ . Let  $I_d$  and  $0_d$  be, respectively, the identity matrix and the zero matrix of order  $d$ . Let  $\mathbf{1}_d$  be a  $d$ -dimensional vector with all coordinates set to 1. We use  $A_1 \otimes A_2$  to denote the Kronecker

product of any two vectors or matrices  $A_1$  and  $A_2$ . Finally, we represent the  $k^{\text{th}}$  standard basis in  $\mathbb{R}^{|\mathcal{Y}|}$  by  $e_k$ , i.e.,  $e_k$  has coordinate  $k$  set to 1 and all the others set to 0.

With the above notations, and letting  $\tilde{w} = \text{vec}(W)$  and  $\tilde{v} = \text{vec}(V)$ , we can show that the saddle-point problem in (9) can be expressed as

$$\min_{\tilde{w} \in \mathbb{R}^{Nn}} \min_{\tilde{v} \in \mathbb{R}^{bn}} \tilde{f}(\tilde{w}) + \langle \tilde{B}\tilde{w}, \tilde{v} \rangle - \tilde{g}^*(\tilde{v}),$$

with appropriate definitions of  $\tilde{f}$ ,  $\tilde{B}$  and  $\tilde{g}^*$ . First, we have (with some tedious algebra)

$$\lambda \sum_{y \in \mathcal{Y}} \langle B_y W_y, V_y \rangle = \langle \tilde{B}\tilde{w}, \tilde{v} \rangle,$$

where

$$\tilde{B} \triangleq \left( \sum_{k=1}^{|\mathcal{Y}|} e_k e_k^\top \otimes B_k \right) \otimes \lambda I_n. \quad (10)$$

We define  $\text{abs}(z) \triangleq [|z_1|, |z_2|, \dots, |z_k|]$  for any vector  $z \in \mathbb{R}^k$ . We note that  $\sum_{y \in \mathcal{Y}} \lambda g_y^*(V_y)$  is finite (when it is 0) only if  $g_y^*(V_y) = 0$ , i.e. only if  $\|V_y\|_{1,\infty} \leq 1$ , for all  $y \in \mathcal{Y}$ . Moreover, for  $\lambda$  finite and positive, we have  $\lambda g_y^*(V_y) = g_y^*(V_y)$ . This lets us define

$$\tilde{g}^*(\tilde{v}) \triangleq g^*(C_g \text{abs}(\tilde{v})), \quad \text{where}$$

$$C_g = \sum_{k=1}^{|\mathcal{Y}|} e_k e_k^\top \otimes \mathbb{1}_n^\top$$

is a block diagonal matrix with  $|\mathcal{Y}|$  blocks each equal to  $\mathbb{1}_n^\top$ , and  $g^*(z) = 0$  if  $z_k \in [-1, 1]$  for all  $k \in \{1, \dots, |\mathcal{Y}|\}$  and  $\infty$  otherwise.

Finally, we can write the smoothed loss function  $f(W)$  defined in (4) as

$$f(W) = \tilde{f}(\tilde{w}) = \frac{1}{m} \sum_{i=1}^m \tilde{u}(A_i \tilde{w}), \quad (11)$$

where  $\tilde{u}$  is the soft-max function defined in (3), and

$$A_i = C_i (I_N \otimes x_i^\top).$$

Here  $C_i$  is a matrix with  $(N - |J_{y_i}|)$  rows and  $N$  columns. Each row of  $C_i$  corresponds to some  $j \notin J_{y_i}$ , with its  $j(i)$ th coordinate being 1,  $j$ th coordinate being  $-1$ , and all the other coordinates being 0.

### 3.2 BOUNDS ON THE LIPSCHITZ CONSTANTS

In order to choose the step sizes  $\sigma$  and  $\tau$  in the CP algorithm appropriately, we need to estimate the two parameters  $L_f$  and  $L$  that appeared in (8). First, we give an upper bound on  $L_f$ .

**Proposition 1.** *The Lipschitz constant  $L_f$  of  $\nabla \tilde{f}(\tilde{w})$  defined in (11) is bounded as*

$$L_f \leq NR^2,$$

where  $N$  is the total number of prototypes and  $R$  is an upper bound on  $\|x_i\|$  for all  $i = 1, \dots, m$ .

*Proof.* We derive the desired result by bounding the spectral norm of the Hessian matrix of  $\tilde{f}$  defined in (11). It is sufficient to consider the Hessian of each  $\tilde{u}(A_i \tilde{w})$ , which can be written as

$$H_i = A_i^\top \nabla^2 \tilde{u}(A_i \tilde{w}) A_i.$$

It is not hard to check that  $\nabla^2 \tilde{u}(A_i \tilde{w}) \preceq I$ , i.e., the matrix  $I - \nabla^2 \tilde{u}(A_i \tilde{w})$  is positive semidefinite for any  $\tilde{w}$ . Therefore we have

$$H_i \preceq A_i^\top A_i = (I_N \otimes x_i^\top)^\top C_i^\top C_i (I_N \otimes x_i^\top).$$

In terms of their spectral norm, we have

$$\|H_i\| \leq \|C_i^\top C_i\| \|I_N \otimes x_i^\top\|^2. \quad (12)$$

If  $\|x_i\|_2 \leq R$ , then we have  $\|I_N \otimes x_i^\top\| \leq R$ .

It remains to bound  $\|C_i^\top C_i\|$ , which is the same as  $\|C_i C_i^\top\|$ . By construction of  $C_i$  at the end of Section 3.1, we have

$$C_i C_i^\top = I_{d_i} + \mathbb{1}_{d_i} \mathbb{1}_{d_i}^\top$$

where  $d_i = N - |J_{y_i}|$ . Therefore we have

$$\|C_i^\top C_i\| = \|C_i C_i^\top\| = N - |J_{y_i}| + 1 \leq N.$$

Combining with (12), we conclude that  $\|H_i\| \leq NR^2$ . Finally, since  $\|\nabla^2 \tilde{f}(\tilde{w})\| \leq (1/m) \sum_{i=1}^m \|H_i\|$ , we obtain the desired result.  $\square$

Next we derive the precise value of  $L = \|\tilde{B}\|$ .

**Proposition 2.** *The singular values of the matrix  $\tilde{B}$  defined in (10) belong to the set*

$$\left\{ \lambda \sqrt{|J_1|}, \dots, \lambda \sqrt{|J_{|\mathcal{Y}|}|}, 0 \right\}.$$

Therefore, the spectral norm of  $\tilde{B}$  is

$$L = \max_{k=1, \dots, |\mathcal{Y}|} \lambda \sqrt{|J_k|}.$$



*Proof.* We first claim that  $B_k$  has two distinct singular values, viz.,  $\sqrt{|J_k|}$  (with multiplicity  $|J_k| - 1$ ) and 0 (with multiplicity 1). We invoke a characterization of the singular values in terms of the eigen decomposition of positive semidefinite matrix  $B_k^\top B_k \in \mathbb{R}^{|J_k| \times |J_k|}$  to argue for the full spectrum of  $B_k$ . Specifically, if  $\eta^2$  is an eigenvalue of the matrix  $B_k^\top B_k$ , then  $\eta$  is a singular value of  $B_k$ .

Now note that  $B_k^\top B_k$  has all the off-diagonal coordinates equal to -1 and all the diagonal coordinates equal to  $|J_k| - 1$ . This is precisely the Laplacian of the complete graph with  $|J_k|$  nodes, which is known to have the eigenvalue  $|J_k|$  with multiplicity  $|J_k| - 1$  and 0 with multiplicity one. This completes the analysis for the spectrum of  $B_k$ .

Next, we note the term within the parentheses in the definition of  $\tilde{B}$  is a block diagonal matrix with blocks  $B_1, \dots, B_{|\mathcal{Y}|}$ , and therefore, the set of the singular values of this set is simply the union of the singular values of each block, i.e., the set  $\{\sqrt{|J_1|}, \dots, \sqrt{|J_{|\mathcal{Y}|}|}, 0\}$ . Finally, the distinct singular values of  $\tilde{B}$  are  $\{\lambda\sqrt{|J_1|}, \dots, \lambda\sqrt{|J_{|\mathcal{Y}|}|}, 0\}$ , since  $\lambda$  is the unique singular value (multiplicity  $n$ ) of the matrix  $\lambda I_n$ . This follows since every singular value  $\mu_{12}$  of  $A_1 \otimes A_2$  can be expressed as the product of singular values  $\mu_1$  of  $A_1$  and  $\mu_2$  of  $A_2$ .  $\square$

We can rewrite the condition in (8) as

$$\tau \leq \frac{1}{L_f}, \quad \frac{\sigma\tau}{1 - \tau L_f} L^2 \leq 1.$$

Given the bounds for  $L_f$  and  $L$  derived in Proposition 1 and Proposition 2, we can choose the step sizes  $\tau$  and  $\sigma$  to satisfy the conditions above, which lead to  $O(1/t)$  convergence of the CP algorithm. For example, if we start with  $p$  prototypes for each class, then  $|J_k| = p$  for  $k \in \{1, \dots, |\mathcal{Y}|\}$  and  $N = p|\mathcal{Y}|$ . Therefore we have  $L_f \leq p|\mathcal{Y}|R^2$  and  $L^2 = \lambda^2 p$ , and can choose

$$\tau = \frac{1}{2pR^2|\mathcal{Y}|}, \quad \sigma \leq \frac{1}{2\lambda^2 p \tau}.$$

We note that the bound on  $L_f$  can be very loose. Some trial-and-error for tuning the step sizes is expected in practice.

### 3.3 COMPUTING PROXIMAL MAPS

We denote the sign of a real number  $p$  by  $\text{sign}(p) \in \{0, \pm 1\}$ , and the positive part  $\max(0, p)$  by  $(p)_+$ . For clarity of presentation, in this section, we use  $A(i, j)$  to denote the entry of the matrix  $A$  at the  $i$ th row

and  $j$ th column. Our next result shows that the Chambolle-Pock (CP) update for each  $W_y$ ,  $y \in \mathcal{Y}$ , can be computed via a closed form expression.

**Proposition 3.** *The CP update rule for  $W_y$  is*

$$W_y^{t+1}(i, j) = \frac{\text{sign}(U_y^t(i, j))}{\mu\tau\eta + 1} (|U_y^t(i, j)| - \mu\tau)_+,$$

where  $U_y^t \triangleq W_y^t - \tau(\nabla_y f(W^t) + \lambda B_y^\top V_y^{t+1})$ .

*Proof.* In the CP algorithm, the matrix  $W_y$  is updated as

$$W_y^{t+1} = \text{prox}_{\mu\tau h_\eta}(W_y^t - \tau(\nabla_y f(W^t) + \lambda B_y^\top V_y^{t+1})),$$

where  $\nabla_y f(W^t)$  denotes the partial gradient of  $f$  with respect to  $W_y$  at the previous iterate  $W^t$ . We can equivalently write the above proximal mapping as

$$W_y^{t+1} = \underset{Z}{\text{argmin}} \left\{ h_\eta(Z) + \frac{1}{2\mu\tau} \|Z - U_y^t\|_F^2 \right\}. \quad (13)$$

Since  $W_y^{t+1}$  is optimal for the above minimization problem, the (sub-)gradient of the corresponding objective function must vanish. Recall that

$$h_\eta(W_y) = \|W_y\|_{1,1} + (\eta/2) \|W_y\|_F^2.$$

The optimality condition for (13) means that there exists  $Z(i, j) \in \partial|W_y^{t+1}(i, j)|$  such that

$$Z(i, j) + \eta W_y^{t+1}(i, j) + \frac{1}{\mu\tau} (W_y^{t+1}(i, j) - U_y^t(i, j)) = 0.$$

When  $W_y^{t+1}(i, j) = 0$ , we have  $\partial|W_y^{t+1}(i, j)| \in [-1, 1]$ , which implies  $|U_y^t(i, j)| \leq \mu\tau$ . Otherwise, we have  $\partial Z^*(i, j) = \text{sign}(Z^*(i, j))$ , whence

$$|U_y^t(i, j)| > \mu\tau \text{ and } \text{sign}(Z^*(i, j)) = \text{sign}(U_y^t(i, j)).$$

So, we can perform soft thresholding to obtain  $Z^*$ , i.e.  $W_y^{t+1}$ , for all cases, and it turns out as claimed.  $\square$

We next derive an expression for updating  $V_y$ .

**Proposition 4.** *The CP update rule for  $V_y$  is*

$$V_y^{t+1} = \underset{\|Z\|_{1,\infty} \leq 1}{\text{arg min}} \frac{1}{2} \|Z - (V_y^t + \lambda \sigma_t B_y \bar{W}_y^t)\|_F^2.$$

*Proof.* Since the empirical loss term and the sparsity term depend only on  $W$ , the update rule for  $V_y$  is

$$V_y^{t+1} = \underset{Z}{\text{arg max}} \left\{ \lambda (\langle B_y \bar{W}_y^t, Z \rangle - g_y^*(Z)) - \frac{1}{2\sigma} \|Z - V_y^t\|_F^2 \right\}.$$

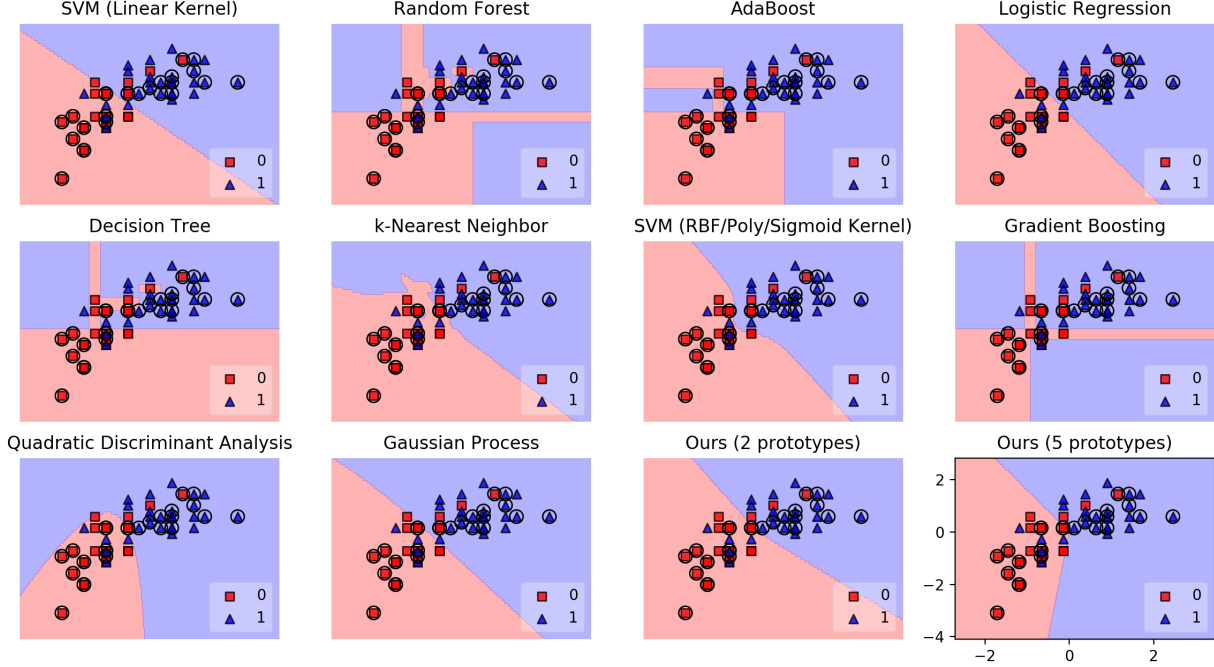


Figure 1: **Visual comparison of the decision boundary of classifiers on a run of the vineyard dataset.** The instances for two classes are shown in red and blue, with the test data, additionally, encircled.

We note that

$$\begin{aligned}
& \lambda(\langle B_y \bar{W}_y^t, Z \rangle - g_y^*(Z)) - \frac{1}{2\sigma} \|Z - V_y^t\|_F^2 \\
&= -\lambda g_y^*(Z) - \frac{1}{2\sigma} \left( \|Z - V_y^t\|_F^2 - 2\sigma \lambda \langle B_y \bar{W}_y^t, Z \rangle \right) \\
&= -\lambda g_y^*(Z) - \frac{1}{2\sigma} \|Z - (V_y^t + \sigma \lambda B_y \bar{W}_y^t)\|_F^2.
\end{aligned}$$

Since  $g_y^*(Z)$  is the indicator function of the unit norm ball  $\|Z\|_{1,\infty}$ , i.e.,  $g_y^*(Z) = 0$  if  $\|Z\|_{1,\infty} \leq 1$  and  $\infty$  otherwise, we have  $V_y^{t+1}$

$$\begin{aligned}
&= \operatorname{argmin}_Z \left\{ \lambda g_y^*(Z) + \frac{1}{2\sigma} \|Z - (V_y^t + \sigma \lambda B_y \bar{W}_y^t)\|_F^2 \right\} \\
&= \operatorname{argmin}_{\|Z\|_{1,\infty} \leq 1} \frac{1}{2} \|Z - (V_y^t + \lambda \sigma_t B_y \bar{W}_y^t)\|_F^2,
\end{aligned}$$

which is what we set out to prove.  $\square$

Using the results of Propositions 3 and 4, we arrive at the customized CP algorithm in Algorithm 2 for solving the SMP problem. For the updates on  $V_y$ , we can compute  $V_y^{t+1}$  by projecting independently the rows of  $(V_y^t + \sigma \lambda B_y \bar{W}_y^t)$  on the  $\ell_1$  unit ball. This can be done efficiently (Brucker, 1984; Pardalos & Kovoor, 1990; Duchi *et al.*, 2008).

## 4 EXPERIMENTS

We conducted several experiments<sup>1</sup> to substantiate the benefits of our framework. Our experiments are designed to convey two salient aspects of our approach. First, we attempt to position our method as an alternative to the standard classification algorithms. Second, we underscore the aptness of our approach as a viable means to obtaining highly sparse yet accurate representations. Before we dive into the details, we provide some visual intuition to differentiate our method from the other models.

We consider the following classifiers for the pictorial depiction: linear SVM (LSVM), SVM with a non-linear kernel (RSVM) selected from radial basis function, polynomial, and sigmoid via cross validation, Logistic Regression (LR), Decision Trees (DT), Random Forest (RF),  $k$ -Nearest Neighbor (kNN), Gaussian Process (GP), Gradient Boosting (GB), AdaBoost (AB), and Quadratic Discriminant Analysis (QDA). Our baselines are popular in the machine learning literature, have varying degrees of

<sup>1</sup>All our experiments used the average loss function from (2) in Algorithm 2 directly (i.e. without any smoothing), and we set  $T = 200$ .

Table 1: Comparison of test accuracy of the different classification algorithms on low dimensional OpenML datasets. The number of prototypes per class for the proposed algorithm, i.e. SMP, was set to 2.

	LSVM	RF	AB	LR	DT	kNN	RSVM	GB	QDA	GP	SMP
<b>sleuth1714</b>	.82±.03	.83±.08	.81±.14	.83±.04	.83±.06	.82±.04	.76±.03	.82±.06	.63±.13	.80±.03	<b>.87±.06</b>
<b>vis_env</b>	.66±.04	.65±.08	.66±.03	.65±.08	.62±.04	.57±.03	.69±.06	.64±.03	.62±.07	.65±.09	<b>.70±.03</b>
<b>sleuth2016</b>	.71±.04	.70±.03	.70±.05	.72±.03	.65±.07	.67±.06	.72±.04	.65±.03	.62±.07	.73±.03	<b>.74±.02</b>
<b>sleuth1605</b>	.66±.09	.70±.06	.66±.07	.70±.07	.63±.09	.66±.05	.65±.09	.65±.09	.62±.05	<b>.72±.07</b>	.70±.06
<b>sleuth2002</b>	.65±.04	.59±.04	.60±.04	.64±.04	.55±.04	.63±.07	.64±.04	.60±.05	.65±.05	.62±.04	<b>.68±.06</b>
<b>rmftsa_cto</b>	.75±.02	.70±.02	.74±.02	.75±.00	.69±.03	.71±.03	.74±.02	.72±.03	<b>.76±.02</b>	.75±.02	<b>.76±.02</b>
<b>rabe266</b>	.93±.04	.90±.04	.91±.04	.92±.04	.91±.03	.92±.03	.93±.04	.90±.04	.94±.03	<b>.95±.04</b>	.94±.04
<b>rabe265</b>	.58±.07	<b>.64±.05</b>	.60±.10	.63±.02	.54±.05	.55±.03	.62±.06	.56±.09	.61±.04	.60±.08	<b>.64±.06</b>
<b>rabe148</b>	.95±.04	.94±.02	.91±.08	.95±.04	.89±.07	.92±.05	.91±.06	.91±.08	.92±.09	.95±.02	<b>.96±.02</b>
<b>prnn_synth</b>	.82±.02	.84±.02	.84±.02	.83±.02	.83±.01	.83±.02	.83±.03	.84±.01	.83±.03	.84±.02	<b>.85±.02</b>
<b>hutsof99</b>	.74±.07	.69±.06	.65±.09	.73±.07	.60±.10	.66±.11	.66±.14	.67±.05	.59±.07	.70±.05	<b>.77±.04</b>
<b>humandev</b>	.88±.03	.86±.02	.85±.03	.89±.04	.85±.03	.87±.04	.88±.03	.86±.03	.88±.03	.88±.02	<b>.89±.04</b>
<b>elusage</b>	.90±.05	.84±.06	.84±.06	.89±.04	.84±.06	.87±.05	.89±.04	.84±.06	.90±.04	.89±.04	<b>.92±.04</b>
<b>basketball</b>	.70±.02	.65±.04	.68±.02	<b>.71±.03</b>	.71±.03	.63±.02	.66±.05	.69±.04	.69±.04	.68±.02	<b>.71±.03</b>
<b>michiganacc</b>	.72±.06	.60±.09	.71±.05	.71±.04	.67±.06	.68±.05	.71±.05	.69±.04	.72±.04	.71±.05	<b>.73±.05</b>
<b>election2000</b>	.92±.04	.91±.04	.91±.03	.92±.02	.91±.03	.92±.01	.90±.07	.92±.02	.72±.06	.92±.03	<b>.93±.02</b>
<b>cyyoung9302</b>	.80±.04	.83±.05	.83±.02	.86±.04	.75±.10	.83±.02	.84±.02	.83±.05	.83±.07	.84±.03	<b>.87±.04</b>
<b>bankruptcy</b>	.84±.07	.84±.06	.82±.04	.90±.05	.80±.05	.78±.07	.89±.06	.81±.05	.78±.15	.90±.05	<b>.95±.02</b>
<b>asbestos</b>	.65±.04	.60±.05	.60±.03	.65±.07	.60±.05	.59±.04	.56±.06	.60±.06	.65±.05	.60±.06	<b>.68±.06</b>
<b>MindCave2</b>	.70±.04	.65±.06	.63±.05	.72±.04	.66±.04	.64±.05	.67±.06	.69±.06	.65±.03	.71±.04	<b>.73±.06</b>

---

**Algorithm 2** Sparse Multi-Prototype Classification

- 1: Choose parameters  $\lambda, \eta, \mu$  and  $\tau, \sigma$
  - 2: Initialize  $W^0 = \{W_y^0\}_{y \in \mathcal{Y}}$  and  $V^0 = \{V_y^0\}_{y \in \mathcal{Y}}$
  - 3: Populate  $B = \{B_y\}_{y \in \mathcal{Y}}$
  - 4:  $\bar{W}^0 = W^0$
  - 5: **for**  $t = 0, 1, \dots, T$  **do**  
    Update  $V^{t+1}$  using  $\ell_1$  projections
  - 6:  $Z_y^t = V_y^t + \lambda \sigma B_y \bar{W}_y^t$
  - 7:  $V_y^{t+1} = \arg \min_{\|Z\|_{1,\infty} \leq 1} (1/2) \|Z - Z_y^t\|_F^2$   
    Update  $W^{t+1}$  using soft thresholding
  - 8:  $U_y^t = W_y^t - \tau \left( \nabla f_y(W_y^t, W_{\setminus y}^t) + \lambda B_y^\top V_y^{t+1} \right)$
  - 9:  $Q_y^t(i, j) = \frac{\text{sign}(U_y^t(i, j))}{\mu \tau \eta + 1}$
  - 10:  $W_y^{t+1}(i, j) = Q_y^t(i, j) (|U_y^t(i, j)| - \mu \tau)_+$   
    Update  $\bar{W}^{t+1}$
  - 11:  $\bar{W}_y^{t+1} = 2W_y^{t+1} - W_y^t$
- 

(non-)linearity, and include models from both the generative and the discriminative families.

Fig. 1 shows the decision boundaries obtained by the different classifiers on the vineyard data. The two classes are depicted in red and blue. The test data have been encircled to distinguish them from the

training instances. We observe that, on this problem, the different instantiations of our model provide a better separation of the two classes compared to the other models. For instance, both the linear classifiers, i.e. Logistic Regression and SVM with linear kernel, seem to underfit the data. On the other hand, the SMP models are able to carve out good decision boundaries. We further observe that our model trained with five prototypes performs better than that trained with two on this data. However, this phenomenon does not hold in general, since having multiple prototypes might lead to overfitting, especially in small datasets, for low values of  $\lambda$ .

#### 4.1 LOW-DIMENSIONAL REGIME (NO SPARSITY)

We found that the SMP performed very well on several low-dimensional (i.e.  $n \leq 20$ ) OpenML datasets.<sup>2</sup> We now describe the results of our experiments with these datasets. We preprocessed all the data to normalize each feature to have zero mean and unit variance. We split each dataset evenly into train and test sets using random partitioning. For SMP, we clustered the training examples in each class into  $p = 2$  clusters using  $k$ -means, and initial-

<sup>2</sup>Available at <https://www.openml.org/>

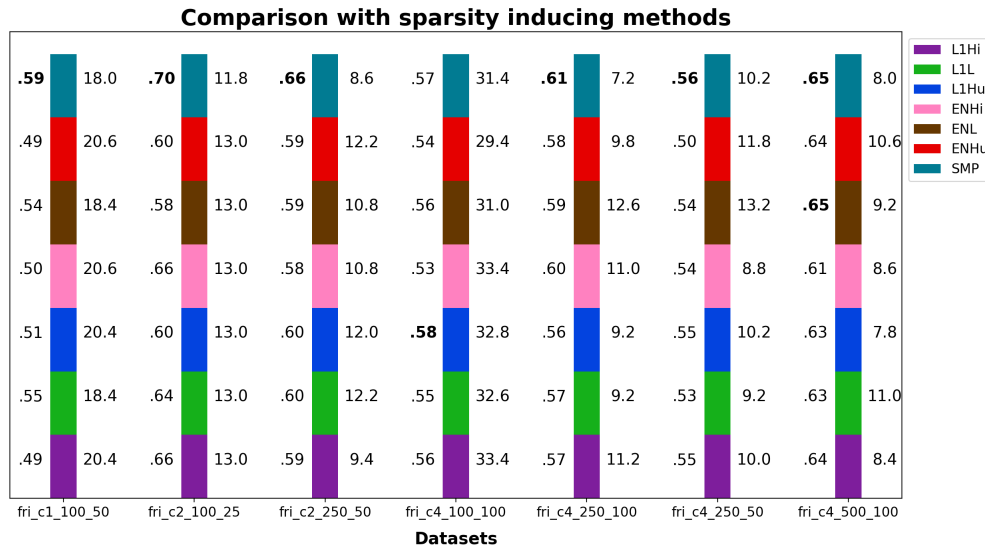


Figure 2: **Comparison on high dimensional OpenML datasets.** Each stacked bar shows two numbers: average test accuracy on the left, and total number of selected features (including multiplicities) on the right.

ized the class prototypes with the cluster centers. We followed 5-fold cross-validation (CV) for each method to obtain a good setting of hyperparameters. The details are given in the Supplementary. We report the average test accuracy and standard deviation over five random partitions per dataset.

The results on test accuracy are documented in Table 1. Evidently, SMP is seen to perform very well on many of these datasets. Note that these results should not be misconstrued as implying that SMP would generally work well with arbitrary data. Indeed, we discovered that the performance of SMP was suboptimal on many other datasets, where algorithms like RSVM and GB performed much better due to highly non-linear structure in the data.

## 4.2 HIGH-DIMENSIONAL REGIME

In this section, we explicate the results of our experiments on high dimensional data, where feature selection becomes especially critical. Our objective is to demonstrate the efficacy of SMP in recovering discriminative sparse features.

We first describe the experimental setup. We compare SMP with six baselines that induce sparsity by minimizing an  $\ell_1$ -regularized loss function (Bach *et al.*, 2012). These baselines minimize a regularized empirical loss function, namely hinge loss, log loss, or the binary-classification Huber loss, where

the regularization consisted of either  $\ell_1$  or elastic net penalty (i.e. both  $\ell_1$  and  $\ell_2$  terms). We call these six baselines as L1Hi ( $\ell_1$ , hinge), L1L ( $\ell_1$ , log), L1Hu ( $\ell_1$ , Huber), ENHi (elastic net, hinge), ENL (elastic net, log) and ENHu (elastic net, huber) respectively.

The amount of sparsity achieved by different baselines at any fixed penalty is method specific. Therefore, we first observed the sparsity obtained with SMP on each method, and then modulated the  $\ell_1$  penalty for other methods to have roughly the same number of selected features. Then, we retrained these classifier using only the selected features, using the same loss (hinge, loss, or log) and an additional  $\ell_2$  penalty. Our procedure ensured that each baseline benefited, in effect, from an *elastic net*-like regularization while having the most important features at its disposal. We followed 5-fold CV to find a good setting of hyperparameters for each method.

Our results on several high dimensional OpenML datasets are summarized in Fig. 2. The first number in the name of a dataset represents the number of instances in the dataset, while the second term represents dimensionality. In SMP, since some features might be selected in more than one prototype, for fairness of evaluation, we included the multiplicity while computing the selected feature count. These results underscore the merits in combining the power of multiple prototypes with sparse representations.

## References

- Aioli, F., & Sperduti, A. 2005. Multiclass Classification with Multi-Prototype Support Vector Machines. *Journal of Machine Learning Research (JMLR)*, **6**, 817–850.
- Bach, F., Jenatton, R., Mairal, J., & Obozinski, G. 2012. Optimization with Sparsity-Inducing Penalties. *Foundations and Trends in Machine Learning*, **4**(1), 1–106.
- Brucker, P. 1984. An  $O(n)$  algorithm for quadratic Knapsack problems. *Operations Research Letters*, **3**(3), 163–166.
- Bucila, B., Caruana, R., & Niculescu-Mizil, A. 2006. Model Compression. *Pages 535–541 of: The Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
- Chambolle, A., & Pock, T. 2011. A First-Order Primal-Dual Algorithm for Convex Problems with Applications to Imaging. *Journal of Mathematical Imaging and Vision*, **40**(1), 120–145.
- Chambolle, A., & Pock, T. 2016. On the ergodic convergence rates of a first-order primal-dual algorithm. *Math. Program.*, **159**(1-2), 253–287.
- Chen, W., Wilson, J., Tyree, S., Weinberger, K., & Chen, Y. 2015. Compressing neural networks with the hashing trick. *Pages 2285–2294 of: ICML*.
- Crammer, K., & Singer, Y. 2001. On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines. *JMLR*, **2**, 265–292.
- Dekel, O., & Singer, Y. 2007. Support vector machines on a budget. *Pages 345–352 of: NIPS*.
- Dekel, O., Shalev-Shwartz, S., & Singer, Y. 2008. The Forgetron: A kernel-based perceptron on a budget. *SIAM Journal on Computing*, **37**(5), 1342–1372.
- Duchi, J., Shalev-Shwartz, S., Singer, Y., & Chandra, T. 2008. Efficient projection onto the  $\ell_1$ -ball for learning in high dimensions. *Pages 272–279 of: ICML*.
- Feng, J., Yuan, X., Wang, Z., Xu, H., & Yan, S. 2012. Auto-Grouped Sparse Representation for Visual Analysis. **23**, 640–653.
- Gupta, C., Suggala, A. S., Goyal, A., Simhadri, H. V., Paranjape, B., Kumar, A., Goyal, S., Udupa, R., Varma, M., & Jain, P. 2017. ProtoNN: Compressed and Accurate kNN for Resource-scarce Devices. *Pages 1331–1340 of: ICML*.
- Han, S., Mao, H., & Dally, W. J. 2016. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *In: ICLR*.
- He, B., Ma, F., & Yuan, X. 2017. An Algorithmic Framework of Generalized Primal-Dual Hybrid Gradient Methods for Saddle Point Problems. *Journal of Mathematical Imaging and Vision*, **58**(2), 279–293.
- Hocking, T. D., Joulin, A., Bach, F., & Vert, J.-P. 2011. Clusterpath an algorithm for clustering using convex fusion penalties. *In: ICML*.
- Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., & Bengio, Y. 2016. Binarized Neural Networks. *Pages 4107–4115 of: NIPS*.
- Kumar, A., Goyal, S., & Varma, M. 2017. Resource-efficient Machine Learning in 2 KB RAM for the Internet of Things. *Pages 1935–1944 of: ICML*.
- Kusner, M., Tyree, S., Weinberger, K. Q., & Agrawal, K. 2014. Stochastic neighbor compression. *Pages 622–630 of: ICML*.
- Luo, J.-H., Wu, J., & Lin, W. 2017. ThiNet: A Filter Level Pruning Method for Deep Neural Network Compression. *Pages 5068–5076 of: ICCV*.
- Nakkiran, P., Alvarez, R., Prabhavalkar, R., & Parada, C. 2015. Compressing deep neural networks using a rank-constrained topology. *In: Sixteenth Annual Conference of the International Speech Communication Association*.
- Nan, F., Wang, J., & Saligrama, V. 2016. Pruning Random Forests for Prediction on a Budget. *Pages 2334–2342 of: NIPS*.
- Pardalos, P. M., & Kuvorov, N. 1990. An algorithm for a singly constrained class of quadratic programs subject to upper and lower bounds. *Mathematical Programming*, **46**, 321–328.
- Sainath, T., Kingsbury, B., Sinhwani, V., Arisoy, E., & Ramabhadran, B. 2013. Low-rank matrix factorization for deep neural network training with high-dimensional output targets. *Pages 6655–6659 of: ICASSP*.
- Tibshirani, R. 1996. Regression Shrinkage and Selection via the lasso. *Journal of the Royal Statistical Society. Series B (methodological)*, **58**(1), 267–288.
- Weston, J., & Watkins, C. 1999. Support vector machines for multi-class pattern recognition. *Pages 219–224 of: Proceedings of the 6th European Symposium on Artificial Neural Networks (ESANN)*.
- Yu, Y., Liu, S., & Pan, S. J. 2017. Communication-Efficient Distributed Primal-Dual Algorithm for Saddle Point Problems. *In: UAI*.

- Zhang, Y., & Xiao, L. 2015. Stochastic primal-dual coordinate method for regularized empirical risk minimization. *Pages 353–361 of: ICML.*
- Zhong, K., Guo, R., Kumar, S., Yan, B., Simcha, D., & Dhillon, I. 2017. Fast Classification with Binary Prototypes. *Pages 1255–1263 of: AISTATS.*
- Zou, H., & Hastie, T. 2005. Regularization and Variable Selection via the Elastic Net. *Journal of the Royal Statistical Society. Series B (methodological)*, **67**(2), 301–320.

---

# Fast Stochastic Quadrature for Approximate Maximum-Likelihood Estimation

---

**Nico Piatkowski**

Department of Computer Science  
AI Group, TU Dortmund  
44221 Dortmund, Germany

**Katharina Morik**

Department of Computer Science  
AI Group, TU Dortmund  
44221 Dortmund, Germany

## Abstract

Recent stochastic quadrature techniques for undirected graphical models rely on near-minimax degree- $k$  polynomial approximations to the model’s potential function for inferring the partition function. While providing desirable statistical guarantees, typical constructions of such approximations are themselves not amenable to efficient inference. Here, we develop a class of Monte Carlo sampling algorithms for efficiently approximating the value of the partition function, as well as the associated pseudo-marginals. More precisely, for pairwise models with  $n$  vertices and  $m$  edges, the complexity can be reduced from  $\mathcal{O}(d^k)$  to  $\mathcal{O}(k^4 + kn + m)$ , where  $d \geq 4m$  is the parameter dimension. We also consider the uses of stochastic quadrature for the problem of maximum-likelihood (ML) parameter estimation. For completely observed data, our analysis gives rise to a probabilistic bound on the log-likelihood of the model. Maximizing this bound yields an approximate ML estimate which, in analogy to the moment-matching of exact ML estimation, can be interpreted in terms of pseudo-moment-matching. We present experimental results illustrating the behavior of this approximate ML estimator.

## 1 INTRODUCTION

The major source of complexity in the course of parameter estimation for undirected graphical models is the #P-hardness of the partition function (Valiant, 1979; Bulatov and Grohe, 2004). This quantity plays a fundamental role in various contexts, including approximate inference, maximum-likelihood (ML) parameter estimation, and large deviations analysis—to mention just a

few. For a general undirected model, exact computation of this partition function is intractable; therefore, developing approximations and bounds is an important problem. The dominant approaches in this area are Markov Chain Monte Carlo (MCMC) sampling approaches (Andrieu et al., 2003) and variational inference (Wainwright and Jordan, 2008). While both directions work very well in practice, theoretical quality guarantees cannot be asserted. Some of the existing techniques indeed deliver error bounds, but the error cannot be quantified without making assumptions that go beyond the ordinary variational principle or sampling procedures.

Our recent stochastic quadrature technique (Piatkowski and Morik, 2016) for undirected graphical models relies on a near-minimax degree- $k$  polynomial approximation to the model’s potential function for inferring the partition function. While providing desirable statistical guarantees, typical constructions of such approximations are themselves not amenable to efficient inference. Here, we develop a class of Monte Carlo sampling algorithms for efficiently approximating the value of the partition function, as well as the associated pseudo-marginals.

Our contributions can be summarized as follows:

- We provide a Monte Carlo sampling procedure that reduces the complexity of the stochastic quadrature inference method from  $\mathcal{O}(d^k)$  to  $\mathcal{O}(k^4 + kn + m)$  when certain combinatorial quantities are precomputed. An empirical evaluation shows that our new method is several orders of magnitude faster than the existing approach.
- We provide the first stochastic quadrature based algorithm for marginal inference, and thus, for approximate maximum-likelihood parameter estimation. Experimental results show that approximate log-likelihood and predicted marginal probabilities are close to their exact counterparts.
- We explain how the stochastic quadrature can be ap-

plied to models with continuous random variables.

- Our results are derived from first-principles and work with any discrete and some continuous exponential family members.

## 2 NOTATION AND BACKGROUND

Let us summarize the notation and background necessary for subsequent development. The set that contains the first  $n$  strictly positive integers is denoted by  $[n] = \{1, 2, \dots, n\}$ .

**Graphical Models:** An undirected graph  $G = (V, E)$  consists of  $n = |V|$  vertices, connected via edges  $(v, w) \in E$ . For each vertex  $v \in V$ , we denote the set of adjacent vertices by  $\mathcal{N}(v)$ . A clique  $C$  is a fully-connected subset of vertices, i.e.,  $\forall v, w \in C : (v, w) \in E$ . The set of all cliques of  $G$  is denoted by  $\mathcal{C}$ . Here, any undirected graph represents the conditional independence structure of an undirected graphical model or Markov random field (MRF). To this end, we identify each vertex  $v \in V$  with a random variable  $\mathbf{X}_v$  taking values in the state space  $\mathcal{X}_v$ . The random vector  $\mathbf{X} = (\mathbf{X}_v : v \in V)$  contains the joint state of all vertices in some arbitrary but fixed order, taking values  $\mathbf{x}$  in the Cartesian product space  $\mathcal{X} = \bigotimes_{v \in V} \mathcal{X}_v$ . Moreover, we allow to access these quantities for any proper subset of variables  $S \subset V$ , i.e.,  $\mathbf{X}_S = (\mathbf{X}_v : v \in S)$ ,  $\mathbf{x}_S$ , and  $\mathcal{X}_S$ , respectively.

**Exponential Families:** Markov random fields with strictly positive density can be represented via exponential family members, which have been studied extensively during the last century, e.g. (Pitman, 1936; Hammersley and Clifford, 1971; Besag, 1975; Wainwright and Jordan, 2008). The probability density of  $\mathbf{X}$  w.r.t. some probability measure  $\mathbb{P}_\theta$  can hence be written as

$$p_\theta(\mathbf{x}) = \exp(\langle \boldsymbol{\theta}, \phi(\mathbf{x}) \rangle - A(\boldsymbol{\theta})) \quad (1)$$

where  $\boldsymbol{\theta} \in \mathbb{R}^d$  is the  $d$ -dimensional parameter vector, and  $\phi(\mathbf{x})$  is a statistic, sufficient for  $\boldsymbol{\theta}$ —it captures all properties of  $\mathbf{X}$  which are relevant for inferring  $\boldsymbol{\theta}$ , i.e.,  $\mathbb{P}(\boldsymbol{\theta} \in \Omega \mid \phi(\mathbf{x})) = \mathbb{P}(\boldsymbol{\theta} \in \Omega \mid \mathbf{x})$  for all  $\Omega \subseteq \mathbb{R}^d$ . Normalization of  $p_\theta$  is guaranteed via  $A(\boldsymbol{\theta}) = \ln Z(\boldsymbol{\theta}) = \ln \int_{\mathcal{X}} \exp(\langle \boldsymbol{\theta}, \phi(\mathbf{x}) \rangle) d\nu(\mathbf{x})$  w.r.t. some base measure  $\nu$ . Different base measures allow for either discrete or continuous random variables  $\mathbf{X}$  (Wainwright and Jordan, 2008). When  $\mathcal{X}$  is discrete, the statistic  $\phi(\mathbf{x})$ , given via  $\phi(\mathbf{x})_{C=\mathbf{y}} = \prod_{v \in V} \delta_{\{\mathbf{x}_C=\mathbf{y}\}}$  with  $\mathbf{y} \in \mathcal{X}_C$ , is always sufficient for  $\boldsymbol{\theta}$ . Here,  $\delta_{\{\text{expression}\}}$  is the indicator function that evaluates to 1 if and only if the expression is true, and 0 otherwise. Note that each dimension of  $\phi$ ,

say  $\phi(\mathbf{x})_i$ , corresponds to  $\phi(\mathbf{x})_{C=\mathbf{y}}$ . That is, we have an equivalence between indices  $i \in [d]$  and pairs of clique  $C \in \mathcal{C}$  and clique-state  $\mathbf{y} \in \mathcal{X}_C$ , in short:  $i \equiv (C, \mathbf{y})$ . Thus, we have  $d = \sum_{C \in \mathcal{C}} |\mathcal{X}_C|$  dimensions in total. This kind of sufficient statistic is also called *overcomplete*. In various applications (Ising, 1925; Sutton and McCallum, 2011), the dimensionality of the model is reduced by assuming a pairwise factorization. Only cliques of size  $\leq 2$  are considered in this case, which implies  $d \leq \sum_{v \in V} |\mathcal{X}_v| + \sum_{\{v, w\} \in E} |\mathcal{X}_v| |\mathcal{X}_w|$ .

**Quadrature:** Whenever integrating a function  $f$  is not tractable, one may resort to numerical methods in order to approximate the definite integral  $\mathcal{I}[f] = \int_l^u f(z) dz$ . A different way of performing numeric integration are general quadrature rules. There, the basic idea is to replace the integrand  $f$  by an approximation  $h \approx f$ , that admits tractable integration. It turns out, that choosing  $h = h_k$  to be a degree- $k$  Chebyshev polynomial approximation of  $f$ , delivers highly accurate results, due to the equioscillation property implied by near-minimax optimality. The general quadrature procedure can be summarized as

$$\int_l^u f(x) dx \approx \int_l^u h_k(x) dx = \sum_{i=0}^k w_i f(x_i) = \mathcal{I}_k[f]$$

where  $w_i$  are certain coefficients and  $x_i$  are certain abscissae in  $[l, u]$  (all to be determined) (Mason and Handscomb, 2002).

In general, any polynomial approximation works. It can be shown that an optimal (w.r.t. the  $l_p$ -norm) degree- $k$  polynomial approximation  $h_k$  of any function  $f$  on a fixed interval  $[l, u]$  always exists and is uniquely characterized by the equioscillation property (Mason and Handscomb, 2002). That is, the error function  $E(z) = f(z) - h_k(z)$  oscillates on  $[l, u]$  and has exactly  $k + 2$  extrema (Jr., 1966).

Due to their oscillation property, Chebyshev polynomials are an important building block in the construction and analysis of minimax optimal approximations. Chebyshev polynomials are specified by the fundamental recurrence relation

$$T_0(z) = 1, T_1(z) = z, T_k(z) = 2zT_{k-1}(z) - T_{k-2}(z).$$

They have an extraordinary large variety of convenient properties, like rapidly decreasing and individually converging coefficients (Gautschi, 1985), which make them ubiquitous in virtually any field of numerical analysis. An excellent discussion of Chebyshev polynomials in general, can be found in (Mason and Handscomb, 2002). Depending on the choice of interpolation points and different kinds of orthogonality properties, Chebyshev



polynomial based quadrature rules are termed Gauss-Chebyshev quadrature, Fejér quadrature or Clenshaw-Curtis quadrature (Clenshaw and Curtis, 1960).

Putting all together, the (deterministic) quadrature approximation to the partition function  $Z(\boldsymbol{\theta})$  is

$$\begin{aligned} Z(\boldsymbol{\theta}) &= \int_{\mathcal{X}} \exp(\langle \boldsymbol{\theta}, \phi(\mathbf{x}) \rangle) d\nu(\mathbf{x}) \\ &\approx \int_{\mathcal{X}} \text{exp}_{\zeta}^k(\langle \boldsymbol{\theta}, \phi(\mathbf{x}) \rangle) d\nu(\mathbf{x}) = \hat{Z}_{\zeta}^k(\boldsymbol{\theta}), \end{aligned} \quad (2)$$

where  $\text{exp}_{\zeta}^k$  is a degree- $k$  Chebyshev approximation to the exponential function on the interval  $[l; u]$ , and  $\zeta$  are the corresponding coefficients. Chebyshev approximations yield the best uniform approximation on  $[l; u]$ .  $\zeta$  can be approximated numerically via discrete cosine transformation or the Remez exchange algorithm (Fraser, 1965). It can be shown that the approximation error  $\varepsilon$  is bounded and exponentially small in  $k \ln k$  (Xiang et al., 2010) when  $l \leq \min_{\mathbf{x}} \langle \boldsymbol{\theta}, \phi(\mathbf{x}) \rangle$  and  $u \geq \max_{\mathbf{x}} \langle \boldsymbol{\theta}, \phi(\mathbf{x}) \rangle$ .

### 3 FAST STOCHASTIC QUADRATURE

In this section, we present the stochastic Clenshaw-Curtis quadrature that yields an  $(1 \pm \varepsilon)$ -approximation to the partition function (Piatkowski and Morik, 2016). We then develop a class of Monte Carlo algorithms designed to perform the actual estimation of  $A(\boldsymbol{\theta})$  efficiently.

**$k$ -Integrable Statistics:** Let  $\phi$  denote the  $d$ -dimensional statistic of the exponential family representation (1) of some undirected graphical model for  $\mathbf{X}$ . Of particular interest are statistics which are  $k$ -integrable—that is, the function

$$\chi_{\phi}^k(\mathbf{j}) = \int_{\mathcal{X}} \prod_{i=1}^k \phi(\mathbf{x})_{j_i} d\nu(\mathbf{x}) \quad (3)$$

admits a polynomial time computable closed-form expression for all index tuples  $\mathbf{j} \in [d]^k$ . It can be shown (Piatkowski and Morik, 2016) that overcomplete sufficient statistics of discrete Markov random fields are all ways  $k$ -integrable. In this case,

$$\chi_{\phi}^k(\mathbf{j}) = \begin{cases} \frac{|\mathcal{X}|}{|\mathcal{X}_{\cup_{i=1}^k C(j_i)}|}, & \mathbf{j} \text{ is realizable} \\ 0, & \text{otherwise.} \end{cases} \quad (4)$$

Here,  $C(j_i)$  denotes the clique that corresponds to the  $i$ -th entry of  $\mathbf{j}$ , i.e.,  $\mathbf{j}_i \equiv (C(j_i), \mathbf{y}(j_i))$ . An index tuple  $\mathbf{j}$  is not realizable, if two (or more) indices imply that the same vertex is in two (or more) different states at the same time.

Let us extend this result by showing that various sufficient statistics for continuous random variables are as well  $k$ -integrable.

**Lemma 1 (Continuous  $k$ -integrability)** *Let  $\mathbf{X}$  be an  $n$ -dimensional continuous random vector. Any statistic  $\phi(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^d$  whose coordinate-wise statistics  $\phi(\mathbf{x})_i$  are (linear transformations of)*

$$\phi(\mathbf{x})_i = \mathbf{x}_j^c, \text{ or } \phi(\mathbf{x})_i = \frac{1}{\mathbf{x}_j^c}, \text{ or } \phi(\mathbf{x})_i = \ln(\mathbf{x}_j)^c,$$

with  $c \in \mathbb{N}$ ,  $j \in [n]$ , is  $k$ -integrable for all  $k \in \mathbb{N}$ .

Details on the integration of elementary functions can be found in (Bronstein, 1990). In fact, the sufficient statistics of the Gaussian, Poisson, exponential, beta, Dirichlet, Pareto, Weibull with known shape, chi-squared, log-normal, beta, and gamma distributions, restricted to the interval  $(0; u]$ , consist only of terms of the form  $1/x^c$ ,  $x^c$ , and  $\ln(x)^c$  which implies their  $k$ -integrability. E.g., assume that  $\phi(x)_1 = x$  and  $\phi(x)_2 = \ln_2(x)^2$  with  $x \in (0; u]$ , then, for  $\mathbf{j} = (1, 2, 1)$ , we have  $\chi_{\phi}^k(\mathbf{j}) = u^3(9 \ln(u)^2 - 6 \ln(u) + 2)/(27 \ln(2)^2)$ , which is indeed a polynomial time computable closed-form.

One may extend Lemma 1 to include statistics of the form  $|x - m|^c$ , which appear in the density of the Laplace distribution. Closed-form expressions exist, but we excluded them here due to the notational clutter that arises when products of such functions are integrated.

**Stochastic Clenshaw-Curtis Quadrature (SCCQ):**

The major ingredient of the stochastic quadrature is a specific probability mass function (pmf) over index tuples  $\mathbf{j} \in [d]^i$  of length  $0 \leq i \leq k$ . For ease of notation, we assume that indices of  $(k + 1)$ -dimensional objects start at 0. Suppose  $\phi : \mathcal{X} \rightarrow \mathbb{R}_+^d$  is a non-negative,  $k$ -integrable statistic. Let  $\|\chi_{\phi}^i\|_1$  denote the 1-norm of the function  $\chi_{\phi}^i$ . Moreover, for any  $(k + 1)$ -dimensional real-valued vector  $\zeta$ , let  $|\zeta|$  denote the element-wise absolute value of  $\zeta$ , i.e.,  $\|\chi_{\phi}^i\|_1 = \sum_{j \in [d]^i} |\chi_{\phi}^i(\mathbf{j})|$ .

Let further  $(\mathbf{J}, I)$  be the discrete random variable with state space  $[d]^k \otimes ([k] \cup \{0\})$  and pmf  $\mathbb{P}_{\zeta, \phi}(\mathbf{J} = \mathbf{j}, I = i) = \mathbb{P}_{\phi}(\mathbf{J} = \mathbf{j} \mid I = i) \mathbb{P}_{\zeta, \phi}(I = i)$  with

$$\mathbb{P}_{\zeta, \phi}(I = i) = \frac{|\zeta_i| \|\chi_{\phi}^i\|_1}{\sum_{j=0}^k |\zeta_j| \|\chi_{\phi}^j\|_1} \quad (5)$$

and

$$\mathbb{P}_{\phi}(\mathbf{J} = \mathbf{j} \mid I = i) = \frac{\chi_{\phi}^i(\mathbf{j})}{\|\chi_{\phi}^i\|_1}. \quad (6)$$

We call  $\mathbb{P}_{\zeta, \phi}$  the *tuple mass* with parameter  $(\zeta, \phi)$ .

Now, we define the random variable which constitutes the core of SCCQ.

---

**Algorithm 1:** Stochastic Clenshaw-Curtis Quadrature

---

**input**  $\boldsymbol{\theta}, \zeta, k, N$

**output** Approximate partition function  $\hat{Z}_{\zeta}^{N,k}(\boldsymbol{\theta})$

- 1:  $S \leftarrow 0$
  - 2: **for**  $l = 1$  to  $N$  **do**
  - 3:    $(\mathbf{j}, i) \sim \mathbb{P}_{\zeta, \phi}$
  - 4:    $S \leftarrow S + \hat{Z}_{\mathbf{j}, i}^k(\boldsymbol{\theta})$
  - 5: **end for**
  - 6:  $\hat{Z}_{\zeta}^{N,k}(\boldsymbol{\theta}) \leftarrow \frac{1}{N} S$
- 

**Definition 1 (1-SCCQ)** Let  $k \in \mathbb{N}$ ,  $\boldsymbol{\theta} \in \mathbb{R}^d$ , and let  $\mathbf{J}$  be a random index tuple of random length  $I$ , both having joint tuple mass  $\mathbb{P}_{\zeta, \phi}$ . The random variable

$$\hat{Z}_{\mathbf{J}, I}^k(\boldsymbol{\theta}) = \tau \operatorname{sgn}(\zeta_I) \prod_{r=0}^I \boldsymbol{\theta}_{\mathbf{j}_r}$$

with  $\tau = \sum_{j=0}^k |\zeta_j| \|\chi_{\phi}^j\|_1$  is called 1-SCCQ.

Surprisingly, this random variable is closely related to the quadrature approximation to  $Z(\boldsymbol{\theta})$  from equation (2).

**Theorem 1 (Unbiasedness of SCCQ)** Let  $\zeta$  be the coefficient vector  $\zeta$  of a degree- $k$  polynomial approximation to  $\exp$  over some arbitrary but fixed interval  $[l; u]$ , and let  $\phi$  be a non-negative and  $k$ -integrable statistic. The random variable  $\hat{Z}_{\mathbf{J}, I}^k(\boldsymbol{\theta})$  is an unbiased estimator for  $\hat{Z}_{\zeta}^k(\boldsymbol{\theta}) = \int_{\mathcal{X}} \exp_{\zeta}^k(\langle \boldsymbol{\theta}, \phi(\mathbf{x}) \rangle) d\nu(\mathbf{x})$ .

**Proof.** Using equations (3), (5), and (6), as well as Definition 1, it follows that

$$\begin{aligned} & \mathbb{E} \left[ \hat{Z}_{\mathbf{J}, I}^k(\boldsymbol{\theta}) \right] \\ &= \sum_{i=0}^k \sum_{\mathbf{j} \in [d]^k} \mathbb{P}_{\zeta, \phi}(\mathbf{J} = \mathbf{j}, I = i) \tau \operatorname{sgn}(\zeta_i) \prod_{r=0}^i \boldsymbol{\theta}_{\mathbf{j}_r} \\ &= \sum_{i=0}^k \zeta_i \sum_{\mathbf{j} \in [d]^i} \prod_{r=0}^i \boldsymbol{\theta}_{\mathbf{j}_r} \int_{\mathcal{X}} \prod_{r=0}^i \phi(\mathbf{x})_{\mathbf{j}_r} d\nu(\mathbf{x}) \\ &= \int_{\mathcal{X}} \sum_{i=0}^k \zeta_i \langle \boldsymbol{\theta}, \phi(\mathbf{x}) \rangle^i d\nu(\mathbf{x}) = \hat{Z}_{\zeta}^k(\boldsymbol{\theta}), \end{aligned}$$

where the last line stems from the fact that

$$\langle \boldsymbol{\theta}, \phi(\mathbf{x}) \rangle^i = \sum_{j_1=1}^d \sum_{j_2=1}^d \cdots \sum_{j_i=1}^d \prod_{l=0}^i \boldsymbol{\theta}_{\mathbf{j}_l} \prod_{r=0}^i \phi(\mathbf{x})_{\mathbf{j}_r} .$$

■

Based on the theorem, we devise a Monte Carlo procedure, called  $N$ -SCCQ or simply SCCQ, shown in Algorithm 1. By combining the error  $\varepsilon$  that is introduced by

the polynomial approximation with the error that is introduced by the sampling procedure, an overall error bound can be derived<sup>1</sup>.

**Theorem 2 (SCCQ Error Bound)** Let  $\zeta$  be the coefficient vector of a degree- $k$  Chebyshev approximation to  $\exp$  on  $[l; u] = [-\|\boldsymbol{\theta}\|_1; +\|\boldsymbol{\theta}\|_1]$  with worst-case error  $\varepsilon$ . Let  $\hat{Z}_{\zeta}^{N,k}(\boldsymbol{\theta})$  be the output of Algorithm 1. Furthermore, let  $\delta \in (0, 1]$ ,  $\epsilon > 0$ ,  $N = (\ln 2/\delta)\tau^2 2\|\boldsymbol{\theta}\|_{\infty}^{2k'} \varepsilon^{-2} |\mathcal{X}|^{-2}$ , with  $(k-1)k! \geq 8 \exp(2\|\boldsymbol{\theta}\|_1)/(\pi\epsilon)$ , and  $k' = 1$  if  $\|\boldsymbol{\theta}\|_{\infty} < 1$  or otherwise  $k' = k$ . Then,

$$\mathbb{P}[|\hat{Z}_{\zeta}^{N,k}(\boldsymbol{\theta}) - Z(\boldsymbol{\theta})| < \epsilon Z(\boldsymbol{\theta})] \geq 1 - \delta .$$

### 3.1 COMPUTATIONAL COMPLEXITY

While SCCQ enjoys a bounded error and an apparently simple algorithm, the actual sampling of index tuples from  $\mathbb{P}_{\zeta, \phi}$  (line 3 in Algorithm 1) and the computation of  $\hat{Z}_{\mathbf{j}, I}^k(\boldsymbol{\theta})$  (line 4 in Algorithm 1) turn out to be computationally hard. Computing  $\hat{Z}_{\mathbf{j}, I}^k(\boldsymbol{\theta})$  requires the  $\|\chi_{\phi}^i\|_1$  values. In (Piatkowski and Morik, 2016), the authors assume that the values of  $\|\chi_{\phi}^i\|_1$  for all  $0 \leq i \leq k$  are pre-computed, which requires  $\mathcal{O}(d^k)$  additions. While rather small polynomial degrees ( $k \approx 8$ ) suffice to achieve reasonable results, the overcomplete dimension  $d$  of a  $10 \times 10$  binary Ising grid model is 720. Hence, at least  $d^k = 720^8 > 2^{75}$  additions are required to compute  $\|\chi_{\phi}^i\|_1$ . In our initial work on SCCQ (Piatkowski and Morik, 2016), rejection sampling was used to generate the samples from  $\mathbb{P}_{\zeta, \phi}$  with a uniform proposal  $\mathbb{Q}$  on  $[d]^k \otimes ([k] \cup \{0\})$ . Since the ratio  $\mathbb{P}_{\zeta, \phi}(\mathbf{j}, i)/\mathbb{Q}(\mathbf{j}, i) = (k+1)d^k \mathbb{P}_{\zeta, \phi}(\mathbf{j}, i)$  is large, one shall expect that many samples will be rejected.

### 3.2 NORMALIZING THE TUPLE MASS

To alleviate the complexity issues of SCCQ, we now present a closed-form expression for  $\|\chi_{\phi}^i\|_1$ . Our result relies on the closed-form of  $k$ -integrable statistics, which is given by equation (4) for discrete state space models. We restrict ourselves to discrete models, since a general closed-form that covers all continuous state space models does not exist. However, the general methodology can be transferred to the continuous case.

The forthcoming results make heavy use of equivalence classes of index tuples  $\mathbf{j} \in [d]^i$  and their cardinalities. In

<sup>1</sup>An earlier result can be found in (Piatkowski and Morik, 2016). There, the bound on the error of the polynomial approximation uses an inequality which is originally designed for complex-valued functions. Here, we apply a recent inequality due to Trefethen (Trefethen, 2008). Both results are qualitatively equivalent w.r.t.  $N$  and  $k$ . Nevertheless, the new proof is simplified.

this context, it is important to recall that any index  $j \in [d]$  corresponds to a pair of clique and state:  $i \equiv (C, \mathbf{y})$ . Consequently, a tuple of indices corresponds to a tuple of cliques and states.

**Definition 2 (Sub-Alphabets)** Let  $\mathcal{A}$  be some set of objects or symbols— $\mathcal{A}$  is an alphabet—and let  $\mathcal{P}(\mathcal{A})$  be its power set. The set  $\mathcal{P}(\mathcal{A}, n) \subseteq \mathcal{P}(\mathcal{A})$  contains all subsets of  $\mathcal{A}$  with at most  $n$  elements, i.e.,

$$\mathcal{P}(\mathcal{A}, n) = \{S \in \mathcal{P}(\mathcal{A}) \mid |S| \leq n\}.$$

The size of  $\mathcal{P}(\mathcal{A}, n)$  is thus

$$|\mathcal{P}(\mathcal{A}, n)| = \sum_{i=1}^n \binom{|\mathcal{A}|}{i}.$$

**Definition 3 (Tuple Classes)** Let  $i \in \mathbb{N}$ , and denote the clique tuple that corresponds to an index tuple  $\mathbf{j} \in [d]^i$  by  $\mathcal{C}(\mathbf{j}) \in \mathcal{C}^i$ . Two or more index tuples  $\mathbf{j}, \mathbf{j}'$  may correspond to the same clique tuple, i.e.,  $\mathcal{C}(\mathbf{j}) = \mathcal{C}(\mathbf{j}')$ . The equivalence class of all index tuples that correspond to the same clique tuple is denoted by

$$[\mathbf{j}] = \{\mathbf{j}' \in [d]^i \mid \mathcal{C}(\mathbf{j}) = \mathcal{C}(\mathbf{j}')\}.$$

Similarly, two or more clique tuples  $\mathcal{C}, \mathcal{C}'$  may correspond to the same set of cliques. The equivalence class of clique tuples that correspond to the same set of cliques is denoted by

$$[\mathcal{C}] = \left\{ \mathcal{C}' \in \mathcal{C}^i \mid \bigcup_{c \in \mathcal{C}} \{c\} = \bigcup_{c' \in \mathcal{C}'} \{c'\} \right\}.$$

Combining both, the equivalence class of all index tuples, whose corresponding clique tuples are in the same equivalence class, is denoted by

$$[\mathbf{j}]^* = \{\mathbf{j}' \in [d]^i \mid \mathcal{C}(\mathbf{j}') \in [\mathcal{C}(\mathbf{j})]\}.$$

Note that all members of a specific clique tuple equivalence class  $[\mathcal{C}]$  are determined by a unique set of cliques which come from the alphabet  $\mathcal{C}$ . Hence, we identify each class  $[\mathcal{C}]$  with this unique set of cliques and treat each  $[\mathcal{C}]$  as an element of  $\mathcal{P}(\mathcal{C}, i)$ . Moreover, there are  $|\mathcal{P}(\mathcal{C}, i)|$  distinct size- $i$  clique tuple equivalence classes.

In the remainder, it will be important to know how large these equivalence classes are.

**Lemma 2 (Counting Tuples)** Let  $i, j \in \mathbb{N}$ ,  $\mathbf{j} \in [d]^j$ ,  $\mathcal{C} \in \mathcal{C}^i$ , and consider the equivalence classes defined above. Then,

$$\begin{aligned} |[\mathbf{j}]| &= \prod_{l=1}^i |\mathcal{X}_{\mathcal{C}(\mathbf{j})_l}|, & |[\mathcal{C}]| &= h(\mathcal{C})! \binom{i}{h(\mathcal{C})}, \\ |[\mathbf{j}]^*| &= |[\mathcal{C}(\mathbf{j})]| |[\mathbf{j}]| \end{aligned}$$

where  $h(\mathcal{C})$  is the number of distinct cliques which appear in the tuple  $\mathcal{C}$ ,  $n!$  is the factorial, and  $\{n \ k\}^\top$  is the Stirling number of second kind.

It will be helpful to define equivalence classes of index tuples w.r.t. some  $k$ -integrable statistics. Here, equivalence w.r.t.  $\chi_\phi^i$  is established by the value that each member of an equivalence class contributes to  $\|\chi_\phi^i\|_1$ .

**Definition 4 (Tuple Classes and  $k$ -integrability)** Let  $\phi$  be a  $k$ -integrable statistic,  $i \in \mathbb{N}$ , and  $\mathbf{j} \in [d]^i$ . The equivalence class of all index tuples which correspond to the same clique tuple and have non-zero  $\chi_\phi^i$ -value is denoted by

$$[\mathbf{j}]_\phi = \{\mathbf{j}' \in [d]^i \mid \mathbf{j}' \in [\mathbf{j}] \wedge \chi_\phi^i(\mathbf{j}') \neq 0\}.$$

The corresponding extension to equivalence classes of clique tuples, is denoted by

$$[\mathbf{j}]_\phi^* = \{\mathbf{j}' \in [d]^i \mid \mathbf{j}' \in [\mathbf{j}]^* \wedge \chi_\phi^i(\mathbf{j}') \neq 0\}.$$

Up to now, we made no use of the fact that our state space is discrete. The above definitions and lemmas are valid for any exponential family model with positive  $k$ -integrable statistic. However, the proof of the next lemma makes use of equation (4). In order to extend our results to continuous random variables, one has to invoke Lemma 1 to derive a closed-form for  $\chi_\phi^i$ .

**Lemma 3 (Counting Realizable Tuples)** Suppose  $\phi$  is the binary, overcomplete sufficient statistic of discrete MRFs. Then,

$$|[\mathbf{j}]_\phi| = |\mathcal{X}_{\mathcal{C}(\mathbf{j})}|, \quad \text{and} \quad |[\mathbf{j}]_\phi^*| = |[\mathcal{C}(\mathbf{j})]| |[\mathbf{j}]_\phi|,$$

with  $\mathcal{X}_{\mathcal{C}(\mathbf{j})} = \mathcal{X}_{\cup_{l=1}^i \mathcal{C}(\mathbf{j})_l}$  and  $\mathcal{C}(\mathbf{j}) \in \mathcal{C}^i$ .

Now, we have gathered all terms and definitions to devise an improved procedure for the normalization of the index tuple mass.

**Theorem 3 (Tuple Mass Normalization)** Suppose  $\phi$  is the binary, overcomplete sufficient statistic of discrete MRFs. The conditional index tuple mass  $\mathbb{P}_{\zeta, \phi}(\mathbf{J} = \mathbf{j} \mid I = i)$  (equation (6)) can be normalized in  $\mathcal{O}(1)$  steps. More precisely,

$$\|\chi_\phi^i\|_1 = |\mathcal{X}| \sum_{l=0}^i \binom{i}{l} \binom{|\mathcal{C}|}{l} l! = |\mathcal{X}| |\mathcal{C}|^i. \quad (7)$$

The complexity  $\mathcal{O}(1)$  provided in the theorem is an overwhelming improvement, compared to the naive summation, i.e.,  $\mathcal{O}(d^k)$ . Since we need the normalization  $\|\chi_\phi^i\|_1$  for all  $1 \leq i \leq k$  tuple lengths,  $\hat{Z}_{\mathbf{J}, I}^k(\boldsymbol{\theta})$  can be computed in  $\mathcal{O}(k)$  steps when a pair  $(\mathbf{j}, i)$  is given.

---

**Algorithm 2:** Fast Index Tuple Sampler
 

---

**input** Tuple length  $i$ 
**output** Sample  $\mathbf{j} \mid I = i$  from  $\mathbb{P}_{\zeta, \phi}$ 

- 1:  $l \sim \mathbb{P}(l \mid i)$  // See Theorem 4
  - 2:  $a \sim \mathbb{U}(1; \mathbf{binom}(|\mathcal{C}|, l))$
  - 3:  $b \sim \mathbb{U}(1; \mathbf{Stirling2}(i, l) \times \mathbf{factorial}(l))$
  - 4:  $[\mathcal{C}] \leftarrow$  compute  $a$ -th  $l$ -combination of  $\{1, 2, \dots, |\mathcal{C}|\}$  // via (Buckles and Lybanon, 1977)
  - 5:  $\mathcal{C} \leftarrow$  compute  $b$ -th composition of  $\{1, 2, \dots, i\}$  with  $l$  subsets // via (Ehrlich, 1973)
  - 6:  $S \leftarrow \bigcup_{h=1}^i \mathcal{C}_h$
  - 7:  $c \sim \mathbb{U}(1; \prod_{v \in S} |\mathcal{X}_v|)$
  - 8:  $\mathbf{y} \leftarrow$  compute  $c$ -th joint state of all vertices in  $S$
  - 9: **return**  $\mathbf{j}$  that corresponds to  $\mathcal{C} = \mathbf{y}$
- 

### 3.3 FAST INDEX TUPLE SAMPLER

Based on the insights that we gained so far, we derive a direct sampling scheme for index tuples that circumvents any rejection step (Algorithm 2).

Given our results from the last subsection, drawing a random tuple length from  $\mathbb{P}_{\zeta}(I)$  can be done efficiently—it is a draw from a categorical distribution with state space size  $k$  (which is rather small). Sampling from the tuple mass  $\mathbb{P}_{\zeta, \phi}(\mathbf{J} = \mathbf{j} \mid I = i)$  can be more involved, which motivates the derivation of a specialized sampling scheme. Our algorithm is motivated by inversion sampling: For any fixed  $i$ , inversion sampling of  $\mathbf{j}$  then consists of drawing a uniform random number  $u$  in  $(0; 1)$ , and finding the smallest  $L \in \mathbb{N}$ , such that the sum of the first  $L$  tuple masses exceeds  $u$ . The  $L$ -th tuple is then the sample. The worst-case runtime complexity is then  $\mathcal{O}(d^k)$  per sample, which can be prohibitively expensive whenever the dimension  $d$  of the model is large. Based on the equivalence classes that we exploited already for the normalization of the tuple mass, we derive a factorization of  $\mathbb{P}_{\zeta, \phi}(\mathbf{J} = \mathbf{j} \mid I = i)$  which in turn implies an efficient stagewise sampling procedure.

To this end, let  $\prec$  be an any arbitrary but fixed strict total ordering on the equivalence classes of clique tuples. I.e.,  $\forall \mathbf{A}, \mathbf{B} \in \mathcal{P}(\mathcal{C}, i)$  with  $\mathbf{A} \neq \mathbf{B}$ , either  $[\mathbf{A}] \prec [\mathbf{B}]$  or  $[\mathbf{B}] \prec [\mathbf{A}]$ —by definition, each element of  $\mathcal{P}(\mathcal{C}, i)$  corresponds to a unique equivalence class. This order induces an order on clique tuples and index tuples, i.e.,  $\mathbf{j}, \mathbf{j}' \in [d]^i$ ,  $\mathbf{j} \leq \mathbf{j}' \Leftrightarrow [\mathcal{C}(\mathbf{j})] \preceq [\mathcal{C}(\mathbf{j}')]$ . Within each equivalence class, we assume that tuples are ordered lexicographically.

**Theorem 4 (Tuple Mass Factorization)** *Suppose that  $\phi$  is the binary, overcomplete sufficient statistic of a dis-*

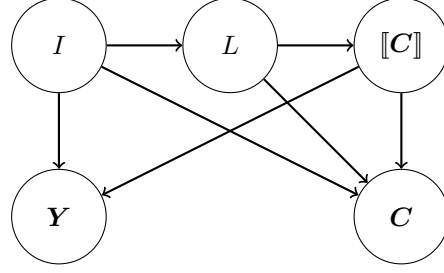


Figure 1: Directed graphical model for the factorization of the tuple mass  $\mathbb{P}_{\zeta, \phi}(\mathbf{J} = \mathbf{j}, I = i)$ . Any index tuple  $\mathbf{j}$  can be identified with some pair  $(\mathcal{C}, \mathbf{y})$  of clique tuple and state tuple.

crete state MRF. The tuple mass of any  $\mathbf{j}$  factorizes:

$$\begin{aligned} \mathbb{P}_{\zeta, \phi}(\mathbf{J} = \mathbf{j} \mid I = i) \\ = \mathbb{P}(\mathcal{C} \mid [\mathcal{C}], l, i) \mathbb{P}(\mathbf{y} \mid [\mathcal{C}], i) \mathbb{P}([\mathcal{C}] \mid l) \mathbb{P}(l \mid i) \end{aligned}$$

with

$$\begin{aligned} \mathbb{P}(l \mid i) &= |\mathcal{C}|^{-i} \binom{i}{l} \binom{|\mathcal{C}|}{l} l! \\ \mathbb{P}([\mathcal{C}] \mid l) &= \frac{1}{\binom{|\mathcal{C}|}{l}} \mathbb{P}(\mathcal{C} \mid [\mathcal{C}], l, i) = \frac{1}{\binom{i}{l} l!} \\ \mathbb{P}(\mathbf{y} \mid [\mathcal{C}], i) &= \begin{cases} \frac{1}{|\mathcal{X}_{[\mathcal{C}]}|} & , \mathbf{y} \in \mathcal{X}_{[\mathcal{C}]} \\ 0 & , \text{otherwise,} \end{cases} \end{aligned}$$

where  $l$  denotes the number of distinct cliques in the clique tuple  $\mathcal{C}$ ,  $[\mathcal{C}]$  denotes the equivalence class that contains  $\mathcal{C}$ , and  $\mathbf{y}$  is the joint state of all cliques in the tuple  $\mathcal{C}$ .

While the proof is rather simple, it is not obvious how to come up with this factorization. The idea is to first draw the equivalence class  $[\mathcal{C}]$ , then a uniform member  $\mathcal{C}$  of this class, then a uniform joint state  $\mathbf{y}$  of all cliques in  $\mathcal{C}$ . Notice that the sampling steps for  $[\mathcal{C}]$ ,  $\mathcal{C}$  and  $\mathbf{y}$  are uniform, while the probability mass of the number  $l$  of distinct cliques that will appear in the tuple is a function of  $l$ . Let us now investigate the complexity of our new method.

**Theorem 5 (Complexity of Tuple Sampling)**

*Algorithm 2 samples an index tuple  $\mathbf{j}$  of given length  $i$  from  $\mathbb{P}_{\zeta, \phi}$  in*

$$\mathcal{O}(k^4 + kn + |\mathcal{C}| + \{i l\}^\top + l!)$$

*steps. When permutations and partitions are precomputed, the runtime reduces to*

$$\mathcal{O}(k^4 + kn + |\mathcal{C}|)$$

per sample. Here,  $k$  is the polynomial degree,  $l \leq i$  is the number of distinct cliques in the generated tuple, and  $n = |V|$ .

Thus, we found a Monte Carlo algorithm to sample from  $\mathbb{P}_{\zeta, \phi}$  without any rejection step. Since the algorithm does not use a Markov chain, the generated samples are truly independent. Any number of samples can thus be generated in parallel. Because no data has to be exchanged, the overall runtime scales linearly with the number of processors. This is a superior property compared to MCMC methods, where sampling cannot be parallelized and consecutive samples are not independent. Moreover, the theorem tells us how the complexity of stochastic quadrature is related to the graphical structure and the polynomial degree. The runtime is independent of the parameter dimension  $d$  and the state space sizes. In contrast, the runtime of loopy belief propagation (Pearl, 1988; Kschischang et al., 2001) and similar variational techniques (like TRW-BP (Wainwright et al., 2003)) is at least quadratic in the vertex state space sizes.

## 4 APPROXIMATE ML ESTIMATION

An important feature of maximum-likelihood parameter estimation is that the solution is specified by moment-matching. To illustrate this notion, suppose that we are given an i.i.d. data set  $\mathcal{D} = \{\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^N\}$  from some unknown measure  $\mathbb{P}_{\theta^*}$ . By using an exponential family model (which is exact whenever the state space  $\mathcal{X}$  is discrete), the log-likelihood of  $\theta$  on  $\mathcal{D}$  is given by:

$$\ell(\theta) = \frac{1}{N} \sum_{i=1}^N \ln \mathbb{P}_{\theta}(\mathbf{x}^i) = \langle \theta, \tilde{\mu} \rangle - A(\theta)$$

with  $\tilde{\mu} = (1/N) \sum_{i=1}^N \phi(\mathbf{x}^i)$ . Taking the derivatives of  $\ell$  w.r.t. some  $\theta_i$ , we find that  $(1/N) \sum_{i=1}^N \phi(\mathbf{x}^i) = \mathbb{E}_{\theta}[\phi(\mathbf{X})_i]$  at any critical point  $\theta$  where  $\mathbb{E}_{\theta}$  denotes the expectation under  $\mathbb{P}_{\theta}$ . That is, the maximum-likelihood solution has its moments matched to the empirical average  $\tilde{\mu}$ . In this section, we show how SCCQ can be used to develop a method for approximate ML estimation that, in analogy to this exact moment-matching, performs a type of pseudo-moment matching. To this end, a means of computing  $\nabla A(\theta) = \nabla \ln Z(\theta) = \mathbb{E}_{\theta}[\phi(\mathbf{X})]$  via SCCQ is required. Recalling that  $i \equiv (C, \mathbf{y})$  and that  $\phi(\mathbf{y})$  is binary in discrete models reveals that  $\mathbb{E}_{\theta}[\phi(\mathbf{X})_i] = \mathbb{P}_{\theta}(\phi(\mathbf{X})_i = 1) = \mathbb{P}_{\theta}(\mathbf{X}_C = \mathbf{y})$ . Since  $\mathbb{P}_{\theta}(\mathbf{X}_C = \mathbf{y})$  is the marginal probability mass of the event  $\{C = \mathbf{y}\}$ , the problem of computing  $\mathbb{E}_{\theta}[\phi(\mathbf{X})_i]$  is also called *marginal inference*.

## 4.1 MARGINAL INFERENCE

For any subset  $U \subseteq V$  of variables, and any joint state  $\mathbf{x}_U$ , the marginal density is defined by

$$\begin{aligned} \mathbb{P}_{\theta}(\mathbf{X}_U = \mathbf{x}_U) &= \int_{\mathcal{X}_{V \setminus U}} \mathbb{P}_{\theta}(\mathbf{x}_U, \mathbf{x}_{V \setminus U}) \, d\nu(\mathbf{x}) \\ &= \frac{1}{Z(\theta)} \int_{\mathcal{X}_{V \setminus U}} \exp(\langle \theta, \phi(\mathbf{x}) \rangle) \, d\nu(\mathbf{x}), \end{aligned}$$

with  $\mathbf{x} = (\mathbf{x}_U, \mathbf{x}_{V \setminus U})$ . Especially the last integral is reminiscent of the partition function. In fact, it can be interpreted as the partition function of another model with state space  $\mathcal{X}_{V \setminus U}$ . It is this sum that will be approximated via SCCQ to estimate the marginal. To formalize this idea, we provide adjusted definitions of the SCCQ core concepts. First, we adapt the notion of  $k$ -integrability to marginal densities. In accordance to equation (3), we call  $\phi$  marginally  $k$ -integrable, if

$$\chi_{\phi, U}^k(\mathbf{j}, \mathbf{x}_U) = \int_{\mathcal{X}_{V \setminus U}} \prod_{i=1}^k \phi(\mathbf{x}_U, \mathbf{x}_{V \setminus U})_{\mathbf{j}_i} \, d\nu(\mathbf{x}_{V \setminus U})$$

admits a polynomial time computable closed-form expression for all  $\mathbf{j} \in [d]^k$ , for all  $U \subseteq V$ , and for all  $\mathbf{x}_U \in \mathcal{X}_{V \setminus U}$ . The difference to ordinary  $k$ -integrability is merely symbolical. In fact, all  $k$ -integrable statistics that are mentioned in this paper are also marginally  $k$ -integrable. Moreover, marginally  $k$ -integrable statistics give rise to the marginal tuple mass  $\mathbb{P}_{\zeta, \phi}(\mathbf{J} = \mathbf{j}, \mathbf{X}_U = \mathbf{x}_U, I = i)$  in the same way how the ordinary tuple mass from equation (6) arises from ordinary  $k$ -integrability. Moreover, the marginal tuple mass factorizes.

### Corollary 1 (Marginal Tuple Mass Factorization)

Suppose that  $\phi$  is the binary, overcomplete sufficient statistic. The marginal tuple mass factorizes:

$$\begin{aligned} &\mathbb{P}_{\zeta, \phi}(\mathbf{J} = \mathbf{j}, I = i, \mathbf{X}_U = \mathbf{x}_U) \\ &= \mathbb{P}(\mathbf{C} \mid \llbracket \mathbf{C} \rrbracket, l, i) \mathbb{P}(\mathbf{y}, \mathbf{x}_U \mid \llbracket \mathbf{C} \rrbracket, i) \mathbb{P}(\llbracket \mathbf{C} \rrbracket \mid l) \mathbb{P}(l \mid i) p(i) \end{aligned}$$

where  $\mathbb{P}(l \mid i)$ ,  $\mathbb{P}(\llbracket \mathbf{C} \rrbracket \mid l)$ , and  $\mathbb{P}(\mathbf{C} \mid \llbracket \mathbf{C} \rrbracket, l, i)$  are given by Theorem 4, and

$$\mathbb{P}(\mathbf{y}, \mathbf{x}_U \mid \llbracket \mathbf{C} \rrbracket, i) = \begin{cases} \frac{1}{|\mathcal{X}_{\llbracket \mathbf{C} \rrbracket \cup U}|} & , \mathbf{y} \in \mathcal{X}_{\llbracket \mathbf{C} \rrbracket} \\ & \wedge \nexists v \in U : \mathbf{x}_v \neq \mathbf{y}_v \\ 0 & , \text{otherwise} . \end{cases}$$

The ordinary tuple mass  $\mathbb{P}_{\zeta, \phi}(\mathbf{J} = \mathbf{j}, I = i)$  and the marginal tuple mass  $\mathbb{P}_{\zeta, \phi}(\mathbf{J} = \mathbf{j}, \mathbf{X}_U = \mathbf{x}_U, I = i)$  differ only in the factor  $\mathbb{P}(\mathbf{y}, \mathbf{x}_U \mid \llbracket \mathbf{C} \rrbracket, i)$ . We may hence use their quotient as importance weight to convert SCCQ samples for the partition function into SCCQ samples for

---

**Algorithm 3:** SCCQ Marginal Inference

---

**input**  $\theta, k, \zeta, N$ **output** Pseudo marginals  $\hat{\mu}$ 

```
1:  $\mathbf{S} \leftarrow \mathbf{0}, \mathbf{m} \leftarrow \mathbf{0}$ , done  $\leftarrow$  false
2: while  $\exists i : m_i < N$  do
3:    $(j, i) \sim \mathbb{P}_{\zeta, \phi}(\cdot, \mathbf{x}_C)$ 
4:   for  $C \in \mathcal{C}$  do
5:     for  $\mathbf{x}_C \in \mathcal{X}_C$  do
6:       if agree( $j, i, C, \mathbf{x}$ )  $\wedge m_i < N$  then
7:          $\mathbf{S}_l \leftarrow \mathbf{S}_l + \hat{Z}_{j,i}^k(\theta) \frac{p(\mathbf{y}(j), \mathbf{x}_C | \llbracket C(j) \rrbracket, i)}{p(\mathbf{y}(j) | \llbracket C(j) \rrbracket, i)}$ 
8:          $m_i \leftarrow m_i + 1$ 
9:       end if
10:    end for
11:  end for
12: end while
13: for  $C \in \mathcal{C}$  do
14:   for  $\mathbf{x}_C \in \mathcal{X}_C$  do
15:      $\hat{\mu}_{C=\mathbf{x}_C} \leftarrow \frac{\mathbf{S}_{C=\mathbf{x}_C}}{\sum_{\mathbf{x}_C \in \mathcal{X}_C} \mathbf{S}_{C=\mathbf{x}_C}}$ 
16:   end for
17: end for
```

---

marginal probabilities. We have

$$\begin{aligned} & \mathbb{E}_{\mathbf{J}, I} \left[ \frac{p(\mathbf{y}(\mathbf{J}), \mathbf{x}_U | \llbracket C(\mathbf{J}) \rrbracket, I)}{p(\mathbf{y}(\mathbf{J}) | \llbracket C(\mathbf{J}) \rrbracket, I)} \hat{Z}_{\mathbf{J}, I}^k(\theta) \right] \\ &= \sum_{i=0}^k \sum_{j \in [d]^i} \mathbb{P}_{\zeta, \phi}(\mathbf{J} = \mathbf{j}, I = i) w_{j, i, U} \hat{Z}_{j, i}^k(\theta) \\ &= \sum_{i=0}^k \sum_{j \in [d]^i} \mathbb{P}_{\zeta, \phi}(\mathbf{J} = \mathbf{j}, I = i, \mathbf{X}_U = \mathbf{x}_U) \hat{Z}_{j, i}^k(\theta) \\ &= \mathbb{E}_{\mathbf{J}, I, \mathbf{X}_U = \mathbf{x}_U} \left[ \hat{Z}_{\mathbf{J}, I}^k(\theta) \right], \end{aligned}$$

with importance weight  $w_{j, i, U} = \frac{p(\mathbf{y}(j), \mathbf{x}_U | \llbracket C(j) \rrbracket, i)}{p(\mathbf{y}(j) | \llbracket C(j) \rrbracket, i)}$ .

Now, we gathered all parts which are required for marginal inference. The corresponding inference procedure is provided in Algorithm 3. While the main idea is to perform  $d$  separate runs of Algorithm 1, such a naive approach would result in an unnecessary high runtime. Instead, we make use of Corollary 1 to propose an importance sampling approach, in which each SCCQ sample is shared among all marginals. For each marginal  $p(\mathbf{X}_C = \mathbf{x}_C)$ , we validate if the pair  $(j, i)$  that is sampled in line 3 agrees with the assignment  $\mathbf{x}_C$  (line 6)—otherwise, its marginal tuple mass is zero. If they agree, we reweigh the sample, perform the summation and count the number of successes in lines 7 and 8. In lines 13–17, the estimated sums are normalized and written to  $\hat{\mu}$ .

## 4.2 PARAMETER ESTIMATION

With Algorithm 3, we can compute the log-likelihood's gradient  $\nabla \ell(\theta)$ , and employ any first-order method to estimate the parameters. To measure the progress of parameter estimation, it is convenient to estimate the log-likelihood of the model, which inherits its computational complexity from the log-partition function. Before we proceed to some experimental results, we close this section by translating the SCCQ error bound from Theorem 2 to an error bound on the log-likelihood.

**Theorem 6 (SCCQ Log-Likelihood Error)** *Assume that the preconditions of Theorem 2 hold. Let  $\hat{\ell}(\theta) = \langle \theta, \hat{\mu} \rangle - \ln \hat{Z}_{\zeta}^{N, k}(\theta)$  be the SCCQ approximation to the log-likelihood. Whenever the outcome  $\hat{Z}_{\zeta}^{N, k}(\theta)$  of Algorithm 1 is positive, we have*

$$\mathbb{P} \left[ |\hat{\ell}(\theta) - \ell(\theta)| < \frac{\epsilon Z(\theta)}{\min\{\hat{Z}_{\zeta}^{N, k}(\theta), Z(\theta)\}} \right] \geq 1 - \delta.$$

That is, with probability of at least  $1 - \delta$ , the absolute error in the approximated log-likelihood is roughly  $\epsilon$  when  $\hat{Z}_{\zeta}^{N, k}(\theta)$  and  $Z(\theta)$  have the same order of magnitude.

## 5 EXPERIMENTAL DEMONSTRATION

Theoretical insights from the previous sections do provably reduce the computational complexity. Moreover, pseudo marginals, based on unbiased estimates of the quadrature approximation to the partition function, facilitate approximate maximum-likelihood estimation. We conduct a small set of experiments to assess our methods empirically and answer the following questions:

- Q1** What is the runtime improvement when  $\|\chi_{\phi}^k\|$  is computed via Theorem 3 instead of naive summation?
- Q2** What is the runtime improvement when index tuples are sampled with Algorithm 2 instead of rejection sampling?
- Q3** Does SCCQ-based approximate maximum-likelihood estimation work in practice?

To answer **Q1**, we measure the runtime in nanoseconds for computing  $\|\chi_{\phi}^k\|$  via Theorem 3 and via naive summation on a  $4 \times 4$  binary Ising grid for polynomial degree  $k \in \{1, 2, 3, 4\}$ . The results are depicted in the leftmost plot of Figure 2. All results are averaged over 10 independent runs and error-bars show the standard deviation (if any). The runtime is shown in log-scale. Normalizing the tuple mass via Theorem 3 is several orders

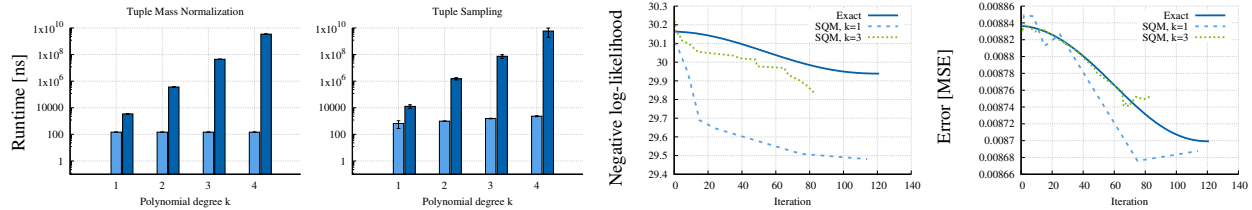


Figure 2: From left to right: (1) Runtime in log-scale for computing  $\|\chi_\phi^k\|$  with Theorem 3 (light blue) and naive summation (dark blue). Runtime in log-scale for drawing a single index tuple via Algorithm 2 (light blue) and rejection sampling (dark blue). (3) and (4): Progress of (approximate) negative log-likelihood  $-\ell(\theta)$  and mean squared error (MSE) between predicted and empirical marginals during parameter estimation on the mushroom data set. The solid line indicates the exact outcome, while the dashed lines represent SCCQ results with  $N = 10^4$  samples and  $k \in \{1, 3\}$ .

of magnitude faster than the standard approach, as expected. Regarding **Q2**, the situation looks similar. The corresponding results are depicted in the second plot of Figure 2. We see that increasing the polynomial degree and thus the maximal tuple length increases the runtime of rejection sampling. Clearly, the proportion of rejected samples increases when the state space size of the random tuples increases. On the other hand, the runtime of Algorithm 2 is almost constant in practice. To answer **Q3**, a regularized maximum-likelihood estimation on the mushroom data set<sup>2</sup> is conducted. The set contains 5644 fully observed training instances. Each data point  $x$  consists of 23 categorical features with up to 9 different states, representing properties of mushrooms. In total,  $|\mathcal{X}| \approx 2^{43}$ . To facilitate exact computation of likelihood and marginals, we use the Chow-Liu tree (Chow and Liu, 1968) as the conditional independence structure of the model. Note, however, that SCCQ is completely oblivious of the graphical structure. Hence, the reported results are valid for intractable non-tree-structured models as well. To prevent the model parameters from becoming too large,  $l_1$ -regularization with  $\lambda = 1/2$  is applied. The actual parameter estimation is carried out via the fast iterative shrinkage-thresholding algorithm (FISTA) (Beck and Teboulle, 2009) with stepsize  $1/L$  where  $L$  is an upper bound on the log-likelihood’s gradient’s Lipschitz constant. We run SCCQ with  $N = 10^4$  Monte Carlo samples. In each training iteration, we assess the (approximate) negative log-likelihood and the mean squared error (MSE) between predicted and empirical marginal probabilities. The last two plots of Figure 2 show the corresponding results. Each line corresponds to one parameter estimation. Since the runs converge in different iterations, the three lines have slightly different lengths. The results show that even the very coarse linear ( $k = 1$ ) approximation yields a reasonable approximate log-likelihood and approximate marginals. The learning

process evolves similar to the exact computation. When the polynomial degree is increased to  $k = 3$ , the approximation is even closer to the exact outcome as predicted by the theory. Especially the SCCQ marginal probabilities are often indistinguishable from the exact marginals.

## 6 CONCLUSION

We presented the first complete framework for SCCQ-based parameter learning for undirected graphical models. Quadrature-based inference provides bounds on the partition function. However, the complexity of existing algorithms is exponential in the degree of the underlying polynomial approximation and polynomial in the dimension of the model’s parameter vector—the accompanying computational complexity is not practical. We provide accelerated SCCQ algorithms whose complexity is independent of the dimension. Our empirical evaluation shows that the new algorithms are several orders of magnitude faster. In addition, we provide the first algorithm for SCCQ-based marginal inference whose practical speed and accuracy are sufficient to be used for approximate maximum-likelihood estimation. Hence, SCCQ is a highly parallel drop-in replacement for MCMC and message-passing whenever the parameter norm is bounded (e.g., via regularization). Finally, we explained how the stochastic quadrature can be applied to models with continuous random variables, which opens new research opportunities, e.g., inference in exponential family models with mixed domains, where some dimensions are discrete and others are continuous.

### Acknowledgements

This work has been supported by Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center SFB 876 “Providing Information by Resource-Constrained Analysis”, project A1.

<sup>2</sup><https://archive.ics.uci.edu/ml/datasets/mushroom>

## References

- Christophe Andrieu, Nando de Freitas, Arnaud Doucet, and Michael I. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50(1-2):5–43, 2003. doi: 10.1023/A:1020281327116.
- Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009. doi: 10.1137/080716542.
- Julian Besag. Statistical analysis of non-lattice data. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 24(3):179–195, 1975. ISSN 00390526, 14679884. doi: 10.2307/2987782.
- Manuel Bronstein. Integration of elementary functions. *Journal of Symbolic Computation*, 9(2):177–173, 1990. doi: 10.1016/S0747-7171(08)80027-2.
- Bill P. Buckles and M. Lybanon. Algorithm 515: Generation of a vector from the lexicographical index [g6]. *ACM Transactions on Mathematical Software*, 3(2):180–182, June 1977. ISSN 0098-3500. doi: 10.1145/355732.355739.
- Andrei Bulatov and Martin Grohe. The complexity of partition functions. In Josep Díaz, Juhani Karhumäki, Arto Lepistö, and Donald Sannella, editors, *Automata, Languages and Programming*, volume 3142 of *Lecture Notes in Computer Science*, pages 294–306. Springer, Heidelberg, Germany, 2004.
- C.K. Chow and C.N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968. ISSN 0018-9448. doi: 10.1109/TIT.1968.1054142.
- C.W. Clenshaw and A.R. Curtis. A method for numerical integration on an automatic computer. *Numerische Mathematik*, 2(1):197–205, 1960.
- Gideon Ehrlich. Loopless algorithms for generating permutations, combinations, and other combinatorial configurations. *Journal of the ACM*, 20(3):500–513, 1973. doi: 10.1145/321765.321781.
- W. Fraser. A survey of methods of computing min-max and near-minimax polynomial approximations for functions of a single independent variable. *Journal of the ACM*, 12(3):295–314, July 1965.
- W. Gautschi. Questions of numerical condition related to polynomials. *Studies in Numerical Analysis*, (24):140–177, 1985.
- John Michael Hammersley and Peter Clifford. Markov fields on finite graphs and lattices. *Unpublished manuscript*, 1971.
- Ernst Ising. Beitrag zur Theorie des Ferromagnetismus. *Zeitschrift für Physik*, 31:253–258, 1925.
- Elliott Ward Cheney Jr. *Introduction to Approximation Theory*. Amer Mathematical Society, 2nd edition, 1966. ISBN 978-0821813744.
- Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.
- J.C. Mason and David C. Handscomb. *Chebyshev polynomials*. Chapman and Hall/CRC, 1st edition, 2002. ISBN 9780849303555.
- Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers, Burlington, MA, USA, 1988.
- Nico Piatkowski and Katharina Morik. Stochastic discrete clenshaw-curtis quadrature. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 3000–3009. JMLR.org, 2016.
- Edwin James George Pitman. Sufficient statistics and intrinsic accuracy. *Mathematical Proceedings of the Cambridge Philosophical Society*, 32:567–579, 1936.
- Charles Sutton and Andrew McCallum. An introduction to conditional random fields. *Foundations and Trends in Machine Learning*, 4(4):267–373, 2011.
- Lloyd N. Trefethen. Is gauss quadrature better than clenshaw-curtis? *SIAM Review*, 50(1):67–87, 2008. doi: 10.1137/060659831.
- Leslie Gabriel Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.
- Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008.
- Martin J. Wainwright, Tommi S. Jaakkola, and Alan S. Willsky. Tree-reweighted belief propagation algorithms and approximate ML estimation by pseudo-moment matching. In Christopher M. Bishop and Brendan J. Frey, editors, *9th Workshop on Artificial Intelligence and Statistics*. Society for Artificial Intelligence and Statistics, Key West, FL, 2003.
- Shuhuang Xiang, Xiaojun Chen, and Haiyong Wang. Error bounds for approximation in Chebyshev points. *Numerische Mathematik*, 116(3):463–491, 2010.



---

# Finite-sample Bounds for Marginal MAP

---

**Qi Lou**

University of California, Irvine  
Irvine, CA 92697, USA  
qlou@ics.uci.edu

**Rina Dechter**

University of California, Irvine  
Irvine, CA 92697, USA  
dechter@ics.uci.edu

**Alexander Ihler**

University of California, Irvine  
Irvine, CA 92697, USA  
ihler@ics.uci.edu

## Abstract

Marginal MAP is a key task in Bayesian inference and decision-making, and known to be very challenging in general. In this paper, we present an algorithm that blends heuristic search and importance sampling to provide anytime finite-sample bounds for marginal MAP along with predicted MAP solutions. We convert bounding marginal MAP to a surrogate task of bounding a series of summation problems of an augmented graphical model, and then adapt dynamic importance sampling [Lou *et al.*, 2017b], a recent advance in bounding the partition function, to provide finite-sample bounds for the surrogate task. Those bounds are guaranteed to be tight given enough time, and the values of the predicted MAP solutions will converge to the optimum. Our algorithm runs in an anytime/anyspace manner, which gives flexible trade-offs between memory, time, and solution quality. We demonstrate the effectiveness of our approach empirically on multiple challenging benchmarks in comparison with some state-of-the-art search algorithms.

## 1 INTRODUCTION

Probabilistic graphical models, including Bayesian networks and Markov random fields, provide a framework for representing and reasoning with probabilistic and deterministic information [Dechter, 2013; Dechter *et al.*, 2010; Darwiche, 2009]. Typical inference queries in graphical models include *maximum a posteriori* (MAP) that aims to find an assignment of MAP (or MAX) variables with the highest value, the *partition function* that is the normalizing constant ensuring a proper probability measure over all variables, and marginal MAP (MMAP) that

generalizes the aforementioned two tasks by maximizing over a subset of variables with the remaining variables marginalized, which arises in many scenarios such as latent variable models [Ping *et al.*, 2014] and decision-making tasks [Kiselev and Poupart, 2014].

MMAP has complexity  $\text{NP}^{\text{PP}}$  [Park, 2002], commonly believed to be more challenging than either max inference (NP-complete [Darwiche, 2009]) or sum inference ( $\#\text{P}$ -hard [Valiant, 1979]), and can be intractable even for tree-structured models [Park, 2002]. Because of the inherent difficulty of MMAP, recent works on MMAP often focus on approximate schemes. Among these, approximations with deterministic or probabilistic guarantees are of particular interest because they quantify bounds on the approximation errors. We also prefer approaches with an anytime behavior because they allow users to trade off computational resources with solution quality.

Approaches that offer deterministic bounds are typically based on search or variational methods. Some early works [Park and Darwiche, 2003; Yuan and Hansen, 2009] in search solve MMAP exactly based on depth-first branch and bound. Marinescu *et al.* [2014] outperformed its predecessors by introducing AND/OR search spaces and high-quality variational heuristics; this was further improved using best-first search variants, including weighted heuristic search [Lee *et al.*, 2016b], and alternating depth-first and best-first AND/OR search (AAOBF [Marinescu *et al.*, 2017]). However, these methods typically require regular evaluation of internal summation problems when traversing the MAP space; when these internal sums are difficult, the search process may stall completely. One way to avoid this issue is to unify the summation with the MAP search in a single, best-first search framework (UBFS [Lou *et al.*, 2018]), which allows the bounds to improve as the summation is performed, and switch to other MAP configurations when appropriate. However, another promising approach is to make use of probabilistic bounds (e.g., Lou *et al.* [2017b]), which hold with a user-selected probability, and can be significantly faster

and tighter than deterministic bounds. However, since each MAP configuration is associated with an independent summation problem, comparing MAP configurations using probabilistic bounds must compensate for the presence of many uncertain tests (in effect, a multiple hypothesis testing problem), and is thus non-trivial to adapt to the MAP search, which may contain exponentially many such configurations.

Variational methods [Wainwright and Jordan, 2008] offer another class of deterministic bounds for MMAP. However, these bounds are often not anytime (e.g., [Liu and Ihler, 2013]), and those, such as [Ping *et al.*, 2015], are often not “any-space”, meaning that their quality depends heavily on the available memory and may not continue to improve without more. Other types of algorithms can provide anytime deterministic bounds for MMAP as well, for example, one based on factor set elimination [Mauá and de Campos, 2012]; however, the factor sets that it maintains tend to grow very large, which limits its practical use to problems with relatively small induced widths (see Marinescu *et al.* [2017] or Lou *et al.* [2018]).

Some Monte Carlo approaches are able to provide probabilistic bounds; for example, Xue *et al.* [2016] proposes a random hashing based algorithm that provides a constant factor approximation. However, this approach can have difficulty on large scale problem instances (see [Lou *et al.*, 2018]). Other Monte Carlo methods may have no bound guarantees at all, e.g., those based on Markov chain Monte Carlo [Yuan *et al.*, 2004; Doucet *et al.*, 2002].

To some extent, the intrinsic hardness of MMAP arises from the non-commutativity of the sum and max operations. One natural idea to alleviate this issue is to convert the mixed inference task to a pure sum or a pure max one first. For example, Cheng *et al.* [2012] constructs an explicit factorized approximation of the marginalized distribution using a form of approximate variable elimination, which results in a structured MAP problem.

**Our Contributions.** In this paper, we present an approach that provides anytime finite-sample bounds (i.e., they hold with probability  $1 - \delta$  for some confidence parameter  $\delta$ ) for MMAP, that enjoys the benefits of both heuristic search and importance sampling. Briefly speaking, we follow Doucet *et al.* [2002] to construct an augmented graphical model from the original model by replicating the marginalized variables and potential functions. From this augmented model, we derive a sequence of decreasing summation objectives that bound the MMAP optimum raised to some fixed power. Then, we adapt dynamic importance sampling [Lou *et al.*, 2017b] to bound these summation objectives and provide finite-sample bounds of the MMAP optimum.

Our framework has several key advantages: 1) it provides anytime probabilistic upper and lower bounds that are guaranteed to be tight given enough time. 2) it is able to predict high-quality MAP solutions whose values converge to the optimum; the exploration-exploitation trade-off of searching MAP solutions is controlled by the number of replicates of the marginalized variables. 3) it runs in an anytime/anyspace manner, which gives flexible trade-offs between memory, time, and solution quality.

## 2 BACKGROUND

Let  $X = (X_1, \dots, X_N)$  be a vector of random variables, where each  $X_i$  takes values in a discrete domain  $\mathcal{X}_i$ ; we use lower case letters, e.g.  $x_i \in \mathcal{X}_i$ , to indicate a value of  $X_i$ , and  $x$  to indicate an assignment of  $X$ . A graphical model over  $X$  consists of a set of factors  $\mathcal{F} = \{f_\alpha(X_\alpha) \mid \alpha \in \mathcal{I}\}$ , where each factor (a.k.a. potential function)  $f_\alpha$  is defined on a subset  $X_\alpha = \{X_i \mid i \in \alpha\}$  of  $X$ , called its scope.

We associate an undirected graph  $\mathcal{G} = (V, E)$ , or *primal graph*, with  $\mathcal{F}$ , where each node  $i \in V$  corresponds to a variable  $X_i$  and we connect two nodes,  $(i, j) \in E$ , iff  $\{i, j\} \subseteq \alpha$  for some  $\alpha$ . Then,

$$f(x) = \prod_{\alpha \in \mathcal{I}} f_\alpha(x_\alpha)$$

defines an unnormalized probability measure over  $X$ .

Let  $X_M$  be a subset of  $X$  called MAX variables, and  $X_S = X \setminus X_M$  SUM variables. The MMAP task seeks an assignment  $x_M^*$  of  $X_M$  with the largest marginal probability:

$$x_M^* = \operatorname{argmax}_{x_M} \pi(x_M) \quad (1)$$

where

$$\pi(x_M) = \sum_{x_S} f(x).$$

If  $X_M$  is an empty set, the MMAP task reduces to computing the normalizing constant (a.k.a. partition function); if  $X_S$  is empty, it becomes the standard MAP inference task. We use  $\mathcal{X}_M$  to denote the MAP space, i.e., the Cartesian product of all  $\mathcal{X}_i$ 's where  $X_i$  is a MAX variable. We will assume in the sequel that  $x_M^*$  is unique for convenience, though our algorithm and analysis still hold without this assumption.

### 2.1 AND/OR Search Spaces

An AND/OR search space is a generalization of the standard (“OR”) search space, that enables us to exploit conditional independence structure during search [Dechter and

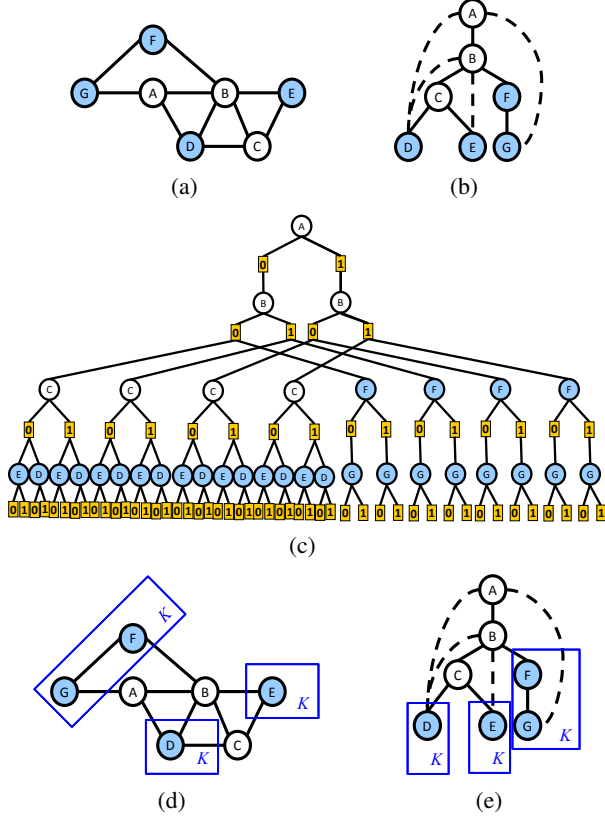


Figure 1: (a) A primal graph of a graphical model over 7 variables (A, B, C are MAX variables and D, E, F, G are SUM variables) with unary and pairwise potential functions. (b) A valid pseudo tree for the primal graph. (c) An AND/OR search tree guided by the pseudo tree. (d) An augmented model created by replicating SUM variables and factors of the model in (a). Plate notations used here. (e) A valid pseudo tree for the augmented model.

Mateescu, 2007]. The AND/OR search space for a graphical model is defined relative to a *pseudo tree* that captures problem decomposition along a fixed search order.

**Definition 1 (pseudo tree).** A *pseudo tree* of a primal graph  $\mathcal{G} = (V, E)$  is a directed tree  $\mathcal{T} = (V, E')$  sharing the same set of nodes as  $\mathcal{G}$ . The tree edges  $E'$  form a subset of  $E$ , and each edge  $(i, j) \in E \setminus E'$  are required to be a “back edge”, i.e., the path from the root of  $\mathcal{T}$  to  $j$  passes through  $i$  (denoted  $i \leq j$ ).

If a tree node of a pseudo tree corresponds to a MAX variable in the associated graphical model of the pseudo tree, we call it MAX node, otherwise we call it SUM node. A pseudo tree is called *valid* for an MMAP task if there is *no* MAX variable descended from any SUM variable. Thus, all MAX variables of a valid pseudo tree form a subtree (assuming a dummy MAX root) that contains the

root. We assume valid pseudo trees in the sequel.

Guided by a pseudo tree, we can construct an AND/OR search tree consisting of alternating levels of OR and AND nodes for a graphical model. Each OR node  $s$  is associated with a variable, which we lightly abuse notation to denote  $X_s$ ; the children of  $s$ ,  $ch(s)$ , are AND nodes corresponding to the possible values of  $X_s$ . If an OR node is associated with some MAX variable, it is called OR-MAX node. Notions of OR-SUM, AND-MAX, AND-SUM nodes are defined analogously. The root  $\emptyset$  of the AND/OR search tree corresponds to the root of the pseudo tree. Let  $pa(c) = s$  indicate the parent of  $c$  in the AND/OR tree, and  $an(c) = \{n \mid n \leq c\}$  indicate the ancestors of  $c$  (including itself) in the tree.

In an AND/OR tree, any AND node  $c$  corresponds to a partial configuration  $x_{\leq c}$  of  $X$ , defined by its assignment and that of its ancestors:  $x_{\leq c} = x_{\leq p} \cup \{X_s = x_c\}$ , where  $s = pa(c)$ ,  $p = pa(s)$ . For completeness, we also define  $x_{\leq s}$  for any OR node  $s$ , which is the same as that of its AND parent, i.e.,  $x_{\leq s} = x_{\leq pa(s)}$ . For any node  $n$ , the corresponding variables of  $x_{\leq n}$  are denoted as  $X_{\leq n}$ . Let  $de(X_n)$  be the set of variables below  $X_n$  in the pseudo tree; we define  $X_{>n} = de(X_n)$  if  $n$  is an AND node;  $X_{>n} = de(X_n) \cup \{X_n\}$  if  $n$  is an OR node.

We also associate a weight  $w_c$  with each AND node, defined to be the product of all factors  $f_\alpha$  that are instantiated at  $c$  but not before:

$$w_c = \prod_{\alpha \in \mathcal{I}_c} f_\alpha(x_\alpha), \quad \mathcal{I}_c = \{\alpha \mid X_{pa(c)} \in X_\alpha \subseteq X_{an(c)}\}.$$

**Example.** Fig. 1(a) shows the primal graph of a pairwise model. Variables A, B, C are MAX variables, and the rest SUM. Fig. 1(b) shows one valid pseudo tree of the model. Fig. 1(c) shows the AND/OR search tree that respects the pseudo tree.

## 2.2 SEARCH IN AND/OR SEARCH TREES

Finally, the purpose of the search tree is to compute some inference quantity for the model, such as the MAP optimum  $\max_x f(x)$ , the partition function  $Z = \sum_x f(x)$  and the MMAP optimum  $\max_{x_M} \sum_{x_S} f(x)$ . To this end, we associate a “value”  $v_n$  with each node  $n$  in the AND/OR search tree, which represents the inference task’s value on the unexpanded portion of the search space below node  $n$ . The value  $v_n$  can be defined recursively in terms of its children and grandchildren as follows. We first define  $v_l = 1$  for any leaf (since no part of the model remains uninstantiated). Let  $n$  be a non-leaf node; for maximization tasks, we have

$$\text{Max: } v_n = \begin{cases} \prod_{c \in ch(n)} v_c, & \text{if AND node } n. \\ \max_{c \in ch(n)} w_c v_c, & \text{if OR node } n. \end{cases}$$

while for summation, the recursion defining  $v_n$  for  $n$  is

$$\text{Sum: } v_n = \begin{cases} \prod_{c \in \text{ch}(n)} v_c, & \text{if AND node } n. \\ \sum_{c \in \text{ch}(n)} w_c v_c, & \text{if OR node } n. \end{cases}$$

For MMAP tasks, the recursion for AND nodes is the same as the aforementioned tasks, while the recursion for OR nodes is more involved:

$$\text{MMAP: } v_n = \begin{cases} \max_{c \in \text{ch}(n)} w_c v_c, & \text{if OR-MAX node } n. \\ \sum_{c \in \text{ch}(n)} w_c v_c, & \text{if OR-SUM node } n. \end{cases}$$

Any search algorithm for reasoning about the model can be thought of as maintaining upper (and/or lower) bounds on these quantities at each node. In particular, for heuristic search, we assume that we have a heuristic function  $h_n$  that gives upper (or lower) bound on  $v_n$ . These heuristics typically are more accurate deeper in the search tree, and therefore their updates can be propagated upwards to the root to yield tighter bounds to the overall inference value. Any search algorithm is then defined by the order of expansion of the search tree.

A typical example of this kind of search algorithms is AOBFS [Lou *et al.*, 2017a], a best-first search algorithm that can provide anytime upper (and/or lower) bounds for the summation task. Since AOBFS will be a component of our proposed algorithm, we briefly present some of its essence here. AOBFS maintains an explicit AND/OR search tree of visited nodes, denoted  $\mathcal{S}$ . For each node  $n$  in the AND/OR search tree, AOBFS maintains  $u_n$ , an upper bound on  $v_n$ , initialized via a pre-compiled heuristic  $v_n \leq h_n^+$ , and subsequently updated during search using information propagated from the frontier:

$$u_n = \begin{cases} \prod_{c \in \text{ch}(n)} u_c, & \text{if AND node } n. \\ \sum_{c \in \text{ch}(n)} w_c u_c, & \text{if OR node } n. \end{cases}$$

Thus, the upper bound at the root,  $u_\emptyset$ , is an anytime deterministic upper bound of the partition function. Note that this upper bound depends on the current search tree  $\mathcal{S}$ , so we write  $U^\mathcal{S} = u_\emptyset$ .

If all nodes below  $n$  have been visited, then  $u_n = v_n$ ; we call  $n$  *solved* and can remove the subtree below  $n$  from memory. Hence we can partition the frontier nodes into two sets: solved frontier nodes,  $\text{SOLVED}(\mathcal{S})$ , and unsolved ones,  $\text{OPEN}(\mathcal{S})$ .

### 2.3 DYNAMIC IMPORTANCE SAMPLING

Our work can be viewed as a generalization of dynamic importance sampling (DIS) [Lou *et al.*, 2017b], a recent

advance in bounding the partition function with finite-sample bounds (see also [Liu *et al.*, 2015]), which we briefly introduce here to make our paper self-contained.

DIS interleaves search with sampling: search, as it improves the deterministic upper bound of the partition function by expanding nodes in the AND/OR search tree, also induces a sequence of importance sampling proposal distributions with bounded importance weights that are unbiased estimators of the partition function. Meanwhile, samples are drawn independently from those improving proposal distributions. By averaging those importance weights based on their corresponding upper bounds, DIS constructs an unbiased estimator of the partition function  $Z$  with strong probabilistic guarantees.

To be more specific, DIS applies AOBFS with its ‘‘upper priority’’ to quickly drive down the deterministic upper bound. The current search tree  $\mathcal{S}$  induces a proposal distribution  $q^\mathcal{S}$ ; importance weights  $f(x)/q^\mathcal{S}(x)$  are bounded by  $U^\mathcal{S}$  and give an unbiased estimator of  $Z$ :

$$f(x)/q^\mathcal{S}(x) \leq U^\mathcal{S}, \quad \mathbb{E} \left[ f(x)/q^\mathcal{S}(x) \right] = Z.$$

Drawing a sample from  $q^\mathcal{S}$  can be described as a ‘‘two-step’’ top-down sampling process from the root:

**Step 1** For an internal node  $n \in \mathcal{S}$ : if it is an AND node, all its children are selected; if  $n$  is an OR node, one child  $c \in \text{ch}(n)$  is randomly selected with probability  $w_c u_c / u_n$ .

**Step 2** When an unsolved frontier node  $n \in \text{OPEN}(\mathcal{S})$  is reached, draw a sample of its descendant variables  $X_{>n}$  in the pseudo tree according to the mixture proposal  $q(x_{>n} | x_{\leq n})$  derived from *weighted mini-bucket* (WMB, [Liu and Ihler, 2011]).

DIS introduces an unbiased estimator  $\widehat{Z}$  of  $Z$ :

$$\widehat{Z} = \frac{\text{HM}(\mathbf{U})}{N} \sum_{i=1}^N \frac{\widehat{Z}_i}{U_i}, \quad \text{HM}(\mathbf{U}) = \left[ \frac{1}{N} \sum_{i=1}^N \frac{1}{U_i} \right]^{-1}.$$

where  $\{\widehat{Z}_i = f(x^i)/q^{\mathcal{S}_i}(x^i)\}_{i=1}^N$  are importance weights from samples  $\{x^i | x^i \sim q^{\mathcal{S}_i}(x)\}$  with  $\{\mathcal{S}_i\}$  the corresponding search trees, and  $\{U_i = U^{\mathcal{S}_i}\}_{i=1}^N$  the corresponding upper bounds on the importance weights respectively. By defining

$$\Delta = \text{HM}(\mathbf{U}) \left[ \sqrt{\frac{2\widehat{\text{Var}}(\{\widehat{Z}_i/U_i\}_{i=1}^N) \ln(2/\delta)}{N}} + \frac{7 \ln(2/\delta)}{3(N-1)} \right]$$

where  $\widehat{\text{Var}}(\{\widehat{Z}_i/U_i\}_{i=1}^N)$  is the unbiased empirical variance of  $\{\widehat{Z}_i/U_i\}_{i=1}^N$ ,  $\widehat{Z}$  enjoys the finite-sample guarantees: with probability at least  $1 - \delta$ ,  $\widehat{Z} + \Delta$  and  $\widehat{Z} - \Delta$  are upper and lower bounds of  $Z$ , respectively, i.e.,  $\Pr[Z \leq \widehat{Z} + \Delta] \geq 1 - \delta$  and  $\Pr[Z \geq \widehat{Z} - \Delta] \geq 1 - \delta$ .

### 3 OUR ALGORITHM

In this section, we introduce our algorithm, the general idea of which is to first bound the mixed inference objective with a series of sum inference objectives whose finite-sample bounds can be established by generalizing DIS, and then translate the bounds back to those of the original objective in which we are interested.

#### 3.1 AN AUGMENTED GRAPHICAL MODEL

We first introduce an augmented graphical model which connects the MMAP optimum to a series of summation tasks. The augmented graphical model is built from the original model by replicating the SUM variables and the factors. Note that the idea of introducing an augmented space on which we perform inference is adopted from Doucet *et al.* [2002].

Let  $X_{\text{aug}} = (X_M, X_S^1, \dots, X_S^K)$  be all the variables of the augmented model where  $X_S^1, \dots, X_S^K$  are  $K$  replicates of the SUM variables  $X_S$ . The overall function  $f_{\text{aug}}$  of the augmented model is defined as

$$f_{\text{aug}}(x_{\text{aug}}) = \prod_{k=1}^K f(x_M, x_S^k).$$

Thus, the partition function of the augmented model is

$$Z_{\text{aug}} = \sum_{x_M, x_S^1, \dots, x_S^K} \prod_{k=1}^K f(x_M, x_S^k) = \sum_{x_M} \pi^K(x_M).$$

Considering that  $\pi^K(x_M^*)$  (see (1)) is the largest term in the sum on the r.h.s., we have

$$Z_{\text{aug}}/|\mathcal{X}_M| \leq \pi^K(x_M^*) \leq Z_{\text{aug}}, \quad (2)$$

that is to say,

$$(Z_{\text{aug}}/|\mathcal{X}_M|)^{1/K} \leq \pi(x_M^*) \leq Z_{\text{aug}}^{1/K},$$

where  $|\mathcal{X}_M|$  is the size of the MAP space. The above inequalities are actually well-known boundedness relations between the  $\infty$ -norm and  $p$ -norms of the Euclidean space  $\mathbb{R}^{|\mathcal{X}_M|}$ . These bounds are monotonic in  $K$ , i.e., they improve as  $K$  increases, and become tight as  $K$  goes to infinity. In other words,  $K$  acts as a “reverse temperature” parameter. The lower bound is negatively impacted by the domain sizes of the MAX variables, which can be quite loose if  $|\mathcal{X}_M|$  is large compared to the scale of  $K$ .

The significance of (2) is that it connects the MMAP optimum to a summation quantity  $Z_{\text{aug}}$  that can be easily approximated using Monte Carlo methods such as importance sampling.

**Example.** Fig. 1(d) shows an augmented graphical model created from the model of Fig. 1(a). Fig. 1(e) shows one valid pseudo tree for the augmented model.

#### 3.2 MIXED DYNAMIC IMPORTANCE SAMPLING

A straightforward idea is to apply DIS to bound  $Z_{\text{aug}}$  whose finite-sample bounds can then be translated to those of  $\pi(x_M^*)$ . However, several key issues remain to be addressed for this idea to work well.

The first issue is about how to adapt DIS to the augmented model in an efficient manner. Since the augmented model might have many more variables compared to the original model, a naïve construction of AND/OR trees leads to an excessively large search space. Note that any  $X_S^k$  in the augmented model is an identical copy of  $X_S$ ; we thus do not necessarily distinguish those  $X_S$  copies during search. That is to say, when search instantiates those factors involving SUM variables, it behaves as usual but takes into account the effect of replication when using information propagated from SUM nodes. We can also apply an analogous idea to construct weighted mini-bucket (WMB) [Liu and Ihler, 2011] heuristics to ensure that they are still compatible with the new search process. In a nutshell, search for the augmented model can enjoy the same complexity as that for the original model.

Meanwhile, the proposal distribution  $q_{\text{aug}}^S(x_{\text{aug}})$  associated with a search tree  $\mathcal{S}$  has a decomposition property:

$$q_{\text{aug}}^S(x_{\text{aug}}) = q_{\text{aug}}^S(x_M) \prod_{k=1}^K q_{\text{aug}}^S(x_S^k|x_M), \quad (3)$$

with  $q_{\text{aug}}^S(x_S^k|x_M)$  are identical conditional distributions. Its importance weights also share the boundedness property:

$$f_{\text{aug}}(x_{\text{aug}})/q_{\text{aug}}^S(x_{\text{aug}}) \leq U_{\text{aug}}^S,$$

where  $U_{\text{aug}}^S$  is the upper bound associated with  $\mathcal{S}$ . Note that sampling from  $q_{\text{aug}}^S$  can also be done via a two-step sampling procedure analogous to that in DIS.

One point worth mentioning is that

$$\pi(x_M) = \mathbb{E}[f(x_M, x_S^k)/q_{\text{aug}}^S(x_S^k|x_M)]$$

implies that we can estimate the value of each sampled MAP configuration  $x_M$  along the way.

Another issue is that if  $Z_{\text{aug}}$  is much larger than  $\pi^K(x_M^*)$ , even high-quality bounds of  $Z_{\text{aug}}$  might not result in reasonably good bounds of  $\pi(x_M^*)$ , let alone those bounds will never be tight in general for  $\pi(x_M^*)$  with a finite  $K$ . One way to alleviate this issue is based on the following key observation: for any subset  $\mathcal{A}$  of  $\mathcal{X}_M$  that contains  $x_M^*$ , we have

$$Z_{\text{aug}}^{\mathcal{A}}/|\mathcal{A}| \leq \pi^K(x_M^*) \leq Z_{\text{aug}}^{\mathcal{A}}, \quad (4)$$

---

**Algorithm 1** Mixed Dynamic Importance Sampling

---

**Require:** Control parameters  $K, N_d, N_l$ ; confidence parameter  $\delta$ ; memory budget, time budget.

**Ensure:**  $\widehat{Z}_{\text{aug}}, \Delta, \text{HM}(\mathbf{U}), \text{HM}(\mathbf{U}/|\mathcal{A}|)$ .

- 1: Construct WMB heuristics for the augmented model.
  - 2: Initialize  $\mathcal{S} \leftarrow \{\emptyset\}$  with the root  $\emptyset$ .
  - 3: **while** within the time budget
  - 4:     *// update  $\mathcal{S}, U^{\mathcal{S}}, \mathcal{A}^{\mathcal{S}}$  during search.*
  - 5:     **if** within the memory budget
  - 6:         Expand  $N_d$  nodes via AOBFS (Alg. 1 of Lou *et al.* [2017a]) with its “upper priority”.
  - 7:     **else**
  - 8:         Expand  $N_d$  nodes via depth-first search.
  - 9:     **end if**
  - 10:     Draw  $N_l$  samples from  $q_{\text{aug}}^{\mathcal{S}}$  (see (3)).
  - 11:     After drawing each sample:
  - 12:         Update  $N, \widehat{Z}_{\text{aug}}, \text{HM}(\mathbf{U}), \text{HM}(\mathbf{U}/|\mathcal{A}|), \widehat{\text{Var}}, \Delta$  via (5), (6), (11), (12).
  - 13: **end while**
- 

where

$$Z_{\text{aug}}^{\mathcal{A}} = \sum_{x_M \in \mathcal{A}} \pi^K(x_M).$$

The above inequalities tell us that if we know an instantiation of  $X_M$  is not optimal, we can mute its contribution to  $Z_{\text{aug}}$  and use the resulting smaller summation quantity to bound  $\pi^K(x_M^*)$ .

This observation enables pruning during search: any node ruled out from being associated with the optimal configuration can be removed from memory. Such pruning is particularly useful to prune MAX nodes: for any AND-MAX node with its sub-problem beneath solved, if it holds the highest value among its siblings, all its siblings (solved or not) and their descendants can be pruned immediately. Thus, as pruning proceeds along with search,  $\mathcal{A}$  shrinks towards  $\{x_M^*\}$ . We use  $\mathcal{A}^{\mathcal{S}}$  to denote the remaining MAP space associated with the search tree  $\mathcal{S}$ .

Note that when we approach the memory limit, we switch the default best-first search to a depth-first search (DFS) that is also compatible with the sampling procedure, and leads to a complete search algorithm with the capability to identify  $x_M^*$  and its value given enough time. By interleaving search and sampling, we derive our algorithm named *mixed dynamic importance sampling* (MDIS) and present it in Alg. 1.

**Remarks on Alg. 1.**

1)  $K$  as the number of replicates of the SUM variables controls the exploration-exploitation trade-off. When  $K$  is small, we draw a small number of samples for the

SUM variables in each iteration, which allows us to evaluate each sampled MAP configuration fast, however introduces more randomness when assessing the MAP configuration; when  $K$  is large, we have more accurate estimate of a MAP configuration being sampled, but also slow down exploration of the MAP space.

2) To predict MAP solutions in an anytime manner, one can simply choose the one with the highest estimated value among those configurations that have been sampled.

**3.2.1 Finite-sample Bounds for Marginal MAP**

In MDIS, each sample not only comes from a different proposal distribution but also gives importance weights corresponding to a different expectation, which is more complicated than in DIS.

Let  $\{x_{\text{aug}}^i\}_{i=1}^N$  be a series of samples drawn via Alg. 1, with  $\{\mathcal{S}_i\}$  the corresponding search trees,  $\{\widehat{Z}_{\text{aug}}^i = f_{\text{aug}}(x_{\text{aug}}^i)/q_{\text{aug}}^{\mathcal{S}_i}(x_{\text{aug}}^i)\}_{i=1}^N$  the corresponding importance weights, and  $\{U_i = U_{\text{aug}}^{\mathcal{S}_i}\}_{i=1}^N$  the corresponding upper bounds associated with those search trees respectively. We denote  $\mathcal{A}_i$  as the MAP space preserved in  $\mathcal{S}_i$ . Thus,

$$\mathbb{E}[\widehat{Z}_{\text{aug}}^i] = Z_{\text{aug}}^{\mathcal{A}_i}.$$

That is to say, the importance weights have different (in fact, decreasing) expectations; this differs from the case of DIS where any importance weight has the same expectation (the partition function). We propose an estimate  $\widehat{Z}_{\text{aug}}$  whose expectation is again an upper bound of  $\pi^K(x_M^*)$  in the following way:

$$\widehat{Z}_{\text{aug}} = \frac{\text{HM}(\mathbf{U})}{N} \sum_{i=1}^N \frac{\widehat{Z}_{\text{aug}}^i}{U_i}, \quad (5)$$

where

$$\text{HM}(\mathbf{U}) = \left[ \frac{1}{N} \sum_{i=1}^N \frac{1}{U_i} \right]^{-1} \quad (6)$$

is the harmonic mean of the upper bounds  $\{U_i\}_{i=1}^N$ . Thus,  $\widehat{Z}_{\text{aug}}$  upweights the terms  $\widehat{Z}_{\text{aug}}^i$  whose expectations are closer to  $\pi^K(x_M^*)$ . The expectation of  $\widehat{Z}_{\text{aug}}$  is

$$\mathbb{E}[\widehat{Z}_{\text{aug}}] = \frac{\text{HM}(\mathbf{U})}{N} \sum_{i=1}^N \frac{Z_{\text{aug}}^{\mathcal{A}_i}}{U_i}.$$

$\mathbb{E}[\widehat{Z}_{\text{aug}}]$  is a convex combination of  $\{Z_{\text{aug}}^{\mathcal{A}_i}\}_{i=1}^N$  with coefficients  $\{\frac{\text{HM}(\mathbf{U})}{NU_i}\}_{i=1}^N$ , shrinking towards  $\pi^K(x_M^*)$  as search proceeds.

According to (4), since  $\pi^K(x_M^*) \leq Z_{\text{aug}}^{\mathcal{A}_i}$ , we know

$$\pi^K(x_M^*) \leq \mathbb{E}[\widehat{Z}_{\text{aug}}], \quad (7)$$

and from  $Z_{\text{aug}}^{\mathcal{A}_i} \leq |\mathcal{A}_i| \pi^K(x_M^*)$ , we know

$$\mathbb{E} [\widehat{Z}_{\text{aug}}] \leq \pi^K(x_M^*) \frac{\text{HM}(\mathbf{U})}{N} \sum_{i=1}^N \frac{|\mathcal{A}_i|}{U_i}. \quad (8)$$

By combining (6), (7), and (8), we derive two-sided bounds for  $\pi^K(x_M^*)$  involving  $\mathbb{E} [\widehat{Z}_{\text{aug}}]$ :

$$\frac{\sum_{i=1}^N 1/U_i}{\sum_{i=1}^N |\mathcal{A}_i|/U_i} \mathbb{E} [\widehat{Z}_{\text{aug}}] \leq \pi^K(x_M^*) \leq \mathbb{E} [\widehat{Z}_{\text{aug}}]. \quad (9)$$

From the above, we can see that the bounds get tight only when  $\mathcal{A}_i$  approaches  $\{x_M^*\}$ . To be concise, we re-arrange the L.H.S. of (9) to derive:

$$\frac{\text{HM}(\mathbf{U}/|\mathcal{A}|)}{\text{HM}(\mathbf{U})} \mathbb{E} [\widehat{Z}_{\text{aug}}] \leq \pi^K(x_M^*) \leq \mathbb{E} [\widehat{Z}_{\text{aug}}], \quad (10)$$

where

$$\text{HM}(\mathbf{U}/|\mathcal{A}|) = \left[ \frac{1}{N} \sum_{i=1}^N \frac{|\mathcal{A}_i|}{U_i} \right]^{-1} \quad (11)$$

is the harmonic mean of  $\{U_i/|\mathcal{A}_i|\}_{i=1}^N$ .

Considering  $\widehat{Z}_{\text{aug}}^i$  are independent, and  $\mathbb{E} \widehat{Z}_{\text{aug}}/\text{HM}(\mathbf{U})$ ,  $\widehat{Z}_{\text{aug}}/\text{HM}(\mathbf{U})$ ,  $\widehat{Z}_{\text{aug}}^i/U_i$  are all within the interval  $[0, 1]$ , we can apply an empirical Bernstein bound [Maurer and Pontil, 2009] to derive finite-sample bounds on  $\mathbb{E} \widehat{Z}_{\text{aug}}$  and translate those bounds to  $\pi(x_M^*)$  based on (10).

**Theorem 1.** For any  $\delta \in (0, 1)$ , we define

$$\Delta = \text{HM}(\mathbf{U}) \left( \sqrt{\frac{2\widehat{\text{Var}} \ln(2/\delta)}{N}} + \frac{7 \ln(2/\delta)}{3(N-1)} \right), \quad (12)$$

where  $\widehat{\text{Var}}$  is the unbiased empirical variance of  $\{\widehat{Z}_{\text{aug}}^i/U_i\}_{i=1}^N$ . Then, the following probabilistic bounds hold for  $\pi(x_M^*)$ :

$$\begin{aligned} \Pr [\pi(x_M^*) \leq (\widehat{Z}_{\text{aug}} + \Delta)^{\frac{1}{K}}] &\geq 1 - \delta, \\ \Pr [\pi(x_M^*) \geq \left( \frac{(\widehat{Z}_{\text{aug}} - \Delta) \text{HM}(\mathbf{U}/|\mathcal{A}|)}{\text{HM}(\mathbf{U})} \right)^{\frac{1}{K}}] &\geq 1 - \delta, \end{aligned}$$

i.e.,  $(\widehat{Z}_{\text{aug}} + \Delta)^{\frac{1}{K}}$  and  $\left( \frac{(\widehat{Z}_{\text{aug}} - \Delta) \text{HM}(\mathbf{U}/|\mathcal{A}|)}{\text{HM}(\mathbf{U})} \right)^{\frac{1}{K}}$  are upper and lower bounds of  $\pi(x_M^*)$  with probability at least  $1 - \delta$ , respectively.

Note that it is possible that  $\widehat{Z}_{\text{aug}} - \Delta < 0$  early on; if so, we may replace  $\widehat{Z}_{\text{aug}} - \Delta$  with any non-trivial lower bound of  $Z_{\text{aug}}$ . In the experiments, we use  $\delta \widehat{Z}_{\text{aug}}$ , a  $(1 - \delta)$  probabilistic bound by the Markov inequality [Gogate and Dechter, 2011]. We can replace  $\widehat{Z}_{\text{aug}} + \Delta$  with the current deterministic upper bound if the latter is tighter.

## 4 EXPERIMENTS

We evaluate our proposed approach (MDIS) against two baseline methods on five benchmarks. The baselines include UBFS [Lou *et al.*, 2018], a unified best-first search algorithm that emphasizes rapidly tightening the upper bound, and AAOBF [Marinescu *et al.*, 2017], a best-first/depth-first hybrid search algorithm that balances upper bound quality with generating and evaluating potential solutions. These two are state-of-the-art algorithms for anytime upper and lower bounds respectively. We do not compare to XOR\_MMAP [Xue *et al.*, 2016] and AFSE [Mauá and de Campos, 2012] due to their limitations to relatively easy problem instances as shown in Lou *et al.* [2018].

Three benchmarks are formed by problem instances from recent UAI competitions: `grid`- 50 grid networks with size no smaller than 25 by 25, `promedas`- 50 medical diagnosis expert systems, `protein`- 44 instances made from the ‘‘small’’ protein side-chains of [Yanover and Weiss, 2002]. Since the original UAI instances are pure MAP tasks, we generate MMAP instances by randomly selecting 10% of the variables as MAP variables. The fourth benchmark is `planning`, formed by 15 instances from probabilistic conformant planning with a finite-time horizon [Lee *et al.*, 2016a]. On these four benchmarks, we compare anytime bounds. Some statistics of the four benchmarks are shown in Table 1. These benchmarks are selected to illustrate different problem characteristics; for example, `protein` instances are relatively small but high cardinality, while `planning` instances have more variables and higher induced width, but lower cardinality. The fifth benchmark, which we will describe in detail later, is created from an image denoising model in order to evaluate quality of the predicted MAP solutions.

The time budget is set to 1 hour for the experiments on the first four benchmarks. We allot 4GB memory to all algorithms, with 1GB extra memory to AAOBF for caching. For our experiments, we use the weighted mini-bucket [Liu and Ihler, 2011] heuristics, whose memory usage is roughly controlled by an *ibound* parameter. For a given memory budget, we first compute the largest *ibound* that fits in memory, then use the remaining memory for search. Since all the competing algorithms use weighted mini-bucket heuristics, the same *ibound* is shared during heuristic construction. We set  $N_d = 100$  and  $N_l = 1$  (see Alg. 1) as suggested by the experimental results in Lou *et al.* [2017b]. We set  $\delta = 0.025$ . All implementations are in C/C++ courtesy of the original authors.

**Anytime bounds for individual instances.** Fig. 2 shows the anytime behavior of all the methods on instances from four benchmarks. In terms of lower bounds,

Table 1: Statistics of the four evaluated benchmarks. The first three benchmarks are formed by problem instances from recent UAI competitions, where 10% of variables are randomly selected as MAX variables. “avg. ind. width of sum” in the last row stands for the average induced width of the internal summation problems.

	grid	promedas	protein	planning
# instances	50	50	44	15
avg. # variables	1248.20	982.10	109.55	1122.33
avg. % of MAX vars	10%	10%	10%	12%
avg. # of factors	1248.20	994.76	394.64	1127.67
avg. max domain size	2.00	2.00	81.00	3.00
avg. max scope	3.00	3.00	2.00	5.00
avg. induced width	124.82	108.14	15.84	165.00
avg. pseudo tree depth	228.92	158.78	33.52	799.33
avg. ind. width of sum	43.44	40.32	10.20	49.67

our approach can always provide decent lower bounds even when the internal summation problems are quite challenging, while AAOBF may not work well since it relies on exact evaluation of those internal summation problems, e.g., on those shown in Fig. 2(b)-2(d). When the internal summation problems are relatively easy, their exact evaluation is cheap; thus AAOBF might perform better than ours. Fig. 2(a) gives a typical example. In terms of upper bounds, our bound quality is often eventually comparable to UBFS, e.g., Fig. 2(b)-2(d). UBFS typically performs better than MDIS early on, while MDIS quickly catches up and becomes comparable. Improvement in AAOBF on upper bounds also requires fast exact evaluation of the internal summation problems, which might not be possible in many cases. So, AAOBF is usually not as competitive as the other two methods on upper bounds.

**Anytime bounds across benchmarks.** We present the anytime performance across the four benchmarks in Table 2 and 3 where we compare anytime bounds at three different timestamps: 1 minute, 10 minutes and 1 hour. From Table 2, we can observe that MDIS with  $K=5$  is dominant at any of these timestamp/benchmark combinations for lower bounds. MDIS with  $K=10$  performs less well, perhaps because it requires more time to draw one full sample compared to when  $K=5$ , leading the empirical Bernstein lower bounds to kick in relatively late; this phenomenon can be also observed in all the plots in Fig. 2. UBFS provides the best upper bounds as shown in Table 3. However, our algorithm generally performs better than AAOBF in terms of upper bounds.

**Empirical evaluation of solution quality.** To evaluate the MAP solution quality predicted by our algorithm, we create an image denoising task from the MNIST database<sup>1</sup>

<sup>1</sup><http://yann.lecun.com/exdb/mnist/>

Table 2: Number of instances that an algorithm achieves the best *lower bounds* at each timestamp (1 min, 10 min, and 1 hour) for each benchmark. The best for each setting is bolded. Entries for UBFS are blank because UBFS does not provide lower bounds.

	grid	promedas	protein	planning
# instances	50	50	44	15
Timestamp: 1min/10min/1hr				
MDIS ( $K=5$ )	<b>47/44/45</b>	<b>32/34/31</b>	<b>31/27/28</b>	<b>14/13/13</b>
MDIS ( $K=10$ )	3/2/1	4/5/6	11/13/14	1/2/2
UBFS	-/-/-	-/-/-	-/-/-	-/-/-
AAOBF	0/4/4	16/21/24	2/4/4	0/0/0

Table 3: Number of instances that an algorithm achieves the best *upper bounds* at each timestamp (1 min, 10 min, and 1 hour) for each benchmark. The best for each setting is bolded.

	grid	promedas	protein	planning
# instances	50	50	44	15
Timestamp: 1min/10min/1hr				
MDIS ( $K=5$ )	0/0/0	9/12/13	5/9/15	1/1/1
MDIS ( $K=10$ )	0/0/0	10/13/14	9/10/13	1/2/3
UBFS	<b>50/50/50</b>	<b>50/50/50</b>	<b>36/32/26</b>	<b>14/14/13</b>
AAOBF	0/0/1	2/4/6	2/2/2	1/1/1

of handwritten digits [LeCun *et al.*, 1998]. We binarize each image, resize it to 14 by 14, and then randomly flip 5% of the pixels to generate a corrupt one. We train a conditional restricted Boltzmann machine (CRBM) [Mnih *et al.*, 2011] model with 64 hidden units and 196 visible units using mixed-product BP [Ping and Ihler, 2017; Liu and Ihler, 2013] for the denoising task. The resulting graphical model thus has 64 SUM variables and 196 MAX variables. Fig. 3(c) gives an illustration of this model. The advantage of this model is that we can easily evaluate any MAP configuration since the internal summation problem only contains singleton potentials; thus this model favors AAOBF since AAOBF is able to evaluate MAP configurations at a very low cost. We set  $K$  to 5 and runtime to 10 minutes for convenience. We test on 100 images with 10 images per digit. Fig. 3(a) compares the denoising results among all the algorithms for one instance per digit. Fig. 3(b) gives an example of the quality of the predicted MAP solutions of our algorithm. In general, the quality of predicted MAP solutions for our algorithm are better than the other two baselines in 51 of 100 instances, which is generally as good as AAOBF (47/100) despite the model being well-suited to AAOBF. A possible reason is that our algorithm is able to traverse the MAP space very quickly and get cheap stochastic estimates of the most promising MAP solutions.



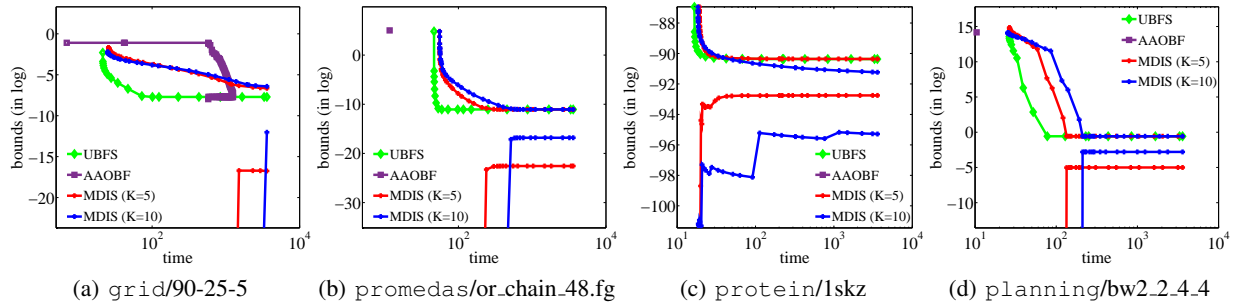


Figure 2: Anytime bounds for MMAP on instances from four benchmarks. The max domain sizes of those instances from (a)-(d) are 2, 2, 81, 3 respectively, and the induced widths of the internal summation problems are 25, 28, 8, 24 respectively. Curves for some bounds may be (partially) missing because they are not in a reasonable scope. UBFS only provides upper bounds. The time limit is 1 hour.

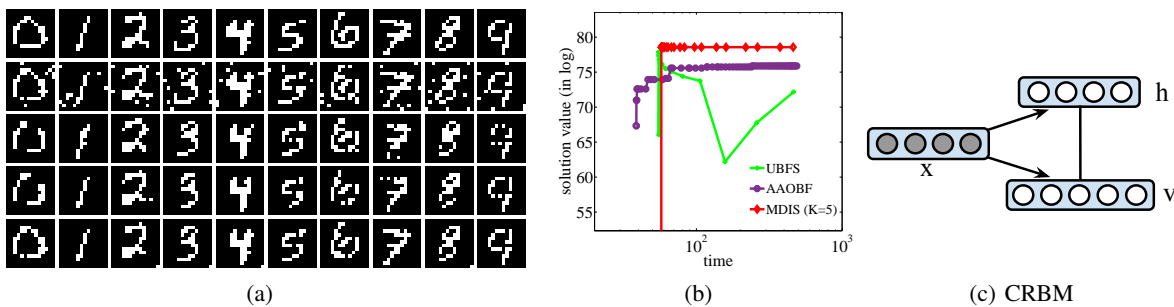


Figure 3: (a) Image denoising results for one instance per digit. The first row is for the ground truth images. The second row is for the noisy inputs created from the ground truth by randomly flipping 5% pixels. Below the first two rows are denoised images from UBFS, AAOBF, MDIS ( $K=5$ ) respectively. (b) An example on MAP solution quality comparison. (c) Illustration of the conditional restricted Boltzmann machine (CRBM) model used for the image denoising task. When conditioned on an input “X”, this model has a bipartite graph structure between hidden units “h” (SUM variables) and visible units “v” (MAX variables).

## 5 CONCLUSION

In this paper, we propose an approach that provides anytime finite-sample upper and lower bounds for MMAP, which enjoys the merits of both heuristic search and importance sampling. Our approach is particularly useful for problem instances whose internal summation problems are challenging. It predicts high-quality MAP solutions along with their estimated values. It runs in an anytime/anyspace manner, which gives flexible trade-offs between memory, time, and solution quality.

### Acknowledgements

We thank all the reviewers for their helpful feedback. We also thank Wei Ping for assistance with the experiments.

This work is sponsored in part by NSF grants IIS-1526842 and IIS-1254071, the U.S. Air Force (Contract FA9453-16-C-0508), and DARPA (Contract W911NF-18-C-0015).

## References

- Qiang Cheng, Feng Chen, Jianwu Dong, Wenli Xu, and Alexander Ihler. Approximating the sum operation for marginal-MAP inference. In *AAAI*, pages 1882–1887, 2012.
- Anan Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.
- Rina Dechter and Robert Mateescu. AND/OR search spaces for graphical models. *Artificial Intelligence*, 171(2-3):73–106, 2007.
- Rina Dechter, Hector Geffner, and Joseph Y Halpern. *Heuristics, Probability and Causality. A Tribute to Judea Pearl*. College Publications, 2010.
- Rina Dechter. Reasoning with probabilistic and deterministic graphical models: Exact algorithms. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 7(3):1–191, 2013.
- Arnaud Doucet, Simon J. Godsill, and Christian P. Robert. Marginal maximum a posteriori estimation using Markov chain Monte Carlo. *Statistics and Computing*, 12(1):77–84, 2002.
- Vibhav Gogate and Rina Dechter. Sampling-based lower bounds

- for counting queries. *Intelligenza Artificiale*, 5(2):171–188, 2011.
- Igor Kiselev and Pascal Poupart. Policy optimization by marginal-MAP probabilistic inference in generative models. In *AAMAS*, pages 1611–1612, 2014.
- Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- Junkyu Lee, Radu Marinescu, and Rina Dechter. Applying search based probabilistic inference algorithms to probabilistic conformant planning: Preliminary results. In *ISAIM*, 2016.
- Junkyu Lee, Radu Marinescu, Rina Dechter, and Alexander Ihler. From exact to anytime solutions for marginal MAP. In *AAAI*, pages 3255–3262, 2016.
- Qiang Liu and Alexander Ihler. Bounding the partition function using Hölder’s inequality. In *ICML*, pages 849–856, 2011.
- Qiang Liu and Alexander Ihler. Variational algorithms for marginal MAP. *Journal of Machine Learning Research*, 14(1):3165–3200, 2013.
- Qiang Liu, John W. Fisher, III, and Alexander Ihler. Probabilistic variational bounds for graphical models. In *NIPS*, pages 1432–1440, 2015.
- Qi Lou, Rina Dechter, and Alexander Ihler. Anytime anysace AND/OR search for bounding the partition function. In *AAAI*, pages 860–867, 2017.
- Qi Lou, Rina Dechter, and Alexander Ihler. Dynamic importance sampling for anytime bounds of the partition function. In *NIPS*, pages 3198–3206, 2017.
- Qi Lou, Rina Dechter, and Alexander Ihler. Anytime anysace and/or best-first search for bounding marginal MAP. In *AAAI*, 2018.
- Radu Marinescu, Rina Dechter, and Alexander Ihler. AND/OR search for marginal MAP. In *UAI*, pages 563–572, 2014.
- Radu Marinescu, Junkyu Lee, Alexander Ihler, and Rina Dechter. Anytime best+ depth-first search for bounding marginal MAP. In *AAAI*, pages 3775–3782, 2017.
- Denis Mauá and Cassio de Campos. Anytime marginal maximum a posteriori inference. In *ICML*, pages 1395–1402, 2012.
- Andreas Maurer and Massimiliano Pontil. Empirical Bernstein bounds and sample variance penalization. In *COLT*, 2009.
- Volodymyr Mnih, Hugo Larochelle, and Geoffrey Hinton. Conditional restricted Boltzmann machines for structured output prediction. In *UAI*, pages 514–522, 2011.
- James Park and Adnan Darwiche. Solving MAP exactly using systematic search. In *UAI*, pages 459–468, 2003.
- James Park. MAP complexity results and approximation methods. In *UAI*, pages 388–396, 2002.
- Wei Ping and Alex Ihler. Belief propagation in conditional RBMs for structured prediction. In *AISTATS*, pages 1141–1149, 2017.
- Wei Ping, Qiang Liu, and Alexander Ihler. Marginal structured SVM with hidden variables. In *ICML*, pages 190–198, 2014.
- Wei Ping, Qiang Liu, and Alexander Ihler. Decomposition bounds for marginal MAP. In *NIPS*, pages 3267–3275, 2015.
- L.G. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8(2):189 – 201, 1979.
- M.J. Wainwright and M. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305, 2008.
- Yexiang Xue, Zhiyuan Li, Stefano Ermon, Carla P. Gomes, and Bart Selman. Solving marginal MAP problems with NP oracles and parity constraints. In *NIPS*, pages 1127–1135, 2016.
- Chen Yanover and Yair Weiss. Approximate inference and protein-folding. In *NIPS*, pages 1457–1464, 2002.
- Changhe Yuan and Eric A. Hansen. Efficient computation of jointree bounds for systematic MAP search. In *IJCAI*, pages 1982–1989, 2009.
- Changhe Yuan, Tsai-Ching Lu, and Marek J. Druzdzel. Annealed MAP. In *UAI*, pages 628–635, 2004.

---

# Acyclic Linear SEMs Obey the Nested Markov Property

---

**Ilya Shpitser**

Department of Computer Science  
Johns Hopkins University  
ilyas@cs.jhu.edu

**Robin J. Evans**

Department of Statistics  
University of Oxford  
evans@stats.ox.ac.uk

**Thomas S. Richardson**

Department of Statistics  
University of Washington  
thomasr@u.washington.edu

## Abstract

The conditional independence structure induced on the observed marginal distribution by a hidden variable directed acyclic graph (DAG) may be represented by a graphical model represented by mixed graphs called maximal ancestral graphs (MAGs). This model has a number of desirable properties, in particular the set of Gaussian distributions can be parameterized by viewing the graph as a path diagram. Models represented by MAGs have been used for causal discovery [22], and identification theory for causal effects [28].

In addition to ordinary conditional independence constraints, hidden variable DAGs also induce generalized independence constraints. These constraints form the nested Markov property [20]. We first show that acyclic linear SEMs obey this property. Further we show that a natural parameterization for all Gaussian distributions obeying the nested Markov property arises from a generalization of maximal ancestral graphs that we call maximal arid graphs (MArG). We show that every nested Markov model can be associated with a MArG; viewed as a path diagram this MArG parametrizes the Gaussian nested Markov model. This leads directly to methods for ML fitting and computing BIC scores for Gaussian nested models.

## 1 INTRODUCTION

Causal models associated with graphs have a long history in statistics, beginning with the seminal work of Wright in pedigree analysis [27], Haavelmo's work on simultaneous equations in econometrics [13] and the more recent synthesis of earlier work into a general causal modeling framework due to Pearl [17]. Causal graphical

models are widely used in a variety of disciplines, with many theoretical developments and applications.

An important parametric subclass of causal graphical models are the linear structural equation models with correlated errors (SEMs). In fact, Wright and to some extent Haavelmo's work was originally within the SEM class. SEMs are defined over a class of mixed graphs containing directed ( $\rightarrow$ ) edges representing direct causation, and bidirected ( $\leftrightarrow$ ) edges representing correlated errors. Mixed graphs of this type without directed cycles—an assumption that rules out cyclic causation—are called *acyclic directed mixed graphs* (ADMGs).

Given an ADMG  $\mathcal{G}$ , the *linear structural equation model with correlated errors (SEM)* associated with  $\mathcal{G}$  is formally defined as the set of multivariate normal distributions with covariance matrices of the form

$$\Sigma = (I - B)^{-T} \Omega (I - B)^{-1},$$

where  $\omega_{ij} = \omega_{ji} = 0$  unless  $i \leftrightarrow j$  exists in  $\mathcal{G}$ , and  $b_{ij} = 0$  unless  $i \rightarrow j$  exists in  $\mathcal{G}$ . The matrix  $\Omega$ —and therefore  $\Sigma$ —is assumed to be positive definite. We denote this set by  $\mathcal{P}_{\text{sem}}(\mathcal{G})$ , and the set of Gaussians with arbitrary covariances  $\mathcal{N}$ .

It is easy to show that this model is equivalent to assuming that each variable  $X_i$  is a linear function of its parents with coefficients  $b_{ji}$  together with an additive error term. The error terms are assumed to have a multivariate normal distribution with covariance matrix given by  $\Omega$ . If  $\Omega = I$ , error terms are uncorrelated and the SEM corresponds to a directed acyclic graph (DAG).

Elements of  $\mathcal{P}_{\text{sem}}(\mathcal{G})$  are known to obey the global Markov property for  $\mathcal{G}$  given by a criterion called *m-separation* [15, 19, 23]; this is the natural extension of d-separation to mixed graphs—see the Appendix for a definition. This criterion implies that absences of certain edges in  $\mathcal{G}$  correspond to conditional independences in elements of  $\mathcal{P}_{\text{sem}}(\mathcal{G})$ . Densities that obey this global Markov property are said to be in the *ordinary Markov model* of  $\mathcal{G}$ , a set of densities we denote  $\mathcal{P}_o(\mathcal{G})$  [9].

Hence  $\mathcal{P}_{\text{sem}}(\mathcal{G}) \subseteq \mathcal{P}_o(\mathcal{G}) \cap \mathcal{N}$ ; that is elements of the SEM for  $\mathcal{G}$  are multivariate normal and are in the ordinary Markov model of  $\mathcal{G}$ .

Given a DAG  $\mathcal{D}$  with observed variables  $O$  and hidden variables  $H$ , a simple operation, called the latent projection [26], maps it to an ADMG  $\mathcal{G}$  with only observed variables  $O$ , such that d-separation applied to any variables in  $O$  in  $\mathcal{D}$ , and m-separation applied to  $\mathcal{G}$  yields the same set of conditional independence relations on  $O$ . Thus, distributions Markov relative to a hidden variable DAG yield marginal distributions in  $\mathcal{P}_o(\mathcal{G})$  for a latent projection  $\mathcal{G}$ .

However, more recent work has shown that these marginal distributions also obey certain *generalized conditional independence constraints*, sometimes called *Verma constraints* [6]. These define a model known as the *nested Markov model*, also associated with  $\mathcal{G}$ , and denoted  $\mathcal{P}_n(\mathcal{G})$  [20]; generally  $\mathcal{P}_n(\mathcal{G}) \subseteq \mathcal{P}_o(\mathcal{G})$ , since the nested model implies all the constraints of m-separation. In this paper we show that distributions in the SEM for  $\mathcal{G}$  also obey the additional constraints of the nested Markov model, so  $\mathcal{P}_{\text{sem}}(\mathcal{G}) \subseteq \mathcal{P}_n(\mathcal{G}) \cap \mathcal{N}$ .

Although well-studied, general SEMs possess many complexities that make them potentially difficult to work with. The models may not be everywhere identifiable, and may contain singularities that prevent convergence of fitting algorithms [5]. No general distributional equivalence result is available for SEMs; see [25] for recent developments. In addition, while characterization of identifiability of causal effects is known for non-parametric structural equations [14, 21], a similar result is not known for SEMs despite decades of work [1, 2, 3, 4, 8, 11, 12, 24].

It is known that SEMs are everywhere identified if and only if they are associated with ADMGs in a special class [7]; in this paper we call this class *arid graphs*. We show that in a further subclass called *maximal arid graphs* (MARGs), it is the case that  $\mathcal{P}_{\text{sem}}(\mathcal{G}) = \mathcal{P}_n(\mathcal{G}) \cap \mathcal{N}$ . Moreover, we show that restricting to maximal arid graphs is without loss of generality in the sense that for any ADMG  $\mathcal{G}$ , there exists a maximal arid graph  $\mathcal{G}^\dagger$  such that  $\mathcal{P}_n(\mathcal{G}) = \mathcal{P}_n(\mathcal{G}^\dagger)$ . We also provide an algorithm for obtaining this *maximal arid projection* from  $\mathcal{G}$ , and show that  $\mathcal{G}^\dagger$  has the same ancestral relations as  $\mathcal{G}$ .

Our results immediately imply that the nested Markov model over multivariate normal densities is a curved exponential family of known dimension, and is everywhere identifiable. They also imply that Gaussian nested models can be fitted efficiently with existing algorithms, such as RICF [5], applied to the SEM associated with  $\mathcal{G}^\dagger$ . Conversely, our results imply that every SEM obeys all the generalized independence constraints implied by  $\mathcal{P}_n(\mathcal{G})$ .

MARGs form a natural subclass of ADMGs for the purposes of nested Markov model search methods, which could be used for causal discovery. This would be a more powerful alternative to model search with maximal ancestral graphs (MAGs), since nested models are more fine-grained and therefore make more unambiguous causal information available. Though the results in this paper make significant progress towards causal structure learning with nested Markov models, more work is required. In particular, a natural next step would be to fully describe equivalence classes of nested Markov models, and develop a constraint based model search algorithm that is akin to the FCI algorithm [22], but that also takes generalized conditional independence constraints into account.

The remainder of the paper is organized as follows. Section 2 gives some preliminary definitions, including that of acyclic directed mixed graphs (ADMGs). In Section 3 we define the nested Markov model associated with ADMGs formally, including the central notion of ‘fixing’. Section 4 shows that the class of nested models can be represented, without loss of generality, by the class of maximal arid graphs. Section 5 characterizes fixing in terms of zeroes of SEM parameters; this leads to the result in Section 6, which shows that for maximal arid graphs, the nested model and SEM coincide. We conclude with an example in Section 7, and discussion in Section 8. The proofs of certain results that are not essential to the presentation are deferred to the Appendix.

## 2 PRELIMINARIES

In this paper, we consider mixed graphs with directed ( $\rightarrow$ ) and bidirected ( $\leftrightarrow$ ) arrows connecting pairs of distinct vertices. There is at most one edge of each type between any pair of vertices, and we forbid directed cycles (i.e. sequences of the form  $v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_k \rightarrow v_1$  for  $k \geq 2$ ). Graphs in this class are called *acyclic directed mixed graphs* (ADMGs). ADMGs may contain *bows*, where both  $a \rightarrow b$  and  $a \leftrightarrow b$ , but this is the only circumstance in which more than one edge may be present between two vertices. See Fig. 2, 3 and 4 for examples of ADMGs.

We will use standard genealogical terminology for relations between vertices. Given a vertex  $v$  in an ADMG  $\mathcal{G}$  with a vertex set  $V$ , define the sets of *parents*, *children*, *ancestors*, *descendants*, and *siblings* of  $v$  as

$$\begin{aligned} \text{pa}_{\mathcal{G}}(v) &\equiv \{w : w \rightarrow v \text{ in } \mathcal{G}\} \\ \text{ch}_{\mathcal{G}}(v) &\equiv \{w : v \rightarrow w \text{ in } \mathcal{G}\} \\ \text{an}_{\mathcal{G}}(v) &\equiv \{w : w = v \text{ or } w \rightarrow \dots \rightarrow v \text{ in } \mathcal{G}\} \\ \text{de}_{\mathcal{G}}(v) &\equiv \{w : w = v \text{ or } v \rightarrow \dots \rightarrow w \text{ in } \mathcal{G}\} \\ \text{sib}_{\mathcal{G}}(v) &\equiv \{w : w \leftrightarrow v \text{ in } \mathcal{G}\}. \end{aligned}$$

respectively. Define also the *non-descendants* of  $v$  to be  $\text{nd}_{\mathcal{G}}(v) \equiv V \setminus \text{de}_{\mathcal{G}}(v)$ . The definitions apply disjointly to sets, e.g. for a set of vertices  $W \subseteq V$ ,  $\text{pa}_{\mathcal{G}}(W) \equiv \bigcup_{w \in W} \text{pa}_{\mathcal{G}}(w)$ . In addition, we define the *district* of  $v$  to be the set

$$\text{dis}_{\mathcal{G}}(v) \equiv \{w : w \leftrightarrow \dots \leftrightarrow v \text{ in } \mathcal{G}\}.$$

The set of districts of an ADMG  $\mathcal{G}$ , which we denote by  $\mathcal{D}(\mathcal{G})$ , always partitions the set of vertices in  $\mathcal{G}$ .

An internal vertex  $v$  on a path is a *collider* (on the path) if both adjacent edges have an arrowhead at  $v$ . A path from  $w$  to  $v$  in  $\mathcal{G}$  is called a *collider path* if every internal vertex is a collider on the path. For example  $w \rightarrow z \leftrightarrow m \leftarrow v$  is a collider path, while  $w \rightarrow z \rightarrow m \rightarrow v$  is not.

Given an ADMG  $\mathcal{G}$ , and a subset  $S$  of vertices  $V$  in  $\mathcal{G}$ , the *induced subgraph*  $\mathcal{G}_S$  is the graph with vertex set  $S$ , and those edges in  $\mathcal{G}$  between elements in  $S$ . A set  $S$  is called *bidirected-connected* in  $\mathcal{G}$  if  $\mathcal{D}(\mathcal{G}_S)$  contains a single set.

### 3 NESTED MARKOV MODELS

Nested Markov models are a class of graphical models associated with ADMGs, and defined by generalized independence constraints. We consider random variables  $X_V \equiv (X_v : v \in V)$  taking values in the product space  $\mathfrak{X}_V = \times_{v \in V} \mathfrak{X}_v$ , for finite dimensional sets  $\mathfrak{X}_v$ . For any  $A \subseteq V$  we denote the subset  $(X_v : v \in A)$  by  $X_A$ .

A *kernel*  $q_V(x_V | x_W)$  is a collection of densities over  $X_V$ , indexed by  $x_W \in \mathfrak{X}_W$ . Conditional densities are kernels, but not all kernels are obtained by conditioning; we give some examples later. Conditioning and marginalization are defined in the usual way in kernels.

A joint density  $p(x_V)$  over  $X_V$  is said to be *nested Markov* with respect to an ADMG  $\mathcal{G}$  if it obeys certain independence constraints in kernels derived from  $p(x_V)$  using a ‘fixing’ operation. These constraints are implied by the m-separation criterion applied to *conditional ADMGs* (*CADMGs*) obtained from  $\mathcal{G}$  by an analogous fixing operation. We now define these terms, and the nested Markov model, precisely.

A CADMG  $\mathcal{G}(V, W)$  is an ADMG with a set of *random* vertices  $V$  and *fixed* vertices  $W$ , with the property that  $\text{sib}_{\mathcal{G}}(w) \cup \text{pa}_{\mathcal{G}}(w) = \emptyset$  for every  $w \in W$ . An example can be found in Fig. 1(b); note that we depict fixed vertices with rectangular nodes, and random vertices with round nodes. Vertices  $V$  in a CADMG correspond to random variables, as in standard graphical models, while vertices in  $W$  correspond to variables that were fixed to a specific value by some operation, such as conditioning or causal interventions. The genealogical relations in

Section 2 generalize in a straightforward way to CADMGs by ignoring the distinction between  $V$  and  $W$ ; the only exception is that districts are only defined for random vertices, so  $\mathcal{D}(\mathcal{G}(V, W))$  partitions  $V$ .

#### 3.1 Fixing

A vertex  $r \in V$  is said to be *fixable* in a CADMG  $\mathcal{G}(V, W)$  if  $\text{dis}_{\mathcal{G}}(r) \cap \text{de}_{\mathcal{G}}(r) = \emptyset$ . Given a CADMG  $\mathcal{G}(V, W)$ , and a fixable  $r \in V$ , the fixing operation  $\phi_r(\mathcal{G})$  yields a new CADMG  $\tilde{\mathcal{G}}(V \setminus \{r\}, W \cup \{r\})$  obtained from  $\mathcal{G}(V, W)$  by removing all edges of the form  $\rightarrow r$  and  $\leftrightarrow r$ , and keeping all other edges. Given a kernel  $q_V(x_V | x_W)$  associated with a CADMG  $\mathcal{G}(V, W)$ , and a fixable  $r \in V$ , the fixing operation  $\phi_r(q_V; \mathcal{G})$  yields a new kernel

$$\tilde{q}_{V \setminus \{r\}}(x_{V \setminus \{r\}} | x_W, x_r) \equiv \frac{q_V(x_V | x_W)}{q_V(x_r | x_{\text{nd}_{\mathcal{G}}(r)})}.$$

A sequence  $r_1, \dots, r_k$  of vertices in  $V$  is said to be *fixable* if  $r_1$  is fixable in  $\mathcal{G}$ ,  $r_2$  is fixable in  $\phi_{r_1}(\mathcal{G})$ , etc. A result in [20] states that for any  $p(x_V) \in \mathcal{P}_n(\mathcal{G})$ , two valid fixing sequences on the same set of variables  $R$  yield the same CADMG and kernel. We therefore unambiguously define

$$\phi_R(\mathcal{G}) \equiv \phi_{r_k}(\dots \phi_{r_2}(\phi_{r_1}(\mathcal{G})) \dots),$$

and similarly the kernel  $\phi_R(p; \mathcal{G})$ .

If a fixing sequence exists for a set  $R \subseteq V$  in  $\mathcal{G}(V, W)$ , we say  $V \setminus R$  is a *reachable set*. Such a set is called *intrinsic* if the vertices in  $V \setminus R$  are bidirected-connected (so that  $\mathcal{D}(\phi_R(\mathcal{G}))$  has a single element). We denote the collections of reachable and intrinsic sets in  $\mathcal{G}$  respectively by  $\mathcal{R}(\mathcal{G})$  and  $\mathcal{I}(\mathcal{G})$ .

For any  $v \in V$ , such that  $\text{ch}_{\mathcal{G}}(v) = \emptyset$ , the *Markov blanket* of  $v$  in a CADMG  $\mathcal{G}(V, W)$  is defined as

$$\text{mb}_{\mathcal{G}}(v) \equiv (\text{dis}_{\mathcal{G}}(v) \cup \text{pa}_{\mathcal{G}}(\text{dis}_{\mathcal{G}}(v))) \setminus \{v\},$$

this is the set of vertices that are connected to  $v$  by collider paths. For brevity, we will denote  $\text{mb}_{\phi_{V \setminus S}(\mathcal{G})}(v)$  by  $\text{mb}_{\mathcal{G}}(v, S)$ .

We are now ready to define the nested Markov model  $\mathcal{P}_n(\mathcal{G})$ . Given an ADMG  $\mathcal{G}$ , let  $\prec$  be any topological ordering on  $V$ . A distribution  $p(x_V)$  is in the nested Markov model associated with  $\mathcal{G}$  if, for each intrinsic  $S \subseteq V$  and  $\prec$ -maximal  $v \in S$ ,

$$X_v \perp\!\!\!\perp X_{V \setminus (\{v\} \cup \text{mb}_{\mathcal{G}}(v, S))} \mid X_{\text{mb}_{\mathcal{G}}(v, S)}$$

holds in  $\phi_{V \setminus S}(p(x_V); \mathcal{G})$ . This is known as the *ordered local Markov property* for nested models. As a consequence, under the nested model, fixing  $r$

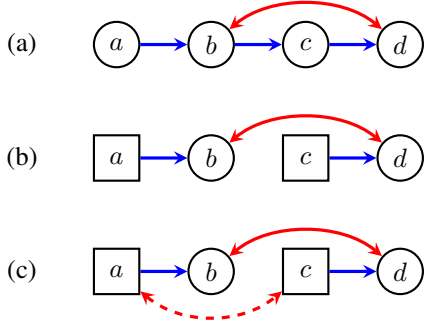


Figure 1: (a) An ADMG  $\mathcal{G}$  that is not ancestral; (b) a CADMG obtained from  $\mathcal{G}$  in (a) by fixing  $a$  and  $c$ ; (c) the graph obtained from (b) that is used to check conditional independence statements associated with  $\phi_{\{a,c\}}(p(a,b,c,d); \mathcal{G})$ .

within any  $R \in \mathcal{R}(\mathcal{G})$  may be redefined as dividing by  $q_R(x_r | x_{\text{mb}_{\mathcal{G}}(r,R)})$ , instead of dividing by  $q_R(x_r | x_{\text{nd}_{\mathcal{G}}(r)})$  (see section 2.11 in [20]). Nested models can be equivalently defined by a *global nested Markov property* obtained by applying the m-separation criterion to each reachable graph  $\phi_{V \setminus S}(\mathcal{G})$  after adding bidirected edges between all pairs of fixed vertices; adding these bidirected edges ensures no independences are implied between vertices in  $W$ . These m-separations imply independences in the kernel  $\phi_{V \setminus S}(p; \mathcal{G})$ ; see [20].

**Example 1.** Consider the ADMG in Fig. 1(a). The vertices  $a$ ,  $c$  and  $d$  all satisfy the condition of being fixable, but  $b$  does not since  $d$  is both a descendant of, and in the same district as,  $b$ . The CADMG  $\mathcal{G}(\{b, d\}, \{a, c\})$  obtained after fixing  $a$  and  $c$  is shown in Fig. 1(b). Notice that fixing  $c$  removes the edge  $b \rightarrow c$ , but that the edge  $c \rightarrow d$  is preserved. Applying m-separation to the graph shown in Fig. 1 (c), obtained from Fig. 1 (b) by connecting  $a, c$  by a bidirected edge, yields

$$X_d \perp\!\!\!\perp X_a \mid X_c \text{ in } \phi_{\{a,c\}}(p(x_{\{a,b,c,d\}}); \mathcal{G}).$$

In addition, one can see easily that if an edge  $a \rightarrow d$  had been present in the original graph, then we would not have obtained this m-separation.

## 4 ARID GRAPHS

The main result of this section is that the nested Markov model associated with *any* ADMG  $\mathcal{G}$  can be associated, without loss of generality, with a closely related *maximal arid graph* (MARG)  $\mathcal{G}^\dagger$ .

Arid graphs lack certain structures called C-trees [21] (aka convergent arborescences [7]) that present difficulties for identifiability. As a result, any linear SEM associated with an arid graph is everywhere identifiable [7].

In addition, maximal arid graphs are analogous to (but a strict superset of) maximal ancestral graphs (MAGs), a class of ADMGs also used for causal discovery.

The section proceeds as follows. In Section 4.1 we define the reachable closure of a set which is the smallest reachable superset of that set. These structures will be used in the proofs of our results, and to define C-trees and (maximal) arid graphs in Section 4.2. In Section 4.3 we define a projection operation which constructs, for any ADMG, its maximal arid graph counterpart. In Section 4.4, we show a number of useful graphical properties remain invariant between the original ADMG, and its maximal arid graph, leading to the proof of our main result in Section 4.5.

### 4.1 Reachable Closures

For a CADMG  $\mathcal{G}(V, W)$ , a (reachable) subset  $C \subseteq V$  is called a *reachable closure* for  $S \subseteq C$  if the set of fixable vertices in  $\phi_{V \setminus C}(\mathcal{G})$  is a subset of  $S$ . Every set  $S$  in  $\mathcal{G}$  has a reachable closure.

**Proposition 2.** *If  $A, B \in \mathcal{R}(\mathcal{G})$ , then  $A \cap B \in \mathcal{R}(\mathcal{G})$ .*

*Proof.* This follows from the fact that if a vertex is fixable, it remains fixable after fixing other vertices (see Lemma 27 of [20]).  $\square$

**Proposition 3.** *For any set of random vertices  $S$  in a CADMG  $\mathcal{G}$ , there is a unique reachable closure.*

*Proof.* Assume there are two such distinct closures  $W_1, W_2$ . Since both  $W_1$  and  $W_2$  are reachable, so is  $W_1 \cap W_2$ , by Proposition 2. Since  $S \subseteq W_1$  and  $S \subseteq W_2$ ,  $S \subseteq W_1 \cap W_2$ . Consider a fixing sequence  $\sigma_1$  for  $V \setminus W_1$ . Then there exists a fixing sequence  $\sigma_2$  for  $V \setminus (W_1 \cap W_2)$  which contains  $\sigma_1$  as a prefix. Note that  $W_1$  being reachable implies that  $W_1 \not\subseteq W_2$ , by the same argument as in the proof of Proposition 2; hence  $\sigma_2$  is non-empty. But this implies  $W_1$  is not a reachable closure for  $S$ , since the next element in  $\sigma_2$  after the  $\sigma_1$  prefix cannot lie in  $S$ . This is a contradiction.  $\square$

In light of Proposition 3, we denote the unique reachable closure of a set  $S$  in  $\mathcal{G}$  by  $\langle S \rangle_{\mathcal{G}}$ . By definition  $\langle S \rangle_{\mathcal{G}} \in \mathcal{R}(\mathcal{G})$  for any  $S$ , and if  $S \in \mathcal{R}(\mathcal{G})$  then  $\langle S \rangle_{\mathcal{G}} = S$ . To avoid clutter, if  $S = \{s\}$ , we write  $\langle \{s\} \rangle_{\mathcal{G}}$  as  $\langle s \rangle_{\mathcal{G}}$ .

**Proposition 4.** *Let  $A \subseteq B$  with  $B$  a reachable set; then  $\langle A \rangle_{\phi_{V \setminus B}(\mathcal{G})} = \langle A \rangle_{\mathcal{G}}$ .*

**Lemma 5.**  $\langle S \rangle_{\mathcal{G}} \subseteq S \cup \text{pa}_{\mathcal{G}}(\langle S \rangle_{\mathcal{G}})$ .

*Proof.* If  $s \in \langle S \rangle_{\mathcal{G}} \setminus S$  then  $s$  has a child in  $\langle S \rangle_{\mathcal{G}}$  since otherwise  $s$  is fixable, which is a contradiction.  $\square$

## 4.2 C-Trees and Arid Graphs

For any  $v \in V$  in an ADMG  $\mathcal{G}$ , the induced subgraph  $\mathcal{G}_{\langle v \rangle}$  is called a  $v$ -rooted C-tree [21] or an *arborescence converging on  $v$*  [7]. These subgraphs are particularly important because of the following result.

**Theorem 6** ([7], Theorem 2). *The SEM for an ADMG  $\mathcal{G}$  is everywhere identifiable if and only if  $\langle v \rangle_{\mathcal{G}} = \{v\}$  for all  $v \in V$ .*

In other words, the SEM parameterization is identifiable everywhere if and only if  $\mathcal{G}$  does not contain any non-trivial converging arborescences. We call such graphs ‘arid’, since they do not contain such ‘trees’.

**Definition 7.** An ADMG  $\mathcal{G}$  is called *arid* if for every vertex  $v$  in  $\mathcal{G}$ ,  $\langle v \rangle_{\mathcal{G}} = \{v\}$ .

Arid ADMGs are “DAG-like,” in the sense that in any DAG  $\mathcal{G}$ , it is also the case that  $\langle v \rangle_{\mathcal{G}} = \{v\}$ . The central result of this section is that the nested Markov model  $\mathcal{P}_n(\mathcal{G})$ , a statistical model with desirable properties, may be associated without loss of generality with arid graphs.

The ordinary Markov model  $\mathcal{P}_o(\mathcal{G})$  has previously been associated with another special class of ADMGs called *ancestral graphs* in [18].

**Definition 8.** An ADMG  $\mathcal{G}$  is called *ancestral* if for every vertex  $v$  in  $\mathcal{G}$ ,  $\text{sib}_{\mathcal{G}}(v) \cap \text{an}_{\mathcal{G}}(v) = \emptyset$ .

Arid graphs may be viewed as a strict generalization of ancestral graphs of [18], due to the following property of C-trees.

**Proposition 9.** *If  $\langle a \rangle_{\mathcal{G}}$  contains more than one element, then there exists  $b \in \langle a \rangle_{\mathcal{G}}$  with  $a \leftrightarrow b$  in  $\mathcal{G}$ .*

*Proof.* If no such element exists then every element in  $\text{pa}_{\mathcal{G}}(a) \cap \langle a \rangle_{\mathcal{G}}$  is fixable in  $\phi_{V \setminus \langle a \rangle_{\mathcal{G}}}(\mathcal{G})$ , which is a contradiction.  $\square$

**Proposition 10.** *Ancestral graphs are arid.*

*Proof.* Follows immediately by the contrapositive application of Proposition 9 and  $\langle a \rangle_{\mathcal{G}} \subseteq \text{an}_{\mathcal{G}}(a)$ .  $\square$

**Proposition 11.** *An arid graph with at least two vertices contains at least two fixable vertices.*

*Proof.* Since the graph is acyclic, there is some childless  $v$  that is therefore fixable. Since the graph is arid,  $\langle v \rangle_{\mathcal{G}} = \{v\} \subset V$ , and so there is also some other vertex that can be fixed to make  $\{v\}$  reachable.  $\square$

## 4.3 Maximal Arid Projection

To prove that  $\mathcal{P}_n(\mathcal{G})$  can always be associated with an arid graph, we define the *maximal arid projection* operation which, for every ADMG  $\mathcal{G}$ , yields a closely related

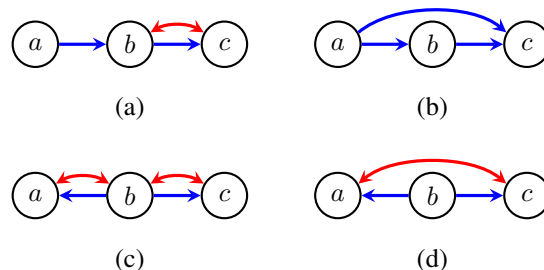


Figure 2: Graphs illustrating maximal arid projection. The graphs (a) and (c) are not arid, but have maximal arid projections given by (b) and (d) respectively.

graph  $\mathcal{G}^\dagger$  that is arid, and ultimately show that  $\mathcal{G}$  and  $\mathcal{G}^\dagger$  yield the same nested model.

In this section we define this projection operation and derive several of its properties, culminating in a proof that the projection and fixing operations commute. We first need a preliminary definition.

**Definition 12.** A pair of vertices  $a \neq b$  in an ADMG  $\mathcal{G}$  is *densely connected* if either  $a \in \text{pa}_{\mathcal{G}}(\langle b \rangle_{\mathcal{G}})$ , or  $b \in \text{pa}_{\mathcal{G}}(\langle a \rangle_{\mathcal{G}})$ , or  $\langle \{a, b\} \rangle_{\mathcal{G}}$  is a bidirected-connected set.

A CADMG  $\mathcal{G}$  is called *maximal* if every pair of densely connected vertices in  $\mathcal{G}$  are adjacent.

Densely connected vertex pairs form the nested Markov analogue of *inducing paths* [26]. Just as the existence of an inducing path between a pair of vertices prevents m-separation by any set, so does the existence of dense connectedness between a pair of vertices prevents m-separation by any set within any CADMG corresponding to a reachable set. In effect, a densely connected pair cannot be made independent, by any combination of conditioning and fixing operations.

**Definition 13.** For a CADMG  $\mathcal{G}$ , we define the *maximal arid projection* of  $\mathcal{G}$ , denoted  $\mathcal{G}^\dagger$ , to be the graph that shares the vertex sets  $V, W$  with  $\mathcal{G}$ , and that contains the following edges:

- for  $b \in V$ , the edge  $a \rightarrow b$  exists in  $\mathcal{G}^\dagger$  if  $a \in \text{pa}_{\mathcal{G}}(\langle b \rangle_{\mathcal{G}})$ ,
- for  $a, b \in V$ , the edge  $a \leftrightarrow b$  exists in  $\mathcal{G}^\dagger$  if neither  $a \in \text{pa}_{\mathcal{G}}(\langle b \rangle_{\mathcal{G}})$ , nor  $b \in \text{pa}_{\mathcal{G}}(\langle a \rangle_{\mathcal{G}})$ , but  $\langle \{a, b\} \rangle_{\mathcal{G}}$  is a bidirected-connected set.

Fig. 2 provides some elementary examples of the maximal arid projection. In each of (a) and (c) we have a dense inducing path between the vertices  $a$  and  $c$ . For (a) we insert the edge  $a \rightarrow c$  to represent this (yielding (b)), while in (c) we add  $a \leftrightarrow c$  (yielding (d)). In each case the bow arcs are replaced by directed edges.

We provide several results characterizing the output of the maximal arid projection operation, first noting that pairs of vertices adjacent in  $\mathcal{G}$  are also adjacent in  $\mathcal{G}^\dagger$ .

**Proposition 14.**

- (i) If  $a \in \text{pa}_{\mathcal{G}}(b)$ , then  $a \in \text{pa}_{\mathcal{G}^\dagger}(b)$ .
- (ii) If  $a \in \text{sib}_{\mathcal{G}}(b)$ , then either  $a \in \text{pa}_{\mathcal{G}^\dagger}(b)$  or  $a \in \text{sib}_{\mathcal{G}^\dagger}(b)$  or  $b \in \text{pa}_{\mathcal{G}^\dagger}(a)$ .

Ancestral relationships are also preserved in  $\mathcal{G}^\dagger$ .

**Proposition 15.**  $a \in \text{an}_{\mathcal{G}}(b)$  if and only if  $a \in \text{an}_{\mathcal{G}^\dagger}(b)$ .

*Proof.* If  $a \in \text{an}_{\mathcal{G}}(b)$ , then  $a \in \text{an}_{\mathcal{G}^\dagger}(b)$  follows by an inductive application of Proposition 14(i). If  $a \in \text{an}_{\mathcal{G}^\dagger}(b)$ , then fix a directed path  $a \rightarrow w_1 \rightarrow \dots \rightarrow w_k \rightarrow b$  in  $\mathcal{G}^\dagger$ . Each directed edge on this path from  $c$  to  $d$  is due to  $c \in \text{pa}_{\mathcal{G}}(\langle d \rangle_{\mathcal{G}})$  being true. But since every element  $\langle d \rangle_{\mathcal{G}}$  is an ancestor of  $d$  in  $\mathcal{G}$ , this implies the existence of a directed path from  $c$  to  $d$  in  $\mathcal{G}$ . Thus, there is a directed path from  $a$  to  $b$  in  $\mathcal{G}$ .  $\square$

**Proposition 16.** If  $\mathcal{G}$  is a (C)ADMG, then so is  $\mathcal{G}^\dagger$ .

*Proof.* Acyclicity of  $\mathcal{G}^\dagger$  follows from Proposition 15; in addition, in a CADMG it is clear from the definition that no arrowheads are introduced into  $W$ .  $\square$

If  $\mathcal{G}$  is acyclic then  $\mathcal{G}^\dagger$  is simple, i.e. contains at most one edge between each pair of vertices, so if  $\mathcal{G}$  is an ADMG then  $\mathcal{G}^\dagger$  is an example of a bow-free acyclic path diagram (BAP) [5, 16].

**Proposition 17.**  $\mathcal{D}(\mathcal{G}^\dagger)$  is a sub-partition of  $\mathcal{D}(\mathcal{G})$ . Further, for any  $S$  reachable in  $\mathcal{G}$  and  $\mathcal{G}^\dagger$ ,  $\mathcal{D}(\phi_{V \setminus S}(\mathcal{G}^\dagger))$  forms a sub-partition of  $\mathcal{D}(\phi_{V \setminus S}(\mathcal{G}))$ .

Note that it will follow from Theorem 19 that if  $S$  is reachable in  $\mathcal{G}$  then it is also reachable in  $\mathcal{G}^\dagger$ .

**Lemma 18.** Let  $v$  be fixable in  $\mathcal{G}$ . For any  $a, b \in V$  there is a directed path from  $a$  to  $b$  in  $\mathcal{G}$  with no intermediate vertex being  $v$ , if and only if there is such a path in  $\mathcal{G}^\dagger$ .

**Theorem 19.** If  $S$  is reachable in an ADMG  $\mathcal{G}$ , then it is also reachable in  $\mathcal{G}^\dagger$  via the same fixing sequence. In this case,  $(\phi_{V \setminus S}(\mathcal{G}))^\dagger = \phi_{V \setminus S}(\mathcal{G}^\dagger)$ .

**Corollary 20.**  $\langle S \rangle_{\mathcal{G}^\dagger} \subseteq \langle S \rangle_{\mathcal{G}}$  for any set  $S$ .

**Proposition 21.**  $\mathcal{G}^\dagger$  is a maximal arid graph.

#### 4.4 Invariance Results In Maximal Arid Projections

A key result will be that the nested Markov model associated with a maximal arid projection is the same as that for the original graph, and this will be proven by showing that the Markov blankets in the two graphs are the same.

**Lemma 22.** Suppose that  $w \in \text{pa}_{\mathcal{G}}(\langle v \rangle_{\mathcal{G}})$ , and that  $\langle \{v, w\} \rangle_{\mathcal{G}}$  is bidirected-connected. Then  $\langle \{v, w\} \rangle_{\mathcal{G}} = \langle v \rangle_{\mathcal{G}}$  and in particular  $w \in \langle v \rangle_{\mathcal{G}}$ .

**Lemma 23.** If  $v, w \in V$  are connected by a collider path  $\pi$  in  $\mathcal{G}$  then they are connected by a collider path  $\pi^\dagger$  in  $\mathcal{G}^\dagger$  that uses a subset of the internal vertices of  $\pi$ . In addition, if  $\pi$  starts with an edge  $v \rightarrow$ , then so does  $\pi^\dagger$ .

This follows by definition of  $\mathcal{G}^\dagger$ , and properties of closures of sets of vertices of size 1 and 2. A detailed proof is in the Appendix.

As an example of this result, notice that the path  $t \rightarrow x \leftrightarrow bp \leftrightarrow y$  in Fig. 4 (a) is replaced by  $t \rightarrow bp \leftrightarrow y$  in the maximal arid projection in Fig. 4 (b).

We provide a partial converse.

**Lemma 24.** If  $v, w \in V$  are connected by a collider path  $\pi^\dagger$  in  $\mathcal{G}^\dagger$ , then they are also connected by a collider path in  $\mathcal{G}$ .

*Proof.*  $\pi^\dagger$  is of the form  $\rightarrow \leftrightarrow \dots \leftrightarrow \leftarrow$  (possibly without the directed edges). Each  $\leftrightarrow$  represents a bidirected path in  $\mathcal{G}$ . A directed edge in  $\mathcal{G}^\dagger$ , say  $v \rightarrow t$ , represents an edge  $v \rightarrow s$  for  $s \in \langle t \rangle_{\mathcal{G}}$ . Since  $\langle t \rangle_{\mathcal{G}}$  is bidirected-connected, there is a path of the form  $v \rightarrow \leftrightarrow \dots \leftrightarrow t$  in  $\mathcal{G}$ . Concatenating these paths (and possibly shortening) gives another collider path.  $\square$

**Theorem 25.** Let  $S$  be a reachable set in  $\mathcal{G}$ . Then  $\text{mb}_{\mathcal{G}}(v, S) = \text{mb}_{\mathcal{G}^\dagger}(v, S)$ .

*Proof.* First note that  $v$  is childless in  $\phi_{V \setminus S}(\mathcal{G})$  if and only if it is so in  $\phi_{V \setminus S}(\mathcal{G}^\dagger)$ , so the statement is well-defined. Theorem 19 shows that it is enough to show this for  $S = V$ . The result is then a direct consequence of Lemmas 23 and 24, since the Markov blanket is just the set of vertices connected to  $v$  by collider paths.  $\square$

**Proposition 26.** There is a one-to-one correspondence between intrinsic sets in  $\mathcal{G}$  and in  $\mathcal{G}^\dagger$ .

**Remark 27.** The set  $H$  is referred to by [10] as the recursive head associated with  $S$ . A consequence of the argument in the proof above is that the discrete parameterization given by [10] is identical for  $\mathcal{G}$  and  $\mathcal{G}^\dagger$ .

An important result is that fixing corresponds to the same probabilistic operation in  $\mathcal{G}$  and  $\mathcal{G}^\dagger$ .

**Proposition 28.** If  $S \in \mathcal{R}(\mathcal{G})$ , then any fixing sequence  $\sigma$  for  $V \setminus S$  valid in  $\mathcal{G}$  consists of the same set of fixing operations when applied to  $p(x_V)$  using  $\mathcal{G}$  and when applied to  $p(x_V)$  using  $\mathcal{G}^\dagger$ .

*Proof.* Recall that fixing is division of  $\phi_W(p(x_V); \mathcal{G}) \equiv q_V(x_V | x_W)$  by  $q_V(x_v | x_{\text{mb}_{\mathcal{G}}(v, S) \cup W})$ . By Theorem 25,  $\text{mb}_{\mathcal{G}}(v, S) = \text{mb}_{\mathcal{G}^\dagger}(v, S)$ , and a simple induction gives the result.  $\square$



Thus, for any  $S \in \mathcal{R}(\mathcal{G}^\dagger)$ ,  $q_S$  is well defined without specifying the particular sequence of fixing operations in  $\mathcal{G}^\dagger$ . Some fixing sequences may be valid in  $\mathcal{G}^\dagger$  but not in  $\mathcal{G}$ ; however the kernels reached in  $\mathcal{G}^\dagger$  are related to those reachable in  $\mathcal{G}$  by the following result.

**Lemma 29.** *Suppose  $S \in \mathcal{I}(\mathcal{G}^\dagger)$ , and let  $S^\dagger = \langle S \rangle_{\mathcal{G}}$ . Let  $v$  be the maximal element of  $S$ . Any independences involving the full conditional of  $v$  hold in  $q_S$  if and only if they hold in  $q_{S^\dagger}$ .*

*Proof.* For simplicity assume  $S^\dagger = V$ , so we write  $\mathcal{G}$  and  $\mathcal{G}^\dagger$  in place of  $\phi_{V \setminus S^\dagger}(\mathcal{G})$  and  $\phi_{V \setminus S^\dagger}(\mathcal{G}^\dagger)$  respectively. This is justified by Theorem 19.

Suppose that  $s \in S^\dagger \setminus S$  is fixable in  $\mathcal{G}^\dagger$  but not in  $\mathcal{G}$ . Then  $s \in \text{dis}_{\mathcal{G}}(v)$  and is an ancestor of some  $r \in S$  such that  $\text{ch}_{\mathcal{G}}(r) = \emptyset$  (in both  $\mathcal{G}$  and  $\mathcal{G}^\dagger$ ).

The vertices  $r$  and  $v$  are connected by a collider path in  $\mathcal{G}$ , and so also are in  $\mathcal{G}^\dagger$ . Further, since they have no children in  $\mathcal{G}$ , they also have no children in  $\mathcal{G}^\dagger$ , and these paths are therefore made up entirely of bidirected edges in both graphs; in other words,  $r$  and  $v$  are in the same district in both graphs. Since  $s$  is fixable in  $\mathcal{G}^\dagger$  but not in  $\mathcal{G}$ , and since ancestor relations are preserved, it follows that  $s$  is in the same district as  $r$  and  $v$  in  $\mathcal{G}$ , but not in  $\mathcal{G}^\dagger$ .

Fixing  $s$  involves division by  $q_{S^\dagger}(x_s | x_{\text{mb}_{\mathcal{G}^\dagger}(s)})$ . Since  $v$  is in a different district to  $s$  and has no children, then  $v \notin \text{mb}_{\mathcal{G}^\dagger}(s)$ , and so by Lemma 10 of [20] we have  $q_{S^\dagger}(x_v | x_{S^\dagger \setminus \{v\}}, x_W) = q_{S^\dagger \setminus \{s\}}(x_v | x_{S^\dagger \setminus \{v\}}, x_W)$ . Any further vertices in  $S^\dagger \setminus S$  are also not in the same district as  $v$  for the same reason, so  $v$  never appears in their Markov blankets and hence this is also the same as the full conditional  $q_S(x_v | x_{S \setminus \{v\}}, x_{W \cup (S^\dagger \setminus S)})$ . The result follows.  $\square$

#### 4.5 Any ADMG And Its Maximal Arid Projection Define The Same Nested Model

We are now ready to state and prove the main result of this section.

**Theorem 30.**  $\mathcal{P}_n(\mathcal{G}) = \mathcal{P}_n(\mathcal{G}^\dagger)$ .

*Proof.* Let  $\prec$  be a topological order and consider any pair  $S, S^\dagger$  as defined in Proposition 26. We will show that the corresponding independences for the ordered local nested Markov property are equivalent. Let  $v$  be the  $\prec$ -maximal element of  $S$  (and therefore of  $S^\dagger$ ). Then the two independences are

$$(X_v \perp\!\!\!\perp X_{V \setminus (\text{mb}_{\mathcal{G}}(v, S) \cup \{v\})} \mid X_{\text{mb}_{\mathcal{G}}(v, S)})$$

in  $\phi_{V \setminus S}(p(x_V); \mathcal{G})$ , and

$$(X_v \perp\!\!\!\perp X_{V \setminus (\text{mb}_{\mathcal{G}^\dagger}(v, S^\dagger) \cup \{v\})} \mid X_{\text{mb}_{\mathcal{G}^\dagger}(v, S^\dagger)})$$

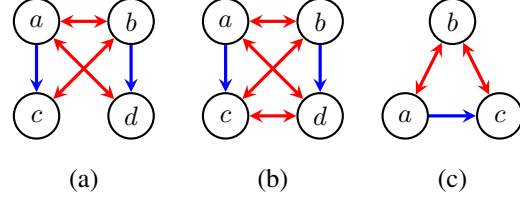


Figure 3: (a) A graph with a non-nested SEM constraint, and (b) A nested Markov equivalent graph. (c) A graph in which the parameter  $\omega_{bc}$  is not identifiable after any fixing.

in  $\phi_{V \setminus S^\dagger}(p(x_V); \mathcal{G}^\dagger)$ . Since—as follows from the proof of Proposition 26—we have  $\text{mb}_{\mathcal{G}}(v, S) = \text{mb}_{\mathcal{G}^\dagger}(v, S^\dagger)$ , it only remains to bridge the difference between the kernels. But this is an independence on the full conditional of  $\phi_{V \setminus S^\dagger}(p(x_V); \mathcal{G})$ , so by Lemma 29, it holds in that kernel if and only if it holds in  $\phi_{V \setminus S}(p(x_V); \mathcal{G}^\dagger)$ .  $\square$

## 5 THE FIXING OPERATION IN STRUCTURAL EQUATION MODELS

If  $v$  is fixable in an ADMG  $\mathcal{G}$ , the kernel  $q_{V \setminus \{v\}}$  resulting from fixing  $v$  is obtained by dividing  $p(x_V)$  by the conditional distribution  $p(x_v | x_{\text{nd}(v)})$ . Hence

$$\begin{aligned} q_{V \setminus \{v\}}(x_{V \setminus \{v\}} | x_v) \\ \equiv p(x_{\text{nd}(v)}) \cdot p(x_{\text{de}(v) \setminus \{v\}} | x_{\text{nd}(v) \cup \{v\}}), \end{aligned}$$

and therefore  $q_{V \setminus \{v\}}$  preserves both the marginal distribution of  $X_{\text{nd}(v)}$  and the conditional distribution of  $X_{\text{de}(v) \setminus \{v\}}$  given  $X_{\text{nd}(v) \cup \{v\}}$ .

**Remark 31.** A Gaussian kernel  $q(x_S | x_{V \setminus S})$  is parameterized via a set of means  $E[x_S | x_{V \setminus S}]$  indexed by  $x_{V \setminus S}$  and variances  $\text{Cov}[x_S | x_{V \setminus S}]$ . There is a distribution naturally associated with  $q(x_S | x_{V \setminus S})$  given by:

$$p_S^*(x_V) \equiv q_S(x_S | x_{V \setminus S}) \prod_{v \in V \setminus S} q_v^*(x_v),$$

where  $q_v^*(x_v)$  is an arbitrary marginal distribution.

In what follows we will consider kernels  $q_S(x_S | x_{V \setminus S})$  derived from a mean zero Gaussian distribution  $p(x_V)$ , hence parameterized via  $\text{Cov}[x_V]$ . We will then take  $q_v^*(x_v)$  to be the univariate normal distribution  $p(x_v)$ . It then follows that the Gaussian distribution  $p_S^*(x_V)$  corresponding to  $q_S(x_S | x_{V \setminus S})$  will also be parameterized via a covariance matrix  $\text{Cov}_S^*[x_V]$ .

**Proposition 32.** *Every conditional independence that holds in  $q_S$  also holds in  $p_S^*$ .*

We now show that fixing in the linear SEM corresponds to setting all coefficients corresponding to incoming edges to zero, but keeping all other parameters constant.

**Lemma 33.** *Let  $v$  be fixable in an ADMG  $\mathcal{G}$ . Then in an SEM corresponding to  $\mathcal{G}$ , setting  $b_{wv} = \omega_{vw} = \omega_{wv} = 0$  for all  $w \neq v$  is equivalent to dividing by  $p(x_v | x_{\text{nd}(v)})/q_v^*(x_v)$ .*

*Proof.* We show that setting parameters to zero as indicated leaves the marginal distribution  $p(x_{\text{nd}(v)})$  and the conditional distribution  $p(x_{\text{de}(v) \setminus \{v\}} | x_{\text{nd}(v)}, x_v)$  unchanged. The former follows easily from the trek rule (see, for example, [7]), since no edge in any trek between non-descendants of  $v$  is altered. Similarly, since we choose  $\text{Var } X_v = \omega_{vv}$  (see Remark 31), and the only trek from  $v$  to itself in the fixed graph is the trivial trek; hence  $\omega_{vv}$  is also preserved.

It remains to show that the same holds for the conditional distribution of the strict descendants given  $\{v\} \cup \text{nd}(v)$ . To see this, note that it is equivalent to check that the concentration  $k_{ij} = (\Sigma^{-1})_{ij}$  remains the same whenever either  $i$  or  $j$  is a descendant of  $v$ . Without loss of generality we may assume that  $j$  has no descendants (by marginalizing anything which is not an ancestor of  $i, j, v$ ). Then we have

$$K = \Sigma^{-1} = (I - B)^T \Omega^{-1} (I - B);$$

denote  $\omega^{ij} = (\Omega^{-1})_{ij}$ . By definition of the model,  $\Omega$  is a block-diagonal matrix with blocks corresponding to districts in the graph  $\mathcal{G}$ , and therefore so is  $\Omega^{-1}$ . Hence  $k_{ij}$  (including the case  $i = j$ ) can be written as

$$k_{ij} = \sum_{d=1}^p b_{id} \omega^{dj} + \omega^{ij} = \sum_{d \in \text{dis}(j)} b_{id} \omega^{dj} + \omega^{ij}, \quad (1)$$

(corresponding to paths of the form  $i \rightarrow d \leftrightarrow \dots \leftrightarrow j$  and  $i \leftrightarrow \dots \leftrightarrow j$  respectively). We claim none of the quantities in (1) are modified by setting the parameters  $b_{wv}$  and  $\omega_{vw} = \omega_{wv}$  to zero.

Suppose for a contradiction that  $b_{id}$  for  $d \in \text{dis}_{\mathcal{G}}(j)$  is one of the parameters set to zero; this could happen only if  $d = v$ . Now, if  $i$  is the descendant of  $v$  then this would imply a cycle, and if  $j$  is the descendant of  $v$  then  $d \in \text{dis}_{\mathcal{G}}(j)$  implies  $v$  is not fixable which is also a contradiction. Hence  $b_{id}$  is not set to zero.

Next consider  $\omega^{dj}$ . If  $d, j$  are in different districts then  $\omega^{dj} = 0$ . Since  $\Omega^{-1}$  is block diagonal, this parameter will only change if  $v$  is in the same district as  $j$  and  $d$ . If  $j$  is a descendant of  $v$  then  $v \notin \text{dis}_{\mathcal{G}}(j)$  since  $v$  is fixable. If  $i$  is a descendant of  $v$  then so is  $d$ , and therefore  $v \notin \text{dis}(j) = \text{dis}(d)$  for the same reason. Therefore the quantities  $\omega^{ij}, \omega^{dj}$  all remain unchanged, as they are a function only of the block of  $\Omega$  corresponding to  $\text{dis}(j)$ .

Hence if either  $i$  or  $j$  is a descendant of  $v$ , then none of the terms in (1) is changed by setting  $b_{wv} = \omega_{vw} = \omega_{wv} = 0$  for all  $w \neq v$ .  $\square$

Thus in the context of a linear SEM fixing  $v$  corresponds to setting the parameters  $b_{wv}$  and  $\omega_{vw} = \omega_{wv}$  to zero. We have the following result as a direct consequence:

**Theorem 34.** *Let  $\mathcal{G}$  be an ADMG then  $\mathcal{P}_{\text{sem}}(\mathcal{G}) \subseteq \mathcal{P}_n(\mathcal{G}) \cap \mathcal{N}$ .*

Recall that  $\mathcal{N}$  is the set of multivariate Gaussian distributions with positive definite covariance matrix.

## 6 ARID SEMS REPRESENT ALL GAUSSIAN NESTED MODELS

The Gaussian nested Markov model associated with an ADMG  $\mathcal{G}$  is exactly the linear SEM corresponding to the maximal arid projection  $\mathcal{G}^\dagger$  of  $\mathcal{G}$ :

**Theorem 35.** *Let  $\mathcal{G}$  be an ADMG. Then  $\mathcal{P}_{\text{sem}}(\mathcal{G}^\dagger) = \mathcal{P}_n(\mathcal{G}) \cap \mathcal{N}$ .*

*Proof.* By Theorem 34  $\mathcal{P}_{\text{sem}}(\mathcal{G}^\dagger) \subseteq \mathcal{P}_n(\mathcal{G}) \cap \mathcal{N}$ . Further, by Theorem 30,  $\mathcal{P}_n(\mathcal{G}) = \mathcal{P}_n(\mathcal{G}^\dagger)$ . Thus it suffices to prove that  $\mathcal{P}_n(\mathcal{G}) \cap \mathcal{N} \subseteq \mathcal{P}_{\text{sem}}(\mathcal{G})$  where  $\mathcal{G}$  is maximal and arid.

In order to facilitate our inductive argument, we extend the definitions of  $\mathcal{P}_n(\mathcal{G})$  and  $\mathcal{P}_{\text{sem}}(\mathcal{G})$  and the result to CADMGs and kernels. Specifically, if  $\mathcal{G}(V, W)$  is a CADMG and  $\prec$  is a topological ordering on  $V$ , then kernel  $q_V \in \mathcal{P}_n(\mathcal{G})$  if, for each intrinsic  $S \subseteq V$  and  $\prec$ -maximal  $v \in S$ ,

$$X_v \perp\!\!\!\perp X_{V \setminus (\{v\} \cup \text{mb}_{\mathcal{G}}(v, S))} \mid X_{\text{mb}_{\mathcal{G}}(v, S)}$$

holds in  $\phi_{V \setminus S}(q_V(x_V | x_W); \mathcal{G})$ . Similarly,  $\mathcal{P}_{\text{sem}}(\mathcal{G})$  represents a SEM where, if there is an edge  $w \rightarrow v$  with  $w \in W, v \in V$  then the equation for  $X_v$  contains  $b_{wv}x_w$  as a summand. We now claim that if  $\mathcal{G}$  is a CADMG then  $\mathcal{P}_{\text{sem}}(\mathcal{G}^\dagger) = \mathcal{P}_n(\mathcal{G}) \cap \mathcal{N}$ , where  $\mathcal{N}$  is the set of Gaussian kernels. This is clearly sufficient.

Suppose  $p \in \mathcal{P}_n(\mathcal{G}) \cap \mathcal{N}$ , where  $\mathcal{G}(V, W)$  is a CADMG with topological ordering  $\prec$ .

If  $|V| = 1$  then the result follows by regression on  $X_{\text{pa}_{\mathcal{G}}(v)}$ . Otherwise we proceed by induction on  $|V|$ . Let  $v$  be the maximal vertex under  $\prec$ .

For any fixable  $w$  in  $\mathcal{G}$  we obtain by the induction hypothesis that  $q_{V \setminus \{w\}} \in \mathcal{P}_n(\phi_w(\mathcal{G})) = \mathcal{P}_{\text{sem}}(\phi_w(\mathcal{G}))$ . Hence we can identify parameters for edges not involving  $v$  by fixing  $v$ . Any directed edge parameter  $b_{ij}$  can be identified provided  $j$  is not fixed; if  $|V| \geq 2$  then, since  $\mathcal{G}$  is arid, it contains at least two fixable vertices by Proposition 11. Hence we can identify every  $b_{ij}$  in this manner. [Since valid fixings commute, all such results will agree.]

Similarly we can identify any bidirected edge this way except possibly  $\omega_{vw}$  if  $w$  and  $v$  are the only two fixable vertices in  $\mathcal{G}$ . In this case,  $\mathcal{G}$  contains only one district and every vertex is an ancestor of  $v$  or  $w$ .

Since we have identified every other parameter, let  $\tilde{p}_\gamma$  be the distribution obtained from all the other parameters with  $\omega_{vw} = \gamma$ . Then by construction,  $\tilde{p}_\gamma$  and  $p$  have the same margins over  $V \setminus \{v\}$  and  $V \setminus \{w\}$ . If we can choose  $\gamma$  so that the covariance  $\sigma_{vw}$  matches that in  $p$ , then  $p = \tilde{p}_\gamma$ . It is not hard to see that  $\sigma_{vw}$  is a monotonic function of  $\omega_{vw}$ , so the only restriction is on the positive definiteness of the relevant covariance matrices. Since the set of positive definite matrices is convex, the set of valid  $\omega_{vw}$  is an interval; in addition,  $\Omega$  is positive definite if and only if  $\Sigma$  is positive definite. Hence, by the intermediate value theorem there exists a  $\gamma$  that maps to the appropriate  $\sigma_{vw}$ .  $\square$

**Example 36.** To see the difficulty with the induction in the previous proof, consider the graph in Fig. 3 (c). There are two fixable vertices,  $b$  and  $c$ , and in either case the fixing corresponds to marginalizing over the corresponding random variable. This means that, in either case, the edge parameter  $\omega_{bc}$  is not identifiable. Every other parameter can be identified inductively, and we may finally use  $\sigma_{bc}$  to identify  $\omega_{bc}$ .

## 7 EXAMPLE

Consider the following simplified medical trial to examine the effect of diet and exercise on diabetes, adapted from [5]. At baseline, patients are randomly assigned to perform  $t$  hours of exercise in a week, but actually perform  $x$  hours. At the end of the week their blood pressure (bp) is measured, this is assumed to depend upon  $x$ , but also to be confounded with it by lifestyle factors. In the second phase of the trial, patients are assigned to lose  $\Delta\text{bmi}$  kilograms in weight; the value of  $\Delta\text{bmi}$  is random, but for ethical reasons depends linearly on  $x$  and bp. Finally, at the end of the trial, triglyceride levels ( $y$ ) are measured, which is used to diagnose diabetes; these are assumed to be correlated with blood pressure, and dependent on exercise and weight loss. This causal structure naturally yields the ADMG shown in Fig. 4(a).

Though a perfectly reasonable causal description of the model, Fig. 4(a) contains a bow and therefore the associated model is non-smooth and not everywhere identifiable. Performing maximal arid projection gives the graph  $\mathcal{G}^\dagger$  in Fig. 4(b), which gives an SEM that induces a curved exponential family and is nested Markov equivalent to the SEM corresponding to the original graph. Note that the resulting graph is not an ancestral graph; indeed  $\mathcal{G}^\dagger$  preserves more of the structure than the corresponding MAG.

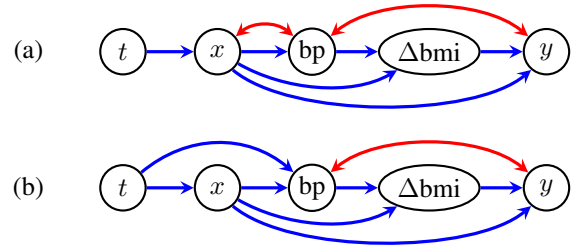


Figure 4: (a) A graph representing a clinical trial for interventions in diabetes; the associated SEM is non-smooth. (b) A nested Markov equivalent graph whose SEM represents a curved exponential family.

## 8 DISCUSSION

We have presented a subclass of ADMGs—the maximal arid graphs (MARGs)—that fully represents the class of nested Markov models. We have shown that any linear SEM associated with a MARG is precisely equal to the class of Gaussian densities in the nested Markov model for that MARG.

We remark that the results on arid graphs we derived in Section 4 are completely non-parametric, and apply not only to the Gaussian models that we study here, but to any model; we showed that any nested Markov equivalence class contains a MARG which, since MARGs are maximal and simple graphs, is a canonical representative of the class to use in search procedures within the set of nested models. In this sense MARGs serve a similar role to MAGs for scoring-based searches for ordinary Markov models corresponding to ADMGs.

### Acknowledgments

The authors would like to thank the American Institute of Mathematics for supporting this research via the SQuARE program. Evans was supported by an EPSRC First Grant. Shpitser was supported in part by NIH grants R01 AI104459-01A1 and R01 AI127271-01A1. Richardson was supported by the U.S. Office of Naval Research grant N00014-15-1-2672.

## References

- [1] C. Brito and J. Pearl. A graphical criterion for the identification of causal effects in linear models. In *Eighteenth National Conference on Artificial Intelligence*, pages 533–538, 2002.
- [2] C. Brito and J. Pearl. Graphical condition for identification in recursive SEM. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, pages 47–54, 2006.
- [3] B. Chen. Identification and overidentification of linear structural equation models. In *Advances in Neural Information Processing Systems*, volume 29, pages 1579–1587, 2016.
- [4] B. Chen, J. Tian, and J. Pearl. Testable implications of linear structural equation models. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 2424–2430, 2014.
- [5] M. Drton, M. Eichler, and T. S. Richardson. Computing maximum likelihood estimates in recursive linear models with correlated errors. *Journal of Machine Learning Research*, 10(Oct):2329–2348, 2009.
- [6] M. Drton, C. Fox, and A. Käuffl. Comments on: Sequences of regressions and their independencies. *Test*, 21(2):255–261, 2012.
- [7] M. Drton, R. Foygel, and S. Sullivant. Global identifiability of linear structural equation models. *Annals of Statistics*, 39(2):865–886, 2011.
- [8] M. Drton and L. Weihs. Generic identifiability of linear structural equation models by ancestor decomposition. *Scand. J. Statist.*, 2016.
- [9] R. J. Evans and T. S. Richardson. Markovian acyclic directed mixed graphs for discrete data. *Annals of Statistics*, pages 1–30, 2014.
- [10] R. J. Evans and T. S. Richardson. Smooth, identifiable supermodels of discrete DAG models with latent variables. *Bernoulli*, 2018. (to appear).
- [11] R. Foygel, J. Draisma, and M. Drton. Half-trek criterion for generic identifiability of linear structural equation models. *Annals of Statistics*, 40(3):1682–1713, 2012.
- [12] L. D. Garcia-Puente, S. Spielvogel, and S. Sullivant. Identifying causal effects with computer algebra. In *Proceedings of the Twenty-sixth Conference on Uncertainty in Artificial Intelligence (UAI)*. AUAI Press, 2010.
- [13] T. Haavelmo. The statistical implications of a system of simultaneous equations. *Econometrica*, 11:1–12, 1943.
- [14] Y. Huang and M. Valtorta. Pearl’s calculus of interventions is complete. In *Twenty Second Conference On Uncertainty in Artificial Intelligence*, 2006.
- [15] J. T. Koster. On the validity of the Markov interpretation of path diagrams of gaussian structural equations systems with correlated errors. *Scandinavian Journal of Statistics*, 26(3):413–431, 1999.
- [16] C. Nowzohour, M. H. Maathuis, R. J. Evans, and P. Bühlmann. Distributional equivalence and structure learning for bow-free acyclic path diagrams. *Electronic Journal of Statistics*, 11(2):5342–5374, 2017.
- [17] J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2 edition, 2009.
- [18] T. Richardson and P. Spirtes. Ancestral graph Markov models. *Annals of Statistics*, 30:962–1030, 2002.
- [19] T. S. Richardson. Markov properties for acyclic directed mixed graphs. *Scandinavian Journal of Statistics*, 30(1):145–157, 2003.
- [20] T. S. Richardson, R. J. Evans, J. M. Robins, and I. Shpitser. Nested Markov properties for acyclic directed mixed graphs. Working paper, <https://arxiv.org/abs/1701.06686v2>, 2017.
- [21] I. Shpitser and J. Pearl. Identification of joint interventional distributions in recursive semi-Markovian causal models. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI-06)*. AAAI Press, Palo Alto, 2006.
- [22] P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. Springer Verlag, New York, second edition, 2001.
- [23] P. Spirtes, T. S. Richardson, C. Meek, R. Scheines, and C. Glymour. Using path diagrams as a structural equation modeling tool. *Sociological Methods & Research*, 27(2):182–225, 1998.
- [24] J. Tian. Parameter identification in a class of linear structural equation models. In *Proceedings of the Twenty-first International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1970–1975. AAAI Press, Palo Alto, CA, 2009.
- [25] T. van Ommen and J. M. Mooij. Algebraic equivalence of linear structural equation models. In *Proceedings of the 33rd Annual Conference on Uncertainty in Artificial Intelligence (UAI-17)*, 2017.
- [26] T. S. Verma and J. Pearl. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence (UAI-90)*, 1990.

- [27] S. Wright. Correlation and causation. *Journal of Agricultural Research*, 20:557–585, 1921.
- [28] J. Zhang. Generalized do-calculus with testable causal assumptions. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, pages 667–674, 2007.

---

# A Unified Particle-Optimization Framework for Scalable Bayesian Sampling

---

Changyou Chen

University at Buffalo (cchangyou@gmail.com)

Ruiyi Zhang, Wenlin Wang, Bai Li, Liqun Chen

Duke University

## Abstract

There has been recent interest in developing scalable Bayesian sampling methods such as stochastic gradient MCMC (SG-MCMC) and Stein variational gradient descent (SVGD) for big-data analysis. A standard SG-MCMC algorithm simulates samples from a discrete-time Markov chain to approximate a target distribution, thus samples could be highly correlated, an undesired property for SG-MCMC. In contrary, SVGD directly optimizes a set of particles to approximate a target distribution, and thus is able to obtain good approximations with relatively much fewer samples. In this paper, we propose a principle particle-optimization framework based on Wasserstein gradient flows to unify SG-MCMC and SVGD, and to allow new algorithms to be developed. Our framework interprets SG-MCMC as particle optimization on the space of probability measures, revealing a strong connection between SG-MCMC and SVGD. The key component of our framework is several particle-approximate techniques to efficiently solve the original partial differential equations on the space of probability measures. Extensive experiments on both synthetic data and deep neural networks demonstrate the effectiveness and efficiency of our framework for scalable Bayesian sampling.

## 1 INTRODUCTION

Bayesian methods have been playing an important role in modern machine learning, especially in unsupervised learning (Kingma and Welling, 2014; Li et al., 2017), and recently in deep reinforcement learning (Houthoofd et al., 2016; Liu et al., 2017). When dealing with big data, two lines of research directions have been developed to scale

up Bayesian methods, *e.g.*, variational-Bayes-based and sampling-based methods. Stochastic gradient Markov chain Monte Carlo (SG-MCMC) is a family of scalable Bayesian learning algorithms designed to efficiently sample from a target distribution such as a posterior distribution (Welling and Teh, 2011; Chen et al., 2014; Ding et al., 2014; Chen et al., 2015). In principle, SG-MCMC generates samples from a Markov chain, which are used to approximate a target distribution. Under a standard setting, samples from SG-MCMC are able to match a target distribution exactly with an infinite number of samples (Teh et al., 2016; Chen et al., 2015). However, this is practically infeasible, as only a finite number of samples are obtained. Although nonasymptotic approximation bounds w.r.t. the number of samples have been investigated (Teh et al., 2016; Vollmer et al., 2016; Chen et al., 2015), there are no theory/algorithms to guide learning an optimal set of fixed-size samples/particles. This is an undesirable property of SG-MCMC, because in practice one often seeks to learn the optimal samples of a finite size that best approximate a target distribution.

A remedy for this issue is to adopt the idea of particle-based sampling methods, where a set of particles (or samples) are initialized from some simple distribution, followed by iterative updates to better approximate a target distribution. The updating procedure is usually done by optimizing some metrics such as a distance measure between the target distribution and the current approximation. There is not much work in this direction for large-scale Bayesian sampling, with an outstanding representative being the Stein variational gradient descent (SVGD) (Liu and Wang, 2016a). In SVGD, the update of particles are done by optimizing the KL-divergence between the empirical particle distribution and a target distribution, thus the samples are designed to be updated optimally to reduce the KL-divergence in each iteration. Because of this property, SVGD is found to perform better than SG-MCMC when the number of samples used to approximate a target distribution is limited, and has been

applied to other problems such as deep generative models (Feng et al., 2017) and deep reinforcement learning (Liu et al., 2017; Haarnoja et al., 2017; Zhang et al., 2018b).

Though often achieving comparable performance in practice, little work has been done on investigating connections between SG-MCMC and SVGD, and on developing particle-optimization schemes for SG-MCMC. In this paper, adopting ideas from Wasserstein-gradient-flow literature, we propose a unified particle-optimization framework for scalable Bayesian sampling. The idea of our framework is to work directly on the evolution of a density functions on the space of probability measures, *e.g.*, the Fokker-Planck equation in SG-MCMC. To make the evolution solution computationally feasible, particle approximations are adopted for densities, where particles can be optimized during the evolution process. Both SG-MCMC and SVGD are special cases of our framework, and are shown to be highly related. Notably, sampling with SG-MCMC becomes a deterministic particle-optimization problem as SVGD on the space of probability measures, overcoming the aforementioned correlated-sample issue. Furthermore, we are able to develop new *unified* particle-optimization algorithms by combing SG-MCMC and SVGD, which is less prone to high-dimension space and thus obtains better performance for large-scale Bayesian sampling. We conduct extensive experiments on both synthetic data and Bayesian learning of deep neural networks, verifying the effectiveness and efficiency of our proposed framework.

## 2 PRELIMINARIES

In this section, we review related concepts and algorithms for SG-MCMC, SVGD, and Wasserstein gradient flows (WGF) on the space of probability measures.

### 2.1 Stochastic gradient MCMC

**Diffusion-based sampling methods** Generating random samples from a distribution (*e.g.*, a posterior distribution) is one of the fundamental problems in Bayesian statistics, which has many important applications in machine learning. Traditional Markov Chain Monte Carlo methods (MCMC), such as the Metropolis–Hastings algorithm (Metropolis et al., 1953) produces unbiased samples from a desired distribution when the density function is known up to a normalizing constant. However, most of these methods are based on random walk proposals which suffer from high dimensionality and often lead to highly correlated samples. On the other hand, dynamics-based sampling methods such as the Metropolis adjusted Langevin algorithm (MALA) (Xifara et al., 2014) avoid this high degree of correlation by combining dynamical

systems with the Metropolis step. In fact, these dynamical systems are derived from a more general mathematical technique called diffusion process, or more specifically, Itô diffusion (Øksendal, 1985).

Specifically, our objective is to generate random samples from a posterior distribution  $p(\boldsymbol{\theta} | \mathbf{X}) \propto p(\mathbf{X} | \boldsymbol{\theta})p(\boldsymbol{\theta})$ , where  $\boldsymbol{\theta} \in \mathbb{R}^r$  represents the model parameter, and  $\mathbf{X} \triangleq \{\mathbf{x}_i\}_{i=1}^N$  represents the data. The canonical form is  $p(\boldsymbol{\theta} | \mathbf{X}) = (1/Z) \exp(U(\boldsymbol{\theta}))$ , where  $U(\boldsymbol{\theta}) =$

$$\log p(\mathbf{X} | \boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) \triangleq \sum_{i=1}^N \log p(\mathbf{x}_i | \boldsymbol{\theta}) + \log p(\boldsymbol{\theta})$$

is referred to as the potential energy based on an i.i.d. assumption of the model, and  $Z$  is the normalizing constant. In Bayesian sampling, the posterior distribution corresponds to the (marginal) stationary distribution of a (continuous-time) Itô diffusion, defined as a stochastic differential equation of the form:

$$d\boldsymbol{\Theta}_t = F(\boldsymbol{\Theta}_t)dt + g(\boldsymbol{\Theta}_t)d\mathcal{W}_t, \quad (1)$$

where  $t$  is the time index;  $\boldsymbol{\Theta}_t \in \mathbb{R}^p$  represents the full variables in a dynamical system, and  $\boldsymbol{\Theta}_t \supseteq \boldsymbol{\theta}_t$  (thus  $p \geq r$ ) is potentially an augmentation of model parameter  $\boldsymbol{\theta}$ ;  $\mathcal{W}_t \in \mathbb{R}^p$  is  $p$ -dimensional Brownian motion. Functions  $F : \mathbb{R}^p \rightarrow \mathbb{R}^p$  and  $g : \mathbb{R}^p \rightarrow \mathbb{R}^p \times \mathbb{R}^p$  are assumed to satisfy the Lipschitz continuity condition (Ghosh, 2011). By Fokker-Planck equation (or the forward Kolmogorov equation) (Kolmogoroff, 1931; Risken, 1989), when appropriately designing the diffusion-coefficient functions  $F(\cdot)$  and  $g(\cdot)$ , the stationary distribution of the corresponding Itô diffusion equals the posterior distribution of interest,  $p(\boldsymbol{\theta} | \mathbf{X})$ . For example, the 1st-order Langevin dynamic defines  $\boldsymbol{\Theta} = \boldsymbol{\theta}$ , and  $F(\boldsymbol{\Theta}_t) = \frac{1}{2}\nabla_{\boldsymbol{\theta}}U(\boldsymbol{\theta})$ ,  $g(\boldsymbol{\Theta}_t) = \mathbf{I}_r$ ; the 2nd-order Langevin diffusion defines  $\boldsymbol{\Theta} = (\boldsymbol{\theta}, \mathbf{q})$ , and  $F(\boldsymbol{\Theta}_t) = \begin{pmatrix} \mathbf{q} \\ -B\mathbf{q} - \nabla_{\boldsymbol{\theta}}U(\boldsymbol{\theta}) \end{pmatrix}$ ,  $g(\boldsymbol{\Theta}_t) = \sqrt{2B} \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_r \end{pmatrix}$  for a scalar  $B > 0$ ;  $\mathbf{q}$  is an auxiliary variable known as the momentum (Chen et al., 2014; Ding et al., 2014).

Let the density of  $\boldsymbol{\Theta}_t$  be  $\mu_t$ , it is known  $\mu_t$  is characterized by the Fokker-Planck (FP) equation (Risen, 1989):

$$\frac{\partial \mu_t}{\partial t} = -\nabla_{\boldsymbol{\Theta}} \cdot (\mu_t F(\boldsymbol{\Theta}_t)) + \nabla_{\boldsymbol{\Theta}} \nabla_{\boldsymbol{\Theta}} : (\mu_t \Sigma(\boldsymbol{\Theta}_t)) \quad (2)$$

where  $\Sigma(\boldsymbol{\Theta}_t) \triangleq g(\boldsymbol{\Theta}_t)g^\top(\boldsymbol{\Theta}_t)$ ,  $\mathbf{a} \cdot \mathbf{b} \triangleq \mathbf{a}^\top \mathbf{b}$  for vectors  $\mathbf{a}$  and  $\mathbf{b}$ ,  $\mathbf{A} : \mathbf{B} \triangleq \text{trace}(\mathbf{A}^\top \mathbf{B})$  for matrices  $\mathbf{A}$  and  $\mathbf{B}$ . The FP equation is the key to develop our particle-optimization framework for SG-MCMC. In the following, we focus on the simplest case of 1st-order Langevin dynamics if not stated explicitly, though the derivations apply to other variants.

**Stochastic gradient MCMC** SG-MCMC algorithms are discretized numerical approximations of the Itô dif-

fusion (1). They mitigate the slow mixing and non-scalability issues encountered in traditional MCMC algorithms by *i*) adopting gradient information of the posterior distribution, *ii*) using minibatches of the data in each iteration of the algorithm to generate samples, and *iii*) ignoring the rejection step as in standard MCMC. To make the algorithms scalable in a big-data setting, three developments will be implemented based on the Itô diffusion: *i*) define appropriate functions  $F$  and  $g$  in the Itô-diffusion formula so that the (marginal) stationary distributions coincide with the target posterior distribution  $p(\boldsymbol{\theta} | \mathbf{X})$ ; *ii*) replace  $F$  or  $g$  with unbiased stochastic approximations to reduce the computational complexity, *e.g.*, approximating  $F$  with a random subset of the data instead of using the full data. For example, in the 1st-order Langevin dynamics,  $\nabla_{\boldsymbol{\theta}} U(\boldsymbol{\theta})$  could be *approximated* by an unbiased estimator with a subset of data:

$$\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}) \triangleq \nabla \log p(\boldsymbol{\theta}) + \frac{N}{n} \sum_{i=1}^n \nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}_{\pi_i} | \boldsymbol{\theta}) \quad (3)$$

where  $\pi$  is a size- $n$  random subset of  $\{1, 2, \dots, N\}$ , leading to the first SG-MCMC algorithm in machine learning – stochastic gradient Langevin dynamics (SGLD) (Welling and Teh, 2011); and *iii*) solve the generally intractable continuous-time Itô diffusions with a numerical method, *e.g.*, the Euler method (Chen et al., 2015). For example, this leads to the following update in SGLD:

$$\boldsymbol{\theta}_{\ell} = \boldsymbol{\theta}_{\ell-1} + \nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_{\ell-1})h + \sqrt{2h} \boldsymbol{\delta}_{\ell},$$

where  $h$  means the stepsize,  $\ell$  indexes the samples,  $\boldsymbol{\delta}_{\ell} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  is a random sample from an isotropic normal distribution. After running the algorithm for  $L$  steps, the collection of samples  $\{\boldsymbol{\theta}_{\ell}\}_{\ell=1}^L$  are used to approximate the unknown posterior distribution  $\frac{1}{L} e^{U(\boldsymbol{\theta})}$ .

## 2.2 Stein variational gradient descent

Different from SG-MCMC, SVGD initializes a set of particles which are iteratively updated so that the empirical particle distribution approximates the posterior distribution. Specifically, we consider a set of particles  $\{\boldsymbol{\theta}^{(i)}\}_{i=1}^M$  drawn from some distribution  $q$ . SVGD tries to update these particles by doing gradient descent on the interactive particle system via

$$\boldsymbol{\theta}^{(i)} \leftarrow \boldsymbol{\theta}^{(i)} + h\phi(\boldsymbol{\theta}^{(i)}), \quad \phi = \arg \max_{\phi \in \mathcal{F}} \left\{ \frac{\partial}{\partial h} \text{KL}(q_{[h\phi]} || p) \right\}$$

where  $\phi$  is a function perturbation direction chosen to minimize the KL divergence between the updated density  $q_{[h\phi]}$  estimated by the particles and the posterior  $p(\boldsymbol{\theta} | \mathbf{X})$  ( $p$  for short). Since  $\text{KL}(q || p)$  is convex in  $q$ , global optimum of  $q = p$  can be guaranteed. SVGD considers  $\mathcal{F}$  as the unit ball of a vector-valued reproducing kernel Hilbert

space (RKHS)  $\mathcal{H}$  associated with a kernel  $\kappa(\boldsymbol{\theta}, \boldsymbol{\theta}')$ . In such as setting, Liu and Wang (2016b) shown:

$$-\frac{\partial}{\partial h} \text{KL}(q_{[h\phi]} || p)|_{h=0} = \mathbb{E}_{\boldsymbol{\theta} \sim q} [\text{trace}(\Gamma_p \phi(\boldsymbol{\theta}))], \quad (4)$$

with  $\Gamma_p \phi(\boldsymbol{\theta}) \triangleq \nabla_{\boldsymbol{\theta}} U(\boldsymbol{\theta})^{\top} \phi(\boldsymbol{\theta}) + \nabla_{\boldsymbol{\theta}} \cdot \phi(\boldsymbol{\theta})$ ,

where  $\Gamma_p$  is called the Stein operator. Assuming that the update function  $\phi(\boldsymbol{\theta})$  is in a RKHS with kernel  $\kappa(\cdot, \cdot)$ , it was shown in (Liu and Wang, 2016b) that (4) is maximized with:

$$\phi(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{\theta} \sim q} [\kappa(\boldsymbol{\theta}, \boldsymbol{\theta}') \nabla_{\boldsymbol{\theta}} U(\boldsymbol{\theta}) + \nabla_{\boldsymbol{\theta}} \kappa(\boldsymbol{\theta}, \boldsymbol{\theta}')]. \quad (5)$$

When approximating the expectation  $\mathbb{E}_{\boldsymbol{\theta} \sim q}[\cdot]$  with empirical particle distribution and adopting stochastic gradients, we arrive at the following updates for the particles ( $\ell$  denotes the iteration number):  $\boldsymbol{\theta}_{\ell+1}^{(i)} = \boldsymbol{\theta}_{\ell}^{(i)} +$

$$\frac{h}{M} \sum_{j=1}^M \left[ \kappa(\boldsymbol{\theta}_{\ell}^{(j)}, \boldsymbol{\theta}_{\ell}^{(i)}) \nabla_{\boldsymbol{\theta}_{\ell}^{(j)}} \tilde{U}(\boldsymbol{\theta}_{\ell}^{(j)}) + \nabla_{\boldsymbol{\theta}_{\ell}^{(j)}} \kappa(\boldsymbol{\theta}_{\ell}^{(j)}, \boldsymbol{\theta}_{\ell}^{(i)}) \right] \quad (6)$$

SVGD applies updates (6) repeatedly, moving the samples to a target distribution  $p$ .

## 2.3 Wasserstein Gradient Flows

For a better motivation of WGF, we start from gradient flows defined on the Euclidean space.

**Gradient flows on the Euclidean space** For a smooth function  $E : \mathbb{R}^r \rightarrow \mathbb{R}$ , and a starting point  $\boldsymbol{\theta}_0 \in \mathbb{R}^r$ , the gradient flow of  $E(\boldsymbol{\theta})$  is defined as the solution of the differential equation:  $\frac{d\boldsymbol{\theta}}{dt} = -\nabla E(\boldsymbol{\theta}(t))$ , s.t.  $\boldsymbol{\theta}(0) = \boldsymbol{\theta}_0$ . This is a standard Cauchy problem (Rulla, 1996), endowed with a unique solution if  $\nabla E$  is Lipschitz continuous. When  $E$  is non-differentiable, the gradient is replaced with its subgradient, defined as  $\partial E(\boldsymbol{\theta}) \triangleq \{\mathbf{p} \in \mathbb{R}^r : F(\boldsymbol{\theta}') \geq F(\boldsymbol{\theta}) + \mathbf{p} \cdot (\boldsymbol{\theta}' - \boldsymbol{\theta}), \forall \boldsymbol{\theta}' \in \mathbb{R}^r\}$ . Note  $\partial E(\boldsymbol{\theta}) = \{\nabla E(\boldsymbol{\theta})\}$  if  $E$  is differentiable at  $\boldsymbol{\theta}$ . In this case, the gradient flow formula above is replaced with:  $\frac{d\boldsymbol{\theta}}{dt} \in -\partial E(\boldsymbol{\theta}(t))$ .

**Wasserstein gradient flows** Let  $\mathcal{P}(\Omega)$  denote the space of probability measures on  $\Omega \subset \mathbb{R}^r$ . WGF is an extension of gradient flows in Euclidean space by lifting the definition onto the space of probability measures. Formally, let  $\mathcal{P}(\Omega)$  be endowed with a Riemannian geometry induced by the 2nd-order Wasserstein distance, *i.e.*, the curve length between two elements (two distributions) is defined as:

$$W_2^2(\mu, \nu) \triangleq \inf_{\gamma} \left\{ \int_{\Omega \times \Omega} \|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_2^2 d\gamma(\boldsymbol{\theta}, \boldsymbol{\theta}') : \gamma \in \Gamma(\mu, \nu) \right\}$$

where  $\Gamma(\mu, \nu)$  is the set of joint distributions over  $(\boldsymbol{\theta}, \boldsymbol{\theta}')$  such that the two marginals equal  $\mu$  and  $\nu$ , respectively. The Wasserstein distance can be explained as an optimal-transport problem, where one wants to transform elements



in the domain of  $\mu$  to  $\nu$  with a minimum cost (Villani, 2008). The term  $\|\theta - \theta'\|_2^2$  represents the cost to transport  $\theta$  in  $\mu$  to  $\theta'$  in  $\nu$ , and can be replaced by a general metric  $c(\theta, \theta')$  in a metric space. If  $\mu$  is absolutely continuous w.r.t. the Lebesgue measure, there is a unique optimal transport plan from  $\mu$  to  $\nu$ , i.e., a mapping  $\mathcal{T} : \mathbb{R}^r \rightarrow \mathbb{R}^r$  pushing elements in the domain of  $\mu$  onto  $\nu$  satisfying  $\mathcal{T}_\# \mu = \nu$ . Here  $\mathcal{T}_\# \mu$  denotes the pushforward measure of  $\mu$ . The Wasserstein distance is equivalently reformulated as:  $W_2^2(\mu, \nu) \triangleq \inf_{\mathcal{T}} \left\{ \int_{\Omega} c(\theta, \mathcal{T}(\theta)) d\mu(\theta) \right\}$ .

Let  $\{\mu_t\}_{t \in [0,1]}$  be an absolutely continuous curve in  $\mathcal{P}(\Omega)$  with finite second-order moments. We consider to define the change of  $\mu_t$ 's by investigating  $W_2^2(\mu_t, \mu_{t+h})$ . Motivated by the Euclidean-space case, this is reflected by a vector field,  $\mathbf{v}_t(\theta) \triangleq \lim_{h \rightarrow 0} \frac{\mathcal{T}(\theta_t) - \theta_t}{h}$  called the *velocity of the particle*. A gradient flow can be defined on  $\mathcal{P}(\Omega)$  correspondingly (Ambrosio et al., 2005).

**Lemma 1** *Let  $\{\mu_t\}_{t \in [0,1]}$  be an absolutely-continuous curve in  $\mathcal{P}(\Omega)$  with finite second-order moments. Then for a.e.  $t \in [0,1]$ , the above vector field  $\mathbf{v}_t$  defines a gradient flow on  $\mathcal{P}(\Omega)$  as:  $\partial_t \mu_t + \nabla \cdot (\mathbf{v}_t \mu_t) = 0$ .*

The gradient flow describes the evolution of a functional  $E$ , which is a lifted version of the function in the case of Euclidean space in Section 2.3 to the space of probability measures.  $E$  maps a probability measure  $\mu$  to a real value, i.e.,  $E : \mathcal{P}(\Omega) \rightarrow \mathbb{R}$ . We will focus on the case where  $E$  is convex in this paper, which is enough considering gradient flows for SG-MCMC and SVGD, though the theory applies to a more general  $\lambda$ -convex energy functional setting (Ambrosio et al., 2005). It can be shown that  $\mathbf{v}_t$  in Lemma 1 has the form  $\mathbf{v}_t = -\nabla \frac{\delta E}{\delta \mu_t}(\mu_t)$  (Ambrosio et al., 2005), where  $\frac{\delta E}{\delta \mu_t}$  is called the *first variation* of  $E$  at  $\mu_t$ . Based on this, gradient flows on  $\mathcal{P}(\Omega)$  can be written

$$\partial_t \mu_t = -\nabla \cdot (\mathbf{v}_t \mu_t) = \nabla \cdot \left( \mu_t \nabla \left( \frac{\delta E}{\delta \mu_t}(\mu_t) \right) \right). \quad (7)$$

**Remark 1** *Intuitively, an energy functional  $E$  characterizes the landscape structure (appearance) of the corresponding manifold in  $\mathcal{P}(\Omega)$ , and the gradient flow (7) defines a geodesic path on this manifold. Usually, by choosing appropriate  $E$ , the landscape is convex, e.g., for the cases of both SG-MCMC and SVGD described below. This provides a theoretical guarantee on the optimal convergence of a gradient flow.*

### 3 PARTICLE-OPTIMIZATION-BASED SAMPLING

In this section, we interpret the continuous versions of both SG-MCMC and SVGD as WGFs, followed by several techniques for particle optimization in the next section. In the following,  $\mu_t$  denotes the distribution of  $\theta_t$ .

#### 3.1 SVGD as WGF

The continuous-time and infinite-particle limit of SVGD with full gradients, denoted as  $\text{SVGd}^\infty$ , is known to be a special instance of the Vlasov equation in nonlinear partial-differential-equation literature (Liu, 2017):

$$\partial_t \mu_t = \nabla \cdot ((\mathbf{W} * \mu_t) \mu_t), \quad (8)$$

where  $(\mathbf{W} * \mu_t)(\theta) \triangleq \int \mathbf{W}(\theta - \theta') \mu_t(\theta') d\theta'$  is the convolutional operator applied for some function  $\mathbf{W} : \mathbb{R}^r \rightarrow \mathbb{R}$ . To specify  $\text{SVGd}^\infty$ , we generalize the convolutional operator, and consider  $\mathbf{W}$  as a function with two input arguments, i.e.,

$$(\mathbf{W} * \mu_t)(\theta) \triangleq \int \mathbf{W}(\theta, \theta') \mu_t(\theta') d\theta'.$$

Under this setting, we can specify the function  $\mathbf{W}(\cdot, \cdot)$  for  $\text{SVGd}^\infty$  as

$$\begin{aligned} \mathbf{W}(\theta, \theta') &\triangleq \nabla_{\theta'} \log p(\theta' | \mathbf{X}) \kappa(\theta', \theta) + \nabla_{\theta'} \kappa(\theta', \theta) \\ &= \nabla_{\theta'} [p(\theta' | \mathbf{X}) \kappa(\theta, \theta')] / p(\theta'). \end{aligned} \quad (9)$$

As will be shown in Section 4,  $\mathbf{W}$  in (9) naturally leads to the SVGD algorithm, without the need to derive from an RKHS perspective.

**Proposition 2** *The stationary distribution of (8) is  $\lim_{t \rightarrow \infty} \mu_t \triangleq \mu = p(\theta | \mathbf{X})$ .*

To interpret  $\text{SVGd}^\infty$  as a WGF, we need to specify two quantities, the energy functional and an underlying metric to measure distances between density functions.

#### Energy functional and distance metric of $\text{SVGd}^\infty$

There are two ways to derive energy functionals for  $\text{SVGd}^\infty$ , depending on the underlying metrics for probability distributions. When adopting the WGF framework where  $W_2$  is used as the underlying metric, according to (7), the energy functional  $E_s$  must satisfy

$$\begin{aligned} \nabla_{\theta} \left( \frac{\delta E_s}{\delta \mu_t}(\mu_t) \right) &= \mathbf{W}(\theta, \theta') * \mu_t \\ &= \mathbb{E}_{\theta' \sim \mu_t} [\nabla_{\theta'} [p(\theta' | \mathbf{X}) K(\theta, \theta')] / p(\theta' | \mathbf{X})]. \end{aligned} \quad (10)$$

In general, there is no close-form solution for the above equation. Alternatively, Liu (2017) proved another form of the energy functional by defining a different distance metric on the space of probability measures, called  $\mathcal{H}$ -Wasserstein distance:

$$W_{\mathcal{H}}(q_1, q_2) \triangleq \inf_{\phi_t, \mu_t} \left\{ \int_0^1 \|\phi_t\|_{\mathcal{H}} dt, \text{ s.t. } \mu_t = -\nabla_{\theta} \cdot (\phi_t \mu_t), \mu_0 = q_1, \mu_1 = q_2 \right\}, \quad (11)$$

where  $\phi_t \triangleq \mathbf{W} * \mu_t$ , and  $\|\cdot\|_{\mathcal{H}}$  is the norm in the Hilbert space induced by  $\kappa(\cdot, \cdot)$ . Under this metric, the underlying energy functional is proved to be the standard KL-divergence between  $\mu_t$  and  $p$ , e.g.,

$$E_s = \text{KL}(\mu_t, p(\cdot | \mathbf{X})).$$

As can be seen in Section 4, this interpretation allows one to derive SVGD, a particle-optimization-based algorithm to approximate the continuous-time equation (8).

### 3.2 SG-MCMC as WGF

The continuous-time limit of SG-MCMC, when considering gradients to be exact, corresponds to standard Itô diffusions. We consider the Itô diffusion of SGLD for simplicity, *e.g.*,

$$d\boldsymbol{\theta}_t = \frac{1}{2} \nabla U(\boldsymbol{\theta}_t) dt + d\mathcal{W}. \quad (12)$$

**Energy functional** The energy functional for SG-MCMC is easily seen by noting that the corresponding FP equation (2) is in the gradient-flow form of (7). Specifically, the energy functional  $E$  is defined as:

$$E(\mu) \triangleq \underbrace{- \int U(\boldsymbol{\theta}) \mu(\boldsymbol{\theta}) d\boldsymbol{\theta}}_{E_1} + \underbrace{\int \mu(\boldsymbol{\theta}) \log \mu(\boldsymbol{\theta}) d\boldsymbol{\theta}}_{E_2} \quad (13)$$

Note  $E_2$  is the energy functional of a pure Brownian motion (*e.g.*,  $U(\boldsymbol{\theta}) = 0$  in (12)). We can verify (13) by showing that it satisfies that FP equation. According to (7), the first variation of  $E_1$  and  $E_2$  is calculated as

$$\frac{\delta E_1}{\delta \mu} = -U, \quad \frac{\delta E_2}{\delta \mu} = \log \mu + 1. \quad (14)$$

Substituting (14) into (7) recovers the FP equation (2) for the Itô diffusion (12).

## 4 PARTICLE OPTIMIZATION

An efficient way to solve the generally infeasible WGF formula (7) is to adopt numerical methods with particle approximation. With a little abuse of notation but for conciseness, we do not distinguish subscripts  $t$  and  $\ell$  for the particle  $\boldsymbol{\theta}$ , *i.e.*,  $\boldsymbol{\theta}_t$  denotes the continuous-time version of the particle, while  $\boldsymbol{\theta}_\ell$  denotes the discrete-time version. We develop several techniques to approximate different types of WGF for SG-MCMC and SVGD. In particle approximation, the continuous density  $\mu_t$  is approximated by a set of  $M$  particles  $(\boldsymbol{\theta}_t^{(i)})_{i=1}^M$  that evolve over time  $t$  with weights  $(m_i)_{i=1}^M$  such that  $\sum_{i=1}^M m_i = 1$ , *i.e.*,  $\mu_t \approx \sum_{i=1}^M m_i \delta(\boldsymbol{\theta}_t^{(i)})$ , where  $\delta(\boldsymbol{\theta}_t^{(i)}) = 1$  when  $\boldsymbol{\theta} = \boldsymbol{\theta}_t^{(i)}$  and 0 otherwise. Typically  $m_i$ 's are chosen at the beginning and fixed over time, thus we assume  $m_i = \frac{1}{M}$  and rewrite  $\mu_t \approx \frac{1}{M} \sum_{i=1}^M \delta(\boldsymbol{\theta}_t^{(i)})$  in the following for simplicity. We investigate two types of particle-approximation methods in the following, discrete gradient flows and by blob methods.

#### Particle approximation by discrete gradient flows

Denote  $\mathcal{P}_s(\mathbb{R}^r)$  be the space of probability measures with finite 2nd-order moments. Define the following optimization problem with stepsize  $h$ :

$$J_h(\mu) \triangleq \arg \min_{\nu \in \mathcal{P}_s(\mathbb{R}^d)} \left\{ \frac{1}{2h} W_2^2(\mu, \nu) + E(\nu) \right\}. \quad (15)$$

A discrete gradient flow of the continuous one in (7) up to time  $T$  is the composition of a sequence of the solutions  $(\tilde{\mu}_\ell)_{\ell=1}^{T/h}$  of (15), *i.e.*,

$$\tilde{\mu}_\ell \triangleq J_h(\tilde{\mu}_{\ell-1}) = J_h(J_h(\cdots \mu_0)) \triangleq J_h^\ell \mu_0. \quad (16)$$

One can show that when  $h \rightarrow 0$ , the discrete gradient flow (16) converges to the true flow (7) for all  $\ell$ . Specifically, let  $\partial E(\mu)$  be the set of Wasserstein subdifferential of  $E$  at  $\mu$ , *i.e.*,  $\xi \in \partial E(\mu)$  if  $\partial_t \mu = \xi$  is satisfied. Define  $|\partial E|(\mu) = \min\{\|\xi\|_{L^2(\mu)} : \xi \in \partial E(\mu)\}$  to be the minimum norm of the elements in  $\partial E(\mu)$ . We have

**Lemma 3 (Craig (2014))** *Assume  $E$  is proper, coercive and lower semicontinuous (specify in Section B of the Supplementary Material (SM)). For an  $\mu_0$  and  $t \geq 0$ , as  $\frac{T}{h} \rightarrow \infty$ , the discrete gradient sequence  $\tilde{\mu}_{T/h} \triangleq J_h^{T/h} \mu_0$  converge uniformly in  $t$  to a compact subset of  $[0, +\infty)$ , and  $W_2^2(\tilde{\mu}_{T/h}, \mu_T) \leq \sqrt{3} |\partial E|(\mu) \sqrt{T} h$ .*

Lemma 3 suggests the discrete gradient flow can approximate the original WGF arbitrarily well if a small enough stepsize  $h$  is adopted. Consequently, one solves (16) through a sequence of optimization procedures to update the particles. We will derive a particle-approximation method for the  $W_2$  term in (15), which allows us to solve SG-MCMC efficiently. However, this technique is not applicable to SVGD, as we neither have an explicit form of the energy functional in (10) when adopting the  $W_2$  metric, nor have an explicit form for the metric  $W_{\mathcal{H}}$  in (11) when adopting the KL-divergence as the energy functional. Fortunately, this can be solved by the second approximation method called blob methods.

**Particle approximation by blob methods** The name of blob methods comes from the classical fluids literature, where instead of evolving the density in (7), one evolves all particles on a grid with time-spacing  $h$  (Carrillo et al., 2017). Specifically, note the function  $\mathbf{v}_t$  in (7) represents velocity of particles via transportation map  $\mathcal{T}$ , thus solving a WGF is equivalent to evolving the particles along their velocity in each iteration. Formally, one can prove

**Proposition 4 (Craig and Bertozzi (2016))** *Let  $\mu_0 \approx \frac{1}{M} \sum_{i=1}^M \delta(\boldsymbol{\theta}_0^{(i)})$ . Assume  $\mathbf{v}_t$  in (7) is well-defined and continuous w.r.t. each  $\boldsymbol{\theta}_t^{(i)}$  at time  $t$ . Then solving the PDE (7) reduces to solving a system of ordinary differential equations for the locations of the Dirac masses:*

$$d\boldsymbol{\theta}_t^{(i)} / dt = -\mathbf{v}_t(\boldsymbol{\theta}_t^{(i)}). \quad (17)$$

Proposition 4 suggests evolving each particle along the directions defined by  $\mathbf{v}_t$ , eliminating the requirement to know an explicit form of the energy functional. In the following, we apply the above particle-optimization techniques to derive algorithms for SVGD and SG-MCMC.

## 4.1 A particle-optimization algorithm for SVGD

As mentioned above, discrete-gradient-flow approximation does not apply to SVGD. We thus rely on the blob method. From Section 3.1,  $\mathbf{v}_t$  in SVGD is defined as  $\mathbf{v}_t(\boldsymbol{\theta}) = (\mathbf{W} * \mu_t)(\boldsymbol{\theta})$ . When  $\mu_t(\boldsymbol{\theta})$  is approximated by particles,  $\mathbf{v}_t(\boldsymbol{\theta}_t^{(i)})$  is simplified as:

$$\mathbf{v}_t(\boldsymbol{\theta}_t^{(i)}) = \frac{1}{M} \sum_{j=1}^M \mathbf{W}(\boldsymbol{\theta}_t^{(i)}, \boldsymbol{\theta}_t^{(j)}).$$

As a result, with the definition of  $\mathbf{W}$  in (9), updating  $\{\boldsymbol{\theta}_t^{(i)}\}$  by time discretizing (17) recovers the update equations for standard SVGD in (6).

## 4.2 Particle-optimization algorithms for SG-MCMC

Both the discrete-gradient-flow and the blob methods can be applied for SG-MCMC, which are detailed below.

**Particle optimization with discrete gradient flows**  
We first specify Lemma 3 in the case of SG-MCMC in Lemma 5, which is known as the Jordan-Kinderlehrer-Otto scheme (Jordan et al., 1998).

**Lemma 5 (Jordan et al. (1998))** *Assume that  $p(\boldsymbol{\theta}_t | \mathbf{X}) \leq C_1$  is infinitely differentiable, and  $\|\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta} | \mathbf{X})\| \leq C_2 (1 + C_1 - \log p(\boldsymbol{\theta} | \mathbf{X}))$  ( $\forall \boldsymbol{\theta}$ ) for some constants  $\{C_1, C_2\}$ . Let  $T = hK$  with  $K$  the number of iterations,  $\tilde{\mu}_0$  be an arbitrary distribution with same support as  $p(\boldsymbol{\theta} | \mathbf{X})$ , and  $\{\tilde{\mu}_k\}_{k=1}^K$  be the solution of the functional optimization problem:*

$$\tilde{\mu}_k = \arg \min_{\mu \in \mathcal{P}_s(\mathbb{R}^r)} KL(\mu || p) + \frac{1}{2h} W_2^2(\tilde{\mu}_{k-1}, \mu). \quad (18)$$

*Then  $\tilde{\mu}_K$  converges to  $\mu_T$  in the limit of  $h \rightarrow 0$ , i.e.,  $\lim_{h \rightarrow 0} \tilde{\mu}_K = \mu_T$ , where  $\mu_T$  is the solution of the FP equation (2) at time  $T$ .*

According to Lemma 5, it is apparent that SG-MCMC can be implemented by iteratively solving the optimization problem in (18). However, particle approximations for both terms in (18) are challenging. In the following, we develop efficient techniques to solve the problem.

First, rewrite the optimization problem in (18) as

$$\min_{\mu \in \mathcal{P}_s(\mathbb{R}^r)} \underbrace{-\mathbb{E}_{\mu}[\log p(\boldsymbol{\theta} | \mathbf{X})]}_{F_1} + \underbrace{\mathbb{E}_{\mu}[\log \mu] + \frac{1}{2h} W_2^2(\tilde{\mu}_{k-1}, \mu)}_{F_2}$$

We aim at deriving gradient formulas for both the  $F_1$  and  $F_2$  terms under a particle approximation in order to perform gradient descent for the particles. Let  $\mu \approx \frac{1}{M} \sum_{i=1}^M \delta(\boldsymbol{\theta}^{(i)})$ . The gradient of  $F_1$  is easily approximated as

$$\frac{\partial F_1}{\partial \boldsymbol{\theta}^{(i)}} \approx -\nabla_{\boldsymbol{\theta}^{(i)}} \log p(\boldsymbol{\theta}^{(i)} | \mathbf{X}). \quad (19)$$

To approximate the gradient for  $F_2$ , let  $p_{ij}$  denote the joint distribution of the particle-pair  $(\boldsymbol{\theta}^{(i)}, \boldsymbol{\theta}_{k-1}^{(j)})$ . Note  $\mathbb{E}_{\mu}[\log \mu]$  is minimized when the particles  $\{\boldsymbol{\theta}^{(i)}\}$  are uniformly distributed. In other words, the marginal distribution vector  $(\sum_j p_{ij})_i$  is a uniform distribution. Combining  $\mathbb{E}_{\mu}[\log \mu]$  with the definition of  $W_2$ , calculating  $F_2$  is equivalent to solving the following optimization problem:

$$P \triangleq \{p_{ij}\} = \arg \min_{p_{i,j}} \sum_{i,j} p_{ij} d_{ij} \quad (20)$$

$$s.t. \quad \sum_j p_{ij} = \frac{1}{M}, \quad \sum_i p_{ij} = \frac{1}{M},$$

where  $d_{ij} \triangleq \|\boldsymbol{\theta}^{(i)} - \boldsymbol{\theta}_{k-1}^{(j)}\|^2$ . We can further enforce the joint distribution  $\{p_{ij}\}$  to have maximum entropy by introducing a regularization term  $\mathbb{E}_{p_{ij}}[\log p_{ij}]$ , which is stronger than the regularizer enforced for the marginal distribution above. After introducing Lagrangian multipliers  $\{\alpha_i, \beta_j\}$  to deal with the constraints in (20), we arrive at the dual problem:

$$\begin{aligned} \max \mathcal{L}^D(\{p_{ij}\}, \{\alpha_i\}, \{\beta_j\}) &= \lambda \sum_{i,j} p_{ij} \log p_{ij} + p_{ij} d_{ij} \\ &+ \sum_i \alpha_i \left( \sum_j p_{ij} - \frac{1}{M} \right) + \sum_j \beta_j \left( \sum_i p_{ij} - \frac{1}{M} \right), \end{aligned}$$

where  $\lambda$  is the weight for the regularizer. The optimal  $p_{ij}$ 's can be obtained by applying KKT conditions to set the derivative w.r.t.  $p_{ij}$  to be zero, ending up with the following form:

$$p_{ij}^* = u_i e^{-d_{ij}/\lambda} v_j,$$

where  $u_i \triangleq e^{-\frac{1}{2} - \frac{\alpha_i}{\lambda}}$ ,  $v_j \triangleq e^{-\frac{1}{2} - \frac{\beta_j}{\lambda}}$ . As a result, the particle gradients on  $F_2$  can be approximated as

$$\begin{aligned} \frac{\partial F_2}{\partial \boldsymbol{\theta}^{(i)}} &\approx - \frac{\sum_j u_i v_j d_{ij} e^{-d_{ij}/\lambda}}{\partial \boldsymbol{\theta}^{(i)}} \\ &= \sum_j 2u_i v_j \left( \frac{d_{ij}}{\lambda} - 1 \right) e^{-d_{ij}/\lambda} (\boldsymbol{\theta}^{(i)} - \boldsymbol{\theta}_{k-1}^{(j)}). \end{aligned} \quad (21)$$

Theoretically, we need to adaptively update  $\{u_i, v_j\}$  as well to ensure the constraints in (20). In practice, however, we use a fixed scaling factor  $\gamma$  to approximate  $u_i v_j$  for the sake of simplicity.

Particle gradients are obtained by combining (19) and (21), which are then used to update the particles  $\{\boldsymbol{\theta}^{(i)}\}$  by standard gradient descent. Intuitively, (19) encourages particles move to local modes while (21) regularizes particle interactions. Different from SVGD, our scheme imposes both attractive and repulsive forces for the particles. Specifically, by inspecting (21), we can conclude that: *i)* When  $\boldsymbol{\theta}^{(i)}$  is far from a previous particle  $\boldsymbol{\theta}_k^{(j)}$ , i.e.,  $\frac{d_{ij}}{\lambda} > 1$ ,  $\boldsymbol{\theta}^{(i)}$  is pulled close to  $\{\boldsymbol{\theta}_k^{(j)}\}$  with force proportional to  $(\frac{d_{ij}}{\lambda} - 1)e^{-d_{ij}/\lambda}$ ; *ii)* when  $\boldsymbol{\theta}^{(i)}$  is close enough to a previous particle  $\boldsymbol{\theta}_k^{(j)}$ , i.e.,  $\frac{d_{ij}}{\lambda} < 1$ ,  $\boldsymbol{\theta}^{(i)}$  is pushed away, preventing it from collapsing to  $\boldsymbol{\theta}_k^{(j)}$ .

**Particle optimization with blob methods** The idea of blob methods can also be applied to particle approximation for SG-MCMC, which require the velocity vector field  $\mathbf{v}_t$ . According to (13), this is calculated as:  $\mathbf{v}_t(\boldsymbol{\theta}) = -\nabla_{\boldsymbol{\theta}} \frac{\delta(E_1+E_2)}{\delta\mu} = -\nabla_{\boldsymbol{\theta}} U - \nabla_{\boldsymbol{\theta}} \mu / \mu$ . Unfortunately, direct application of particle approximation is infeasible because the term  $\nabla_{\boldsymbol{\theta}} \mu$  is undefined with discrete  $\mu$ . To tackle this problem, we adopt the idea in Carrillo et al. (2017) to approximate the energy functional  $E_2$  in (13) as:  $E_2 \approx \int \mu(\boldsymbol{\theta}) \log(\mu * K)(\boldsymbol{\theta}) d\boldsymbol{\theta}$ , where  $K(\cdot, \cdot)$  is another kernel function to smooth out  $\mu$ . Consequently, based on Carrillo et al. (2017), the velocity  $\mathbf{v}_t$  can be calculated as (details in Section C of the SM):

$$\mathbf{v}_t(\boldsymbol{\theta}) = -\nabla_{\boldsymbol{\theta}} U - \sum_{j=1}^n \nabla_{\boldsymbol{\theta}_t^{(j)}} K(\boldsymbol{\theta}, \boldsymbol{\theta}_t^{(j)}) / \sum_k K(\boldsymbol{\theta}_t^{(j)}, \boldsymbol{\theta}_t^{(k)}) - \sum_{j=1}^n \nabla_{\boldsymbol{\theta}_t^{(j)}} K(\boldsymbol{\theta}, \boldsymbol{\theta}_t^{(j)}) / \sum_{k=1}^n K(\boldsymbol{\theta}, \boldsymbol{\theta}_t^{(k)}) \quad (22)$$

Given  $\mathbf{v}_t$ , particle updates can be obtained by solving (17) numerically as in SVGD. By inspecting the formula of  $\mathbf{v}_t$  in (22), the last two terms both act as repulsive forces. Interestingly, the mechanism is similar to SVGD, but with adaptive force between different particle pairs.

## 5 THE GENERAL RECIPE

Based on the above development, a more general particle-optimization framework is proposed by combining the PDEs of both SG-MCMC and SVGD. As a result, we propose the following PDE to drive evolution of densities

$$\frac{\partial \mu_t}{\partial t} = -\nabla_{\boldsymbol{\theta}} \cdot (\mu_t F(\boldsymbol{\theta}_t)) + \lambda_1 \nabla_{\boldsymbol{\theta}} \cdot ((\mathbf{W} * \mu_t) \mu_t) + \lambda_2 \nabla_{\boldsymbol{\theta}} \nabla_{\boldsymbol{\theta}} : (\mu_t g(\boldsymbol{\theta}_t) g^{\top}(\boldsymbol{\theta}_t)) , \quad (23)$$

where  $\lambda_1$  and  $\lambda_2$  are two constants. It is easily seen that to ensure the stationary distribution of (23) to be equal to  $p(\boldsymbol{\theta} | \mathbf{X})$ , the following condition must be satisfied:

$$\nabla_{\boldsymbol{\theta}} \cdot (p(\boldsymbol{\theta} | \mathbf{X}) F(\boldsymbol{\theta})) = \lambda_1 \nabla_{\boldsymbol{\theta}} \cdot ((\mathbf{W} * p(\boldsymbol{\theta} | \mathbf{X})) p(\boldsymbol{\theta} | \mathbf{X})) + \lambda_2 \nabla_{\boldsymbol{\theta}} \nabla_{\boldsymbol{\theta}} : (p(\boldsymbol{\theta} | \mathbf{X}) g(\boldsymbol{\theta}) g^{\top}(\boldsymbol{\theta})) \quad (24)$$

There are many feasible choices for the functions and parameters  $\{F(\boldsymbol{\theta}), \mathbf{W}, g(\boldsymbol{\theta}), \lambda_1, \lambda_2\}$  to satisfy (24). However, the verification procedure might be complicated given the present of a convolutional term in (24). We recommend the following choices for simplicity:

- $F(\boldsymbol{\theta}) = \frac{1}{2}U(\boldsymbol{\theta})$ ,  $\mathbf{W} = 0$ ,  $g(\boldsymbol{\theta}) = \mathbf{I}$  and  $\lambda_2 = 1$ : this reduces to the Wasserstein-based SGLD with particle optimization. Specifically, when the discrete-gradient-flow approximation is adopted, the algorithm is denoted as  $w$ -SGLD; whereas when the blob method is adopted, it is denoted as  $w$ -SGLD-B.

- $F(\boldsymbol{\theta}) = 0$ ,  $g(\boldsymbol{\theta}) = 0$ ,  $\mathbf{W}$  is defined as (9): this reduces to standard SVGD.

- $F(\boldsymbol{\theta}) = \frac{1}{2}U(\boldsymbol{\theta})$ ,  $g(\boldsymbol{\theta}) = \mathbf{I}$ ,  $\mathbf{W}$  is defined as (9), and  $\lambda_2 = 1$ : this is the combination of SGLD and SVGD, and is called particle interactive SGLD, denoted as PI-SGLD or  $\pi$ -SGLD.

It is easy to verify that condition (24) is satisfied for all the above three particle-optimization algorithms. Furthermore, particle updates are readily developed by applying either the discrete-gradient-flow or blob-based methods.

## 6 RELATED PARTICLE-BASED MCMC METHODS

There have been related particle-based MCMC algorithms. Representative methods are sequential Monte Carlo (SMC) (Moral et al., 2006), particle MCMC (PMCMC) (Andrieu et al., 2010) and many variants. In SMC, particles are sample from a proposal distribution, and the corresponding weights are updated by a resampling step. PMCMC extends SMC by sampling from an extended distribution interacted with a MH-rejection step. Compared to our framework, their proposal distributions are typically hard to choose; furthermore, optimality of the particles from both methods can not be guaranteed. Furthermore, the methods are typically much more computationally expensive. Recently, Dai et al. (2016) proposed a particle-based MCMC algorithm by approximating a target distribution with weighted kernel density estimator, which updates particle weights based on likelihoods of the corresponding particles. This approach is theoretically sound but lacks an underlying geometry interpretation. Finally, we note that  $w$ -SGLD has been successfully applied to reinforcement learning recently for improved policy optimization (Zhang et al., 2018a).

## 7 EXPERIMENTS

We verify our framework on a set of experiments, including a number of toy experiments and applications to Bayesian sampling of deep neural networks (DNNs).

### 7.1 Demonstrations

**Toy Distributions** We compare various sampling methods on multi-mode toy examples, *i.e.*, SGLD, SVGD,  $w$ -SGLD,  $w$ -SGLD-B and  $\pi$ -SGLD. We aim to sample from four unnormalized 2D densities  $p(z) \propto \exp\{U(z)\}$ , with detailed functional form provided in the SM. We optimize/sample 2000 particles to approximate target distributions. The results are shown in Figure 1. It can be seen from Figure 1 that though SGLD maintains good asymptotic properties, it is inaccurate to approximate distributions with only a few samples; in some case, the

samples cannot even cover all the modes. Interestingly, all other particle-optimization-based algorithms successfully find all the modes and fit the distributions well.  $w$ -SGLD is good at finding modes, but worse at modeling the correct variance due to difficulty of controlling the balance between attractive and repulsive forces between particles.  $w$ -SGLD-B is better than  $w$ -SGLD at modeling the distribution variance, performing similarly to SVGD and  $\pi$ -SGLD. Even though, we note that  $w$ -SGLD is very useful when the number of particles is small, which fits a distribution better, as shown in Section E of the SM.

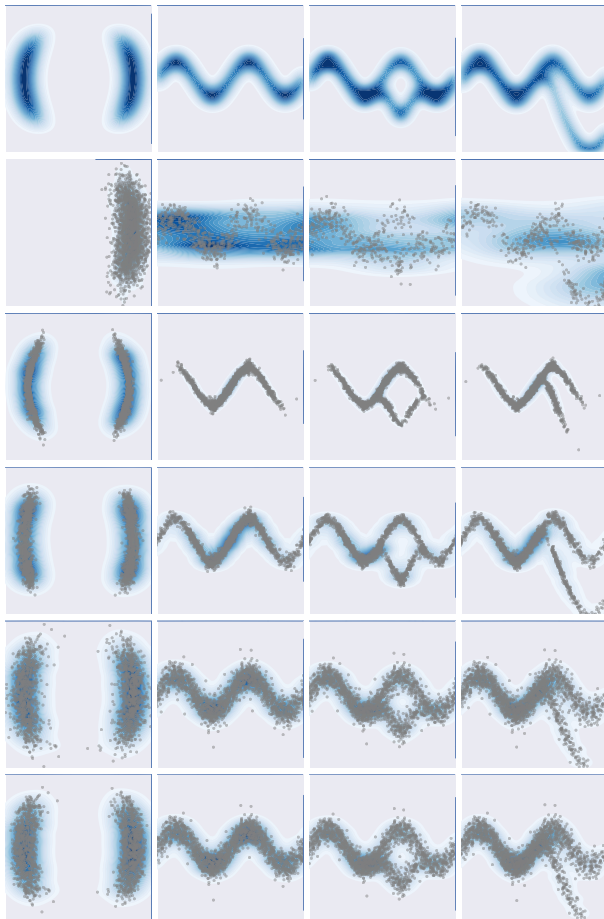


Figure 1: Illustration of different algorithms on toy distributions. Each column is a distribution case. 1st row: Ground truth; 2nd row: standard SGLD; 3rd row:  $w$ -SGLD; 4th row:  $w$ -SGLD-B; 5th row: SVGD; 6th row:  $\pi$ -SGLD.

**Bayesian Logistic regression** We next compare the three variants of our framework (*i.e.* SVGD,  $w$ -SGLD and  $w$ -SGLD-B) on a simple logistic-regression task with quantitative evaluations. We use the same model, data and experimental settings as Liu and Wang (2016a). The Covertypes dataset contains 581,012 data points and 54 features. We perform 5 runs for each setting and report the mean of testing accuracies/log-likelihoods. Figure 2 plots both test accuracies and test log-likelihoods w.r.t. the

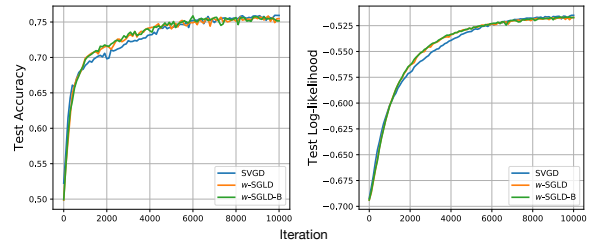


Figure 2: Test accuracies (left) and log-likelihoods (right) v.s. iterations for SVGD,  $w$ -SGLD and  $w$ -SGLD-B.

number of training iterations. It is clearly that while all methods converge to the same accuracy/likelihood level, both  $w$ -SGLD and  $w$ -SGLD-B converge slightly faster than SVGD. In addition,  $w$ -SGLD and  $w$ -SGLD-B have similar convergence behaviors, thus we only use  $w$ -SGLD in the DNN experiments below.

### Parameter Sensitivity

Now we study the role of hyper-parameters in  $\pi$ -SGLD: the number of particles  $M$  and the scaling factor  $\gamma$  to

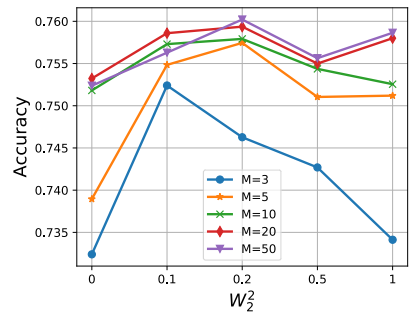


Figure 3: Impact of  $W_2^2$  factor  $\gamma$  and particle number  $M$ .

replace the  $u_i v_j$ -term in (21). We use the same dataset and model as the above experiment. Figure 3 plots test accuracies along with different parameter settings. As expected, the best performance is achieved with appropriate scale of  $W_2^2$ . The performance keep improving with increasing particles. Interestingly, the Wasserstein regularization is more important when the number of particles is small, demonstrating the superiority when approximate distributions with very few particles.

## 7.2 Applications on deep neural networks

We conduct experiments for Bayesian learning of DNNs. Different from traditional optimization for DNNs, we are interested in modeling weight uncertainty of neural networks, an important topic that has been well explored (Hernández-Lobato and Adams, 2015; Blundell et al., 2015a; Li et al., 2016; Louizos and Welling, 2016). We assign priors to the weights, which are simple isotropic Gaussian priors in our case, and perform posterior sampling with the proposed particle-optimization-based algorithms, as well as other standard algorithms such as SGLD and SGD. We use the RMSprop optimizer for feed-forward networks (FNN), and Adam for for convolutional neural networks (CNNs) and recurrent neural

networks (RNNs). For all methods, we use a RBF kernel  $K(\theta, \theta') = \exp(-\|\theta - \theta'\|_2^2/h)$ , with the bandwidth set to  $h = \text{med}^2/\log M$ . Here med is the median of the pairwise distance between particles. All experiments are conducted on a single TITAN X GPU.

**Feed-forward Neural Networks** We perform the classification tasks on the standard MNIST dataset. A two-layer model 784-X-X-10 with ReLU activation function is used, with X being the number of hidden units for each layer. The training epoch is set to 100. The test errors are reported in Table 1. Not surprisingly, Bayesian methods generally perform better than their optimization counterparts. The new  $\pi$ -SGLD which combines  $w$ -SGLD and SVGD improves both methods with little computational overhead. In addition,  $w$ -SGLD seems to perform better than SVGD in this case, partially due to a better asymptotic property mentioned in (Liu, 2017). Furthermore, standard SGLD which is based on MCMC obtains higher test errors compared to particle-optimization-based algorithms, partially due to the correlated-sample issue discussed in the introduction. See (Blundell et al., 2015b) for details on the other methods in Table 1.

Table 1: Classification error of FNN on MNIST.

Method	Test Error	
	400-400	800-800
$\pi$ -SGLD	<b>1.36%</b>	<b>1.30%</b>
$w$ -SGLD	1.44%	1.37%
SVGD	1.53%	1.40%
SGLD	1.64%	1.41%
RMSprop	1.59%	1.43%
RMSspectral	1.65%	1.56%
SGD	1.72%	1.47%
BPB, Gaussian	1.82%	1.99%
SGD, dropout	1.51%	1.33%

**Convolution Neural Networks** We use the CIFAR-10 dataset to test our framework on CNNs. We adopt a CNN of three convolution layers, using  $3 \times 3$  filter size with C64-C128-C256 channels, and  $2 \times 2$  max-pooling after each convolution layer. Our implementation adopts batch normalization, drop out and data augmentation to improve the performance. Training losses and test accuracies are presented in Table 2. Consistently,  $\pi$ -SGLD outperforms all other algorithms in terms of test accuracy. ADAM obtains a better training loss but worse test accuracy, indicating worse generalization ability of the optimization-based methods compared to Bayesian methods.

**Recurrent Neural Networks** For RNNs, we run standard language models. Experiments are presented on three publicly available corpora: APNEWS, IMDB and BNC. APNEWS is a collection of Associated Press news articles from 2009 to 2016. IMDB is a set of movie re-

Table 2: Classification error of CNN on CIFAR-10.

Method	Training Loss	Test Accuracy
ADAM	23.80	86.76%
SVGD	30.57	88.72%
SGLD	28.52	88.64%
$w$ -SGLD	31.26	88.80%
$\pi$ -SGLD	25.06	<b>89.52%</b>

views collected by Maas et al. (2011), and BNC BNC Consortium (2007) is the written portion of the British National Corpus, which contains excerpts from journals, books, letters, essays, memoranda, news and other types of text. These datasets can be downloaded from Github\*.

Table 3: Perplexity of language model on three corpora.

Method	APNEWS	IMDB	BNC
SGD	64.13	72.14	102.89
SGLD	63.01	68.12	95.13
SVGD	61.64	69.25	94.99
$w$ -SGLD	61.22	67.41	93.68
$\pi$ -SGLD	<b>59.83</b>	<b>67.04</b>	<b>92.33</b>

We follow the standard set up as Wang et al. (2017). Specifically, we lower case all the word tokens and filter out word tokens that occur less than 10 times. All the datasets are divided into training, development and testing sets. For the language model set up, we consider a 1-layer LSTM model with 600 hidden units. The sequence length is fixed to be 30. In order to alleviate overfitting, dropout with a rate of 0.4 is used in each LSTM layer. Results in terms of test perplexities are presented in Table 3. Again, we see that  $\pi$ -SGLD performs best among all algorithms, and  $w$ -SGLD is slightly better than SVGD, both of which are better than other algorithms.

## 8 CONCLUSION

We propose a unified particle-optimization framework for large-scale Bayesian sampling. Our framework defines gradient flows on the space of probability measures, and uses particles to approximate the corresponding densities. Consequently, solving gradient flows reduces to optimizing particles on the parameter space. Our framework includes the standard SVGD as a special case, and also allows us to develop efficient particle-optimization algorithms for SG-MCMC, which is highly related to SVGD. Extensive experiments are conducted, demonstrating the effectiveness and efficiency of our proposed framework. Interesting future work includes designing more practically efficient variants of the proposed particle-optimization framework, and developing theory to study general convergence behaviors of the algorithms, in addition to the asymptotic results presented in (Liu, 2017).

\*<https://github.com/jhlau/topically-driven-language-model>

## REFERENCES

- Ambrosio, L., Gigli, N., and Savaré, G. (2005). *Gradient Flows in Metric Spaces and in the Space of Probability Measures*. Lectures in Mathematics ETH Zürich.
- Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B*, 72(3):269–342.
- Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015a). Weight uncertainty in neural networks. In *ICML*.
- Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015b). Weight uncertainty in neural networks. In *ICML*.
- BNC Consortium, B. (2007). The british national corpus, version 3 (bnc xml edition). *Distributed by Bodleian Libraries, University of Oxford, on behalf of the BNC Consortium*. URL:<http://www.natcorp.ox.ac.uk/>.
- Carrillo, J. A., Craig, K., and Patacchini, F. S. (2017). A blob method for diffusion. (arXiv:1709.09195).
- Chen, C., Ding, N., and Carin, L. (2015). On the convergence of stochastic gradient MCMC algorithms with high-order integrators. In *NIPS*.
- Chen, T., Fox, E. B., and Guestrin, C. (2014). Stochastic gradient Hamiltonian Monte Carlo. In *ICML*.
- Craig, K., editor (2014). *The Exponential Formula for the Wasserstein Metric*. PhD thesis, The State University of New Jersey.
- Craig, K. and Bertozzi, A. L. (2016). A blob method for the aggregation equation. *Mathematics of Computation*, 85(300):1681–1717.
- Dai, B., He, N., Dai, H., and Song, L. (2016). Provable Bayesian inference via particle mirror descent. In *AISTATS*.
- Ding, N., Fang, Y., Babbush, R., Chen, C., Skeel, R. D., and Neven, H. (2014). Bayesian sampling using stochastic gradient thermostats. In *NIPS*.
- Feng, Y., Wang, D., and Liu, Q. (2017). Learning to draw samples with amortized stein variational gradient descent. In *UAI*.
- Ghosh, A. P. (2011). *Backward and Forward Equations for Diffusion Processes*. Wiley Encyclopedia of Operations Research and Management Science.
- Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. (2017). Reinforcement learning with deep energy-based policies. In *ICML*.
- Hernández-Lobato, J. M. and Adams, R. P. (2015). Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *ICML*.
- Houthoofd, R., Chen, X., Duan, Y., Schulman, J., De Turck, F., and Abbeel, P. (2016). Vime: Variational information maximizing exploration. In *NIPS*.
- Jordan, R., Kinderlehrer, D., and Otto, F. (1998). The variational formulation of the Fokker-Planck equation. *SIAM Journal on Mathematical Analysis*, 29(1):1–17.
- Kingma, D. P. and Welling, M. (2014). Auto-encoding variational Bayes. In *ICLR*.
- Kolmogoroff, A. (1931). Some studies in machine learning using the game of checkers. *Mathematische Annalen*, 104(1):415–458.
- Li, C., Chen, C., Carlson, D., and Carin, L. (2016). Preconditioned stochastic gradient Langevin dynamics for deep neural networks. In *AAAI*.
- Li, C., Liu, H., Chen, C., Pu, Y., Chen, L., Henao, R., and Carin, L. (2017). ALICE: Towards understanding adversarial learning for joint distribution matching. In *NIPS*.
- Liu, Q. (2017). Stein variational gradient descent as gradient flow. In *NIPS*.
- Liu, Q. and Wang, D. (2016a). Stein variational gradient descent: A general purpose Bayesian inference algorithm. In *NIPS*.
- Liu, Q. and Wang, D. (2016b). Stein variational gradient descent: A general purpose bayesian inference algorithm. In *NIPS*.
- Liu, Y., Ramachandran, P., Liu, Q., and Peng, J. (2017). Stein variational policy gradient. In *UAI*.
- Louizos, C. and Welling, M. (2016). Structured and efficient variational deep learning with matrix Gaussian posteriors. In *ICML*.
- Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. (2011). Learning word vectors for sentiment analysis. In *ACL*.
- Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *Journal of Chemical Physics*, 21:1087–1092.
- Moral, P. D., Doucet, A., and Jasra, A. (2006). Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B*, 68(3):411–436.
- Øksendal, B., editor (1985). *Stochastic Differential Equations*. Springer-Verlag, Berlin.
- Risken, H. (1989). *The Fokker-Planck equation*. Springer-Verlag, New York.
- Rulla, J. (1996). Error analysis for implicit approximations to solutions to Cauchy problems. *SIAM Journal on Numerical Analysis*, 33(1):68–87.
- Teh, Y. W., Thiery, A. H., and Vollmer, S. J. (2016). Consistency and fluctuations for stochastic gradient Langevin dynamics. *JMLR*, 17(1):193–225.
- Villani, C. (2008). *Optimal transport: old and new*. Springer Science & Business Media.
- Vollmer, S. J., Zygalakis, K. C., and Teh, Y. W. (2016). (exploration of the (Non-)asymptotic bias and variance of stochastic gradient Langevin dynamics. *JMLR*, 1:1–48.
- Wang, W., Gan, Z., Wang, W., Shen, D., Huang, J., Ping, W., Satheesh, S., and Carin, L. (2017). Topic compositional neural language model. *arXiv preprint arXiv:1712.09783*.
- Welling, M. and Teh, Y. W. (2011). Bayesian learning via stochastic gradient Langevin dynamics. In *ICML*.
- Xifara, T., Sherlock, C., Livingstone, S., Byrne, S., and Girolami, M. (2014). Langevin diffusions and the Metropolis-adjusted Langevin algorithm. *Statistics & Probability Letters*, 91:14–19.
- Zhang, R., Chen, C., Li, C., and Carin, L. (2018a). Policy optimization as wasserstein gradient flows. In *ICML*.
- Zhang, R., Li, C., Chen, C., and Carin, L. (2018b). Learning structural weight uncertainty for sequential decision-making. In *AISTATS*.

---

# An Efficient Quantile Spatial Scan Statistic for Finding Unusual Regions in Continuous Spatial Data with Covariates

---

**Travis Moore**  
School of EECS  
Oregon State University  
Corvallis, OR 97331  
moortrav@eecs.oregonstate.edu

**Weng-Keen Wong**  
School of EECS  
Oregon State University  
Corvallis, OR 97331  
wongwe@eecs.oregonstate.edu

## Abstract

Domains such as citizen science biodiversity monitoring and real estate sales are producing spatial data with a continuous response and a vector of covariates associated with each spatial data point. A common data analysis task involves finding unusual regions that differ from the surrounding area. Existing techniques compare regions according to the means of their distributions to measure unusualness. Comparing means is not only vulnerable to outliers, but it is also restrictive as an analyst may want to compare other parts of the probability distributions. For instance, an analyst interested in unusual areas for high-end homes would be more interested in the 90th percentile of home sale prices than in the mean. We introduce the Quantile Spatial Scan Statistic (QSSS), which finds unusual regions in spatial data by comparing quantiles of data distributions while accounting for covariates at each data point. We also develop an exact incremental update of the hypothesis test used by the QSSS, which results in a massive speedup over a naive implementation.

## 1 INTRODUCTION

Analysis of spatial data often involves finding spatial regions that are different from the surrounding area. For example, epidemiologists are interested in finding regions with an unusually high incidence of disease while criminologists are interested in identifying crime hotspots. The spatial scan statistic (SSS) (Kulldorff, 1997) is a widely used technique to discover unusual regions from a Bernoulli or Poisson point process. The SSS searches over a given set of regions, scoring each

region according to how a quantity of interest (eg. the disease rate) inside the region differs from outside the region. Finally, the SSS computes the p-value of the highest scoring region using a randomization test.

Many spatial data sets, however, are more complex than point processes, which focus on the spatial locations of the data. Real-world spatial data sets from domains such as citizen science biodiversity monitoring and real estate associate a response value with each point as well as a set of covariates (i.e. features). For example, in a real estate data set, each data point has a location, a sale price, and associated features such as square footage, number of bedrooms, age, etc. Formally, we represent the  $i$ th data point of dataset  $D$  as a tuple  $(Y_i, X_{i,1}, \dots, X_{i,p}, L_{i,1}, \dots, L_{i,d})$ , where  $Y_i$  is a continuous response,  $(X_{i,1}, \dots, X_{i,p})$  are the  $p$  covariates and  $(L_{i,1}, \dots, L_{i,d})$  are coordinates in  $d$ -dimensions; for simplicity, we assume  $d = 2$ . In later sections, we will refer to the data as  $D = \{Y, X, L\}$  to represent the distinct aspects of response, covariates and locations.

We can follow the SSS framework to find unusual regions in this more complex setting. For each region, we fit a model that captures the relationship between the features and the response variable. Then, we use a scoring function to compare the models from the “inside” versus the “outside” regions, by using a hypothesis test that compares the means of the models. While such an approach seems reasonable, there are two shortcomings. First, the approach is not robust as the mean is well known to be vulnerable to outliers and the models for each region can be badly skewed by outliers with extreme values for the response variable (Rousseeuw and Leroy, 1987). Second, many real-world tasks involve comparing spatial regions using other parts of their distributions besides the mean. For instance, a real-estate agent interested in high-end homes may want to compare regions based on the 90th percentile of the sale price distribution. To overcome both of these problems, we develop a novel method for comparing quantiles of spatial regions.



To accomplish our goal of comparing quantiles of spatial regions, we modify the proposed SSS variant by fitting quantile regression models to the “inside” and “outside” regions. Unfortunately, this naive approach is computationally expensive; fitting a quantile regression requires a linear program and this step would be required in the inner loop of the algorithm. To make the algorithm efficient, we replace the likelihood ratio test with the rank test, which is a non-parametric hypothesis test that avoids the need to fit quantile regressions to the “inside” regions. However, performing a rank test from scratch every time we score a new region is also computationally expensive. Instead, we develop an incremental version of the rank test that allows the rank test from a smaller region to be updated when the region is grown to include more spatial data points.

The contributions of our work are as follows. First, we introduce the Quantile Spatial Scan Statistic (QSSS), which discovers unusual regions for continuous spatial data with covariates. The comparison between regions to determine unusualness is based on a comparison of the  $\tau$ -th quantile of the response variable distributions. To our knowledge, no such version of the SSS currently exists in the literature. This algorithm is also robust to outliers, unlike an analogous algorithm that makes comparisons based on the mean of a region. Second, we show how to make the QSSS over an order of magnitude faster than a naive implementation by introducing an incremental update to the rank test. This update is exact and not an approximation. Finally, we evaluate the QSSS on simulated data and also show interesting results from case studies on three real-world datasets.

## 2 RELATED WORK AND BACKGROUND

We first discuss related work and then provide some background needed to understand our approach. A large body of work that is seemingly related to our task has focused on producing disease maps that illustrate how disease cases vary across space (eg. (Best et al., 2005)). Researchers have also investigated spatial quantile regression (eg. (Reich et al., 2011; Macmillan, 2013)). These modeling approaches generally produce a probabilistic surface, which results in a useful visualization but does not directly solve our goal of identifying specific unusual regions. Achieving this goal requires a human to inspect the probabilistic surface, manually segment it into unusual regions and rank these regions according to some unusualness criterion. This human intervention is not desirable when the spatial region is large and also if the goal is to create an automated monitoring system. Our QSSS algorithm essentially automates these steps in a compu-

tationally efficient manner.

### 2.1 THE SPATIAL SCAN STATISTIC

The Spatial Scan Statistic, introduced by Kulldorff (1997) is a widely used approach for finding anomalous regions. For the SSS, each spatial data point is represented by a tuple  $(c_i, b_i)$  along with its location. The value  $c_i$  corresponds to a count at location  $i$  (e.g. the number of disease cases) and  $b_i$  is the baseline value (e.g. the population) at location  $i$ . The value  $c_i$  is Poisson distributed with mean  $qb_i$ , where  $q$  is the probability of an event of interest occurring.

The original SSS used a scanning window in the shape of a circle to discover unusual regions. While in theory the search should be over all circular regions, the search is, in practice, often limited to circles with centers determined by a fixed grid superimposed on the spatial area. Let  $\mathcal{C}$  be the set of all circular regions searched by the SSS and let  $C \in \mathcal{C}$  be the region under consideration. For a region  $C$  under consideration, let  $c_{in} = \sum_{i \in C} c_i$ ,  $c_{out} = \sum_{i \notin C} c_i$ ,  $b_{in} = \sum_{i \in C} b_i$ ,  $b_{out} = \sum_{i \notin C} b_i$ . Let  $q_{in}$  be the event probability inside the region  $C$  and let  $q_{out}$  be the probability outside the region  $C$ .

Under the null hypothesis  $H_0$ , the event probability is uniform across the entire area i.e.  $q_{in} = q_{out}$ . Under the alternate hypothesis  $H_1(C)$ ,  $q_{in} > q_{out}$ . We estimate  $q_{in}$  and  $q_{out}$  using maximum likelihood estimation. The SSS uses the likelihood ratio test to score a region  $C$ :

$$\begin{aligned} \text{Score}(C) &= \frac{P(\mathbf{D}|H_1(C))}{P(\mathbf{D}|H_0)} \\ &= \left(\frac{c_{in}}{b_{in}}\right)^{c_{in}} \left(\frac{c_{out}}{b_{out}}\right)^{c_{out}} \left(\frac{c_{in} + c_{out}}{b_{in} + b_{out}}\right)^{-(c_{in} + c_{out})} \end{aligned}$$

if  $\left(\frac{c_{in}}{b_{in}}\right) > \left(\frac{c_{out}}{b_{out}}\right)$  and 1 otherwise.

The SSS then selects the region with the highest score i.e.  $C^* = \underset{C \in \mathcal{C}}{\operatorname{argmax}} \text{Score}(C)$ . Due to the multiple hypothesis testing problem, we cannot interpret the score from the likelihood ratio test as a true p-value. Instead, we estimate the p-value through a randomization test. In each replication of the randomization test, we maintain the same underlying population as the original problem, but generate events assuming a uniform probability. Then, the search for the best scoring region is performed. The process is repeated for  $R$  replications to produce an empirical distribution which determines how likely it is to obtain a best score of  $C^*$ .

Many researchers have extended the original SSS approach, including using scanning windows that are arbitrarily shaped (Duczmal and Assuncao, 2004) and incorporating mobility patterns (Lan et al., 2014). One variant

goes beyond shifts in means by discovering which sub-population is most affected by a treatment (McFowland et al., 2018). We point out that performing a quantile-based comparison results in a fundamentally different type of optimization problem and past work on speeding up the SSS (eg. (Neill and Moore, 2004; Neill, 2012)) is not readily applicable. Finally, Moore and Wong (2015) use the SSS to find species rich hotspots but they do not compare quantiles of distributions.

## 2.2 QUANTILE REGRESSION

Suppose we have a continuous random variable  $Y$  with distribution function  $F(Y) = P(Y \leq y)$ . The  $\tau$ -th quantile  $q(\tau)$ , with  $0 < \tau < 1$ , is defined as  $q(\tau) = F^{-1}(\tau) = \inf_y \{F(y) \geq \tau\}$ . For example, when  $\tau = 0.5$ , we get the median. Given a dataset  $Y_1, \dots, Y_n$ , the  $\tau$ -th sample quantile  $\hat{q}(\tau)$ , can be computed by solving the optimization problem:

$$\hat{q}(\tau) = \underset{q}{\operatorname{argmin}} \sum_{i=1}^n \rho_\tau(Y_i - q)$$

where  $\rho_\tau(r) = r(\tau - I(r < 0))$ .

Quantile regression, introduced by Koenker and Bassett (1978), fits a regression to the conditional  $\tau$ -th quantile of the response variable. Given a dataset  $\mathbf{D} = \{(Y_1, \mathbf{X}_1), \dots, (Y_n, \mathbf{X}_n)\}$  where  $Y_i$  is the response variable and  $\mathbf{X}_i$  are the covariates, fitting a quantile regression involves solving:

$$\hat{\beta}(\tau) = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^n \rho_\tau(Y_i - \mathbf{X}_i \beta)$$

The solution  $\hat{\beta}(\tau)$  produces a conditional quantile function  $Q_Y(\tau | \mathbf{X} = \mathbf{x}) = \mathbf{x}' \hat{\beta}(\tau)$ , similar to how a standard regression produces the conditional mean when the coefficients are multiplied with the covariate values.

Quantile regression is a useful tool for analyzing specific parts of a distribution. It can model the data extremes by setting  $\tau$  close to either 1 or 0, or it can reduce the influence of these points by modeling  $\tau$  close to 0.5.

There are several methods for comparing two quantile regression models. These test include applications of Wald's test, the Likelihood Ratio test, and Rank test (Koenker and Machado, 1999). Mood's median test (Mood, 1950) can also be adapted to perform a comparison at a given quantile. While fast to compute, this version of Mood's test lacks the power of the Wald, Likelihood Ratio, and Rank alternatives. Any of these methods are still usable when the covariate set  $\mathbf{X}$  is empty by using the quantiles of  $\mathbf{Y}$ . We use the Rank test, as

it can be implemented without repeatedly re-estimating the quantile regression coefficients for each data subset, thereby reducing its computation time without sacrificing power. In the following section we explain the Rank test for quantile regression.

## 2.3 RANK TEST FOR QUANTILE REGRESSION

Let the regression model for the  $\tau$ th quantile have the form  $Y = \mathbf{X} \beta_1 + \tilde{\mathbf{X}} \beta_2$  where each row  $\mathbf{X}_i$  corresponds to a data point. For a given data subset  $\mathbf{C} \subseteq \mathbf{D}$ ,  $\tilde{\mathbf{X}}_i = \mathbf{X}_i$  if  $\mathbf{X}_i \in \mathbf{C}$  and  $\tilde{\mathbf{X}}_i = \mathbf{0}$  if  $\mathbf{X}_i \notin \mathbf{C}$ . This model will simultaneously fit a regression to  $\mathbf{C}$  and  $\mathbf{D} \setminus \mathbf{C}$ . In the spatial scan context  $\mathbf{C}$  is the region inside our circle and  $\mathbf{D} \setminus \mathbf{C}$  is the region outside. The goal is then to test the null hypothesis  $H_0 : \beta_2 = \mathbf{0}$  against the alternative  $H_1 : \beta_2 \neq \mathbf{0}$  to see if the subset  $\mathbf{C}$  is sufficiently different from the full distribution of  $\mathbf{D}$ .

The Rank test is an application of the score test, using a score function and ranking process to estimate the data distribution when the true likelihood is unknown. In general terms, the score test statistic is composed of the product of the square of a score vector, an approximation of the derivative using a score function in place of the true likelihood, and the inverse of the Fischer information. The Rank test statistic takes the following form when applied to quantile regression for the null hypothesis above (Gutenbrunner et al., 1993).

$$T = \mathbf{S}' \mathbf{M}^{-1} \mathbf{S} / \Psi^2 \quad (1)$$

We include the following definitions along with the dimensions of each term in braces for clarity.

$$\begin{aligned} \mathbf{S}_{[p \times 1]} &= n^{-1/2} (\tilde{\mathbf{X}} - \mathbf{H} \tilde{\mathbf{X}})' \hat{\mathbf{b}} \\ \mathbf{H}_{[n \times n]} &= \mathbf{X} (\mathbf{X}' \mathbf{X})^{-1} \mathbf{X}' \\ \hat{\mathbf{b}}_{[n \times 1]} &= - \int \psi(t) d\hat{\mathbf{a}}(t) \\ \mathbf{M}_{[p \times p]} &= n^{-1} (\tilde{\mathbf{X}} - \mathbf{H} \tilde{\mathbf{X}})' (\tilde{\mathbf{X}} - \mathbf{H} \tilde{\mathbf{X}}) \end{aligned}$$

We now provide an intuitive explanation for each term.  $\mathbf{S}$  is the score vector for the test. It represents an approximate derivative of  $\beta$  under the null hypothesis. By formulating  $\mathbf{S}$  with the matrix  $\tilde{\mathbf{X}} - \mathbf{H} \tilde{\mathbf{X}}$ , the influence of  $\mathbf{X}$  (and  $\beta_1$ ) is removed, focusing the approximate derivative on  $\beta_2$ , the parameters of interest. When  $\mathbf{S}$  is large, it indicates that the null hypothesis is ill-suited to the data.

With the true likelihood unknown,  $\mathbf{S}$  is calculated using  $\hat{\mathbf{b}}$ , an  $n$  vector of scores calculated for each data point. These scores are computed by integrating the score function  $\psi(t)$  with respect to the regression rankscores  $\hat{\mathbf{a}}$  de-

finied by Gutenbrunner and Jureckov (1992). The regression rankscores allow the rank test to be applied to quantile regression by converting the multi-dimensional data into a single-dimensional ranking for the chosen quantile.  $\hat{\alpha}$  is equal to the dual solution of the quantile regression under the null hypothesis, and can be calculated using the primal solution  $\beta_1$ . The value  $\hat{\alpha}_i = 1$  if  $\beta_1 \mathbf{X}_i > 0$ , 0 if  $\beta_1 \mathbf{X}_i < 0$ , and between 0 and 1 if  $\beta_1 \mathbf{X}_i = 0$ , satisfying  $\mathbf{X}'\hat{\alpha} = (1 - \tau)\mathbf{X}'\mathbf{1}$ .

$\Psi^2 = \int (\psi(t) - \bar{\psi})^2 dt$  is an additional normalization term for the covariance of the score function. Koenker and Machado (1999) highlight the quantile score function  $\psi(t) = \tau - I(t < \tau)$ , which focuses the test on a specific quantile. This gives us  $\hat{\mathbf{b}}_i = \hat{\alpha}_i(\tau) - (1 - \tau)$  and  $\Psi^2 = \tau(1 - \tau)$ . With this choice of score function,  $\hat{\mathbf{b}}_i$  is either  $\tau$  if  $\beta_1 \mathbf{X}_i > 0$ ,  $\tau - 1$  if  $\beta_1 \mathbf{X}_i < 0$ , or a value inbetween otherwise.

The test statistic  $T$  follows a Chi-squared distribution with  $p$  degrees of freedom under the null hypothesis. It has the desirable properties of not depending on the error distribution, and not needing to learn the model under the alternative hypothesis. The values  $\hat{\mathbf{b}}$  are calculated under the null hypothesis that  $\beta_2 = \mathbf{0}$ .

### 3 METHODOLOGY

We start with a high level overview of our QSSS<sup>1</sup>. Given a dataset  $\mathbf{D} = \{\mathbf{Y}, \mathbf{X}, \mathbf{L}\}$ , and a list of starting locations  $\mathbf{P}$ , the QSSS searches over circular areas in  $\mathbf{L}$ , beginning at each starting location in  $\mathbf{P}$  and growing the regions one data point at a time, starting from some minimum number of points. The regions are grown as circles of increasing radius. Each time the region grows, we calculate its test statistic using our Incremental Rank test (Section 3.1). Once the region cannot be grown any larger, or reaches a maximum size, we move on to the next starting point in  $\mathbf{P}$ . After all starting points have been exhausted, an adjusted p-value is calculated for the region with the highest test statistic using a Gumbel correction (Section 3.2). We chose the Gumbel correction because it is much faster than the traditional randomization test. If the adjusted p-value is significant then the algorithm returns the region, otherwise it says that no significant region was found.

#### 3.1 FASTER RANK TEST FOR QSSS

In the QSSS framework, the Rank test needs to be performed for every circular subset  $\mathbf{C} \subseteq \mathbf{D}$ . We can choose a set of starting points (either each data point or a grid

<sup>1</sup>Matlab code for our experiments and algorithms can be found at <https://github.com/moortrav/QSSS>

formed over  $\mathbf{L}$ ) for the regions and grow each one, recalculating our hypothesis test each time the region overlaps a new data point. The inclusion of a new data point  $i$  into the region will change the  $i$ th row of  $\tilde{\mathbf{X}}$  from a row of zeros to the  $i$ th row of  $\mathbf{X}$ . Under the framework of the Rank test,  $\mathbf{X}$ ,  $\mathbf{H}$ , and  $\hat{\mathbf{b}}$  will be the same for every choice of region  $\mathbf{C}$ . Thus our only task is to update  $T$  as  $\tilde{\mathbf{X}}$  changes.

The primary bottlenecks in updating  $T$  are in updating  $\mathbf{S}$  and recomputing  $\mathbf{M}^{-1}$ .  $\mathbf{M}^{-1}$  can be updated incrementally using applications of the Sherman-Morrison formula (Sherman and Morrison, 1950), but a more efficient update can be performed by leveraging the special structure of  $T$ . Note that we can re-write  $T$  as

$$T = \hat{\mathbf{b}}\mathbf{Z}(\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\hat{\mathbf{b}}/\Psi^2 \quad (2)$$

where  $\mathbf{Z} = \tilde{\mathbf{X}} - \mathbf{H}\tilde{\mathbf{X}}$ .  $\mathbf{Z}(\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'$  is by definition a projection matrix onto the space of  $\mathbf{Z}$ . If we let  $\mathbf{U}$  be an  $n \times p$  orthonormal column basis of  $\mathbf{Z}$ , then

$$T = \hat{\mathbf{b}}\mathbf{U}\mathbf{U}'\hat{\mathbf{b}}/\Psi^2 \quad (3)$$

The inverse in Equation 2 is a normalization term. Since  $\mathbf{U}$  is already normalized, the formulation of Equation 3 allows us to forgo the matrix inverse calculation. Thus we can quickly recalculate  $T$  by performing incremental updates to our orthonormal basis  $\mathbf{U}$  as  $\tilde{\mathbf{X}}$  changes.

##### 3.1.1 Incremental Orthogonalization of Rank Test

Our goal is to take an existing orthonormal basis at iteration  $t$  (i.e.  $\mathbf{U}^t$ ), and calculate  $\mathbf{U}^{t+1}$  based on the (small) change in  $\tilde{\mathbf{X}}$  when a new data point is added to the inside region. To do this efficiently, we leverage the QR decomposibility of  $\mathbf{Z}^t$ , which enables a rank one update. However, we also need to efficiently preserve the QR decomposibility of  $\mathbf{Z}^{t+1}$ , which we do through a series of Givens rotations.

Let  $\mathbf{K}_{[n \times n]}$  be a row selector matrix, where  $K_{[j,j]} = 1$  if point  $j$  is in  $\mathbf{C}$ , and all other values are zero. For a current region  $\mathbf{C}^t$  at iteration  $t$ ,  $\tilde{\mathbf{X}} = \mathbf{K}^t \mathbf{X}$ . If we add point  $i$  to  $\tilde{\mathbf{X}}$  during iteration  $t + 1$ , then this is equivalent to changing the  $i$ th diagonal of  $\mathbf{K}^t$  from 0 to 1. We can express this change as a matrix sum  $\mathbf{K}^{t+1} = \mathbf{K}^t + \mathbf{K}_i$  where  $\mathbf{K}_i$  is zero except for the  $i$ th diagonal. This allows us to decompose the change in  $\mathbf{Z}^{t+1}$  as follows:

$$\mathbf{Z}^{t+1} = (\mathbf{K}^t + \mathbf{K}_i)\mathbf{X} - \mathbf{H}(\mathbf{K}^t + \mathbf{K}_i)\mathbf{X} \quad (4)$$

$$= \mathbf{Z}^t + \mathbf{K}_i\mathbf{X} - \mathbf{H}\mathbf{K}_i\mathbf{X} \quad (5)$$

$$= \mathbf{Z}^t + (\mathbf{e}_i - \mathbf{H}_i)'\mathbf{X}_i \quad (6)$$

where  $\mathbf{e}_i$  is the  $i$ th unit basis vector of size  $1 \times n$ . Note that  $\mathbf{e}_i'\mathbf{X}_i$  is an outer product producing a matrix of size  $n \times p$ .  $\mathbf{H}$  is a symmetric matrix, so we use the row vector  $\mathbf{H}_i$  to keep our notation consistent. In Equation 6 we have reduced the update to  $\mathbf{Z}^t$  to the product of a column and row vector i.e.  $(\mathbf{e}_i - \mathbf{H}_i)'\mathbf{X}_i$ . This means that the matrix added to  $\mathbf{Z}^t$  has a rank of one, and the change to each column of  $\mathbf{Z}^t$  is a multiple of the same column vector  $(\mathbf{e}_i - \mathbf{H}_i)'$ . We can use this special update structure in an algorithm to find  $\mathbf{U}^{t+1}$  efficiently.

If the QR factorization of  $\mathbf{Z}^t$  is known, where  $\mathbf{R}$  is an upper triangular matrix and  $\mathbf{Q} = \mathbf{U}^t$  is an orthonormal column basis, then the factorization for  $\mathbf{Z}^{t+1}$  can be found with the rank one update algorithm detailed in section 12.5.1 of Golub and Loan (2012). This algorithm lets us find the factorization  $\mathbf{Z}^{t+1} = \mathbf{Q}^{t+1}\mathbf{R}^{t+1}$  using the previous factorization  $\mathbf{Z}^t = \mathbf{Q}^t\mathbf{R}^t$ , giving us  $\mathbf{U}^{t+1} = \mathbf{Q}^{t+1}$  for our update to the test statistic  $T$ .

Let  $\mathbf{v} = \mathbf{e}_i - \mathbf{H}_i$ . We start by refactoring the update as

$$\mathbf{Z}^{t+1} = \mathbf{Q}^t\mathbf{R}^t + \mathbf{v}'\mathbf{X}_i = \mathbf{Q}^t(\mathbf{R}^t + \mathbf{w}'\mathbf{X}_i) \quad (7)$$

Where  $\mathbf{w}' = (\mathbf{Q}^t)^{-1}\mathbf{v}' = (\mathbf{Q}^t)'\mathbf{v}'$ . Our goal is to turn  $\mathbf{Z}^{t+1}$  into the product of an orthonormal matrix (which will be  $\mathbf{Q}^{t+1}$ ) and an upper triangular matrix to be produced from  $(\mathbf{R}^t + \mathbf{w}'\mathbf{X}_i)$  through Givens rotations. The details of the Golub and Loan (2012) algorithm that does this can be found in the supplemental materials.

This algorithm is not ideal in its current form, because creating the upper triangular matrix takes  $O(n)$  Givens rotations, a result of  $\mathbf{Q}$  being  $n \times n$ . However, the first  $p$  columns of  $\mathbf{Q}$  and  $p$  rows of  $\mathbf{R}$ , denoted as  $\mathbf{Q}_{[:,1:p]}$  and  $\mathbf{R}_{[1:p,:]}$ , are sufficient to reconstruct  $\mathbf{Z}$ , as  $\mathbf{Z} = \mathbf{Q}_{[:,1:p]}\mathbf{R}_{[1:p,:]} = \mathbf{Q}\mathbf{R}$ . Working with this reduced factorization would reduce the storage and number of Givens rotations required for the algorithm.

Unfortunately this representation is insufficient to perform the update. If we were to compute the vector  $\mathbf{w}$  from Equation 7 with  $\mathbf{Q}_{[:,1:p]}$ , then  $\mathbf{w}' = \mathbf{Q}'_{[:,1:p]}\mathbf{v}' = \mathbf{Q}'_{[:,1:p]}\mathbf{e}_i' - \mathbf{Q}'_{[:,1:p]}\mathbf{H}_i' = \mathbf{0}$ . To see this, note that  $\mathbf{Q}'_{[:,1:p]}\mathbf{e}_i'$  is zero because the  $i$ th row of  $\mathbf{Z}^t$  and  $\mathbf{Q}$  is zero, since the  $i$ th data point has not been added to the inside region yet.  $\mathbf{Q}'_{[:,1:p]}\mathbf{H}_i'$  is also zero because  $\mathbf{H}_i$  is perpendicular to  $\mathbf{Z}^t$  and thus perpendicular to  $\mathbf{Q}_{[:,1:p]}$ . With  $\mathbf{w}' = \mathbf{0}$ , Equation 7 becomes  $\mathbf{Z}^{t+1} = \mathbf{Q}^t_{[:,1:p]}\mathbf{R}^t_{[1:p,:]}$ ,

which completely ignores the update term. Intuitively speaking, we cannot update the column basis of  $\mathbf{Z}^t$  by only considering that basis.

Fortunately, there is a way to summarize the influence of the last  $n-p$  columns of  $\mathbf{Q}$ , denoted  $\mathbf{Q}_{[:,(p+1):n]}$ , into a single vector. When the Givens rotations zero out element  $j$  in  $\mathbf{w}$ , it changes element  $j - 1$  to  $\sqrt{w_{j-1}^2 + w_j^2}$ . Consequently, the result of rotations  $\mathbf{J}'_{p+1} \dots \mathbf{J}'_{n-1}$  will set  $w_{p+1} = \sqrt{w_{p+1}^2 + \dots + w_n^2} = \sqrt{\sum_{j=p+1}^n (\mathbf{Q}'_j\mathbf{H}_i)^2} = |\mathbf{Q}_{[:,(p+1):n]}\mathbf{H}_i|$ . Because  $\mathbf{Q}_{[:,1:p]}$  is perpendicular to  $\mathbf{H}_i$ , the columns  $\mathbf{Q}_{[:,p+1]} \dots \mathbf{Q}_{[:,n]}$  are an orthonormal basis of  $\mathbf{H}_i$ . Projecting  $\mathbf{H}_i$  onto its own basis will preserve its length, giving us  $|\mathbf{Q}_{[:,(p+1):n]}\mathbf{H}_i| = |\mathbf{H}_i|$ . Thus, we can summarize all  $n-p$  Givens rotations with a single vector  $\mathbf{q}$  such that  $\mathbf{q}\mathbf{H}_i = |\mathbf{H}_i|$ , which gives us  $\mathbf{q} = \mathbf{H}_i/|\mathbf{H}_i|$ . If we append  $\mathbf{q}$  as a new column of  $\mathbf{Q}_{[:,1:p]}$  to produce  $\tilde{\mathbf{Q}}_{[:,1:p]}$  and a zero row to the bottom of  $\mathbf{R}_{[1:p,:]}$  to produce  $\tilde{\mathbf{R}}_{[1:p,:]}$  then we can run the algorithm with only  $O(p)$  Givens rotations and still produce the same result. Since  $\mathbf{q}$  is normalized and perpendicular to  $\mathbf{Q}_{[:,1:p]}$ ,  $\tilde{\mathbf{Q}}_{[:,1:p]}$  is still orthonormal.

We can perform the rank one update on  $\tilde{\mathbf{Q}}_{[:,1:p]}^t$  and  $\tilde{\mathbf{R}}_{[1:p,:]}^t$  using the algorithm in Golub and Loan (2012). The first  $p$  columns of  $\tilde{\mathbf{Q}}_{[:,1:p]}^{t+1}$  make our new orthonormal column basis  $\mathbf{U}^{t+1}$  used to calculate our test statistic  $T$ .

---

#### Algorithm 1 Incremental Rank Test

---

Inputs:  $\mathbf{X}, \mathbf{H}, \hat{\mathbf{b}}, \mathbf{Q}, \mathbf{R}, \tau, i$   
 $\mathbf{v} = \mathbf{e}_i - \mathbf{H}_i$   
 $\mathbf{Q}, \mathbf{R} = \text{qr\_update}(\mathbf{Q}, \mathbf{R}, \mathbf{v}, \mathbf{X}_i)$   
 $T = \hat{\mathbf{b}}'\mathbf{Q}\mathbf{Q}'\hat{\mathbf{b}}/(\tau(1 - \tau))$   
Return( $T$ )

---

Algorithm 1 shows the incremental rank test which calls `qr_update`. It takes the index  $i$  of the datapoint being added to the region, along with the QR factorization for the previous iteration as inputs. The details of `qr_update` can be found in the supplementary materials. We can run the algorithm with either the full QR factorization, or the abridged form represented by  $\tilde{\mathbf{Q}}_{[:,1:p]}$  and  $\tilde{\mathbf{R}}_{[1:p,:]}$

Note that our incremental rank test is not an approximation as it computes the test statistic (Equation 1) exactly.

### 3.1.2 Update Runtime

With our compact representation for  $\tilde{\mathbf{Q}}_{[:,1:p]}$  and  $\tilde{\mathbf{R}}_{[1:p,:]}$ , the rank one update to our QR factorization takes  $O(np)$  time. Each Givens rotation is an  $O(n)$  operation, and we perform  $O(p)$  of them in total. Once  $\mathbf{U}^{t+1}$  is found,

$T^{t+1}$  can be calculated in  $O(np)$  time by computing  $\hat{\mathbf{b}}\mathbf{U}^{t+1} = \mathbf{u}$ , and then finding  $T^{t+1} = \mathbf{u}\mathbf{u}'$ . Thus the entire update to  $T$  can be performed in  $O(np)$  time when a single point is added to  $\tilde{\mathbf{X}}$ .

### 3.2 MULTIPLE HYPOTHESIS TEST CORRECTION

To account for the multiple hypothesis test problem, we perform a correction using the method in Abrams et al. (2010). We generate 1000 simulations of the data under the null hypothesis. The maximum test statistic from each of these simulations are used to fit the parameters  $\mu, \gamma$  of a Gumbel distribution. We calculate the adjusted p-value of a region with test statistic  $T$  as  $1 - g(T|\mu, \gamma)$ , where  $g$  is the CDF of the Gumbel distribution. This tells us the rarity of drawing a value at least as large as  $T$  from the distribution of maximum test statistics. In all of our applications we report the most significant region found by QSSS, provided that the adjusted p-value of the region is less than 0.05. Otherwise no significantly different region is found.

## 4 RESULTS

### 4.1 SYNTHETIC DATA EXPERIMENTS

We begin by demonstrating the speedup from our incremental formulation of the Rank test, followed by a comparison of the Rank test to other possible choices of hypothesis tests. We use synthetic data to evaluate these two criteria, as it is easy to generate in large quantities, and it can contain a verifiable ground truth.

#### 4.1.1 Simulator

The purpose of our simulator is to inject data points in spatial regions where the data distribution is altered at a specific percentile. We start with a default distribution, then modify a specific range of the distribution for a random spatial region. This acts as the target region for the algorithm to identify. A detailed description of our simulator can be found in the supplemental materials.

#### 4.1.2 Incremental Rank Test Timing

Using our simulated data, we compare the runtime of our incremental version of the Rank test to its naive (non-incremental) formulation. For each algorithm we calculated the Rank test statistic  $T$ , starting from a base radius, then expanding to include 100 new points. In Figure 1 we show the average time, in milliseconds, that each algorithm took to calculate  $T$  when a new point was added. These tests were done for increasing values of  $n$  while

keeping  $p$  constant at 5. The two algorithms start out at similar times when  $n = 1000$ , but quickly diverge. At  $n = 16,000$  our incremental Rank test takes only 2.83 ms to compute each update, while the non-incremental version takes 166.9 ms. The incremental speedup for the Rank test makes it usable within the framework of the QSSS, while the naive calculation would take far too much time to be feasible for large  $n$ .

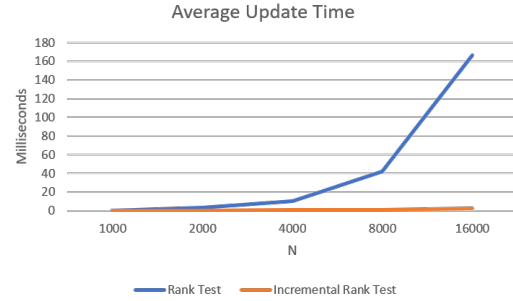


Figure 1: Average time to update the Rank test statistic, for the full and incremental formulations. Times were averaged over 100 updates for randomly generated data with different values of  $n$  and constant  $p = 5$ .

### 4.1.3 Comparison against Other Baselines

We are unaware of other methods that can solve exactly the same problem as the QSSS. As a result, we develop three baseline algorithms that can be used for comparison. We create the first baseline by adapting the SSS to account for covariates by modeling the response for the inside (and outside) region using least squares regression. This baseline compares means (not quantiles) of distributions by using a likelihood ratio test. We add the Mean test to show the ineffectiveness of a mean-based test statistic in finding regions that differ only in specific quantiles. For the second baseline, we modify the SSS to focus on a specific quantile of the distribution. We do this by fitting a  $\tau$ th quantile regression with coefficients  $\beta$  to the entire data set; then, for each test region, this baseline calculates Mood's test at  $\tau$ , which is a statistic from a  $2 \times 2$  Chi-Squared table that compares the number of points above and below the  $\beta$  plane from both inside and outside of the region. Finally, our third baseline is similar to the second baseline but it replaces Mood's test with the more powerful but computationally expensive likelihood ratio test from Koenker and Machado (1999) (LR) for quantile regression. This LR test forms a Chi-squared statistic from the residuals of the quantile regressions fit to the null and alternative models.

Using our simulator, we produced 30 randomly generated data sets with  $n = 1000$ .  $B_2$  (parameters for the

injected data) is the same as  $B_1$  (parameters for the normal data) in these datasets, except between the 70th and 100th percentiles of the distribution. 100 of the points are generated from  $f(B_2)$  and 900 of the points are generated from  $f(B_1)$ . Our Moods, LR, and Rank test search for regions that differ at the 90th percentile. For each algorithm, we look at the most significant region found for each dataset, provided it has a p-value of at most 0.05 after the Gumbel correction. Otherwise we count the algorithm as finding no significant region for that dataset. Note that this experiment setup is extremely challenging. The ground truth region to detect makes up 10% of the total dataset, but only 30% of the points in the region on average indicate that it has a different distribution. Adding in the random noise term further complicates the detection task.

<b>P=3</b>	<b>Moods</b>	<b>LR</b>	<b>Rank</b>	<b>Mean</b>
Precision	0.322	0.499	<b>0.576</b>	0.405
Recall	0.353	<b>0.548</b>	0.500	0.334
F1	0.337	0.522	<b>0.535</b>	0.366
Time (s)	<b>3.32*</b>	350.73	46.64	31.06
<b>P=5</b>	<b>Moods</b>	<b>LR</b>	<b>Rank</b>	<b>Mean</b>
Precision	0.259	0.395	<b>0.508</b>	0.216
Recall	0.320	0.416	<b>0.484</b>	0.110
F1	0.286	0.405	<b>0.495</b>	0.146
Time (s)	<b>3.02*</b>	310.65	84.36	39.25
<b>P=10</b>	<b>Moods</b>	<b>LR</b>	<b>Rank</b>	<b>Mean</b>
Precision	0.286	0.243	<b>0.676*</b>	0.197
Recall	0.344	0.278	<b>0.442</b>	0.169
F1	0.312	0.259	<b>0.535*</b>	0.182
Time (s)	<b>3.26*</b>	379.32	83.31	38.25

Table 1: The precision, recall, F1 and running time of QSSS on synthetic data using various algorithms. The \* indicates statistical significance (paired t-test,  $\alpha = 0.05$ ).

Table 1 shows the results of the simulation experiments. The precision, recall and F1 score of each algorithm is reported in the task of finding the region generated from  $B_2$  in each dataset. These values are calculated on a per data point basis for each dataset, then averaged over the 30 datasets. Three experiment runs were performed, with dimensionality  $p = 3, 5$  and 10. LR and Rank are the two most accurate tests for  $p = 3$  and 5, with Rank being the most accurate for  $p = 10$ . The poor performance of Mood’s and Mean is expected, since Mood’s is a low power test and Mean is ill-suited to find such subtle distributional variations.

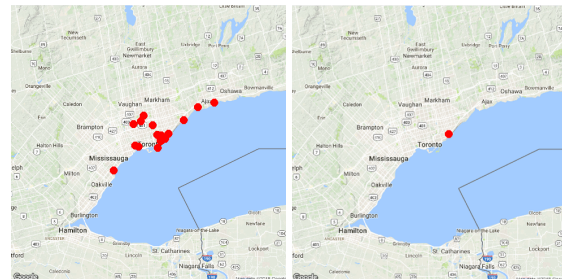
Table 1 also shows the average total runtime of each spatial scan algorithm on the simulation data. This table illustrates the speed of Mood’s test compared to the other hypothesis tests. We can also see that the LR test is sig-

nificantly slower than the others, a result of needing to fit a quantile regression to the alternative model for every test region. In our implementation of LR, we use warm-starting to increase the speed of the quantile regression algorithm as the regions grow point by point. Even with warmstarting, the LR algorithm still takes at least four times longer to run on the simulation data than our other hypothesis choices.

Comparing the accuracy and timing results, we see that while the Rank and LR test are the most accurate, the Rank test offers the best tradeoff in terms of usability between speed and power. We found it infeasible to use the computationally expensive LR test, even on moderately sized datasets. While the Mood’s and Mean tests were faster, neither one was very capable at detecting differences in our simulated data. Due to these result, we primarily use the Rank test in our case studies below; we also include results from the Mean test to illustrate the differences between the two.

## 4.2 ROBUSTNESS TO OUTLIERS

One of the benefits of quantile based analysis is that it is more robust to data outliers than mean-based methods. We illustrate how this can affect spatial scan analysis using eButterfly data as an example use case.



(a) 50th Percentile

(b) Mean

Figure 2: Most significant regions found from eButterfly data, with data points in the region shown as red dots. The figures are zoomed in on the Toronto region for visibility. Figure 2a is from the QSSS algorithm at the 50th percentile, Figure 2b is from the mean regression spatial scan. The region in 2b represents a single location, rather than an area, as all the data from that subset have the same location parameters.

Citizen science biodiversity monitoring programs, such as eButterfly (Prudic et al., 2017), play an important role in ecology as it informs species distribution models and also conservation programs. Citizen scientists participating in these programs submit checklists which record observations of certain types of organisms, such as butterflies in the case of eButterfly, identified by species.

We construct a dataset out of the abundance counts of monarch butterflies (i.e. the number of butterfly individuals observed) in Ontario in 2016. Quantile regression on count data can be addressed using the smoothing method in Machado and Silva (2002), which turns the counts into continuous values by adding uniform noise. This transformation allows us to perform inference with the Rank test as we would with continuous data.

In our analysis, we include the time spent observing for each checklist as the covariate, since there should be a strong correlation between this value and the number of monarchs observed. We ran our QSSS algorithm on several different quantiles and compared the top region for each to the top region found by a mean-based least squares spatial scan.

Figure 2 shows the most significant regions found by the mean regression spatial scan and QSSS at the 50th percentile<sup>2</sup>. Inspection of the data, and verification with domain experts at eButterfly reveal two interesting results identified by the algorithms. Within the data time window there is a single observer who heavily skews the distribution. This observer was involved in a monarch tagging project, and submitted a significant number of very high monarch checklists. The region found by the mean spatial scan only includes the checklists from this observer, all at the same spatial location. When QSSS is run at the 50th percentile, a different trend emerges. The checklists from the observer has much less influence on the model at this level, and the algorithm instead picks up an area of high monarch counts due to migration routes around the great lakes.

If we were limited to only mean-based spatial scans, we would have to filter out the outlier data from the monarch tagging observer to find the desired trends in the dataset. Being able to adjust the percentile of QSSS allows us to reduce the influence of outliers as desired, without explicit removal of outliers from the data.

### 4.3 QUANTILE BASED REGION DETECTION

We now demonstrate the usefulness of detecting unusual spatial regions based on different quantiles.

#### 4.3.1 Education and Unemployment Data

We combine the county-level education and unemployment datasets from the USDA Economic Research Service web page (Parker, 2017). We use the county-level unemployment rates from 2016 as the response variable, and combine the education percentages from 2012-2016

<sup>2</sup>All maps generated using ggmap in R (Kahle and Wickham, 2013)

with median household income (as percentage of state total) values from 2016 as the covariates. The education percentages are the proportion of adults in each county with less than a high school diploma, just a high school diploma, one to three years of college, and four years of college or more. We only use the counties from the continental US.

We ran our QSSS algorithm on the 10th and 90th percentile of the data, along with a mean-based approach using least squares regression. Figure 3 shows the most significant region found by each algorithm. Both the mean and 90th percentile search found the Appalachian region that intersects Kentucky, West Virginia and Virginia, which is well-known to have high unemployment rates with the collapse of the coal industry (Caruthers, 2016). In the 10th percentile region, South Dakota, North Dakota, Nebraska, and Colorado are rated 2,3,4, and 6 in unemployment in the continental US as a whole. This middle region of the country enjoys lower unemployment rates due to the local oil industry and relatively low fallout from the Great Recession (DePillis, 2018). The most significant region discovered at the 10th percentile has a 2 point lower unemployment rate on average, which is abnormally low even when compared to other low unemployment areas.

The unemployment data results highlight the fact that the QSSS, unlike the mean scan, can identify multiple trends in a dataset by changing the modeled quantile.

#### 4.3.2 eBird

The final case study presents the results of applying QSSS to eBird (Sullivan et al., 2014) data. The eBird project collects bird observation checklists from citizen scientists around the world. We compiled two datasets from eBird data collected in 2017 between March and April. These datasets correspond to two different Bird Conservation Regions (BCRs) within the U.S. We divide the data by BCR because they represent cohesive habitats for different bird species. Our choice of March and April is to mitigate the effects of seasonality on the algorithms.

Different from our eButterfly study, we used the total number of species observed from each checklist as our response variable, and the time spent observing as the covariate. Past work has shown that the number of species observed per unit time is highly predictive of the skill level of an observer (Kelling et al., 2015). We use the same count smoothing approach on the eBird data as we did on eButterfly to fit the quantile regression model.

Figure 4 shows the most significant regions found for the mean spatial scan and our QSSS run at the 90th percentile. We corresponded with domain experts from

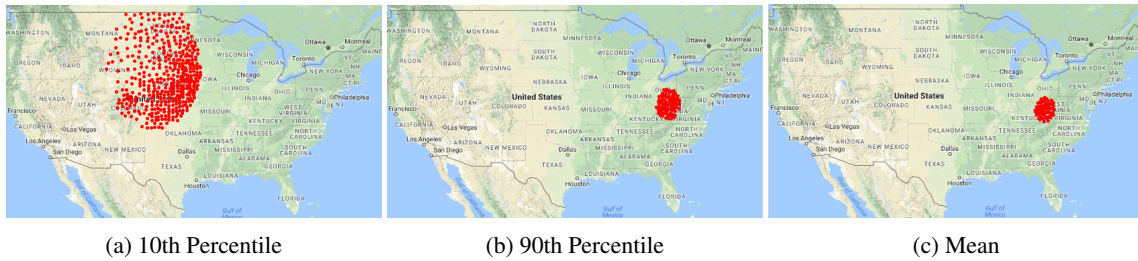


Figure 3: Most significant region found by the QSSS algorithm for the 10th and 90th percentiles of the Education and Unemployment dataset. Most significant region by the mean spatial scan is included for comparison. Regions are illustrated by the centroids of the counties they contain.

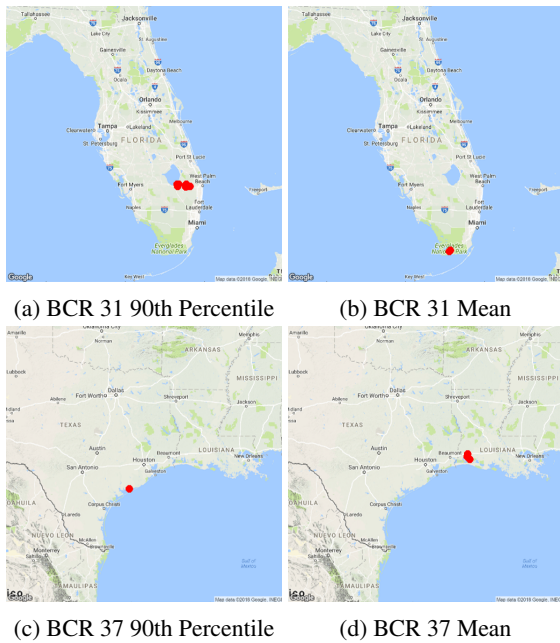


Figure 4: The most significant regions found by QSSS at the 90th percentile and by the mean spatial scan on eBird data from BCRs 31 (Florida) and 37 (Gulf coast). Data points within a region are shown as red dots.

eBird, who offered an analysis on the regions detected.

For BCR 31, the QSSS found an unusual birding location – one that is less frequented by beginners. The birders who visit this region are highly skilled and are able to continue observing a high number of bird species as they stay there. In contrast, the mean scan found a popular species-rich hotspot in the Everglades frequented by both experts and novices. This region has many large wading birds which are easy to see and identify initially.

In BCR 37, the QSSS found a hotspot in Matagorda Bay because it has an unusually high number of bird species along the shoreline that can be readily observed as compared to the surrounding area. Our domain expert com-

mented that the area found by the mean scan was an area that was not particularly high in species. Upon inspecting the models for the inside versus outside region, we found that the models indicate that observers appear to find less species initially inside that area than outside that area.

The mean scan and QSSS algorithms both found very different but meaningful regions for the BCRs. We hypothesize that the QSSS is finding unusual areas in terms of the observation process for more skilled observers (as in BCR 31) and we will continue our analysis on other BCRs in future work.

## 5 FUTURE WORK AND CONCLUSION

The QSSS discovers unusual spatial regions that differ from the surrounding area. The inner loop of the algorithm relies on comparing quantile regressions fit to data from inside and outside a region under consideration. To perform these comparisons efficiently, we developed an incremental rank test, which is over an order of magnitude faster than a naive implementation. Our results on simulated data and on three real-world datasets show that QSSS enables a new type of analysis for spatial data that is different from mean-based methods and that the QSSS is also robust to outliers. For future work, we would like to investigate reporting the top  $K$  most unusual regions rather than the top 1 and we would also like to extend our work to find unusual regions in both space and time.

## ACKNOWLEDGEMENTS

We thank our collaborators from eButterfly (Katy Prudic, Max Larrivee, Jeffrey Oliver and Jeremy Kerr) and eBird (Daniel Fink, Chris Wood). Moore was supported by NSF grant CCF-1521687. This material is based upon work while Wong was serving at NSF. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.



## References

- Abrams, A., Kleinman, K., and Kulldorff, M. (2010). Gumbel based p-value approximations for spatial scan statistics. *Int J Health Geogr*, 9(1):61.
- Best, N., Richardson, S., and Thomson, A. (2005). A comparison of Bayesian spatial models for disease mapping. *Stat Methods Med Res*, 14(1):3559.
- Caruthers, A. (2016). Mapping poverty in the appalachian region. <https://www.communitycommons.org/2016/08/mapping-poverty-in-the-appalachian-region/>.
- DePillis, L. (2018). What america can learn from cities with super-low unemployment. <http://money.cnn.com/2018/01/12/news/economy/cities-unemployment/index.html>.
- Duczmal, L. and Assuncao, R. (2004). A simulated annealing strategy for the detection of arbitrarily shaped spatial clusters. *Comput Stat Data Anal*, 45:269286.
- Golub, G. and Loan, C. V. (2012). *Matrix computations*, volume 3. JHU Press.
- Gutenbrunner, C. and Jureckov, J. (1992). Regression rank scores and regression quantiles. *The Annals of Statistics*, pages 305–330.
- Gutenbrunner, C., Jurekov, J. K. R. S., Koenker, R., and Portnoy, S. (1993). Tests of linear hypotheses based on regression rank scores. *J Nonparametr Stat*, 2(4):307–331.
- Kahle, D. and Wickham, H. (2013). ggmap: Spatial visualization with ggplot2. *The R Journal*, 5(1):144–161.
- Kelling, S., Johnston, A., Hochachka, W. M., Iliff, M., Fink, D., Gerbracht, J., Lagoze, C., Sorte, F. A. L., Moore, T., Wiggins, A., Wong, W.-K., Wood, C., and Yu, J. (2015). Can observation skills of citizen scientists be estimated using species accumulation curves? *PLoS ONE*, 10(10):e0139600.
- Koenker, R. and Bassett, G. (1978). Regression quantiles. *Econometrica*, 46(1):33–50.
- Koenker, R. and Machado, J. A. (1999). Goodness of fit and related inference processes for quantile regression. *J Am Stat Assoc*, 94(448):1296–1310.
- Kulldorff, M. (1997). A spatial scan statistic. *Communications in Statistics: Theory and Methods*, 26(6):1481–1496.
- Lan, L., Malbasa, V., and Vucetic, S. (2014). Spatial scan for disease mapping on a mobile population. In *Proceedings of the 28th AAAI Conference*, pages 431–437.
- Machado, J. and Silva, J. S. (2002). Quantiles for counts. *J Am Stat Assoc*, 100(472):1226–1237.
- Macmillan, D. P. (2013). *Quantile Regression for Spatial Data*. Springer-Verlag, Berlin.
- McFowland, III, E., Somanchi, S., and Neill, D. B. (2018). Efficient Discovery of Heterogeneous Treatment Effects in Randomized Experiments via Anomalous Pattern Detection. *ArXiv e-prints*, 1803.09159.
- Mood, A. (1950). *Introduction to the Theory of Statistics*. McGraw Hill Book Co.
- Moore, T. and Wong, W.-K. (2015). Discovering hotspots and coldspots of species richness in ebird data. In *Proceedings of the AAAI-15 Workshop on Computational Sustainability*.
- Neill, D. and Moore, A. W. (2004). Rapid detection of significant spatial clusters. In *Proceedings of the 10th ACM SIGKDD Conference*, pages 256–265.
- Neill, D. B. (2012). Fast subset scan for spatial pattern detection. *J R Stat Soc Series B Stat Methodol*, 74(2):337–360.
- Parker, T. (2017). Download data. United States Department of Agriculture. <https://www.ers.usda.gov/data-products/county-level-data-sets/county-level-data-sets-download-data/>.
- Prudic, K., McFarland, K., Oliver, J., Hutchinson, R., Long, E., Kerr, J., and Larrive, M. (2017). ebutterfly: leveraging massive online citizen science for butterfly conservation. <http://www.mdpi.com/2075-4450/8/2/53/htm>.
- Reich, B. J., Fuentes, M., and Dunson, D. B. (2011). Bayesian spatial quantile regression. *J Am Stat Assoc*, 106(493):6–20.
- Rousseeuw, P. J. and Leroy, A. M. (1987). *Robust Regression and Outlier Detection*. John Wiley and Sons.
- Sherman, J. and Morrison, W. J. (1950). Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics*, 21(1):124–127.
- Sullivan, B. L., Aycrigg, J. L., Barry, J. H., Bonney, R. E., Bruns, N., Cooper, C. B., Damoulas, T., Dhondt, A. A., Dieterich, T., Farnsworth, A., Fink, D., Fitzpatrick, J. W., Fredericks, T., Gerbracht, J., Gomes, C., Hochachka, W. M., Iliff, M. J., Lagoze, C., Sorte, F. L., Merrifield, M., Morris, W., Phillips, T. B., Reynolds, M., Rodewald, A. D., Rosenberg, K. V., Trautmann, N. M., Wiggins, A., Winkler, D. W., Wong, W.-K., Wood, C. L., Yu, J., and Kelling, S. (2014). The ebird enterprise: An integrated approach to development and application of citizen science. *Biological Conservation*, 169:31–40.

---

# Stable Gradient Descent

---

Yingxue Zhou

Sheng Chen

Arindam Banerjee

{zhou0877@umn.edu, shengc, banerjee@cs.umn.edu}

Department of Computer Science & Engineering  
University of Minnesota, Twin Cities

## Abstract

The goal of many machine learning tasks is to learn a model that has small population risk. While mini-batch stochastic gradient descent (SGD) and variants are popular approaches for achieving this goal, it is hard to prescribe a clear stopping criterion and to establish high probability convergence bounds to the population risk. In this paper, we introduce *Stable Gradient Descent* which validates stochastic gradient computations by splitting data into training and validation sets and reuses samples using a differential private mechanism. StGD comes with a natural upper bound on the number of iterations and has high-probability convergence to the population risk. Experimental results illustrate that StGD is empirically competitive and often better than SGD and GD.

## 1 INTRODUCTION

The primary goal in several machine learning tasks is to learn a model with finite training samples that generalizes well to unseen instances. One typically attempt to solve the following optimization problem which finds a minimizer  $w^*$  of the *population risk*  $F$  over some model class  $\mathcal{W}$ :

$$w^* \in \operatorname{argmin}_{w \in \mathcal{W}} F(w) \triangleq \mathbb{E}_{z \sim \mathcal{P}}[l(w, z)], \quad (1)$$

where  $z \in \mathcal{Z}$  is a data point in domain  $\mathcal{Z}$  following the unknown distribution  $\mathcal{P}$ , and  $l : \mathcal{W} \times \mathcal{Z} \mapsto \mathbb{R}$  is a certain loss function associated with the learning problem. For example, in classification problems  $z = (\mathbf{x}, y)$  is an instance-label pair,  $w$  denotes a classifier, and  $l(w, z)$  can be the hinge loss or logistic loss.

Due to the unavailability of distribution  $\mathcal{P}$ , the challenge of a learning algorithm is to search for an approximate minimizer  $\hat{w}_n$  of the risk  $F$  based *only* on a set of finite samples  $Z_n = \{z_1, z_2, \dots, z_n\}$ . A good criterion for quantifying the quality of  $\hat{w}_n$  is through the *excess risk*:

$$F(\hat{w}_n) - F(w^*). \quad (2)$$

A learning algorithm for obtaining  $\hat{w}_n$  should have small excess risk. Being a function of the random set of samples  $Z_n$ ,  $F(\hat{w}_n)$  is a random variable and a good learning algorithm is expected to have small excess risk with high probability.

In the literature, there are several approaches to tackle the problem. The classical approach is *empirical risk minimization* (ERM) (Shalev-Shwartz and Ben-David [2014]) which aims to find an empirical minimizer defined as  $\hat{w}_n^{ERM} \in \operatorname{argmin}_{w \in \mathcal{W}} \hat{F}(w) \triangleq \frac{1}{n} \sum_{j=1}^n l(w, z_j)$ . Usually first-order iterative optimization methods such as *gradient descent* (GD) are used to obtain the minimizer. However, GD uses all samples to compute gradients in each iteration and is quite slow for large datasets. A popular approach in modern machine learning is *stochastic gradient descent* (SGD) (Zhang [2004]; Rakhlin et al. [2012]; Hazan and Kale [2014]). SGD

minimizes the population risk directly by descending along stochastic gradients, computed based on a single sample or a mini-batch of samples. The stochastic gradient equals the population gradient in expectation (Shalev-Shwartz and Ben-David [2014]). Convergence of SGD is typically analyzed in expectation rather than with high probability with a few notable exceptions (Rakhlin et al. [2012]). In practice, since the improvements in the objective function value is non-monotonic, it is hard to prescribe a theoretically well grounded stopping criterion.

Conceptually, a mini-batch SGD algorithm would be much more well behaved if we had access to  $n$  fresh samples in each iteration. In such a setting, one would be able establish high probability bounds on the sample gradients staying close to the population gradient across all iterations, leading to high probability bounds on the excess risk. In this paper, we introduce *Stable Gradient Descent* (StGD), which behaves similar to the ideal case of  $n$  fresh samples per iteration using ideas from *adaptive data analysis* (Dwork et al. [2015c]) and *differential privacy* (Dwork and Roth [2014]). Each iteration leads to a small privacy loss which, unlike SGD, automatically puts a bound on the number of iterations StGD can be run. We present basic and mini-batch StGD and provide high probability bounds on the excess risk for different types of convex loss functions.

The main idea in StGD is simple: separate the available samples into a training and a validation set, compute stochastic gradient on both, and check if they are close. If they are indeed close, there is confidence in that descent direction and StGD proceeds with the descent. On the other hand, if they are not close, there is lack of confidence in the descent direction, and StGD uses a noisy version of the estimated gradient to do descent. The challenge in naively carrying out the simple idea is that the algorithm may overfit on both the training and the validation set. As a result, StGD carries out the comparison of training and validation gradients using a differentially private mechanism, which allows StGD to reuse the same samples over iterations

but still get high probability bounds across all iterations as if the samples were fresh.

The remainder of the paper is organized as follows. Section 2 describes related work. We present basic StGD in Section 3, and present mini-batch StGD in Section 4, along with high probability excess risk bounds. We present experimental results in Section 5, and conclude in Section 6. All technical proofs are deferred to the supplementary material.

## 2 RELATED WORK

**ERM and SGD:** For ERM, a common algorithm is gradient descent (Shalev-Shwartz and Ben-David [2014]) which computes the full gradient of the empirical risk and takes a step along it at each iteration. The performance of ERM is usually measured in terms of the uniform convergence of  $\hat{F}(w)$  to  $F(w)$  over  $\mathcal{W}$ . (Hardt et al. [2015]) analyzed the stochastic gradient method (SGM) for ERM in terms of stability and optimization error:  $\mathbb{E}[F(\hat{w}_n)] - F(w^*) \leq \epsilon_{opt}(w) + \epsilon_{stab}$ . They demonstrated that for  $L$ -Lipschitz continuous function, SGM has the convergence rate of  $O(L/\sqrt{n})$ . SGD minimizes the population risk by allowing the optimization procedure to take a step along a random direction, as long as the expected value of the direction is the population gradient (Shalev-Shwartz and Ben-David [2014]). In this case, with  $n$  to be the number of stochastic gradient computations. For strongly convex functions, SGD can achieve an expectation risk bound of rate  $O(1/n)$  (Hazan and Kale [2014], Rakhlin et al. [2012]). (Rakhlin et al. [2012]) also presented a similar high probability bound for SGD.

**Differential Privacy:** Informally, differential private analysis ensures that the outcome of analysis on two nearly identical input datasets (different on a single component) should also be nearly identical. As a result, an analyst will not be able to distinguish any single data by comparing the change of output. In the context of machine learning, this randomized algorithm  $\mathcal{M}$  can be a learning algorithm that outputs a classifier  $\mathcal{M}(D) = f$  where  $D$  is the training set. Some work (Chaudhuri et al. [2011], Bass-

ily et al. [2014]) introduced differential privacy to ERM to protect sensitive information about training data. (Dwork et al. [2015b], Dwork et al. [2015a]) introduced differential privacy to adaptive data analysis (ADA). In ADA, analyst test adaptively generated hypotheses on one holdout set where those hypotheses have dependence on the holdout set. To ensure the repeatedly used holdout set to provide valid validations, they designed a *Thresholdout* mechanism which allows the analyst to query the holdout set via a differentially private way. They showed differentially private reused holdout set maintains the statistical nature of fresh sample.

The main contribution of this paper is, we introduce differential privacy to gradient descent by applying *Thresholdout* to the training set. We show that the training set can be reused and maintains the statistical nature of fresh sample in all iterations. Mathematically speaking, gradients computed on the differentially-private reused training set concentrate around the population value with high probability. We exploit the concentration property to derive high-probability risk bounds of StGD.

### 3 STABLE GRADIENT DESCENT

#### 3.1 PRELIMINARIES

We consider the problem of minimizing the population risk defined in Equation 1. We denote  $\nabla l(w, z)$  as the gradient of  $l(w, z)$ . We use  $\nabla_i l(\cdot)$  to denote the  $i$ -th coordinate of  $\nabla l(\cdot)$ . Besides, we use  $g_i(w)$ ,  $\tilde{g}_i(w)$  and  $\hat{g}_i(w)$  to be the  $i$ -th coordinate of  $g(w)$  (population gradient),  $\tilde{g}(w)$  (gradient computed by StGD) and  $\hat{g}(w)$  (empirical gradient), respectively, for  $i \in \{1, \dots, d\}$ , where  $d$  is the dimension of  $w$ . For example, given the sample set  $S$ ,  $\hat{g}(w) = \sum_{z_j \in S} \nabla l(w, z_j) / |S|$ . We consider some special cases of  $F(w)$  with the following assumptions (Boyd and Vandenberghe [2004]):

**1. Convex and  $L$ -Lipschitz:** Function  $F$  is convex with  $L$ -Lipschitz if for all  $w, w' \in \mathcal{W}$  and  $L \geq 0$ :

$$|F(w') - F(w)| \leq L \|w' - w\|.$$

**2. Strongly Convex:** Formally, a function  $F$

---

#### Algorithm 1 Stable Gradient Descent (StGD) Algorithm

---

- 1: **Input:** Dataset  $S$ , certain loss  $l(\cdot)$ , initial point  $w_0$ .
  - 2: **Set:** Noise variance  $\sigma$ , iteration time  $T$ , step size  $\eta$ .
  - 3: Separate  $S$  randomly and evenly into  $S_t$  and  $S_h$ .
  - 4: **for**  $s = 0, \dots, T$  **do**
  - 5:   Run **DPGC**( $S_t, S_h, w_s, \sigma, l(\cdot)$ ) to compute gradient  $\tilde{g}(w_s)$ .
  - 6:    $w_{s+1} = w_s - \eta \tilde{g}(w_s)$ .
  - 7: **end for**
- 

is  $\alpha$ -strongly convex, if for all  $w, w' \in \mathcal{W}$  and any subgradient  $g(w)$  of  $F$  at  $w$ , we have

$$F(w') \geq F(w) + (w' - w)^T g(w) + \frac{\alpha}{2} \|w' - w\|^2.$$

**3. Smooth:** We say a function  $F$  is  $\beta$ -smooth, if  $w, w' \in \mathcal{W}$  and any subgradient  $g(w)$  of  $F$  at  $w$ , we have

$$F(w') \leq F(w) + (w' - w)^T g(w) + \frac{\beta}{2} \|w' - w\|^2.$$

#### 3.2 STGD ALGORITHM

We present StGD in two parts: Algorithm 1 and Algorithm 2 (**DPGC**). To simplify, we suppose there are  $2n$  available samples  $S \sim \mathcal{P}^{2n}$ . The StGD randomly and evenly separates them into two datasets: training set  $S_t$  and validation set  $S_h$ , both of which are of size  $n$ . We set a noise parameter  $\sigma$  and the total iterations  $T$ . We analyze the optimal values of parameters  $\sigma$  and  $T$  in the next section. Starting from initial point  $w_0$ , at each  $s$ -th iteration, StGD runs **DPGC** (Algorithm 2) to query the training set  $S_t$  in order to obtain an estimated gradient  $\tilde{g}(w_s)$ , then updates the  $w_{s+1}$  based on  $\tilde{g}(w_s)$  (line 5, 6 in Algorithm 1).

We present the pseudo-code of **DPGC** in Algorithm 2. **DPGC** unrestrictedly accesses the validation set  $S_h$ , but accesses  $S_t$  via a differentially private way: Given  $w_s$ , **DPGC** first computes gradients on  $S_t$  and  $S_h$ :  $g^t(w_s) = \sum_{z_i \in S_t} \nabla l(w_s, z_i) / |S_t|$ ,  $g^h(w_s) =$

---

**Algorithm 2** Differentially Private Gradient Computation (DPGC)

---

- 1: **Input:** Dataset  $S_t$  and  $S_h$ , parameter  $w_s$ , noise variance  $\sigma$ , loss  $l(\cdot)$ .
  - 2: Compute gradients  $g^t(w_s)$  and  $g^h(w_s)$ :  
 $g^t(w_s) = \sum_{z_i \in S_t} \nabla l(w_s, z_i) / |S_t|$ ,  
 $g^h(w_s) = \sum_{z_i \in S_h} \nabla l(w_s, z_i) / |S_h|$ .
  - 3: **for**  $i = 1, \dots, d$  **do**
  - 4:   Sample  $\xi \sim \text{Lap}(\sigma)$ ,  $\gamma \sim \text{Lap}(2 \cdot \sigma)$ ,  
 $\tau \sim \text{Lap}(4 \cdot \sigma)$ .
  - 5:   **if**  $|g_i^t(w_s) - g_i^h(w_s)| > \gamma + \tau$  **then**
  - 6:      $\tilde{g}_i(w_s) = g_i^t(w_s) + \xi$ .
  - 7:   **else**
  - 8:      $\tilde{g}_i(w_s) = g_i^h(w_s)$ .
  - 9:   **end if**
  - 10: **end for**
  - 11: **Return:**  $\tilde{g}(w_s)$ .
- 

$\sum_{z_i \in S_h} \nabla l(w_s, z_i) / |S_h|$ . Second, for each coordinate  $i \in \{1, \dots, d\}$ , **DPGC** validates  $g_i^t(w_s)$  with  $g_i^h(w_s)$  (line 5-line 8 in Algorithm 2): If their absolute difference is beyond the threshold  $\gamma + \tau$ , **DPGC** outputs  $g_i^t(w_s)$  with noise. Otherwise, **DPGC** returns  $g_i^h(w_s)$ .

### 3.3 CONVERGENCE ANALYSIS

We assume that for every  $i$ -th coordinate, the gradient function  $|\nabla_i l(w, z)| \leq G$  for a fixed constant  $G$ . Given an  $n$ -sample set  $S \in \mathcal{Z}^n$  and a fixed  $w_0$  that is chosen independent of the dataset  $S$ ,  $\hat{g}_i(w_0) = \sum_{z_j \in S} \nabla_i l(w_0, z_j) / n$  and  $g_i(w_0) = \mathbb{E}_{z \sim \mathcal{P}}[\nabla_i l(w_0, z)]$ , by Hoeffding's bound, we have the concentration as

$$P\{|\hat{g}_i(w_0) - g_i(w_0)| \geq \sigma\} \leq 2 \exp\left(\frac{-2n\sigma^2}{4G^2}\right). \quad (3)$$

In general, updating  $w_1$  through typical gradient descent:  $w_1 = w_0 - \eta \hat{g}(w_0)$ , the above concentration bound does not hold for  $\hat{g}_i(w_1) = \sum_{z_j \in S} \nabla_i l(w_1, z_j) / n$ , because  $w_1$  is no longer independent of dataset  $S$ . In the next lemma, we demonstrate that  $w_1, w_2, \dots, w_T$  updated by a differential private mechanism have similar concentration bounds as described in Equation 3.

**Lemma 1.** *Let  $\mathcal{M}$  be an  $\epsilon$ -differentially private gradient descent algorithm and  $S_t \sim \mathcal{P}^n$  be the*

*training set. Let  $w_s = \mathcal{M}(S_t)$  be the corresponding output for  $s \in 1, \dots, T$  and  $\hat{g}(w_s)$  be the empirical gradient on  $S_t$ . For any  $\sigma > 0$ ,  $i \in 1, \dots, d$  and  $s \in 1, \dots, T$ , setting  $\epsilon \leq \frac{\sigma}{2G}$  ensures*

$$P\{|\hat{g}_i(w_s) - g_i(w_s)| \geq \sigma\} \leq 6\sqrt{2} \exp\left(\frac{-n\sigma^2}{4G^2}\right). \quad (4)$$

Lemma 1 illustrates that differential privacy enables the reused training set to maintain the statistical guarantee as a fresh set under the condition that the privacy parameter  $\epsilon$  is bounded by the estimation error  $\sigma$ . Next, we analyze the privacy parameter  $\epsilon$  of StGD.

**Lemma 2.** *StGD satisfies  $\frac{2TG}{n\sigma}$ -differentially private.*

In order to achieve the gradient concentration bound described in Lemma 1 by considering the guarantee of Lemma 2 (i.e. to guarantee that for every  $w_s$ , we have  $P\{|\hat{g}_i(w_s) - g_i(w_s)| \geq \sigma\} \leq 6\sqrt{2} \exp(\frac{-n\sigma^2}{4G^2})$ ), we need to set  $\frac{2TG}{n\sigma} \leq \frac{\sigma}{2G}$  so that we achieve  $\epsilon$ -differential privacy for  $\epsilon \leq \frac{\sigma}{2G}$ . As a result, we get the upper bound of iteration time  $T$  in StGD as  $T = \frac{\sigma^2 n}{4G^2}$ . Next theorem shows that across all iterations, gradients produced by StGD maintain high probability concentration bounds.

**Theorem 1.** *Given parameter  $\sigma > 0$ , let  $w_1, w_2, \dots, w_T$  be the adaptively updated points by StGD and  $\tilde{g}(w_1), \dots, \tilde{g}(w_T)$  be the corresponding output gradient. If we set  $T = \frac{\sigma^2 n}{4G^2}$ , then for all  $s \in 1, \dots, T$  and for all  $t > 0$ , we have*

$$P\left\{ \|\tilde{g}(w_s) - g(w_s)\|^2 \geq d(6t + 1)^2 \sigma^2 \right. \\ \left. \leq 2d \exp(-t) + 6\sqrt{2}d \exp\left(\frac{-n\sigma^2}{4G^2}\right) \right\}. \quad (5)$$

Theorem 1 concludes that the gradient  $\tilde{g}(w_s)$  produced by StGD concentrates to the population gradient  $g(w_s)$  and the concentration error is tightly around  $(6t + 1)^2 \sigma^2$ . Increasing noise parameter  $\sigma$  increases the privacy guarantee as well as the total number of iterations, but also increases the concentration error. Decreasing  $\sigma$  has the opposite effect. We consider StGD in two cases: 1)  $F$  is  $L$ -Lipschitz and  $\alpha$ -strongly convex; 2)  $F$  is  $\beta$ -smooth and  $\alpha$ -strongly convex. For these two cases, we present the best value of  $\sigma$  for the trade-off between statistical rate

and optimization rate which depends on number of iterations in order to achieve the optimal risk bound. To simplify the result, we use the notation  $\rho_{n,d} = \ln n + \ln d$ .

**Theorem 2.** For  $L$ -Lipschitz and  $\alpha$ -strongly convex function  $F$ , given  $2n$  available samples, set noise parameter  $\sigma^2 = 4G^2\rho_{n,d}/\sqrt{n}$ , step size  $\eta_s = \frac{2}{\alpha(s+1)}$  and iteration time  $T = \rho_{n,d}\sqrt{n}$  for StGD. Let  $\hat{w}_n = \sum_{s=0}^T w_s/(T+1)$ , StGD achieves:

$$F(\hat{w}_n) - F(w^*) \leq O\left(\frac{\ln(\sqrt{n}\rho_{n,d})}{\sqrt{n}\rho_{n,d}}\right) + O\left(\frac{d\rho_{n,d}^3}{\sqrt{n}}\right), \quad (6)$$

with probability at least  $1 - O\left(\frac{\rho_{n,d}}{\sqrt{n}}\right)$

The first term of the risk bound in Theorem 2 corresponds to typical strongly convex optimization rate  $O(\ln T/T)$  (Bubeck [2015]) ( $T = \rho_{n,d}\sqrt{n}$  in our case) and is similar to the high probability bound of SGD analyzed in Rakhlin et al. [2012]. The second term comes from the statistical error that depends on available sample size  $n$  and dimension  $d$ .

**Theorem 3.** For  $\beta$ -smooth and  $\alpha$ -strongly convex function  $F$ , given  $2n$  available samples, set noise parameter  $\sigma^2 = \frac{\rho_{n,d}(4G^2\alpha+\beta)^2}{n\alpha\beta}$ , step size  $\eta = \frac{1}{\alpha+\beta}$  and iteration time  $T = (\kappa + \frac{1}{\kappa} + 2)\rho_{n,d}$  where  $\kappa = \beta/\alpha$ . Let  $\hat{w}_n = w_T$  be the output of StGD, we have the following excess risk bound:

$$F(\hat{w}_n) - F(w^*) \leq O\left(\frac{\|w_1 - w^*\|^2}{n}\right) + O\left(\frac{d\rho_{n,d}^3}{n}\right) \quad (7)$$

with probability at least  $1 - O\left(\frac{\rho_{n,d}}{n^{\frac{1}{4}}d^{\frac{1}{3}}}\right)$ .

The risk bound in Theorem 3 is also composed of optimization term and statistical term (same for the subsequent theorems). Factor  $\|w_1 - w^*\|^2$  implies a good initial point can be beneficial. In terms of computational complexity, StGD repeats  $O(\ln n)$  iterations on  $n$  samples. It requires a complexity of  $O(n \ln n)$  gradient computations.

## 4 MINI-BATCH EXTENSIONS

In this section, we extend StGD to its mini-batch version for large-scale machine learning tasks. We first introduce a simple *mini-batch*

---

### Algorithm 3 mini-batch SGD

---

- 1: **Input:** Dataset  $S$ , loss  $l(\cdot)$ , initial point  $w_0$
  - 2: **Set:** Step size  $\eta$ , batch size  $m$ .
  - 3: Separate  $S$  into  $T$  parts:  $S_0, \dots, S_{T-1}$  with  $m$  samples each part.
  - 4: **for**  $s = 0, \dots, T-1$  **do**
  - 5:   Compute gradient  $\hat{g}(w_s)$  on  $S_s$
  - 6:    $w_{s+1} = w_s - \eta_s \cdot \hat{g}(w_s)$
  - 7: **end for**
- 

SGD algorithm, then we present the differentially private algorithm *mini-batch StGD*.

The mini-batch SGD algorithm is described in Algorithm 3. The available set is first partitioned into  $T$  batches with  $m$  samples each batch. Then Algorithm 3 updates  $w_s$  based on the gradient computed on each batch. Mini-batch SGD terminates after a single pass over all batches. The following theorem analyzes the risk bound of mini-batch SGD.

**Theorem 4.** Given  $2n$  available samples, *mini-batch SGD* can achieve the following:

1.  $F$  is  $L$ -Lipschitz and  $\alpha$ -strongly convex: If we set the step size  $\eta_s = \frac{2}{\alpha(s+1)}$ , batch size  $m = \sqrt{n}$  and iteration time  $T = 2n/m$ , output  $\hat{w}_n = \sum_{s=1}^T w_s/T$  of mini-batch SGD satisfies:

$$F(\hat{w}_n) - F(w^*) \leq O\left(\frac{\ln(\sqrt{n}+1)}{\sqrt{n}}\right) + O\left(\frac{d \ln \sqrt{n}}{\sqrt{n}}\right) \quad (8)$$

with probability at least  $1 - d/\sqrt{n}$ .

2.  $F$  is  $\beta$  smooth and  $\alpha$ -strongly convex: If we set the step size  $\eta = \frac{1}{\alpha+\beta}$ ,  $m = \frac{\alpha\beta n}{(\alpha+\beta)^2 \ln n}$ ,  $T = 2n/m$ , output  $\hat{w}_n = w_T$  of StGD satisfies:

$$F(\hat{w}_n) - F(w^*) \leq O\left(\frac{\|w_1 - w^*\|^2}{n}\right) + O\left(\frac{d \ln^2 n}{n}\right) \quad (9)$$

with probability  $1 - O\left(\frac{\ln n}{n}\right)$ .

(Frostig et al. [2015]) proposed a variant of mini-batch SGD that also does a single pass of the available data. For smooth and strongly convex function, they established a convergence rate of  $O(1/n)$  in expectation that is similar to the rate in Theorem 4.

---

**Algorithm 4** mini-batch StGD

---

- 1: **Input:** Dataset  $S$ , loss  $l(\cdot)$ , initial point  $w_0$
  - 2: **Set:** Step size  $\eta$ , batch size  $m$ , inner iterations  $T_1$ , noise  $\sigma$
  - 3: Separate  $S$  into  $T$  parts:  $S_0, \dots, S_{T-1}$  with  $m$  samples each part.
  - 4: **for**  $s = 0, \dots, T - 1$  **do**
  - 5:    $w_{s+1} = \text{StGD}(w_s, S_s, \eta, T_1, \sigma)$
  - 6: **end for**
- 

The mini-batch version of StGD is given in Algorithm 4 (mini-batch StGD) that is also a private version of mini-batch SGD: For each batch  $S_s$ , where  $s \in \{0, \dots, T - 1\}$ , call StGD to query  $S_s$  and update  $w_{s+1}$  as the initial point for next batch  $S_{s+1}$ . In each call, there are  $T_1$  inner iterations that StGD queries  $S_s$  for  $T_1$  times through DPGD. Let  $\tilde{w}_0 = w_s$  as the initial point in each call, then sub-algorithm StGD updates  $\tilde{w}_{k+1} = \tilde{w}_k + \eta \tilde{g}(\tilde{w}_k)$  for  $k = \{0, \dots, T_1 - 1\}$  and  $w_{s+1} = \tilde{w}_{T_1}$ .

**Theorem 5.** *Given  $2n$  available samples, mini-batch StGD can achieve the following:*

1.  **$F$  is  $L$ -Lipschitz and  $\alpha$ -strongly convex:** *If we set the step size  $\eta_s = \frac{2}{\alpha(s+1)}$ , batch size  $m = \sqrt{n}$ ,  $T = 2n/m$ , noise parameter  $\sigma^2 = 8G^2 \ln n / \sqrt{n}$  and  $T_1 = \ln n$ , output  $\hat{w}_n = \sum_{s=1}^T w_s / T$  of mini-batch StGD satisfies:*

$$F(\hat{w}_n) - F(w^*) \leq O\left(\frac{\ln(\sqrt{n}+1)}{\sqrt{n} \ln n}\right) + O\left(\frac{\ln^3 n}{\sqrt{n}}\right) \quad (10)$$

with probability at least  $1 - d/\sqrt{n}$ .

2.  **$F$  is  $\beta$  smooth and  $\alpha$ -strongly convex:** *If we set the step size  $\eta = \frac{1}{\alpha+\beta}$ ,  $m = \frac{\alpha\beta n}{(\alpha+\beta)^2 \ln n}$ ,  $T = 2n/m$ ,  $T_1 = \ln n$ , noise parameter  $\sigma^2 = \frac{4G^2(\alpha+\beta)^2(\ln n)^2}{\alpha\beta n}$ , output  $\hat{w}_n = w_T$  of mini-batch StGD satisfies:*

$$F(\hat{w}_n) - F(w^*) \leq O\left(\frac{\|w_1 - w^*\|^2}{n \ln n}\right) + O\left(\frac{d \ln^4 n}{n}\right) \quad (11)$$

with probability at least  $1 - O\left(\frac{\ln^2 n}{n}\right)$ .

Theorem 5 shows that, compared to the basic mini-batch SGD (Theorem 4), the private ver-

sion improves the rate of the first term for both types of functions.

## 5 EXPERIMENTS

In this section, we conduct experiments to evaluate performances of the proposed algorithms on artificial data and real world data. We divide our experiments into three sets to address questions: (i) How do the StGD and the mini-batch StGD perform regarding the convergence to the population optimum? (ii) For small datasets, how does StGD perform compared to SGD and GD. (iii) For large datasets, does mini-batch StGD outperform SGD and mini-batch SGD? After discussing the experimental setup, we evaluate these questions empirically in Sections 5.2, 5.3, and 5.4 respectively.

### 5.1 EXPERIMENTAL SETTING

**Datasets:** We use both artificial datasets and real-world datasets for our experiments. We discuss the datasets in two categories: the small datasets (i.e., *small artificial dataset, breast cancer, diabetes and german.numer*) for StGD, SGD and GD, and large datasets (i.e., *large artificial dataset, cove type, rcv1 and real-sim*) for mini-batch StGD, SGD and mini-batch SGD. All the real-world datasets are from LIBSVM (Chang and Lin [2011]). The real-world datasets are described in the Table 1. The small artificial dataset, consists of 50 features and one label:  $z_i = (\mathbf{x}_i, y_i) \in \mathbb{R}^{50} \times \{1, -1\}$ . The large artificial dataset consists of 500 features and one label:  $z_i = (\mathbf{x}_i, y_i) \in \mathbb{R}^{500} \times \{1, -1\}$ . The value of each feature is random noise, drawn i.i.d. from normal distribution  $N(0, 1)$ . To generate the label, we first set an optimal minimizer  $w^* \in \mathbb{R}^{50}$  for small datasets and  $w^* \in \mathbb{R}^{500}$  for large datasets. Then, we draw the label  $y_i$  corresponding to  $\mathbf{x}_i$  from the Bernoulli distribution  $y_i \sim B\left(\frac{1}{1+\exp(-w^{*T} \mathbf{x}_i)}, \frac{\exp(-w^{*T} \mathbf{x}_i)}{1+\exp(-w^{*T} \mathbf{x}_i)}\right)$ .

**Evaluation Metrics:** We measure the performance of these algorithms for binary classification problem with linear models. We focus on the smooth and strongly convex loss function case and define the loss function  $F$  to be the

Table 1: Datasets

Datasets	Data size	Features
breast cancer	683	10
diabetes	768	8
german.numer	1000	24
cove type	581012	54
rcv1	697641	47236
real-sim	72309	20958

logistic loss. Thus, the population risk is

$$F(w) = \mathbb{E}[\ln(1 + \exp(-y_i w^T \mathbf{x}_i))]. \quad (12)$$

Given a training set of instance-label pairs  $\{\mathbf{x}_i, y_i\}_{i=1}^n$  the empirical risk is

$$\hat{F}(w) = \frac{1}{n} \sum_{i=1}^n \ln(1 + \exp(-y_i w^T \mathbf{x}_i)). \quad (13)$$

The population optimum is

$$F(w^*) = \mathbb{E}[\ln(1 + \exp(-y_i w^{*T} \mathbf{x}_i))]. \quad (14)$$

We use  $\hat{w}_n$  to be the output trained on  $n$  samples. We evaluate these algorithms in terms of the excess risk  $F(\hat{w}_n) - F(w^*)$  and test error rate for artificial datasets. Since we cannot know the population optimum  $F(w^*)$  out of the expectation, we let the loss computed on a large fresh set (2000 fresh samples for small data case, 20000 fresh samples for large data case) to represent the population risk. As for the real-world datasets, since the population minimizer is unknown, we evaluate these algorithms based on the test loss and test error rate.

**Setup and parameters:** We set  $w_0 = \{1\}^d$  as the initial point for artificial datasets and  $w_0 = \{0\}^d$  for real-world datasets. As for the step size, we use the typical  $\eta = a_1/\sqrt{n}$  for SGD,  $\eta = a_2$  for StGD, mini-batch SGD and mini-batch StGD (which is the default setting in our theoretical analysis) and  $\eta = a_3$  for GD. The above  $a_1$ ,  $a_2$  and  $a_3$  are all constants and we use grid search to find the best values of them for different datasets. As for the iteration times, given the training set with size  $n$  and  $d$  features, we set 500 iterations for GD and  $10 * (\ln n + \ln d)$  for StGD. SGD stops iteration after a single pass over all training samples. Mini-batch StGD has batch size  $n/\ln n$  and  $\ln^2 n$  iterations. Mini-batch SGD has the same batch size, but  $\ln n$

iterations. Finally, we set the noise parameter  $\sigma = (\ln n + \ln d)/n$  for StGD and  $\sigma = \ln^2 n/n$  for mini-batch StGD.

## 5.2 EVALUATIONS OF STGD

In the first set of experiments, we validate the theoretical promise of StGD and mini-batch SGD on artificial datasets. To show the convergence in terms of the sample size  $n$ , we sample a series of artificial datasets with size  $n \in \{50, 100, 150, \dots, 2000\}$  and run these algorithms on those datasets. To show how feature size  $d$  influences the convergence of StGD, we generate samples with feature size  $d = 100$  and  $d = 150$  and report corresponding risks. We repeat the experiment 50 times and report the mean and standard deviation of the results.

We report the population risks (test loss on the large fresh set)  $F(\hat{w}_n)$ , empirical risks (train loss)  $\hat{F}(w_n)$ , population optimum (estimated by the large fresh set) and excess risks  $F(\hat{w}_n) - F(w^*)$ . Fig. 1 (a) and Fig.1 (c) illustrate the convergence rate of StGD and mini-batch SGD respectively. As  $n$  increases, the population risks of StGD and mini-batch StGD converge to population optimum. Fig. 1 (b) and (d) show how the feature size  $d$  influences the convergence rate. Larger  $d$  implies a slower convergence rate.

## 5.3 COMPARISON of STGD, SGD AND GD

In the second set of experiments, we compare StGD, SGD and GD on small datasets in terms of excess risk/test loss and test error rate. For the real-world datasets, we first sample 20% data points from the whole datasets to be the test set, and let the remaining samples to be the train set. Afterwards, we sample a series of datasets with size  $n \in \{10, 20, \dots, 250\}$  (for diabetes and breast cancer) and  $n \in \{20, 40, \dots, 400\}$  (for german.numer) from the remaining train set and run these algorithms on those datasets. The artificial data split is the same as the first set experiment. Given each  $n$ , we train the model and report the loss and error rate on the test set. We repeat the above procedure 10 times and report the mean and standard deviation of



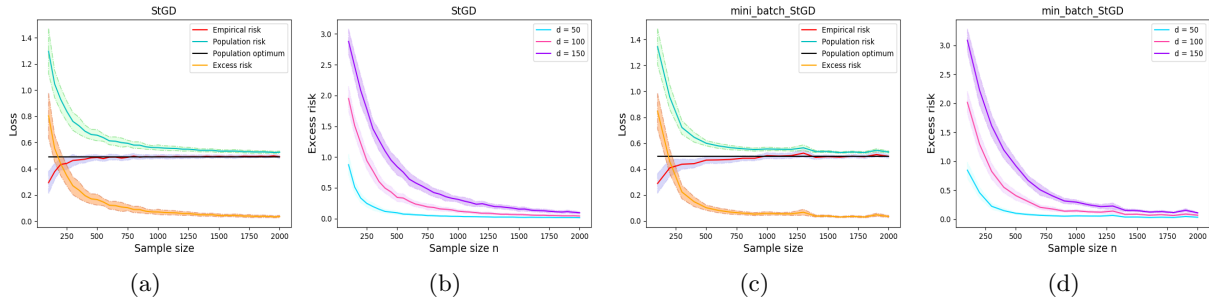


Figure 1: The StGD and mini-batch StGD on artificial data. (a) The risks of StGD. (b) The excess risks of StGD with different data dimension  $d$ . (c) The risks of mini-batch StGD. (d) The excess risks of mini-batch StGD with different data dimension  $d$ . The X-axis is the number of samples, and the Y-axis is the Risk/Loss.

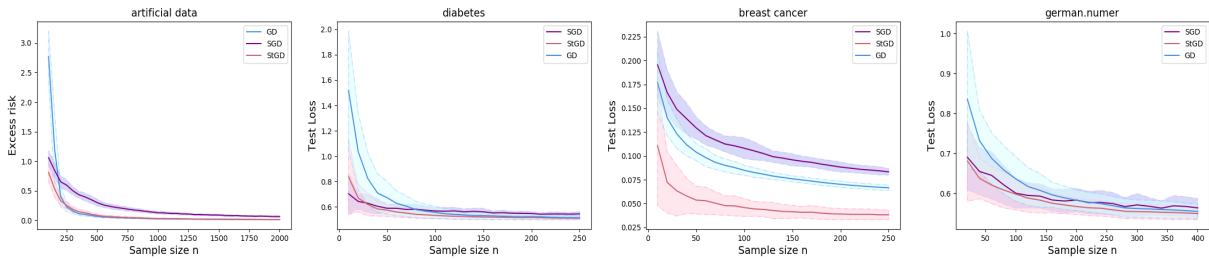


Figure 2: Compare the StGD, SGD and GD on both artificial datasets and small real-world datasets. The X-axis and the Y-axis refer to Fig. 1. The excess risk of StGD converges as fast as GD on artificial dataset. The StGD outperforms GD and SGD in terms of the test loss on real-world datasets.

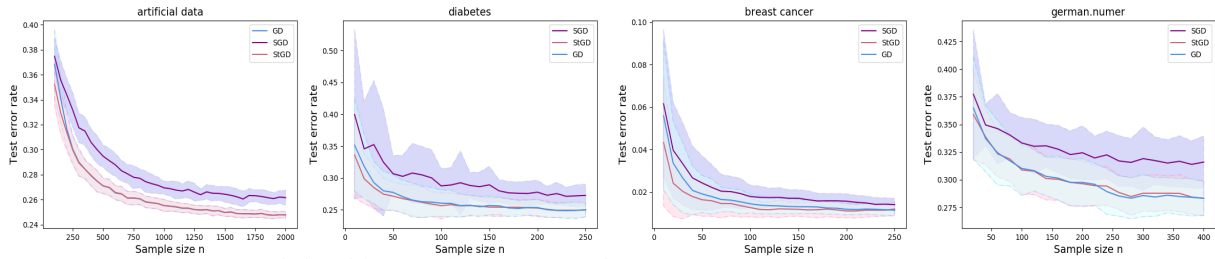


Figure 3: Compare the StGD, SGD and GD on both artificial dataset and small real-world datasets. The X-axis and the Y-axis refer to Fig. 1. The StGD outperforms GD and SGD in terms of the test error rate on artificial dataset and real-world datasets.

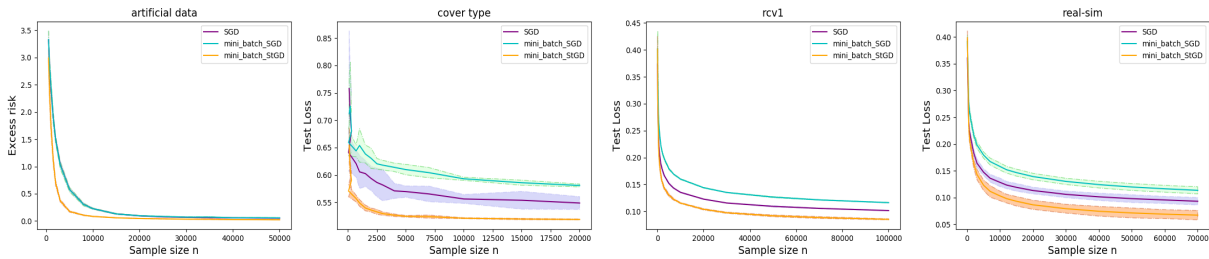


Figure 4: Compare the mini-batch StGD, SGD and mini-batch SGD on both artificial dataset and large real-world datasets. The X-axis and the Y-axis refer to Fig. 1. The excess risk of mini-batch StGD converges as fast as GD on artificial dataset. The mini-batch StGD outperforms SGD and mini-batch SGD in terms of the test loss on real-world datasets.

the results.

Fig. 2 presents the excess risks and test losses on four small datasets of the three algorithms and Fig. 3 compares the test error rates. For artificial datasets, StGD performs nearly the

same as GD in terms of the excess risks and test error rates. For diabetes, breast cancer and german.numer, StGD converges better than GD. In terms of the variance, these three algorithms perform more variance on real-world data

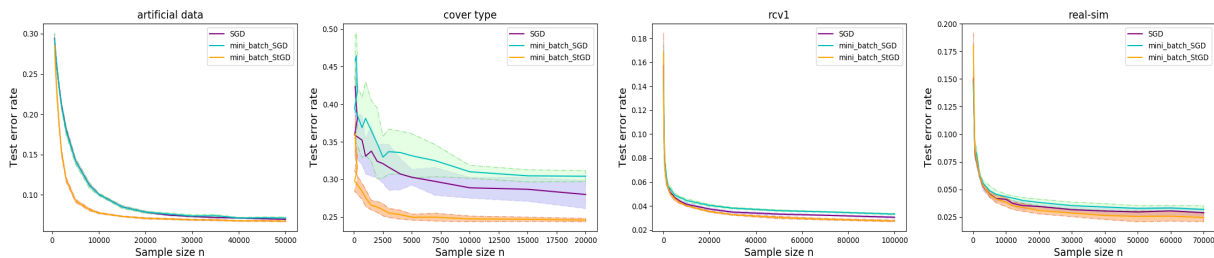


Figure 5: Compare the mini-batch StGD, SGD and mini-batch SGD on both artificial dataset and large real-world datasets. The X-axis and the Y-axis refer to Fig. 1. The test error rate of mini-batch StGD converges as fast as GD on artificial dataset. The mini-batch StGD outperforms SGD and mini-batch SGD in terms of the test error rate on real-world datasets.

than artificial data. The variance of the losses and error rates from repeated runs come from the training data and the algorithm themselves. The noise amount in StGD and the size of the training sample play an important role in variances. Repeating training the models with small dataset brings out large variance. Fig. 2 and Fig. 3 show that the variances decrease as the sample size increases.

#### 5.4 COMPARISON of MINI-BATCH SGD, MINI-BATCH STGD AND SGD

In the third set of experiments, we compare mini-batch StGD, mini-batch SGD and SGD on large datasets. For the real-world datasets, the train and test data split is the same as the second set experiment. From the train set, we sample a series of datasets with size  $n \in \{100, 300, \dots, 50000\}$  for artificial data,  $n \in \{100, 200, \dots, 20000\}$  for cover type,  $n \in \{100, 500, \dots, 100000\}$  for rcv1 and  $n \in \{100, 500, \dots, 70000\}$  for real-sim. Given each  $n$ , we train the model and report the loss and error rate on the test set. We repeat the above procedure 10 times and report the mean and standard deviation of the results.

The excess risks on artificial dataset and the test losses on real-world datasets are shown in Fig. 4 and the test error rates is given in Fig. 5. The results show mini-batch StGD achieves the lowest test loss and test error rate for four datasets and lowest variance for cover type (three algorithms perform low variance in the other three datasets). Compared to training with small datasets (Fig.2 and Fig. 3), we observe less variance with large

datasets.

## 6 CONCLUSION

In this paper, we study the optimization problems in machine learning. Considering the difficulty of obtaining new samples for gradient descent to approximate population gradient, we propose a stable gradient descent algorithm based on adaptive data analysis and differential privacy. We demonstrate StGD works as a basic gradient descent which has access to fresh sample at each iteration. Furthermore, we theoretically analyze that the proposed algorithm converges fast to the population optimum with high probability. Finally, we compare the proposed algorithm with existing methods in experiments. The empirical evaluation illustrate the promise of the proposed algorithm and demonstrated it outperforms existing methods.

**Acknowledgements:** We thank the reviewers for their valuable comments. The research was supported by NSF grants IIS-1563950, IIS-1447566, IIS-1447574, IIS-1422557, CCF-1451986, CNS-1314560, IIS-0953274, IIS-1029711, and NASA grant NNX12AQ39A.

## References

Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *Foundations of Computer Science (FOCS)*, pages 464–473. IEEE, 2014.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

- Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 8(3-4):231–357, 2015.
- Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):27, 2011.
- Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(Mar):1069–1109, 2011.
- Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3–4):211–407, 2014.
- Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toni Pitassi, Omer Reingold, and Aaron Roth. Generalization in adaptive data analysis and holdout reuse. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2350–2358, 2015a.
- Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. The reusable holdout: Preserving validity in adaptive data analysis. *Science*, 349(6248):636–638, 2015b.
- Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Leon Roth. Preserving statistical validity in adaptive data analysis. In *Proceedings of the 47th annual ACM symposium on Theory of computing (STOC)*, pages 117–126. ACM, 2015c.
- Roy Frostig, Rong Ge, Sham M Kakade, and Aaron Sidford. Competing with the empirical risk minimizer in a single pass. In *Conference on learning theory (COLT)*, pages 728–763, 2015.
- Moritz Hardt, Benjamin Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. *arXiv preprint arXiv:1509.01240*, 2015.
- Elad Hazan and Satyen Kale. Beyond the regret minimization barrier: optimal algorithms for stochastic strongly-convex optimization. *Journal of Machine Learning Research*, 15(1):2489–2512, 2014.
- Alexander Rakhlin, Ohad Shamir, and Karthik Sridharan. Making gradient descent optimal for strongly convex stochastic optimization. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, pages 449–456, 2012.
- Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.
- Tong Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the 21st international conference on Machine learning (ICML)*, page 116. ACM, 2004.

---

# Learning to select computations

---

Frederick Callaway<sup>1,a</sup>, Sayan Gul<sup>1,a</sup>, Paul Krueger<sup>a</sup>, Thomas L. Griffiths<sup>a</sup>, & Falk Lieder<sup>1,a,b</sup>

<sup>a</sup> University of California, Berkeley, USA

<sup>b</sup> Max Planck Institute for Intelligent Systems, Tübingen, Germany

<sup>1</sup> These authors contributed equally.

## Abstract

The efficient use of limited computational resources is an essential ingredient of intelligence. Selecting computations optimally according to rational metareasoning would achieve this, but this is computationally intractable. Inspired by psychology and neuroscience, we propose the first concrete and domain-general learning algorithm for approximating the optimal selection of computations: Bayesian metalevel policy search (BMPS). We derive this general, sample-efficient search algorithm for a computation-selecting metalevel policy based on the insight that the value of information lies between the myopic value of information and the value of perfect information. We evaluate BMPS on three increasingly difficult metareasoning problems: when to terminate computation, how to allocate computation between competing options, and planning. Across all three domains, BMPS achieved near-optimal performance and compared favorably to previously proposed metareasoning heuristics. Finally, we demonstrate the practical utility of BMPS in an emergency management scenario, even accounting for the overhead of metareasoning.

## 1 INTRODUCTION

The human brain is the best example of an intelligent system we have so far. One feature that sets it apart from current AI is the remarkable computational efficiency that enables people to effortlessly solve hard problems for which artificial intelligence either under-performs humans or requires superhuman computing power and training time. For instance, to defeat Garry Kasparov

3.5–2.5, Deep Blue had to evaluate 200 000 000 positions per second, whereas Kasparov was able to perform at almost the same level by evaluating only 3 positions per second (Campbell, Hoane, & Hsu, 2002; IBM Research, 1997). This ability to make efficient use of limited computational resources is the essence of intelligence (Russell & Wefald, 1991a). People accomplish this feat by being very selective about when to think and what to think about, choosing computations adaptively and terminating deliberation when its expected benefit falls below its cost (Gershman, Horvitz, & Tenenbaum, 2015; Lieder & Griffiths, 2017; Payne, Bettman, & Johnson, 1988).

Rational metareasoning was introduced to recreate such intelligent control over computation in machines (Horvitz, Cooper, & Heckerman, 1989; Russell & Wefald, 1991b; Hay, Russell, Tolpin, & Shimony, 2012). In principle, rational metareasoning can be used to always select those computations that make optimal use of the agent’s finite computational resources. However, its computational complexity is prohibitive (Hay et al., 2012). The human mind circumvents this computational challenge by learning to select computations through metacognitive reinforcement learning (Krueger, Lieder, & Griffiths, 2017; Lieder & Griffiths, 2017; Wang et al., 2017). Concretely, people appear to learn to predict the value of alternative cognitive operations from features of the task, their current belief state, and the cognitive operations themselves. If humans learn to metareason through metacognitive reinforcement learning, then it should be possible to build intelligent systems that learn to metareason as efficiently as people.

In this paper, we introduce Bayesian metalevel policy search (BMPS), the first domain-general algorithm for learning how to metareason, and evaluate it against existing methods for approximate metareasoning on three increasingly more complex toy problems. Finally, we show that our method makes metareasoning efficient enough to offset its cost in a more realistic emergency management

scenario. In this problem, which we use as a running example, an emergency manager must decide which cities to evacuate in the face of an approaching tornado. She bases her decision on a series of computationally intensive simulations that noisily estimate the impact of the tornado on each city. Because time is short, she is forced to decide which simulations are the most important to run. In the following section, we discuss how to formalize this problem as a sequential decision process.

## 2 BACKGROUND

### 2.1 METAREASONING

If reasoning seeks an answer to the question “what should I do?”, metareasoning seeks to answer the question “how should I decide what to do?”. The theory of rational metareasoning (Russell & Wefald, 1991b; Russell & Subramanian, 1995) frames this problem as selecting computations so as to maximize the sum of the rewards of resulting decisions minus the costs of the computations involved. Concretely, one can formalize reasoning as a metalevel Markov decision process (metalevel MDP) and metareasoning as solving that MDP (Hay et al., 2012). While traditional (object-level) MDPs describe the objects of reasoning—the state of the external environment and how it is affected by physical actions—a metalevel MDP describes reasoning itself. Formally, a metalevel MDP  $M_{\text{meta}} = (\mathcal{B}, \mathcal{A}, T_{\text{meta}}, r_{\text{meta}})$  is an MDP where the states  $\mathcal{B}$  encode the agent’s beliefs, the actions  $\mathcal{A}$  are computations, the transition function  $T_{\text{meta}}$  describes how computations update beliefs, and the reward function  $r_{\text{meta}}$  describes the costs and benefits of computation. A definition table for our notation is included in the Supplementary Material.

A belief state  $b \in \mathcal{B}$  encodes a probability distribution over parameters  $\theta$  of a model of the domain. For example, in the tornado problem described in the introduction,  $\theta$  could be a vector of  $k$  probabilities that each of the  $k$  cities will incur evacuation-warranting damage;  $b$  would thus encode  $k$  distributions over  $[0, 1]$ , e.g.  $k$  Beta distributions. The parameters  $\theta$  determine the utility of acting according to a policy  $\pi$ , that is  $U_{\pi}(\theta)$ . For one-shot decisions,  $U_{\pi}(\theta)$  is the expected reward of taking the single action identified with  $\pi$ . In the tornado problem, for example,  $\pi$  can be represented as a binary vector of length  $k$  indicating whether each city should be evacuated, and  $U_{\pi}(\theta)$  is the cost of making the evacuations plus the expected cost of failing to evacuate cities that incur major damage. In sequential decision-problems,  $U_{\pi}(\theta) = V_{\pi}^{(\theta)}(s)$  is the expected sum of rewards the agent will obtain by acting according to policy  $\pi$  if the environment has the characteristics encoded by  $\theta$ .

$\mathcal{A}$  includes computations  $\mathcal{C}$  that update the belief, as well as a special metalevel action  $\perp$  that terminates deliberation and initiates acting on the current belief. The effects of computations are encoded by  $T_{\text{meta}} : \mathcal{B} \times \mathcal{A} \times \mathcal{B} \rightarrow [0, 1]$  analogously to a standard transition function. The termination action always leads to a unique end state.

The metalevel reward function  $r_{\text{meta}}$  captures the cost of thinking (Shugan, 1980) and the external reward the agent expects to receive from the environment. The computations  $\mathcal{C}$  have no external effects and thus always incur a negative reward  $r_{\text{meta}}(b, c) = -\text{cost}(c)$ . In the problems studied below, all computations that deliberate have the same cost, that is  $\text{cost}(c) = \lambda$  for all  $c \in \mathcal{C}$  whereas  $\text{cost}(\perp) = 0$ . An external reward is received only when the agent terminates deliberation and makes a decision, which is assumed to be optimal given the current belief. The metalevel reward for terminating is thus  $r_{\text{meta}}(b, \perp) = \max_{\pi} \mathbb{E}_{\theta \sim b}[U_{\pi}(\theta)]$ .<sup>1</sup>

Early work on rational metareasoning (Russell & Wefald, 1991b) defined the optimal way to select computations as maximizing the value of computation (VOC):

$$\pi_{\text{meta}}^* = \arg \max_c \text{VOC}(c, b), \quad (1)$$

where  $\text{VOC}(c, b)$  is the expected improvement in decision quality that can be achieved by performing computation  $c$  in belief state  $b$  and continuing optimally, minus the cost of the optimal sequence of computations (Russell & Wefald, 1991b). When no computation has positive value, the policy terminates computation and executes the best object-level action, thus  $\text{VOC}(\perp, b) = 0$ .

### 2.2 APPROXIMATE METAREASONING

Previous work (Russell & Wefald, 1991b; Lin, Kolobov, Kamar, & Horvitz, 2015) has approximated rational metareasoning by the meta-greedy policy  $\arg \max_c \text{VOC}_1(c, b)$  where  $\text{VOC}_1(c, b) = \mathbb{E}_{B' \sim T_{\text{meta}}(b, c, \cdot)} [r_{\text{meta}}(B', \perp)] - r_{\text{meta}}(b, \perp) + r_{\text{meta}}(b, c)$ , is the myopic value of computation (Russell & Wefald, 1991b). The meta-greedy policy selects each computation assuming that it will be the last computation. This policy is optimal when computation provides diminishing returns (i.e. the improvement from each additional computation is less than that from the previous one), but it deliberates too little when this assumption is violated. For example, in the tornado problem (where false negatives have high cost), a single simulation may be unable to ensure that evacuation is unnecessary with sufficient confidence, while two or more could.

<sup>1</sup>If the agent’s model is unbiased, this reward has the same expectation but lower variance than the true external reward.

Hay et al. (2012) approximated rational metareasoning by combining the solutions to smaller metalevel MDPs that formalize the problem of deciding how to decide between one object-level action and the expected return of its best alternative. Each of these smaller metalevel MDPs includes only the computations for reasoning about the expected return of the corresponding object-level action. While this *blinkered* approximation is more accurate than the meta-greedy policy, it is also significantly less scalable and not directly applicable to metareasoning about planning.

These are the main approximations to rational metareasoning. So, to date, there appears to be no accurate and scalable method for solving general metalevel MDPs.

### 2.3 METACOGNITIVE RL

It has been proposed that metareasoning can be made tractable by learning an approximation to the value of computation (Russell & Wefald, 1991b). However, despite some preliminary steps in this direction (Harada & Russell, 1998; Lieder et al., 2014; Lieder, Krueger, & Griffiths, 2017) and related work on meta-learning (Smith-Miles, 2009; Thornton, Hutter, Hoos, & Leyton-Brown, 2013; Wang et al., 2017), learning to approximate bounded optimal information processing remains an unsolved problem in artificial intelligence.

Previous research in cognitive science suggests that people circumvent the intractability of metareasoning by learning a metalevel policy from experience (Lieder & Griffiths, 2017; Cushman & Morris, 2015; Krueger et al., 2017). At least in some cases, the underlying mechanism appears to be model-free reinforcement learning (RL) (Cushman & Morris, 2015; Krueger et al., 2017). This suggests that model-free reinforcement learning might be a promising approach to solving metalevel MDPs. To our knowledge, this approach is yet to be explored in artificial intelligence. Here, we present a proof-of-concept that near-optimal metalevel policies can be learned through metacognitive reinforcement learning.

## 3 BAYESIAN METALEVEL POLICY SEARCH

According to rational metareasoning, an optimal metalevel policy is one that maximizes the VOC (Equation 1). Although the VOC is intractable to compute, it can be bounded. Bayesian metalevel policy search (BMPS) capitalizes on these bounds to dramatically reduce the difficulty of learning near-optimal metalevel policies. Figure 1 illustrates that if the expected decision quality improves monotonically with the number of computa-

tions, then the improvement achieved by the optimal sequence of computations should lie between the benefit of deciding immediately after the first computation and the benefit of obtaining perfect information (Howard, 1966). The former is given by the myopic value of information,<sup>2</sup>

$$\text{VOI}_1(c, b) = \mathbb{E}_{B' \sim T_{\text{meta}}(b, c, \cdot)} [U(B')] - U(b). \quad (2)$$

and the latter is given by the value of perfect information,

$$\text{VPI}(b) = \mathbb{E}_{\theta^* \sim b} [U(B^*(\cdot; \theta^*))] - U(b), \quad (3)$$

where  $U(b) = r_{\text{meta}}(b, \perp)$  is shorthand for the expected value of terminating computation and  $B^*(\theta; \theta^*) = \delta(\theta_i - \theta_i^*)$  is the belief state with perfect knowledge of the true environment parameters  $\theta^*$ .

In problems with many parameters, this upper bound can be very loose because the optimal metalevel policy might reason only about a small subset of relevant parameters. To capture this, we introduce an additional feature  $\text{VPI}_{\text{sub}}(c, b)$  that measures how beneficial it would be to have full information about a subset of the parameters that are most relevant to the given computation. We model relevance with a function  $f(c, i)$  that returns 1 if  $\theta_i$  is relevant to what  $c$  is reasoning about and 0 otherwise. Using this relevance function, we define the value of gaining perfect information about the relevant subset of parameters as

$$\text{VPI}_{\text{sub}}(c, b) = \mathbb{E}_{\theta^* \sim b} [U(B'_{\text{sub}}(\cdot; c, b, \theta^*))] - U(b), \quad (4)$$

<sup>2</sup>The  $\text{VOI}_1$  defined here is equal to the myopic VOC defined by Russell and Wefald (1991) plus the cost of the computation.

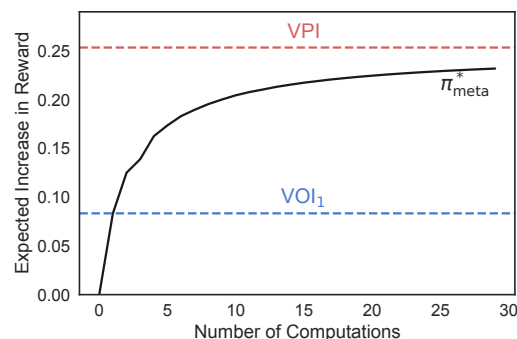


Figure 1: Expected performance in metareasoning about how to choose between three actions increases monotonically with the number of computations, asymptoting at the value of perfect information (VPI). Consequently, the value of executing a single computation must lie between the myopic value of information ( $\text{VOI}_1$ ) and the VPI.

with

$$B'_{\text{sub}}(\theta; c, b, \theta^*) = \prod_i^k B^*(\theta_i; \theta^*)^{f(c,i)} \cdot b(\theta_i)^{1-f(c,i)},$$

where  $k$  is the number of parameters in the agent’s model of the environment. In the tornado problem, for example, each simulation is informative about a single parameter (the probability that the target city will sustain evacuation-warranting damage); thus, we define  $f(c_j, i) = \mathbb{1}(j = i)$ . In the general case, the relevance function is a design choice that affords an easy opportunity to imbue BMPS with domain knowledge. In the simulations reported below, the relevance function associates each  $c$  with the set of parameters that inform the value of the actions (or, in the case of planning, options) that  $c$  reasons about.

Critically, all three VOI features can be computed efficiently or can be efficiently approximated by Monte-Carlo integration (Hammersley, 2013). BMPS thus approximates the VOC by a mixture of VOI features and an estimate of the cost of future computations

$$\begin{aligned} \text{V}\hat{\text{O}}\text{C}(c, b; \mathbf{w}) = & w_1 \cdot \text{VOI}_1(c, b) + w_2 \cdot \text{VPI}(b) \\ & + w_3 \cdot \text{VPI}_{\text{sub}}(c, b) - w_4 \cdot \text{cost}(c), \end{aligned} \quad (5)$$

with the constraints that  $w_1, w_2, w_3 \in [0, 1]$ ,  $w_1 + w_2 + w_3 = 1$ , and  $w_4 \in [1, h]$  where  $h$  is an upper bound on how many computations can be performed. Since the VOC defines the optimal metalevel policy (Equation 1), we can define an approximately optimal policy,  $\pi_{\text{meta}}(b; \mathbf{w}) = \arg \max_c \text{V}\hat{\text{O}}\text{C}(c, b; \mathbf{w})$ .

The parameters  $\mathbf{w}$  of this policy are optimized by maximizing the expected return  $\mathbb{E}[\sum_t r_{\text{meta}}(b_t, \pi_{\text{meta}}(b_t; \mathbf{w}))]$ , i.e. direct policy search. Because there are only three free parameters with the summation constraint, we propose using Bayesian optimization (BO) (Mockus, 2012) to optimize the weights in a sample efficient manner.

The novelty of BMPS lies in leveraging machine learning to approximate the solution to metalevel MDPs and in the discovery of features that make this tractable. As far as we know, BMPS is the first general approach to metacognitive RL. In the following sections, we validate the assumptions of BMPS, evaluate its performance on increasingly complex metareasoning problems, compare it to existing methods, and discuss potential applications.

## 4 EVALUATIONS OF BMPS

We evaluate how accurately BMPS can approximate rational metareasoning against two state-of-the-

art approximations—the meta-greedy policy and the blinkered approximation—on three increasingly difficult metareasoning problems.

### 4.1 WHEN TO STOP DELIBERATING?

How long should an agent deliberate before answering a question? Our evaluation mimics this problem for a binary prediction task (e.g., “Will the price of the stock go up or down?”). Every deliberation incurs a cost and provides probabilistic evidence  $X_t \sim \text{Bernoulli}(\theta)$  in favor of one outcome or the other. At any point the agent can stop deliberating and predict the outcome supported by previous deliberations. The agent receives a reward of +1 if its prediction is correct, or incurs a loss of −1 if it is incorrect. The goal is to maximize the expected reward of this one prediction minus the cost of computation.

#### 4.1.1 Metalevel MDP

We formalize the problem of deciding when to stop thinking as a metalevel MDP  $M_{\text{meta}} = (\mathcal{B}, \mathcal{A}, T_{\text{meta}}, r_{\text{meta}})$  where each belief state  $(\alpha, \beta) \in \mathcal{B}$  defines a beta distribution over the probability  $\theta$  of the first outcome. The metalevel actions  $\mathcal{A}$  are  $\{c_1, \perp\}$  where  $c_1$  refines the belief by sampling, and  $\perp$  terminates deliberation and predicts the outcome that is most likely according to the current belief. The transition probabilities for sampling are defined by the agent’s belief state, that is  $T_{\text{meta}}((\alpha, \beta), c_1, (\alpha + 1, \beta)) = \frac{\alpha}{\alpha + \beta}$  and  $T_{\text{meta}}((\alpha, \beta), c_1, (\alpha, \beta + 1)) = \frac{\beta}{\alpha + \beta}$ . The reward function  $r_{\text{meta}}$  reflects the cost of computation,  $r_{\text{meta}}(b, c_1) = -\lambda$ , and the probability of making the correct prediction,  $r_{\text{meta}}(b, \perp) = +1 \cdot p_{\text{correct}}(\alpha, \beta) - 1 \cdot (1 - p_{\text{correct}}(\alpha, \beta))$ , where  $p_{\text{correct}}(\alpha, \beta) = \max\{\frac{\alpha}{\alpha + \beta}, \frac{\beta}{\alpha + \beta}\}$ . We set the horizon to  $h = 30$ , meaning that the agent can perform at most 29 computations before making a prediction (the 30th metalevel action must be  $\perp$ ).

Since there is only one parameter ( $\theta$  has length one), the  $\text{VPI}_{\text{sub}}$  feature is identical with the VPI feature; thus, we exclude it. For the same reason, the blinkered approximation is equivalent to solving the problem exactly, and we exclude it from the comparison.

#### 4.1.2 Evaluation procedure

We evaluated the potential of BMPS in two steps: First, we performed a regression analysis to evaluate whether the proposed features are sufficient to capture the value of computation, computed exactly by backward induction (Puterman, 2014). Second, we tested whether a near-optimal metalevel policy can be learned by Bayesian optimization of the weights of the metalevel policy. We ran 500 iterations of optimization, estimating

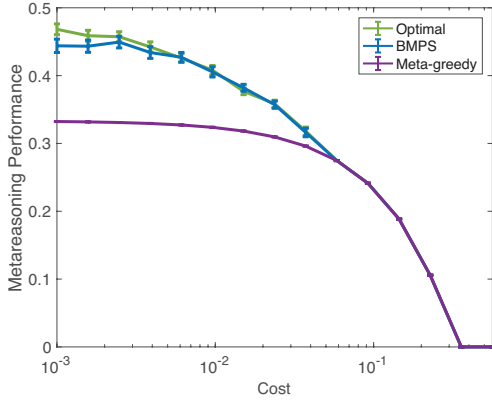


Figure 2: Results of performance evaluation on the problem of metareasoning about when to stop deliberating.

the expected return of the policy entailed by the probed weight vector by its average return across 2500 episodes. The performance of the learned policy was evaluated on an independent test set of 3000 episodes.

#### 4.1.3 Results

First, linear regression analyses confirmed that the three features ( $\text{VOI}_1(c, b)$ ,  $\text{VPI}(c, b)$ , and  $\text{cost}(c)$ ) are sufficient to capture between 90.8% and 100.0% of the variance in the value of computation for performing a simulation ( $\text{VOC}(b, c_1)$ ) across different states  $b$ , depending on the cost of computation. Concretely, as the cost of computation increased from 0.001 to 0.1 the regression weights shifted from  $0.76 \cdot \text{VPI} + 0.46 \cdot \text{VOI}_1 - 4.5 \cdot \text{cost}$  to  $0.00 \cdot \text{VPI} + 1.00 \cdot \text{VOI}_1 - 1.00 \cdot \text{cost}$  and the explained variance increased from 90.8% to 100.0%. The explained variance and the weights remained the same for costs greater than 0.1. Supplementary Figure 1 illustrates this fit for  $\lambda = 0.02$ .

Second, we found that the  $\text{VOI}_1$  and the  $\text{VPI}$  features are sufficient to learn a near-optimal metalevel policy. As shown in Figure 2, the performance of BMPS was at most 5.19% lower than the performance of the optimal metalevel policy across all costs. The difference in performance was largest for the lowest cost  $\lambda = 0.001$  ( $t(2999) = 3.75, p = 0.0002$ ) and decreased with increasing cost so that there was no statistically significant performance difference between BMPS and the optimal metalevel policy for costs greater than  $\lambda = 0.0025$  (all  $p > 0.15$ ). BMPS performed between 6.78% and 35.8% better than the meta-greedy policy across all costs where the optimal policy made more than one observation (all  $p < 0.0001$ ) and 20.3% better on average ( $t(44999) = 42.4, p < 10^{-15}$ ).

## 4.2 META-DECISION-MAKING

How should an agent allocate its limited decision-time across estimating the expected utilities of multiple alternatives? To evaluate how well BMPS can solve this kind of problem, we evaluate it on the *Bernoulli metalevel probability model* introduced by Hay et al. (2012). This problem is similar to the standard multi-armed bandit problem with one critical difference: Only the reward from the final pull counts—the previous “simulated” pulls provide information, but no reward. Like the first problem, the agent takes a single object-level action, choosing arm  $i$  and receiving reward  $r(s, a_i) \sim \text{Bernoulli}(\theta_i)$ . Unlike the first problem, however, the agent must track multiple environment parameters and select among competing computations.

### 4.2.1 Metalevel MDP

The Bernoulli metalevel probability model is a metalevel MDP  $M_{\text{meta}} = (\mathcal{B}, \mathcal{A}, T_{\text{meta}}, r_{\text{meta}}, h)$  where each belief state  $b$  defines  $k$  Beta distributions over the reward probabilities  $\theta_1, \dots, \theta_k$  of the  $k$  possible actions. Thus  $b$  can be represented by  $((\alpha_1, \beta_1), \dots, (\alpha_k, \beta_k))$  where  $b(\theta_i) = \text{Beta}(\theta_i; \alpha_i, \beta_i)$ . For the initial belief state  $b_0$ , these parameters are  $\alpha_i = \beta_i = 1$ . The metalevel actions  $\mathcal{A}$  are  $\{c_1, \dots, c_k, \perp\}$  where  $c_i$  simulates action  $a_i$  and  $\perp$  terminates deliberation and executes the action with the highest expected return. The metalevel transition function  $T_{\text{meta}}$  encodes that performing computation  $c_i$  increments  $\alpha_i$  with probability  $\frac{\alpha_i}{\alpha_i + \beta_i}$  and increments  $\beta_i$  with probability  $\frac{\beta_i}{\alpha_i + \beta_i}$ . The metalevel reward function  $r_{\text{meta}}(b, c)$  is  $-\lambda$  for  $c \in \{c_1, \dots, c_k\}$  and  $r_{\text{meta}}(b, \perp) = \max_i \frac{\alpha_i}{\alpha_i + \beta_i}$ . Finally, the horizon  $h$  is the maximum number of metalevel actions that can be performed and the last metalevel action must be  $\perp$ .

### 4.2.2 Evaluation procedure

We evaluated BMPS on Bernoulli metalevel probability problems with  $k \in \{2, \dots, 5\}$  object-level actions, a horizon of  $h = 25$ , and computational costs ranging from  $10^{-4}$  to  $10^{-1}$ . We compared the policy learned by BMPS with the optimal metalevel policy and three alternative approximations: the meta-greedy heuristic (Russell & Wefald, 1991b), the blinkered approximation (Hay et al., 2012), and the metalevel policy that always deliberates as much as possible. In addition to these, we also trained a Deep-Q-Network (DQN) (Mnih et al., 2015) on the metalevel MDP to compare the performance of our method to baselines achieved by off-the-shelf deep RL methods (Dhariwal et al., 2017).

We trained BMPS as described above, but with 10 iterations of 1000 episodes each. To combat the possibil-



ity of overfitting, we evaluated the average returns of the five best weight vectors over 5000 more episodes and selected the one that performed best. The relevance function for  $VPI_{\text{sub}}$  matches each computation with the single parameter it is informative about, i.e.,  $f(c_j, i) = \mathbb{1}(j = i)$ . The optimal metalevel policy and the blinkered policy were computed using backward induction (Puterman, 2014). The DQN was trained for 5,000,000 steps. Since the episodes have a horizon of  $h = 25$ , this resulted in more than 200,000 training episodes for the DQN. We evaluated the performance of each policy by its average return across 2000 test episodes for each combination of computational cost and number of object-level actions.

### 4.2.3 Results

We found that the BMPS policy attained 99.1% of optimal performance (0.6535 vs. 0.6596,  $t(1998) = -7.43, p < 0.0001$ ) and significantly outperformed the meta-greedy heuristic (0.60,  $t(1998) = 83.9, p < 10^{-15}$ ), the full-deliberation policy (0.20,  $t(1998) = 469.1, p < 10^{-15}$ ), and the DQN (0.58,  $t(1998) = 79.2, p < 10^{-15}$ ). The performance of BMPS (0.6535) and the blinkered approximation (0.6559) differed by only 0.37%.

Figure 3a shows the methods’ average performance as a function of the cost of computation. BMPS outperformed the meta-greedy heuristic for costs smaller than 0.03 (all  $p < 10^{-15}$ ), the full-deliberation policy for costs greater than 0.0003 (all  $p < 0.005$ ), and the DQN for all costs (all  $p < 10^{-15}$ ). For costs below 0.0003, the blinkered policy performed slightly better than BMPS (all  $p < 0.01$ ). For all other costs both methods performed at the same level (all  $p > 0.1$ ). For costs above 0.01, performance of BMPS becomes indistinguishable from the optimal policy’s performance (all  $p > 0.1$ ).

Figure 3b shows the metareasoning performance of each method as a function of the number of options. We found that the performance of BMPS scaled well with the size of the decision problem. For each number of options, the relative performance of the different methods was consistent with the results reported above.

Finally, as illustrated in Supplementary Figure 2, we found that BMPS learned surprisingly quickly, usually discovering near-optimal policies in less than 10 iterations. In particular, BMPS was able to perform significantly better than the DQN, despite being trained on fewer than 20% as many episodes. This demonstrates the value of the proposed VOI features, which dramatically constrain the space of possible metalevel policies to be considered.

## 4.3 METAREASONING ABOUT PLANNING

Having evaluated BMPS on problems of metareasoning about how to make a one-shot decision, we now evaluate its performance at deciding how to plan. To do so, we define the *Bernoulli metalevel tree*, which generalizes the Bernoulli metalevel probability model by replacing the one-shot decision between  $k$  options by a tree-structured sequential decision problem that we will refer to as the *object-level MDP*. The transitions of the object-level MDP are deterministic and known to the agent. The reward associated with each of  $k = 2^{h+1} - 1$  states in the tree is deterministic, but initially unknown;  $r(s, a, s_i) = \theta_i \in \{-1, 1\}$ . The agent can uncover these rewards through reasoning at a cost of  $-\lambda$  per reward. When the agent terminates deliberation, it executes a policy with maximal expected utility. Unlike in the previous domains, this policy entails a sequence of actions rather than a single action.

### 4.3.1 Metalevel MDP

The Bernoulli metalevel tree is a metalevel MDP  $M_{\text{meta}} = (\mathcal{B}, \mathcal{A}, T_{\text{meta}}, r_{\text{meta}})$  where each belief state  $b$  encodes one Bernoulli distribution for each transition’s reward. Thus,  $b$  can be represented as  $(p_1, \dots, p_i)$  such that  $b(\theta_i = 1) = p_i$  and  $b(\theta_i = -1) = 1 - p_i$ . The initial belief  $b_0$  has  $p_i = 0.5$  for all  $i$ . The metalevel actions are defined  $\mathcal{A} = \{c_1, \dots, c_k, \perp\}$  where  $c_i$  reveals the reward at state  $s_i$  and  $\perp$  selects the path with highest expected sum of rewards according to the current belief state. The transition function  $T_{\text{meta}}$  encodes that performing computation  $c_i$  sets  $p_i$  to 1 or 0 with equal probability (unless  $p_i$  has already been updated, in which case  $c_i$  has no effect). The metalevel reward function is defined  $r_{\text{meta}}(b, c) = -\lambda$  for  $c \in \{c_1, \dots, c_k\}$ , and  $r_{\text{meta}}(b, \perp) = \max_{\mathbf{t} \in \mathcal{T}} \sum_{i \in \mathbf{t}} \mathbb{E}[\theta_i | p_i]$  where  $\mathcal{T}$  is the set of possible trajectories  $\mathbf{t}$  through the environment, and  $\mathbb{E}[\theta_i | p_i] = 2p_i - 1$  is the expected reward attained at state  $s_i$ .

### 4.3.3 Evaluation procedure

We evaluated each method’s performance by its average return over 5000 episodes for each combination of tree-height  $h \in \{2, \dots, 6\}$  and computational cost  $\lambda \in \{2^{-7}, \dots, 2^0\}$ . To facilitate comparisons across planning problems with different numbers of steps, we measured the performance of metalevel policies by their expected return divided by the tree-height.

We trained the BMPS policy with 100 iterations of 1000 episodes each. To combat the possibility of overfitting, we evaluated the average returns of the three best weight vectors over 2000 more episodes and selected

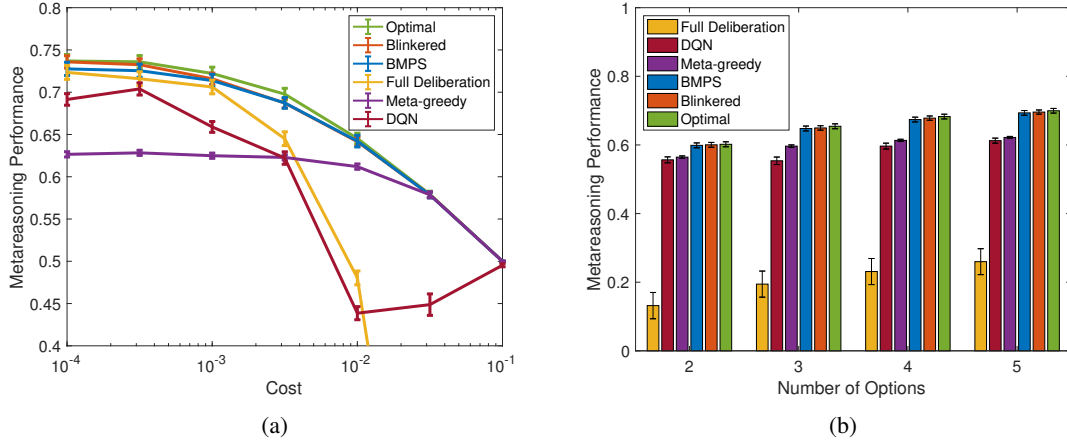


Figure 3: Metareasoning performance of alternative methods on the Bernoulli metalevel probability model (a) as a function of the cost of computation and (b) as a function of the number of actions. Metareasoning performance is defined as the expected reward for the chosen option minus the computational cost of the decision process. Error bars enclose 95% confidence intervals.

the one that performed best. The relevance function for  $VPI_{\text{sub}}$  maps a computation to all the parameters that affect the value of any policy that the initial computation is informative about, i.e.  $f(c_j, i) = \mathbb{1}(i \in \{j\} \cup \text{descendents}(j) \cup \text{ancestors}(j))$

For metareasoning about how to plan in trees of height 2 and 3, we were able to compute the optimal metalevel policy using dynamic programming. But for larger trees, computing the optimal metalevel policy would have taken significantly longer than 6 hours and was therefore not undertaken.

The blinkered policy of Hay et al. (2012) is not directly applicable to planning because of its assumption of “independent actions” which is violated in the Bernoulli metalevel tree. Briefly, the assumption is violated because the reward at a given state affects the value of multiple policies. Thus, we derived a recursive generalization of the blinkered policy to compare with our method. See the Supporting Materials for details.

#### 4.3.4 Results

We first compared BMPS with the optimal policy for  $h \in \{2, 3\}$ , finding that it attained 98.4% of optimal performance (0.367 vs. 0.373,  $t(159998) = -2.87$ ,  $p < 10^{-15}$ ). Metareasoning performance differed significantly across the four methods we evaluated ( $F(3, 799840) = 4625010$ ,  $p < 10^{-15}$ ), and the magnitude of this effect depends on the height of the tree ( $F(12, 799840) = 1110179$ ,  $p < 10^{-15}$ ) and the cost of computation ( $F(21, 799840) = 1266582$ ,  $p < 10^{-15}$ ).

Across all heights and costs, BMPS achieved a metareasoning

performance of 0.392 units of reward per object-level action, thereby outperforming the meta-greedy heuristic (0.307,  $t(399998) = 72.84$ ,  $p < 10^{-15}$ ), the recursively blinkered policy (0.368,  $t(399998) = 20.77$ ,  $p < 10^{-15}$ ), and the full-deliberation policy ( $-1.740$ ,  $t(399998) = 231.18$ ,  $p < 10^{-15}$ ).

As shown in Figure 4a, BMPS performed near-optimally across all computational costs, and its advantage over the meta-greedy heuristic and the tree-blinkered approximation was largest when the cost of computation was low, whereas its benefit over the full-deliberation policy increased with the cost of computation.

Figure 4b shows that the performance of BMPS scaled very well with the size of the planning problem, and that its advantage over the meta-greedy heuristic increased with the height of the tree.

## 5 IS METAREASONING USEFUL?

The costs of metareasoning often outweigh the resulting improvements in object-level reasoning. But here we show that the benefits of BMPS outweigh its costs in a potential application to emergency management.

During severe weather, important decisions—such as which cities to evacuate in the face of an approaching tornado—must be based on a limited number of computationally intense weather simulations that estimate the probability that a city will be severely hit (Baumgart, Bass, Philips, & Kloesel, 2008). Based on these simulations, an emergency manager makes evacuation decisions so as to minimize the risk of false positive errors

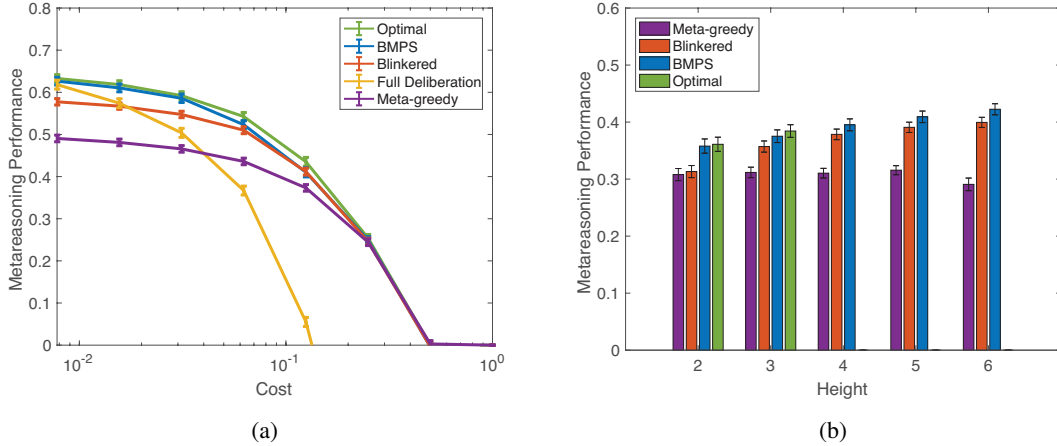


Figure 4: Metareasoning performance of alternative methods on the Bernoulli tree (a) as a function of computational cost (with tree-height 3) and (b) as a function of the number of actions (marginalizing over computational costs between  $10^{-4}$  and  $10^{-1}$ ). Metareasoning performance is normalized by tree height to facilitate comparison. In (b), the optimal policy is only shown for heights at which it can be computed in under six hours and the full observation policy is not shown because its performance is negative for all heights. Error bars enclose 95% confidence intervals.

(evacuating cities that are safe) and false negative errors (failing to evacuate a city the tornado hits). We assume that the manager has access to a single supercomputer, but pays no cost for running each simulation. Thus, the manager has a fixed budget of simulations and her goal is to maximize the expected utility of the final decision.

## 5.1 METHODS

We model the above scenario as follows: There is a finite amount of time  $T$  until evacuation decisions about  $k$  cities have to be made. For each city  $i$ , the emergency manager can run a fine grained, stochastic simulation ( $c_i$ ) of how it will be impacted by the approaching tornado. Each simulation yields a binary outcome, indicating whether the simulated impact would warrant an evacuation or not. The belief state  $b$  and transition function  $T_{\text{meta}}$  of the corresponding metalevel MDP are the same as in the Bernoulli metalevel probability model: Each belief state defines  $k$  Beta distributions that track the probability that the tornado will cause evacuation-warranting damage in each city. The parameters  $\alpha_i$  and  $\beta_i$  correspond to the number of simulations predicting that the tornado {would — would not} be strong enough to warrant an evacuation of city  $i$ . Prior to the first simulation, the parameters for each city  $i$  are initialized as  $\alpha_i = 0.1$  and  $\beta_i = 0.9$  to capture the prior knowledge that evacuations are rarely necessary. The primary formal difference from the Bernoulli metalevel probability model lies in how the final belief state is translated into a decision and reward. Rather than choosing a single option, the agent must make  $k$  independent binary deci-

sions about whether to evacuate each city. Evacuation has a cost,  $\lambda_{\text{evac}} = -1$ , but failing to evacuate a heavily-hit city has a much larger cost,  $\lambda_{\text{fn}} = -20$ . Thus, the metalevel reward function is

$$r_{\text{meta}}(b, \perp) = \sum_{1 \leq i \leq k} \max \left\{ \frac{\alpha_i}{\alpha_i + \beta_i} \cdot \lambda_{\text{fn}}, \lambda_{\text{evac}} \right\}. \quad (6)$$

In contrast to the previous simulations, we now explicitly consider the cost of metareasoning. The decision time  $T$  has to be allocated between reasoning about the cities and metareasoning about which city to reason about so that  $T = n_{\text{sim}} \cdot (t_{\text{MR}} + t_{\text{sim}})$ , where  $n_{\text{sim}}$  is the number of simulations run,  $t_{\text{MR}}$  is the amount of time it takes to choose one simulation to run (i.e. by metareasoning), and  $t_{\text{sim}}$  is the amount of time it takes to run one simulation. Thus, for given values of  $t_{\text{MR}}$  and  $t_{\text{sim}}$  the number of simulations that can be performed is  $n_{\text{sim}} = \left\lfloor \frac{T}{t_{\text{MR}} + t_{\text{sim}}} \right\rfloor$ , where  $\lfloor x \rfloor$  rounds  $x$  down to the closest integer. Note that metalevel policy is computed offline, and thus training time does not factor into the above equation. The simulations reported below use a single BMPS policy optimized for  $k = 20$  and  $n_{\text{sim}} = 50$  to mimic the reuse of pre-computed weights in practical applications; the weights are relatively insensitive to these parameters.

To assess if BMPS could be useful in practice, we compare the utility of evacuation decisions made by its metalevel policy to those made by a baseline metalevel policy that uniformly distributes simulations across the  $k$  cities. Since the BMPS policy has  $t_{\text{MR}} > 0$  while the baseline policy has  $t_{\text{MR}} \approx 0$ , BMPS will typically run

fewer simulations and must make up for this by choosing more valuable ones.

## 5.2 RESULTS

We evaluated the BMPS policy and the uniform computation policy on the tornado problem with  $T = 24$  hours,  $k \in \{10, 30\}$  cities, and a range of plausible values for the duration of each weather simulation ( $t_{\text{sim}} \in [2^{-2}, 2^4]$  hours). For each policy and parameter setting we estimate utility as the mean return over 5000 rollouts.

Empirically, we found that  $t_{\text{MR}} \approx 1$  ms for  $k = 10$  and  $t_{\text{MR}} \approx 3$  ms for  $k = 30$ . Thus, even with a conservative estimate of  $t_{\text{MR}} = 0.001$  hours, metareasoning would cost at most one simulation. Consequently, in our simulations, diverting some of the computational resources to metareasoning was advantageous regardless of how long exactly a tornado simulation might take and the number of cities being considered. As Figure 5 shows, the benefit of metareasoning was larger for the more complex problem with more cities and peaked for an intermediate cost of object-level reasoning.

While this is a hypothetical scenario, it suggests that BMPS could be useful for practical applications. Specifically, we suggest that the method will be most valuable when a metareasoning problem must be faced multiple times (so that the cost of training BMPS offline can be amortized) and object-level computations are expensive (so that the resulting savings in object-level reasoning outweigh the online cost of computing the features used for metareasoning). In follow-up simulations, we explored conditions in which the cost of metareasoning causes a substantial reduction in the number of simulations that can be run. We found that metareasoning continues to be useful as long as object-level computation is substantially more expensive than metareasoning (see Supplementary Material).

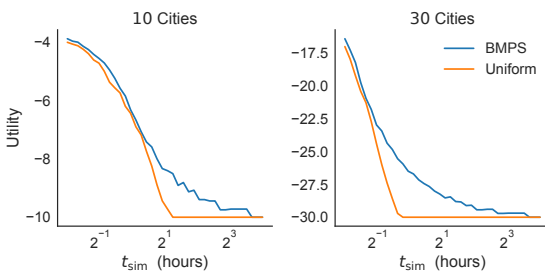


Figure 5: Benefit of metareasoning in the tornado evacuation scenario depending on the duration of each simulation ( $t_{\text{sim}}$ ) and the number of cities considered.

## 6 DISCUSSION

We have introduced a new approach to solving the foundational problem of rational metareasoning: metacognitive reinforcement learning. This approach applies algorithms from RL to metalevel MDPs to learn a policy for selecting computations. Our results show that BMPS can outperform the state of the art for approximate metareasoning. While we illustrated this approach using a policy search algorithm based on Bayesian optimization, there are many other RL algorithms that could be used instead, including policy gradient algorithms, actor-critic methods, and temporal difference learning with function approximation.

Since BMPS approximates the value of computation as a mixture of the myopic VOI and two other VOI features, it can be seen as a generalization of the meta-greedy approximation (Lin et al., 2015; Russell & Wefald, 1991a). It is the combination of these features with RL that makes BMPS tractable and powerful. BMPS works well across a wider range of problems than previous approximations because it reduces arbitrarily complex metalevel MDPs to low-dimensional optimization problems. We predict that metacognitive RL will enable significant advances in artificial intelligence and its applications. In the long view, metacognitive RL may become a foundation for self-improving AI systems that learn how to solve increasingly complex problems with increasing efficiency.

One weakness of our approach is that the time required to compute the value of perfect information by exact integration increases exponentially with the number of parameters in the agent’s model of the environment. Thus, an important direction for future work is developing efficient approximations or alternatives to this feature, and/or discovering new features via deep RL (Mnih et al., 2015). A second limitation is our assumption that the meta-reasoner has an exact model of its own computational architecture in the form of a metalevel MDP. This motivates the incorporation of model-learning mechanisms into a metacognitive RL algorithm.

We have shown that the benefits of metareasoning with our method already more than outweigh its computational costs in scenarios where the object-level computations are very expensive. It might therefore benefit practical applications that involve complex large-scale simulations, active learning problems, hyperparameter search, and the optimization of functions that are very expensive to evaluate. Finally, BMPS could also be applied to derive rational process models of human cognition.

## References

- Baumgart, L. A., Bass, E. J., Philips, B., & Kloesel, K. (2008). Emergency management decision making during severe weather. *Weather and Forecasting*, 23(6), 1268–1279.
- Campbell, M., Hoane, A. J., & Hsu, F.-H. (2002). Deep blue. *Artificial intelligence*, 134(1-2), 57–83.
- Cushman, F., & Morris, A. (2015). Habitual control of goal selection in humans. *Proceedings of the National Academy of Sciences*, 112(45), 13817–13822.
- Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., ... Wu, Y. (2017). *Open AI Baselines*. <https://github.com/openai/baselines>.
- Gershman, S. J., Horvitz, E. J., & Tenenbaum, J. B. (2015). Computational rationality: A converging paradigm for intelligence in brains, minds, and machines. *Science*, 349(6245), 273–278.
- Hammersley, J. (2013). *Monte Carlo methods*. Springer Science & Business Media.
- Harada, D., & Russell, S. J. (1998). Meta-level reinforcement learning. In *NIPS 1998 Workshop on Abstraction and Hierarchy in Reinforcement Learning*.
- Hay, N., Russell, S. J., Tolpin, D., & Shimony, S. (2012). Selecting computations: Theory and applications. In *Proceedings of the 28th Conference of Uncertainty in Artificial Intelligence*.
- Horvitz, E. J., Cooper, G. F., & Heckerman, D. E. (1989). Reflection and action under scarce resources: Theoretical principles and empirical study. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (pp. 1121–1127). San Mateo, CA: Morgan Kaufmann.
- Howard, R. A. (1966). Information value theory. *IEEE Transactions on systems science and cybernetics*, 2(1), 22–26.
- IBM Research. (1997). *Kasparov vs Deep Blue: A contrast in styles*. <http://researchweb.watson.ibm.com/deepblue>.
- Krueger, P. M., Lieder, F., & Griffiths, T. L. (2017). Enhancing metacognitive reinforcement learning using reward structures and feedback. In *Proceedings of the 39th Annual Conference of the Cognitive Science Society*.
- Lieder, F., & Griffiths, T. (2017). Strategy selection as rational metareasoning. *Psychological Review*, 124(6), 762–794.
- Lieder, F., Krueger, P. M., & Griffiths, T. L. (2017). An automatic method for discovering rational heuristics for risky choice. In *Proceedings of the 39th Annual Meeting of the Cognitive Science Society*.
- Lieder, F., Plunkett, D., Hamrick, J. B., Russell, S. J., Hay, N., & Griffiths, T. (2014). Algorithm selection by rational metareasoning as a model of human strategy selection. In *Advances in Neural Information Processing Systems 27* (pp. 2870–2878).
- Lin, C. H., Kolobov, A., Kamar, E., & Horvitz, E. (2015). Metareasoning for planning under uncertainty. In *Proceedings of the 24th International Conference on Artificial Intelligence* (pp. 1601–1609).
- Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... others (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529–533.
- Mockus, J. (2012). *Bayesian approach to global optimization: theory and applications* (Vol. 37). Springer Science & Business Media.
- Payne, J. W., Bettman, J. R., & Johnson, E. J. (1988). Adaptive strategy selection in decision making. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 14(3), 534.
- Puterman, M. L. (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.
- Russell, S. J., & Subramanian, D. (1995). Provably bounded-optimal agents. *Journal of Artificial Intelligence Research*, 2, 575–609.
- Russell, S. J., & Wefald, E. (1991a). *Do the right thing: studies in limited rationality*. Cambridge, MA: MIT press.
- Russell, S. J., & Wefald, E. (1991b). Principles of metareasoning. *Artificial Intelligence*, 49(1-3), 361–395.
- Shugan, S. M. (1980). The cost of thinking. *Journal of consumer Research*, 7(2), 99–111.
- Smith-Miles, K. A. (2009, January). Cross-disciplinary perspectives on meta-learning for algorithm selection. *ACM Comput. Surv.*, 41(1), 6:1–6:25.
- Thornton, C., Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2013). Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 847–855).
- Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., ... Botvinick, M. (2017, January 23). Learning to reinforcement learn. In *Proceedings of the 39th Annual Conference of the Cognitive Science Society*.

---

# Per-decision Multi-step Temporal Difference Learning with Control Variates

---

**Kristopher De Asis**

Department of Computing Science  
University of Alberta  
Edmonton, AB T6G 2E8  
kldeasis@ualberta.ca

**Richard S. Sutton**

Department of Computing Science  
University of Alberta  
Edmonton, AB T6G 2E8  
rsutton@ualberta.ca

## Abstract

Multi-step temporal difference (TD) learning is an important approach in reinforcement learning, as it unifies one-step TD learning with Monte Carlo methods in a way where intermediate algorithms can outperform either extreme. They address a bias-variance trade off between reliance on current estimates, which could be poor, and incorporating longer sampled reward sequences into the updates. Especially in the off-policy setting, where the agent aims to learn about a policy different from the one generating its behaviour, the variance in the updates can cause learning to diverge as the number of sampled rewards used in the estimates increases. In this paper, we introduce per-decision control variates for multi-step TD algorithms, and compare them to existing methods. Our results show that including the control variates can greatly improve performance on both on and off-policy multi-step temporal difference learning tasks.

## 1 TEMPORAL DIFFERENCE LEARNING

*Temporal-difference* (TD) methods (Sutton, 1988) combine ideas from Monte Carlo and dynamic programming methods, and are an important approach in *reinforcement learning*. They allow learning to occur from raw experience in the absence of a model of the environment's dynamics, like with Monte Carlo methods, while computing estimates which bootstrap off of other estimates, like with dynamic programming. TD methods provide a way to learn online and incrementally in both prediction and control settings.

Several TD methods have been proposed. Sarsa (Rum-

mery & Niranjan, 1994; Sutton, 1996) is a classical *on-policy* algorithm, where the policy being learned about, the *target policy*, is identical to the one generating the behaviour, the *behaviour policy*. However, Sarsa can be extended to learn *off-policy*, where the target policy can differ from the behaviour policy, through the use of per-decision *importance sampling* (Precup et al., 2000). Expected Sarsa (van Seijen et al., 2009) is another extension of Sarsa where instead of using the value of the current state-action pair to update the value of the previous state, it uses the expectation of the values of all actions in the current state under the target policy. Since Expected Sarsa takes the expectation under the target policy, it can be used off-policy without importance sampling to correct for the discrepancy between its target and behaviour policies. *Q-learning* (Watkins, 1989) is arguably the most popular off-policy TD control algorithm, as it can also perform off-policy learning without importance sampling, but it is equivalent to Expected Sarsa where the target policy is greedy. The above methods are often described in the one-step case, but they can be extended across multiple time steps.

Multi-step TD methods, such as the  $n$ -step TD and TD( $\lambda$ ) methods, create a spectrum of algorithms where at one end exists one-step TD learning, and at the other, exists Monte Carlo Methods. Intermediate algorithms are created which, due to a bias-variance tradeoff, can outperform either extreme (Jaakkola et al., 1994). Multi-step off-policy algorithms, especially ones with explicit use of importance sampling, have significantly larger variance than their on-policy counterparts (Sutton & Barto, 1998), and several proposals have been made to address this issue in the TD( $\lambda$ ) space of algorithms (Munos et al., 2016; Mahmood et al., 2017).

In this paper, we focus on  $n$ -step TD algorithms as they provide exact computation of the multi-step return, have conceptual clarity, and provide the foundation for TD( $\lambda$ ) methods. We formulate per-decision control variates for existing  $n$ -step TD algorithms, and give insight on their

implications in the TD( $\lambda$ ) space of algorithms. On problems with tabular representations as well as one with function approximation, we show that the introduction of per-decision control variates can improve the performance of existing  $n$ -step TD methods on both on and off-policy prediction and control tasks.

## 2 ONE-STEP TD METHODS

The sequential decision-making problem in reinforcement learning is often modeled as a *Markov decision process* (MDP). Under the MDP framework, an *agent* interacts with an environment over a sequence of discrete time steps. At each time step  $t$ , the agent receives information about the environment's current *state*,  $S_t \in \mathcal{S}$ , where  $\mathcal{S}$  is the set of all possible states in the MDP. The agent is to use this state information to select an *action*,  $A_t \in \mathcal{A}(S_t)$ , where  $\mathcal{A}(s)$  is the set of possible actions in state  $s$ . Based on the environment's current state and the agent's selected action, the agent receives a *reward*,  $R_{t+1} \in \mathbb{R}$ , and gets information about the environment's next state,  $S_{t+1} \in \mathcal{S}$ , according to the *environment model*:  $p(r, s' | s, a) = P(R_{t+1} = r, S_{t+1} = s' | S_t = s, A_t = a)$ .

The agent selects actions according to a *policy*,  $\pi(s, a) = P(A_t = a | S_t = s)$ , which gives a probability distribution across actions  $a \in \mathcal{A}(s)$  for a given state  $s$ . Through policy iteration (Sutton & Barto, 1998), the agent can learn an optimal policy,  $\pi^*$ , where behaving under it will maximize the expected discounted return:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots = \sum_{k=0}^{T-t-1} \gamma^k R_{t+k+1} \quad (1)$$

given a discount factor  $\gamma \in [0, 1]$  and  $T$  equal to the final time step in an episodic task, or  $\gamma \in [0, 1)$  and  $T$  equal to infinity for a continuing task.

*Value-based methods* approach the sequential decision-making problem by computing *value functions*, which provide estimates of what the return will be from a particular state onwards. In prediction problems, also referred to as *policy evaluation*, the goal is to estimate the return under a particular policy as accurately as possible, and a *state-value function* is often estimated. It is defined to be the expected return when starting in state  $s$  and following policy  $\pi$ :  $v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$ . For control problems, the policy which maximizes the expected return is to be learned, and an *action-value function* from which a policy can be derived is instead estimated. It is defined to be the expected return when taking action  $a$  in state  $s$ , and following policy  $\pi$ :

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a] \quad (2)$$

Of note, the action-value function can still be used for prediction problems, and the state-value can be computed as an expectation across action-values under the policy  $\pi$  for a given state:

$$v_\pi(s) = \mathbb{E}_\pi[q_\pi(s, \cdot)] = \sum_a \pi(s, a) q_\pi(s, a) \quad (3)$$

One-step TD methods learn an approximate value function, such as  $Q \approx q_\pi$  for action-values, by computing an estimate of the return,  $\hat{G}_t$ . First, Equation 2 can be written in terms of its successor state-action pairs, also known as the *Bellman equation* for  $q_\pi$ :

$$q_\pi(s, a) = \sum_{r, s'} p(r, s' | s, a) \left( r + \gamma \sum_{a'} \pi(s', a') q_\pi(s', a') \right) \quad (4)$$

Based on Equation 4, one-step TD methods estimate the return by taking an action in the environment according to a policy, sampling the immediate reward, and bootstrapping off of the current estimates in the value function for the remainder of the return. The difference between this *TD target* and the value of the previous state-action pair is then computed, and is often referred to as the *TD error*. The previous state-action pair's value is then updated by taking a step proportional to the TD error with step size  $\alpha \in (0, 1]$ :

$$\hat{G}_t = R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) \quad (5)$$

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha [\hat{G}_t - Q(S_t, A_t)] \quad (6)$$

Equations 5 and 6 correspond to the Sarsa algorithm. It can be seen that in state  $S_{t+1}$ , it samples an action  $A_{t+1}$  according to its behaviour policy, and then bootstraps off of the value of this state-action pair. With a sufficiently small step size, this estimates the expectation under its behaviour policy over the values of successor state-action pairs in Equation 4, allowing for on-policy learning.

In the off-policy case, the discrepancy from  $A_{t+1}$  being drawn from the behaviour policy needs to be corrected. One approach is to correct the affected terms with per-decision *importance sampling*. With actions sampled from a behaviour policy  $\mu$ , and a target policy  $\pi$ , the estimate of the return of off-policy Sarsa with per-decision importance sampling becomes:

$$\rho_t = \frac{\pi(S_t, A_t)}{\mu(S_t, A_t)} \quad (7)$$

$$\hat{G}_t = R_{t+1} + \gamma \rho_{t+1} Q(S_{t+1}, A_{t+1}) \quad (8)$$

Note that in the on-policy case,  $\rho_t$  is always 1, strictly generalizing the original on-policy TD target in Equation 5.

Another approach for the off-policy case is to compute the expectation of all successor state action pairs under

the target policy directly, instead of sampling and correcting the discrepancy. This approach has lower variance and is often preferred in the one-step setting for action-values, and gives the *Expected Sarsa* algorithm (van Seijen et al., 2009) characterized by the following TD target:

$$\hat{G}_t = R_{t+1} + \gamma \mathbb{E}_\pi[Q(S_{t+1}, \cdot)] \quad (9)$$

### 3 MULTI-STEP TD LEARNING

TD algorithms are referred to as one-step TD algorithms when they only incorporate information from a single time step in the estimate of the return that the value function is being updated towards. In multi-step TD methods, a longer sequence of experienced rewards is used to estimate the return. For example, on-policy  $n$ -step Sarsa would update an action-value  $Q(S_t, A_t)$  towards the following estimate:

$$\begin{aligned} \hat{G}_{t:t+n} &= R_{t+1} + \gamma R_{t+2} + \dots + \gamma^n Q(S_{t+n}, A_{t+n}) \\ &= \sum_{k=0}^{n-1} \gamma^k R_{t+k+1} + \gamma^n Q(S_{t+n}, A_{t+n}) \end{aligned} \quad (10)$$

Of note,  $n$ -step Expected Sarsa (Sutton & Barto, 2018) is identical up until the  $n$ -th step, where it instead bootstraps off of the expectation under the target policy:

$$\hat{G}_{t:t+n} = \sum_{k=0}^{n-1} \gamma^k R_{t+k+1} + \gamma^n \mathbb{E}_\pi[Q(S_{t+n}, \cdot)] \quad (11)$$

The  $n$ -step returns can also be written recursively, and is convenient in the more general per-decision off-policy case. If we define the following bootstrapping condition:

$$\hat{G}_{t:t} = Q(S_t, A_t) \quad (12)$$

The  $n$ -step extension of off-policy Sarsa with per-decision importance sampling, as characterized by Equations 7 and 8, can now be written as:

$$\hat{G}_{t:t+n} = R_{t+1} + \gamma \rho_{t+1} \hat{G}_{t+1:t+n} \quad (13)$$

TD algorithms which update towards these  $n$ -step estimates of the return constitute the  $n$ -step TD algorithm family (Sutton & Barto, 2018). Their computational complexity increases with  $n$ , but have the benefit of conceptual clarity, and exact computation of the multi-step return. The  $n$ -step returns also provide the foundation for other multi-step TD algorithms.

Another family of multi-step per-decision TD algorithms,  $TD(\lambda)$ , is also used in practice. They are characterized by computing a geometrically weighted sum of

$n$ -step returns, denoted as the  $\lambda$ -return:

$$\hat{G}_t^\lambda = (1 - \lambda) \sum_{n=1}^{\infty} \lambda^{n-1} \hat{G}_{t:t+n} \quad (14)$$

It introduces a hyperparameter  $\lambda \in [0, 1]$  where  $\lambda = 0$  gives one-step TD, and increasing  $\lambda$  effectively increases the number of sampled rewards included in the estimated return. Substituting the  $n$ -step Sarsa return (13) into Equation 14 gives the  $\lambda$ -return for the Sarsa( $\lambda$ ) algorithm, and assuming  $Q$  does not change, it can be expressed as a sum of one-step Sarsa's TD errors:

$$\begin{aligned} \hat{G}_t &= R_{t+1} + \gamma \rho_{t+1} Q(S_{t+1}, A_{t+1}) \\ \hat{G}_t^\lambda &= Q(S_t, A_t) + \sum_{k=t}^{\infty} (\hat{G}_k - Q(S_k, A_k)) \prod_{i=t+1}^k \gamma \lambda \rho_i \end{aligned} \quad (15)$$

This shows that the  $\lambda$ -return for Sarsa( $\lambda$ ) can be estimated by computing one-step TD errors, and decaying the weight of later TD errors at a rate of  $\gamma \lambda \rho_t$ . Implementing this online and incrementally, an *eligibility trace* vector is maintained to track which state-action pairs led to the current step's TD error. The traces of earlier state-action pairs are decayed at each step by the aforementioned decay rate, and each action-value is adjusted by the current TD error weighted by the trace of the corresponding state-action pair.

Contrasting with  $n$ -step TD methods, the computational complexity of TD( $\lambda$ ) control algorithms scales with the size of the environment,  $|S| \times |A|$ . That is, there is an environment-specific increase in complexity, but it no longer scales with the number of sampled rewards in the estimate of the return.

### 4 PER-DECISION CONTROL VARIATES

When trying to estimate the expectation of some variable  $X$ , control variates are often of the following form (Ross, 2013):

$$X^* = X + c(Y - \mathbb{E}[Y]) \quad (16)$$

where  $Y$  is the outcome of another variable with a known expected value, and  $c$  is a coefficient to be set.  $X^*$  then has the following variance:

$$\text{Var}(X^*) = \text{Var}(X) + c^2 \text{Var}(Y) + 2c \text{Cov}(X, Y) \quad (17)$$

From this, the variance can be minimized with the optimal coefficient  $c^*$ :

$$c^* = -\frac{\text{Cov}(X, Y)}{\text{Var}(Y)} \quad (18)$$



Suppose the  $n$ -step Sarsa algorithm samples the importance sampling-corrected  $n$ -step return, jointly samples the importance sampling-corrected action-value (through the sampled action), and computes the expected action-value under the target policy. We get the following estimate of this term of the multi-step return:

$$\begin{aligned} (\rho_{t+1}\hat{G}_{t+1:t+n})^* &= \rho_{t+1}\hat{G}_{t+1:t+n} \\ &+ c(\rho_{t+1}Q(S_{t+1}, A_{t+1}) - \mathbb{E}_\pi[Q(S_{t+1}, \cdot)]) \end{aligned} \quad (19)$$

Under the assumption that the current estimates are accurate, the action-values represent the expected return. Due to this, the sampled reward sequence and the action-value are, in expectation, perfectly correlated. The covariance term in Equation 18 would then be the variance of the action-value due to the policy, and from this, a reasonable choice for the coefficient would be  $-1$ . This gives:

$$\begin{aligned} (\rho_{t+1}\hat{G}_{t+1:t+n})^* &= \rho_{t+1}\hat{G}_{t+1:t+n} \\ &+ \mathbb{E}_\pi[Q(S_{t+1}, \cdot)] - \rho_{t+1}Q(S_{t+1}, A_{t+1}) \end{aligned} \quad (20)$$

Substituting this estimate into the recursive definition of  $n$ -step Sarsa (13) and maintaining the same bootstrapping condition in Equation 12 gives the following  $n$ -step return:

$$\begin{aligned} \hat{G}_{t:t+n} &= R_{t+1} + \gamma(\rho_{t+1}\hat{G}_{t+1:t+n} \\ &+ \mathbb{E}_\pi[Q(S_{t+1}, \cdot)] - \rho_{t+1}Q(S_{t+1}, A_{t+1})) \end{aligned} \quad (21)$$

Because  $\mathbb{E}_\mu[\mathbb{E}_\pi[Q(S_{t+1}, \cdot)] - \rho_{t+1}Q(S_{t+1}, A_{t+1})] = 0$ , the additional term does not introduce bias into the estimate. To provide an intuition of how it might reduce the variance in the estimate, we can consider some extreme cases of the importance sampling ratio. If  $\rho_{t+1} = 0$ , when the behaviour policy takes an action that the target policy would have never taken, it will bootstrap off of the expectation of its current estimates instead of cutting the return. If  $\rho_{t+1}$  is much greater than 1, an equivalent amount of its current action-value estimate is subtracted to compensate.

In the one-step case, the introduction of this control variate results in one-step Expected Sarsa's target:

$$\begin{aligned} \hat{G}_{t:t+n} &= R_{t+1} + \gamma(\rho_{t+1}Q(S_{t+1}, A_{t+1}) \\ &+ \mathbb{E}_\pi[Q(S_{t+1}, \cdot)] - \rho_{t+1}Q(S_{t+1}, A_{t+1})) \\ \hat{G}_{t:t+n} &= R_{t+1} + \gamma\mathbb{E}_\pi[Q(S_{t+1}, \cdot)] \end{aligned}$$

When applied at the bootstrapping step, it implicitly results in bootstrapping off of the expectation under the target policy as opposed to the importance sampling-corrected action-value. It can be viewed as an alternate generalization of Expected Sarsa to the multi-step setting, where the control variate is applied to the sampled reward sequence in addition to the bootstrapping step.

The control variate can be interpreted as performing an *expectation correction* at each step based on current estimates. Each reward in the trajectory depends on the sampled action at each step, but the algorithm aims to learn the expectation across all possible trajectories under a policy. The importance sampling-corrected action-value is a closer estimate to the sampled return, as the agent knows which action resulted in the immediate reward at each step. Because of this, the action-value is like a guess of what the remainder of the sampled reward sequence will be, and the difference between that and the expectation across all actions provides a per-step estimate of the discrepancy between the sampled reward sequence and the expectation across all reward sequences from the current step onwards.

It can also be seen as implicitly performing adaptive  $n$ -step learning, adjusting the amount of information included based on how accurate its current estimates are. If we rearrange the  $n$ -step return:

$$\begin{aligned} \hat{G}_{t:t+n} &= R_{t+1} + \gamma\mathbb{E}_\pi[Q(S_{t+1}, \cdot)] \\ &+ \gamma(\rho_{t+1}\hat{G}_{t+1:t+n} - \rho_{t+1}Q(S_{t+1}, A_{t+1})) \end{aligned} \quad (22)$$

We get the one-step Expected Sarsa target, along with some difference between the actual sampled rewards and its current estimates. If the value estimates are poor, more rewards will be effectively included in the estimate, and vice-versa. If there is no stochasticity in the environment, it ends up approaching one-step Expected Sarsa as the estimates get close to the true value function.

If we follow similar steps in the state-value case, we arrive at the following  $n$ -step return with a per-decision control variate:

$$\begin{aligned} \hat{G}_{t:t} &= V(S_t) \\ \hat{G}_{t:t+n} &= \rho_t(R_{t+1} + \gamma\hat{G}_{t+1:t+n}) + V(S_t) - \rho_tV(S_t) \\ \hat{G}_{t:t+n} &= \rho_t(R_{t+1} + \gamma\hat{G}_{t+1:t+n}) + (1 - \rho_t)V(S_t) \end{aligned} \quad (23)$$

Of note, the state-value control variate disappears in the on-policy case, but the action-value one does not.

## 5 RELATIONSHIP WITH EXISTING ALGORITHMS

If we substitute the  $n$ -step Sarsa return with the per-decision control variate (21) into the definition of the  $\lambda$ -return in Equation 14, we can rearrange it into a sum of

one-step Expected Sarsa’s TD errors:

$$\begin{aligned}\hat{G}_t &= R_{t+1} + \gamma \mathbb{E}_\pi [Q(S_{t+1}, \cdot)] \\ \hat{G}_t^\lambda &= Q(S_t, A_t) + \sum_{k=t}^{\infty} (\hat{G}_k - Q(S_k, A_k)) \prod_{i=t+1}^k \gamma \lambda \rho_i\end{aligned}\quad (24)$$

This is equivalent to using the eligibility trace decay rate of Sarsa( $\lambda$ ), but backing up the TD error of one-step Expected Sarsa. That is, in the space of action-value TD( $\lambda$ ) algorithms, having the one-step estimates of the return bootstrap off of the expectation under the target policy implicitly induces this per-decision control variate in the corresponding  $n$ -step returns.

An existing algorithm that also uses one-step Expected Sarsa’s TD error in its  $\lambda$ -return is the Tree-backup( $\lambda$ ) algorithm (Precup et al., 2000). Denoting  $\pi_t = \pi(S_t, A_t)$ , Tree-backup( $\lambda$ ) is characterized by the following equations:

$$\begin{aligned}\hat{G}_t &= R_{t+1} + \gamma \mathbb{E}_\pi [Q(S_{t+1}, \cdot)] \\ \hat{G}_t^\lambda &= Q(S_t, A_t) + \sum_{k=t}^{\infty} (\hat{G}_k - Q(S_k, A_k)) \prod_{i=t+1}^k \gamma \lambda \pi_t\end{aligned}\quad (25)$$

If we look at  $n$ -step Tree-backup’s estimate of the return, we can show that it also includes the expectation correction terms:

$$\begin{aligned}\hat{G}_{t:t+n} &= R_{t+1} + \gamma (\pi_{t+1} \hat{G}_{t+1:t+n} \\ &\quad + \sum_{a \neq A_{t+1}} \pi(S_{t+1}, a) Q(S_{t+1}, a)) \\ \hat{G}_{t:t+n} &= R_{t+1} + \gamma (\pi_{t+1} \hat{G}_{t+1:t+n} \\ &\quad + \mathbb{E}_\pi [Q(S_{t+1}, \cdot)] - \pi_{t+1} Q(S_{t+1}, A_{t+1}))\end{aligned}\quad (26)$$

The estimate takes some portion of the sampled reward sequence, and the difference between the expectation under the target policy and an equivalent portion of the sampled action-value estimate.

The introduction of the control variates with the aforementioned choice of the control variate parameter results in an instance of a doubly robust estimator. The use of doubly-robust estimators in off-policy policy evaluation has been investigated by Jiang et al. (2016) and Thomas et al. (2016). However, results when applying the approaches in an online, model-free setting, as well as its view as a multi-step generalization of Expected Sarsa, appear to be novel.

Harutyunyan et al. (2016) has acknowledged the implicit introduction of these terms when using the expectation form of the TD error in action-value TD( $\lambda$ ) algorithms.

However, their work investigated the off-policy correcting effects of including the difference between the expectation under the target policy with an action-value sampled from the behaviour policy (without importance sampling corrections). This work focuses on the effect of explicitly including the additional terms, with importance sampling, in the  $n$ -step setting for both on and off-policy TD learning.

In the state-value setting, combining Equations 23 and 14 gives the following  $\lambda$ -return:

$$\begin{aligned}\hat{G}_t &= R_{t+1} + \gamma V(S_{t+1}) \\ \hat{G}_t^\lambda &= V(S_t) + \rho_t \sum_{k=t}^{\infty} (\hat{G}_k - V(S_k)) \prod_{i=t+1}^k \gamma \lambda \rho_i\end{aligned}\quad (27)$$

which is an intuitive generalization of off-policy per-decision importance sampling for state-values, having an additional importance sampling correction term for the first reward in the sequence. It can be seen that the inclusion of an action-dependent trace decay rate scaling a TD error, as opposed to the return estimate alone, implicitly induces the state-value control variate in the  $n$ -step estimate of the return.

## 6 EXPERIMENTS

In this section, we focus on the action-value setting and investigate the performance of  $n$ -step Sarsa with the per-decision control variate (denoted as *n-step CV Sarsa*) on three problems. The first two are multi-step prediction tasks in a tabular environment, one being off-policy and one being on-policy. The remaining one is a control problem involving function approximation, evaluating the performance of  $n$ -step CV Sarsa beyond the tabular setting, as well as how it handles a changing (greedifying) policy.

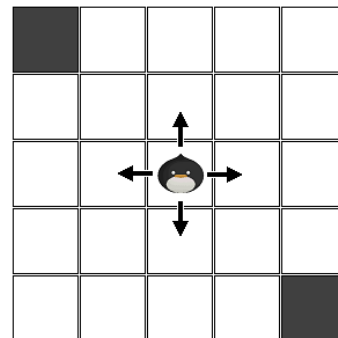


Figure 1: 5×5 Grid World environment. It was set up as an on and off-policy multi-step prediction task where the goal was to estimate the expected return under the target policy as accurately as possible.

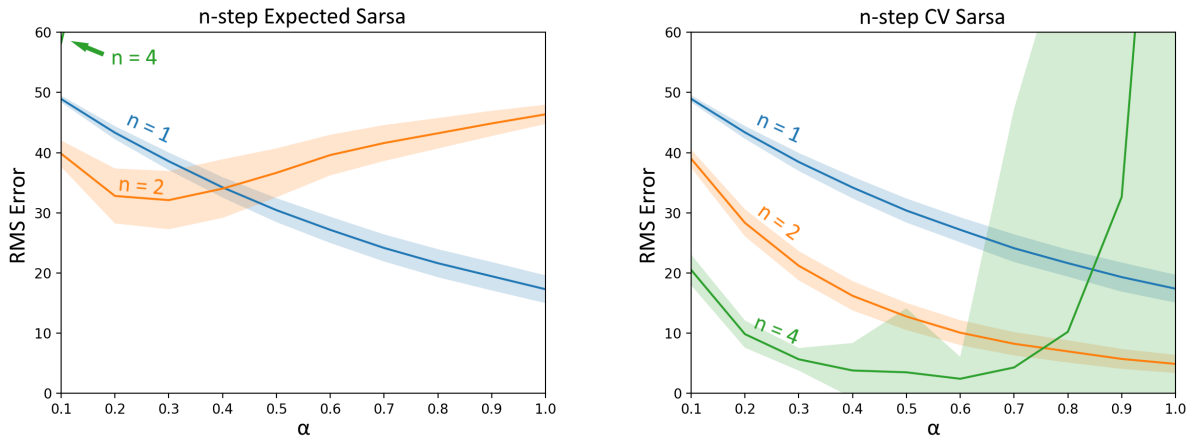


Figure 2: 5x5 Grid World off-policy prediction results. The plot shows the performance of various parameter settings of each algorithm in terms of RMS error after 200 episodes in the learned value function. The shaded region corresponds to one standard deviation, and the results are averaged over 1000 runs.

Since  $n$ -step CV Sarsa ends up bootstrapping off of the expectation over action-values at the end of the reward sequence, we compare the algorithm to  $n$ -step Expected Sarsa as characterized by Equation 11. This allows for examining the effects of the control variate being applied to each reward in the reward sequence in an online and incremental setting.

## 6.1 5x5 GRID WORLD

The 5x5 *Grid World* is a 2-dimensional grid world having terminal states in two opposite corners. The actions consist of 4-directional movement, and moving into a wall transitions the agent to the same state. The agent starts in the center, and a reward of  $-1$  is received at each transition. Experiments were run in both the off-policy and on-policy settings with no discounting ( $\gamma = 1$ ), and the root-mean-square (RMS) error between the learned value function and the true value function were compared.

### 6.1.1 Off-policy Prediction

For the off-policy experiments, the target policy would move north with probability  $1 - \epsilon$ , and select a random action equiprobably otherwise.  $\epsilon$  was set to 0.5, and the behaviour policy was equiprobable random for all states. A parameter study was done for 1, 2, and 4 steps, and the RMS error was measured after 200 episodes. The results are averaged over 1000 runs, and can be seen in Figure 2.

It can be seen that 2-step Expected Sarsa only outperforms 1-step Expected Sarsa for a very limited range of parameters, but is worse otherwise. 4-step Expected

Sarsa was unable to learn for most parameter settings. When the control variate is applied to each reward, we can see that 2-step CV Sarsa outperforms 1-step Expected Sarsa for all parameter settings, and the variance is reduced relative to 2-step Expected Sarsa. Furthermore, 4-step CV Sarsa ends up being able to learn, and can outperform 2-step CV Sarsa for a reasonably wide range of parameters.

### 6.1.2 On-policy Prediction

In the on-policy case, the target policy and behaviour policy were both equiprobable random for all states. The parameters tested are identical to the off-policy experiment with the addition of 8-step instances of each algorithm. The RMS error was measured after 200 episodes, and are also averaged over 1000 runs. The results are summarized in Figure 3.

2-step Expected Sarsa ends up performing better than 1-step Expected Sarsa for a wider range of parameters than in the off-policy case, but the best parameter settings for each perform similarly. Further increasing the number of steps results in relatively poor performance, and doesn't do better than the best parameter setting of 1-step Expected Sarsa. Looking at  $n$ -step CV Sarsa, we can see that performance is drastically improved for all tested settings of  $n$ . Of note, while introducing the per-decision control variate resulted in lower variance for a reasonable range of parameters, assumptions were made regarding the accuracy of the value function when setting the control variate parameter  $c$  in Equation 19. If the number of steps  $n$  and the step size  $\alpha$  get too large, it can result in larger variance and divergence on parameter settings where  $n$ -step Expected Sarsa did not diverge.

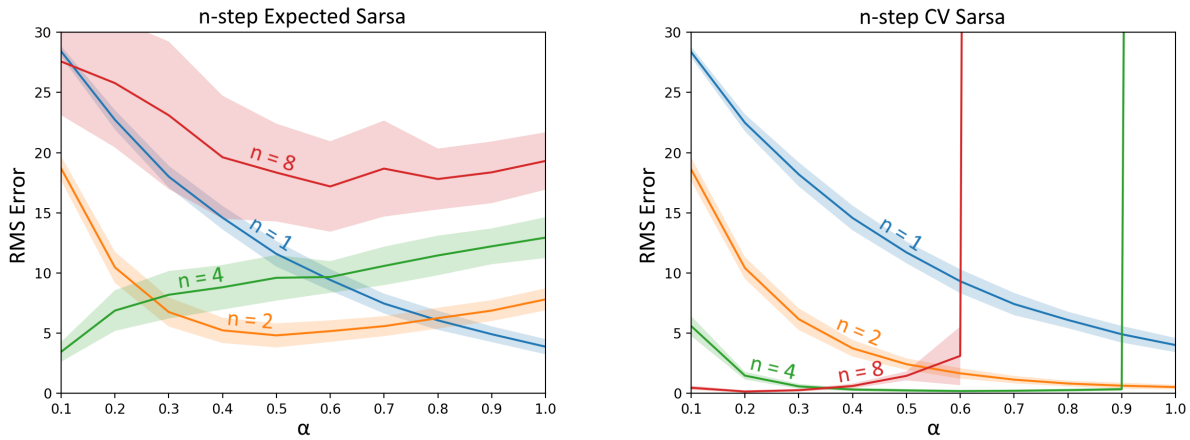


Figure 3: 5x5 Grid World on-policy prediction results. The plot shows the performance of various parameter settings of each algorithm in terms of RMS error after 200 episodes in the learned value function. The shaded region corresponds to one standard deviation, and the results are averaged over 1000 runs.

We did not investigate alternate methods of setting the control variate parameter in this work.

## 6.2 MOUNTAIN CAR

To show that this use of control variates is compatible with function approximation, we ran experiments on *mountain car* as described by Sutton and Barto (1998). A reward of  $-1$  is received at each step, and there is no discounting.

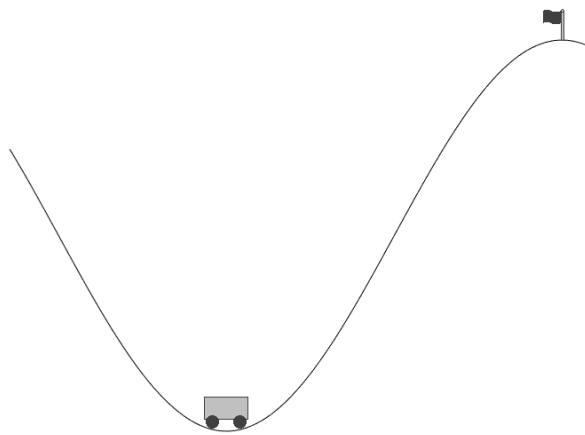


Figure 4: The mountain car environment (Sutton & Barto, 1998). The agent starts at a random location in the valley, receives a reward of  $-1$  at each step, and its goal is to drive past the flag in as few steps as possible.

Because the environment's state space is continuous, we used *tile coding* (Sutton & Barto, 1998) to produce a feature representation for use with linear function approximation. The tile coder used 16 tilings, an asymmetric

offset by consecutive odd numbers, and each tile covered  $1/8$ -th of the feature space in each direction.

We compared  $n$ -step Expected Sarsa and  $n$ -step CV Sarsa with 1, 2, 4, and 8 steps across different step sizes  $\alpha$ . Each algorithm learned on-policy with an  $\epsilon$ -greedy policy which selects an action greedily with respect to its value function with probability  $1 - \epsilon$ , and selected a random action equiprobably otherwise. In this experiment,  $\epsilon$  was set to 0.1. We measured the return per episode up to 100 episodes, and averaged the results over 100 runs. The results for the best parameter setting for each algorithm can be found in Figure 5.

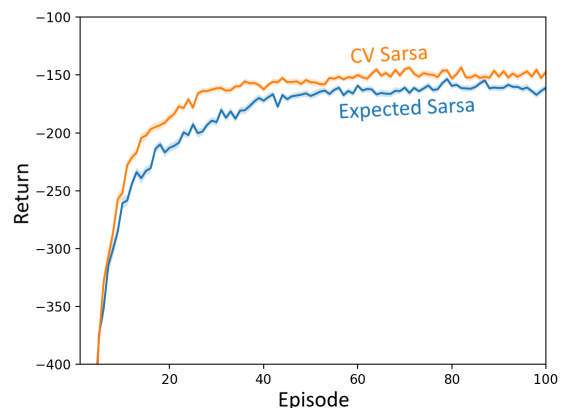


Figure 5: Mountain Car on-policy control results. The plot shows the return per episode of the best parameter setting of each algorithm in terms of the mean return over all episodes. The shaded region corresponds to one standard error, and the results are averaged over 100 runs.

The two algorithms showed a similar trend in the parameters as in the  $5 \times 5$  Grid World environment, but were less pronounced. This is likely due to not requiring accurate value function estimates to perform the task well, and the control variate having less of an effect with greedier target policies, because  $\mathbb{E}_\pi[Q(S_t, \cdot)]$  gets relatively close to  $Q(S_t, A_t)$ . Despite this, as seen in the results for the best parameter settings,  $n$ -step CV Sarsa still outperforms  $n$ -step Expected Sarsa on this task.

## 7 DISCUSSION

From our experiments,  $n$ -step CV Sarsa appears to be an improved multi-step generalization of Expected Sarsa. In both on and off-policy prediction tasks on the  $5 \times 5$  Grid World environment, it generally resulted in lower variance as well as considerably lower error in the estimates compared to  $n$ -step Expected Sarsa, an algorithm which can be interpreted as only applying the control variate at the bootstrapping step. Moreover, when used on a continuous state space control problem with function approximation, applying the control variate on a per-reward level still resulted in greater performance in terms of average return per episode.

Despite the improvement on most of the tested parameter settings, the results also showed that the addition of the per-decision control variates can cause learning to diverge for large  $n$  and large step size  $\alpha$ , even on settings where  $n$ -step Expected Sarsa did not diverge. It is suspected that this is due to assuming the estimates are accurate when setting the control variate parameter in Equation 19. This was not further investigated, but it could be an avenue for future work.

While our results focused on the action-value per-decision control variate, other experiments not included in this paper showed that the state-value per-decision control variate in Equation 23 can also be applied in the off-policy action-value setting. It resulted in performance in between that of  $n$ -step Expected Sarsa and  $n$ -step CV Sarsa, supporting that it is beneficial to add it, but better to use the action-value control variate if the agent is learning action-values.

## 8 CONCLUSIONS

In this paper, we presented a way to derive per-decision control variates in both state-value and action-value  $n$ -step TD methods. The state-value control variate is only present in the off-policy setting, but the action-value control variate affects both on and off-policy learning. In the action-value case, applying the per-decision control variate results in an alternative multi-step extension of Ex-

pected Sarsa. With this control variate perspective, the existing  $n$ -step Expected Sarsa algorithm can be interpreted as only applying a control variate at the bootstrapping step, when it can be applied to the sampled reward sequence as well. Our results on prediction and control problems show that applying it on a per-decision level can greatly improve the accuracy of the learned value function, and consequently perform better when doing TD control.

We also showed how the per-decision control variates relate to TD( $\lambda$ ) algorithms. This provided insight on how minor adjustments in the TD( $\lambda$ ) space can implicitly induce these per-decision control variates in the underlying  $n$ -step returns, resulting in a more unified view of per-decision multi-step TD methods.

Our experiments were limited to the  $n$ -step TD setting without eligibility traces, and focused on learning action-values. We only considered a naive setting of the control variate scaling parameter  $c$ , when our results suggest that the way we set it can negatively affect learning for a few (relatively extreme) parameter combinations. Perhaps insight from the analytical optimal coefficient in Equation 18 can be used to adapt the control variate online to further improve performance.

## Acknowledgements

The authors thank Yi Wan for insights and discussions contributing to the results presented in this paper, and the entire Reinforcement Learning and Artificial Intelligence research group for providing the environment to nurture and support this research. We gratefully acknowledge funding from Alberta Innovates – Technology Futures, Google Deepmind, and from the Natural Sciences and Engineering Research Council of Canada.

## References

- Harutyunyan, A., Bellemare, M. G., Stepleton, T., and Munos, R. (2016).  $Q(\lambda)$  with off-policy corrections. *arXiv:1509.05172*.
- Jaakola, T., Jordan, M. I., and Singh, S. P. (1994). On the convergence of stochastic iterative dynamic programming algorithms. *Neural Computation* 6(6), 1185-1201.
- Jiang, N., and Li, L. (2016). Doubly Robust Off-policy Value Evaluation for Reinforcement Learning. *Proceedings of The 33rd International Conference on Machine Learning*, in PMLR 48:652-661.
- Mahmood, A. R., Yu, H., and Sutton, R. S. (2017). Multi-step off-policy learning without importance sampling ratios. *arXiv:1702.03006*.
- Munos, R., Stepleton, T., Harutyunyan, A., and Bellemare, M. G. (2016). Safe and efficient off-policy reinforcement learning. *arXiv:1606.02647*.
- Precup, D., Sutton, R. S., and Singh, S. (2000). Eligibility traces for off-policy policy evaluation. In *Proceedings of the 17th International Conference on Machine Learning*, pp. 759-766. Morgan Kaufmann.
- Ross, S. M. (2013). *Simulation*. San Diego: Academic Press.
- Rummery, G. A. (1995). *Problem Solving with Reinforcement Learning*. PhD Thesis, Cambridge University.
- Rummery, G. A., and Niranjjan, M. (1994). On-line Q-learning using connectionist systems. Technical Report CUEF/F-INFENG/TR 166, Engineering Department, Cambridge University.
- Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning* 3, 9-44.
- Sutton, R. S. (1996). Generalization in reinforcement learning: Successful examples using sparse coarse coding. In Touretzky, D. S. and Hasselmo, M. E. (eds.), *Advances in Neural Information Processing Systems* 8, pp. 1038-1044. MIT Press.
- Sutton, R. S., and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, Massachusetts.
- Sutton, R. S., and Barto, A. G. (2018). *Reinforcement Learning: An Introduction* (2nd ed.). Manuscript in preparation.
- Thomas, P. and Brunskill, E. (2016). Data-Efficient Off-Policy Policy Evaluation for Reinforcement Learning. *Proceedings of The 33rd International Conference on Machine Learning*, in PMLR 48:2139-2148.
- van Seijen, H., van Hasselt, H., Whiteson, S., and Wiering, M. (2009). A theoretical and empirical analysis of expected sarsa. In *Proceedings of the IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, pp. 177-184.
- Watkins, C. J. C. H. (1989). *Learning from Delayed Rewards*. PhD thesis, Cambridge University.

---

# The Indian Buffet Hawkes Process to Model Evolving Latent Influences

---

Xi Tan<sup>1</sup>, Vinayak Rao<sup>2</sup>, and Jennifer Neville<sup>1,2</sup>

<sup>1</sup>Department of Computer Science and <sup>2</sup>Department of Statistics  
Purdue University  
West Lafayette, IN 47907

## Abstract

Temporal events in the real world often exhibit reinforcing dynamics, where earlier events trigger follow-up activity in the near future. A canonical example of modeling such dynamics is the Hawkes process (HP). However, previous HP models do not capture the rich dynamics of real-world activity—which can be driven by multiple latent triggering factors shared by past and future events, with the latent features themselves exhibiting temporal dependency structures. For instance, rather than view a new document just as a response to other documents in the recent past, it is important to account for the factor-structure underlying all previous documents. This structure itself is not fixed, with the influence of earlier documents decaying with time. To this end, we propose a novel Bayesian nonparametric stochastic point process model, the Indian Buffet Hawkes Processes (IBHP), to learn multiple latent triggering factors underlying streaming document/message data. The IBP facilitates the inclusion of multiple triggering factors in the HP, and the HP allows for modeling latent factor evolution in the IBP. We develop a learning algorithm for the IBHP based on Sequential Monte Carlo and demonstrate the effectiveness of the model. In both synthetic and real data experiments, our model achieves equivalent or higher likelihood and provides interpretable topics and shows their dynamics.

## 1 INTRODUCTION

Temporal activity in real applications exhibit rich dynamics, with past events influencing the future through multiple structured latent factors. For example, the ideas in a research paper may be derived from multiple existing works in the literature, each of which contributes one

or more factors, with only their combination serving to trigger the event. Similarly, a conversation among individuals may heat up or cool down due to the topics being discussed (e.g., politics vs. weather). Communications via email or on social media platforms like Facebook or Twitter may exhibit analogous dynamics. In addition, individual checkin data on platforms like Foursquare or Yelp may depend on combinations of characteristics and activities from previous visited locations. Finally in biological data, pathways are often only activated when a set of genes is expressed together.

Latent feature models (both parametric and nonparametric) have found wide application in settings where exchangeability holds. A canonical model from Bayesian nonparametrics is the Indian Buffet process (IBP). While there has been some work towards relaxing exchangeability assumptions to allow for temporal dynamics, modeling the full richness of interactions remains an open challenge. Our main contribution in this work is a framework that facilitates the modeling of temporal dynamics through a combination of ideas from the IBP with those of Hawkes processes (HP).

In recent years, Hawkes processes [11, 10, 12, 15, 22] have become a popular modeling choice to capture such temporal dynamics [4, 18, 14, 19, 13, 7, 17, 8, 16, 21]. As we outline later, the standard Hawkes process has a number of limitations centering around the fact that each event is triggered by a single observation instead of possibly multiple events and/or factors. To this end, we propose a novel Bayesian nonparametric stochastic point process model, the Indian Buffet Hawkes Processes (IBHP), that synergizes ideas between the IBP and the HP. The contributions of our work include:

1. The use of the IBP to add multiple triggering factors to the HP, which helps to better model dynamics and improves interpretation.
2. Embedding the temporal information from the HP into the IBP to drive the latent factor estimation,

which expands its capability to model factor evolution over time.

3. Developing an efficient and scalable learning algorithm for the IBHP model, based on Sequential Monte Carlo (SMC).
4. Demonstrating the effectiveness of the IBHP on both synthetic and four real-world datasets, where we also show how our framework enables the construction of more flexible (e.g., multi-event) triggering rules.

## 2 PRELIMINARIES

Formally, we are given a sequence of  $N$  observations  $y_n = \{t_n, \mathcal{T}_n\}$ ,  $n = 1, \dots, N$ . For the  $n$ th event,  $t_n$  is the time of occurrence, and  $\mathcal{T}_n$  represents observed attributes attached to it. Since our focus is mostly on settings where events are messages, we will refer to attributes as text. We will model the event times  $t_n$  and event text  $\mathcal{T}_n$  as realizations of a process whose states depend on a hidden state variable  $\mathbf{z}_n$ , summarizing the past observations  $y_{1:(n-1)}$ . Before outlining our model, we first review existing work related to our problem.

### 2.1 HAWKES PROCESSES (HP)

Hawkes processes (HP) [11, 10, 12, 15, 22] are self-exciting point processes [5] where earlier events have a time-decaying influence on future events. Parametrized by a base rate  $\gamma$  and a non-negative triggering kernel  $\kappa(\cdot)$  (the latter models the contribution of each past observation), the rate function at time  $t$  can be written as:

$$\lambda(t) = \gamma + \int_0^t \kappa(t-s) dN(s) \quad (1)$$

where  $N(s)$  is the number of observations within  $[0, s)$ . Given the rate function  $\lambda(t)$  and observation history  $\mathcal{H}_{(0,T]} = (t_1, \dots, t_n)$ , the likelihood function of a Hawkes process is:

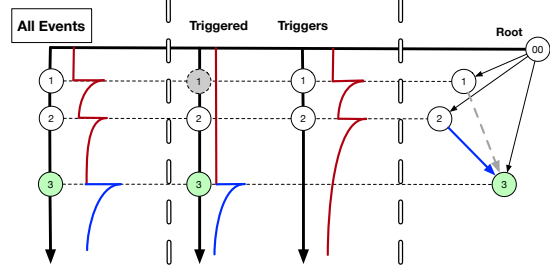
$$\mathcal{L}(\mathcal{H}) = \exp\{-\Lambda(0, T)\} \prod_{i=1}^n \lambda(t_i) \quad (2)$$

where  $\Lambda(0, T) = \int_0^T \lambda(t) dt$  is the cumulative rate. The events in a standard HP are triggered by a single event at a time (see Figure 1).

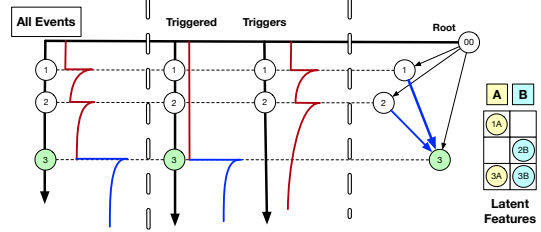
### 2.2 INDIAN BUFFET PROCESSES (IBP)

The Indian Buffet Process (IBP) [9] is a Bayesian non-parametric prior over an infinite dimensional binary matrix whose columns represent exchangeable factors underlying observations. Suppose there are  $N$  customers (observations) arriving sequentially in a restaurant with infinite number of dishes (factors). Each customer is assigned dishes as follows:

- The first customer comes in and helps herself to  $\text{Poisson}(\alpha)$  dishes.



(a) A Hawkes process with single triggers.



(b) A Hawkes process with multiple triggers.

Figure 1: HP with single and multiple triggers. In (a), #3 is triggered by a single event #1, while in (b) it is triggered by #1 and #2. The triggering kernels can be quite different depending on how the triggering has happened. HP with single triggers would fail to model influences from both #1 and #2 at the same time, as shown in (b).

- When the  $n^{\text{th}}$  customer arrives, they independently choose each existing dish with probability  $m_k/n$ , where  $m_k$  is the number of customers that have already sampled dish  $k$  (the popularity of the dish).
- In addition, they sample  $\text{Poisson}(\alpha/n)$  new dishes.

The IBP has several distinctive features: 1) Each observation can have multiple factors; 2) The number of factors grows non-parametrically depending on the size of the dataset; 3) The probability of adding new factors decreases – since the number of new factors follows  $\text{Poisson}(\alpha/n)$  which decreases as  $n$  increases; 4) The rows are exchangeable, with the row sums distributed  $\text{Poisson}(\alpha)$ .

We write the binary feature matrix as  $\mathbf{C}$ . The conditional probability that element  $c_{ik} = 1$  is given by

$$P(c_{ik} = 1 | \mathbf{c}_{-i,k}) = \frac{m_{-ik}}{N} \quad (3)$$

where  $\mathbf{c}_{-ik}$  is the  $k^{\text{th}}$  column without considering the  $i^{\text{th}}$  observation, and  $m_{-ik}$  is the sum of  $\mathbf{c}_{-ik}$ . We need only condition on  $\mathbf{c}_{-ik}$  rather than including other columns because the columns of the matrix are generated independently under this prior. In a Bayesian framework with observations  $\mathbf{X}$ , the posterior can be written as:

$$P(c_{ik} = 1 | \mathbf{C}_{-ik}, \mathbf{X}) \propto P(\mathbf{X} | \mathbf{C}) P(c_{ik} = 1 | \mathbf{c}_{-ik}) \quad (4)$$

where  $P(\mathbf{X} | \mathbf{C})$  is the data likelihood.



### 3 MODEL

Our proposed model, the IBHP, can be viewed as a non-parametric latent state space model, where past events  $y_n = \{t_n, \mathcal{T}_n\}$  influence future observations through latent state variables  $\mathbf{z}_n = \{\mathbf{K}_n, \mathbf{V}_n\}$  (described below). The  $\mathbf{z}_n$ 's summarize information about the past, and themselves evolve following dynamics based on the IBP. Algorithmically, the generative process (see Algorithm 1 for the pseudocode and Figure 2 for an illustrative example) can be described in the following three steps.

#### 3.1 INITIALIZATION ( $\mathcal{M}, \Pi, \Theta$ )

To setup the model, we first specify a triplet  $\mathcal{M} = \{\mathcal{S}, D, L\}$ , with  $\mathcal{S}$  representing the vocabulary of all possible words in the observations,  $D$  representing the length of each document (for simplicity we assume all are equal), and  $L$  representing the number of basis kernels. We also require a pair of hyper parameters  $\Pi = \{\mathbf{w}_0, \mathbf{v}_0\}$  for the priors of the kernel and word distribution weights.

Each latent factor influences the content of future events through a set of *dictionary weights*, which are used to generate text. We write  $\mathbf{v}_k$  for the vector of weights over the words in the vocabulary for the  $k^{\text{th}}$  factor – a length  $|\mathcal{S}|$  vector which sums to one. The weights  $\mathbf{v}_k$  are sampled from a Dirichlet prior (with hyper parameter  $\mathbf{v}_0$ ) whenever a new factor is created (see later). Each latent factor also influences the timing of future events through a *triggering kernel*, and we assume each kernel is a linear combination of a set of  $L$  bases. Throughout, we assume  $L$  exponential basis kernels:

$$\gamma_l(\delta) = \beta_l e^{-\frac{\delta}{\tau_l}}, \quad l = 1, \dots, L. \quad (5)$$

This requires a set of parameters  $\{(\beta_l, \tau_l)\}$ , each of which captures a distinct type of excitation pattern. A binary matrix  $\mathbf{C}$  indicates which factors are associated with each observation. The  $k^{\text{th}}$  factor kernel for the  $i^{\text{th}}$  observation  $\kappa_{ik}$  is a weighted sum of the  $L$  basis kernels:

$$\kappa_{ik}(\delta | \mathbf{w}_k, c_{ik}) = \begin{cases} \sum_{l=1}^L w_{kl} \cdot \gamma_l(\delta), & \text{if } c_{ik} = 1 \\ 0, & \text{if } c_{ik} = 0 \end{cases} \quad (6)$$

The weights  $w_{kl}$ , loadings of the basis kernels for the factors, are sampled from a Dirichlet prior (with hyper parameter  $\mathbf{w}_0$ ) whenever a new factor is created (see later). Thus, immediately after an event (when  $\delta = 0$ ), there is a jump in the event rate with amplitude equal to  $\kappa_{ik} = \mathbf{w}_k^T \beta$ . Observations with the same factor share the factor kernel. We write the model parameters as  $\Theta = \{\lambda_0, \{\beta_l\}, \{\tau_l\}\}$ , where  $\lambda_0$  is a base-rate at which events happen spontaneously.

#### 3.2 THE FIRST EVENT ( $\mathcal{M}, \Theta \rightarrow \mathbf{z}_1 \rightarrow \mathbf{y}_1$ )

To generate the observation  $\mathbf{y}_1 = \{t_1, \mathcal{T}_1\}$ , we first sample the auxiliary variables  $\mathbf{c}_1$  and  $\mathbf{w}_{1:K}$ . The *factor label variable*  $\mathbf{c}_1$  is a binary vector of length  $K$ , where  $K \sim \text{Poisson}(\lambda_0)$  is the number of existing factors.  $c_{nk} = 1$  implies that the  $n^{\text{th}}$  observation has a label of factor  $k$ . Set  $c_{1k} = 1$  for  $k = 1, \dots, K$ . The *kernel weights*  $\mathbf{w}_k$  is a vector of weights for the  $k^{\text{th}}$  factor to load the basis kernels. Each  $\mathbf{w}_k$  is of length  $L$  (the number of basis kernels), and sums to one. Sample  $\mathbf{w}_k \sim \text{Dir}(\mathbf{w} | \mathbf{w}_0)$  for  $k = 1, \dots, K$ .

Given the values of  $\mathbf{c}_1$  and  $\mathbf{w}_{1:K}$ , we can sample the associated latent variables  $\mathbf{z}_1 = \{\mathbf{K}_1, \mathbf{V}_1\}$ . Define the  $1 \times K$  IBHP matrix  $\mathbf{K}_1$ , whose rows are  $\kappa_{1k}$ , with values  $\mathbf{w}_k^T \beta$  (see Equation 6) – since  $\delta = 0$ . For  $n = 1$ , sample  $\mathbf{v}_k \sim \text{Dir}(\mathbf{v} | \mathbf{v}_0)$  for  $k = 1, \dots, K$ , and define the  $|\mathcal{S}| \times K$  matrix  $\mathbf{V}_1$ , whose columns are  $\mathbf{v}_k$ . Conditioned on these state variables  $\mathbf{z}_1$ , we sample the first observation  $\mathbf{y}_1 = \{t_1, \mathcal{T}_1\}$ : The *time stamp*  $t_1$  is sampled from a Poisson process with rate  $\lambda_0$ ; and the *text*  $\mathcal{T}_1$  is sampled from  $\text{Multi}(D, \sum_{k=1}^K \mathbf{v}_k / K)$ , where the weight parameter is the averaged factor weight of the first observation.

#### 3.3 FOLLOW-UP EVENTS ( $\mathbf{z}_{n-1} \rightarrow \mathbf{z}_n \rightarrow \mathbf{y}_n$ )

For a new event, we first decide its associated factors, and sample its time stamp afterwards. Conditioning on  $\mathbf{z}_{n-1}$ , suppose there are  $K$  existing factors, each of which can be represented by an independent Hawkes process. At time  $t_{n-1}$ , the *factor rate* for the  $k^{\text{th}}$  factor is:

$$\lambda_k(t_{n-1}) = \sum_{i=1}^{n-1} \frac{\kappa_{ik}(t_{n-1} - t_i)}{\|\kappa_i\|_0} \quad (7)$$

As with the generation of the initial event, follow-up events ( $n > 1$ ) are also generated by two steps. First, we sample the auxiliary variables  $\mathbf{c}_n$  and set  $\mathbf{w}$  and  $\mathbf{v}$  for any newly generated factors. The first  $K$  components of the *factor label variable*  $\mathbf{c}_n$  is sampled independently from a Bernoulli distribution with probability parameter

$$p_k = \frac{\lambda_k(t_{n-1})}{\lambda_0/K + \lambda_k(t_{n-1})} \quad (8)$$

Meanwhile,  $K^+$  new factors are created by setting  $c_{nk'} = 1$ , for  $k' = K + \{1, \dots, K^+\}$ , where

$$K^+ \sim \text{Poisson} \left( \frac{\lambda_0}{\lambda_0 + \sum_{k=1}^K \lambda_k(t_{n-1})} \right) \quad (9)$$

If  $\kappa$  are binary, which is the case in IBP, and  $\lambda_0 = 1$ , then the mean of  $K^+$  becomes  $1/n$  and  $p_k = (n-1)/n$ , which reduces to the case of IBP with parameter 1:

$$\sum_{k=1}^K \sum_{i=1}^{n-1} \frac{\kappa_{ik}(t_{n-1} - t_i)}{\|\kappa_i\|_0} = \sum_{i=1}^{n-1} \frac{\|\kappa_i\|_0}{\|\kappa_i\|_0} = n-1 \quad (10)$$

For each new factor  $k'$ , we draw from the corresponding priors for  $\mathbf{w}_{k'} \sim \text{Dir}(\mathbf{w}|\mathbf{w}_0)$  and  $\mathbf{v}_{k'} \sim \text{Dir}(\mathbf{v}|\mathbf{v}_0)$ .

Next, we decide the hidden state variables  $\mathbf{z}_n = \{\mathbf{K}_n, \mathbf{V}_n\}$ .  $\mathbf{V}_n$  is constructed by simply adding columns for the  $\mathbf{v}_{k'}$  for newly sampled factors to  $\mathbf{V}_{n-1}$ .  $\mathbf{K}_n$  is constructed by first updating  $\mathbf{K}_{n-1}$  with respect to the new lag time  $\delta = t_n - t_i$ . This step is done *symbolically*, since we do not know  $t_n$  yet. Then we add the rows  $\kappa_{ik'}$  for the newly sampled event based on Equation 6 with  $\delta = 0$ . We emphasize that  $K_n(t_n) : \mathbf{R}^+ \rightarrow \mathbf{R}^{n \times (K+K')}$  at this moment is a symbolic function of  $t_n$ .

Conditioned on these state variables  $\mathbf{z}_n$ , we sample the  $n^{\text{th}}$  observation  $\mathbf{y}_n = \{t_n, \mathcal{T}_n\}$ : The *time stamp*  $t_n$ , depending on its related factors, is sampled from a Poisson process with rate

$$\lambda(t_n) = \sum_{\kappa_{nk} \neq 0} \lambda_k(t_n) = \sum_{\kappa_{nk} \neq 0} \sum_{i=1}^n \frac{\kappa_{ik}(t_n - t_i)}{\|\boldsymbol{\kappa}_i\|_0} \quad (11)$$

The overall rate of IBHP, however, includes the base rate and other factors too:

$$\bar{\lambda}(t_n) = \lambda_0 + \lambda(t_n) + \sum_{\kappa_{nk}=0} \lambda_k(t_n) \quad (12)$$

Now, at this point, since  $t_n$  is known, we can proceed to compute the actual values of  $\mathbf{K}_n$ . Finally, we sample the *dictionary text*  $\mathcal{T}_n$  from  $\text{Multi}(D, \sum_{\kappa_{nk} \neq 0} \mathbf{v}_k / \|\boldsymbol{\kappa}_n\|_0)$ , where the weight parameter is the averaged of all  $\|\boldsymbol{\kappa}_n\|_0$  factor weights associated with the  $n^{\text{th}}$  observation.

## 4 POSTERIOR INFERENCE

### 4.1 SMC FOR IBHP

Sequential Monte Carlo [6] (SMC) methods are powerful and flexible tools for posterior inference in time-series models such as ours. Here, we adapt particle filtering methods to our set up, allowing us to scale our model to large-data regimes. The idea here is to represent the state of the system at any time (from 1 to  $N$ ) with a set of  $F$  particles. We build on ideas from [20], extending them to our more structured setting.

The idea at a high level is to propagate each particle forward by one time step according to the prior, and then reweight each particle by how “compatible” it is with the observation at that time. If the effective number of particles is small (according to their weights), then the algorithm resamples the particles with replacement. Our algorithm to learn the IBHP factors and model parameters can be described as follows (see Algorithm 2 for the pseudocode):

*A. Initialize Particle Weights.* The particle weights are initialized uniformly:  $u_1^f = \frac{1}{F}$ , for  $f = 1, \dots, F$ . Then for each time step  $i = [1 \dots N]$ , we do the following:

---

### 1. Initialization:

- Model specifications:  $\mathcal{M} = \{L, D, \mathcal{S}\}$ ;
- Model hyper parameters:  $\boldsymbol{\Pi} = \{\mathbf{w}_0, \mathbf{v}_0\}$ ;
- Model parameters:  $\boldsymbol{\Theta} = \{\lambda_0, \{\beta_l, \tau_l\}\}$ ;

### 2. Generate the First Event:

- Set  $c_{1,1:K} = 1$ , where  $K \sim \text{Poisson}(\alpha_0)$ ;
- Sample  $\mathbf{w}_k \sim \text{Dir}(\mathbf{w}|\mathbf{w}_0)$  and set  $\boldsymbol{\kappa}_1$ ;
- Sample  $\mathbf{v}_k \sim \text{Dir}(\mathbf{v}|\mathbf{v}_0)$ ;
- Sample  $t_1 \sim \mathcal{PP}(\lambda_0)$ ;
- Sample  $\mathcal{T}_1 \sim \text{Multi}\left(D, \sum_{\kappa_{1k} \neq 0} \mathbf{v}_k / \|\boldsymbol{\kappa}_1\|_0\right)$

### 3. Generate Follow-up Events:

**for**  $n = 2, \dots, N$  **do**

- Sample  $\mathbf{c}_n$  according to Equations 8 and 9.
- Sample  $\mathbf{w}_{k'} \sim \text{Dir}(\mathbf{w}|\mathbf{w}_0)$  and set  $\boldsymbol{\kappa}_n$ ;
- Sample  $\mathbf{v}_{k'} \sim \text{Dir}(\mathbf{v}|\mathbf{v}_0)$ ;
- Sample  $t_n \sim \mathcal{PP}(\lambda(t_n))$  by Equation 11.
- Sample

$$\mathcal{T}_n \sim \text{Multi}\left(D, \sum_{\kappa_{nk} \neq 0} \mathbf{v}_k / \|\boldsymbol{\kappa}_n\|_0\right).$$

**end**

---

**Algorithm 1:** Generative process of IBHP.

*B. Sample Particles.* According to [20], our particles  $\tilde{\mathbf{z}}_i^f = \{\tilde{\mathbf{K}}_i^f, \tilde{\mathbf{V}}_i^f\}$  are sampled based on the conditional distributions  $p(\mathbf{z}_i|\mathbf{z}_{i-1})$  described in Section 3.3.

*C. Sample Model Parameters.* Since the posterior of the model parameter  $\boldsymbol{\Theta} = \{\lambda_0, \{\beta_l\}, \{\tau_l\}\}$  is proportional to the product of its prior and the data likelihood described in Equation 2 and Section 3, we can first sample from its prior, and then use the product of the prior and the HP data likelihood as weights of the samples to approximate the posterior [8]. We update the triggering kernels using the new parameters.

*D. Update Particle Weights.* The importance weight is the ratio between the true posterior and the proposal distribution. Since we use the prior as the proposal, we update the particle weights by  $u_i^f = u_{i-1}^f p(\mathbf{y}_i|\tilde{\mathbf{z}}_i^f, \boldsymbol{\Theta})$  and then normalize them to  $u_j^f = u_j^f / (\sum_{f=1}^F u_j^f)$ .

*E. Resample Particles.* If the effective number of particles is too small, we resample with replacement  $F$  particles from the existing ones with the normalized weights.

### 4.2 COMPLEXITY AND SCALABILITY

The SMC algorithm for the IBHP is easy to implement and scalable. Due to the sequential updating strategy, the time complexity of this algorithm is  $\mathcal{O}(NF)$ , where  $N$  is the number of observations and  $F$  is the number of particles. We will demonstrate and discuss the effectiveness of the algorithm in the experiment section in more detail.

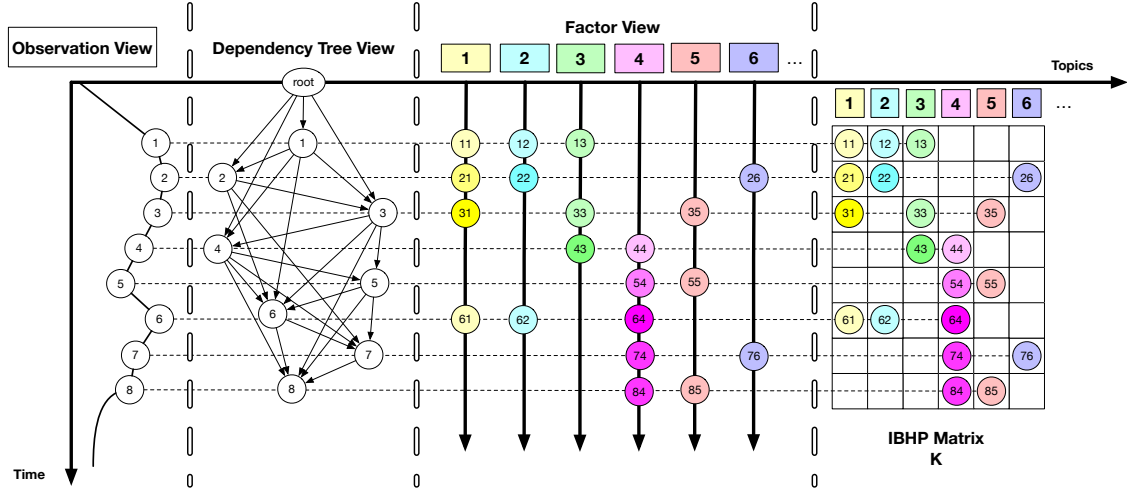


Figure 2: An example of IBHP. In this IBHP realization, the first 8 observations created 6 factors. Each factor has a distinctive color, and color intensities represent instantaneous factor popularities. An observation may be labeled with multiple factors, and are colored in its decomposed factor view accordingly. The dependency tree describes the related events for each observation, where the directed arrows indicate dependency relations. The rate for any *observation* is the aggregation of all its *related factor* rates (see Equation 11), whereas the overall rate at any *time* is the sum of *all factor* rates – so the overall rate can be excited by one observation multiple times through different factors. The overall rate is represented by its height relative to the reference time line. See Section 6.1 for more details.

Initialize the  $F$  uniform particle weights.

**for** each observation  $\mathbf{y}_i = \{t_i, \mathcal{T}_i\}$ ,  $i = 1, \dots, n$   
**do**

**for** each particle  $\mathbf{z}_i^f = \{\mathbf{K}_i, \mathbf{V}_i\}$  of  
    observation  $\mathbf{y}_i$ ,  $f \in \{1, \dots, F\}$  **do**

- Sample the auxiliary variables  $\mathbf{w}_i, \mathbf{c}_i$  and latent factor particles  $\mathbf{z}_i^f = \{\mathbf{K}_i, \mathbf{V}_i\}$ .
- Sample the model parameters  $\Theta = \{\lambda_0, \{\beta_l\}, \{\tau_l\}\}$ .
- Update the triggering kernels.
- Update the particle weights  $\mathbf{u}_i^f$ .

**end**

Normalize the particle weights.

**if**  $\|\mathbf{u}_i\|_2^{-2} < \text{threshold}$ , *i.e.*, the effective number of particles is too low **then**  
    Resample particles with replacement based on the particle weights.

**end**

**end**

**Algorithm 2:** SMC inference algorithm for IBHP.

## 5 RELATED WORK

The idea of considering nonparametric Bayesian models with temporal point processes in a unified framework has been popular in recent years. For example, [4] proposed

a Bayesian nonparametric model that utilizes the Chinese Restaurant Processes (CRP) as a prior for the clusters among individuals, whose rates of communications are modeled by HP. [18] used a similar idea but further extended the model by modeling the jump sizes of HP using Gaussian Processes (GP). HP models with various generalizations of a CRP, such as the distance dependent CRP (ddCRP) [14], the nested CRP (nCRP) [19], and the Chinese Restaurant Franchise Processes (CRFP) [13], have also been explored.

Other attempts have been made by borrowing the ideas from Deep Learning. For example, [7] proposed a model to view the intensity function of a temporal point process as a nonlinear function of the history, and use recurrent neural networks to automatically learn a representation of the influences from the event history. [17] modeled streams of events by constructing a neurally self-modulating multivariate point process where the intensities of multiple event types evolve based on a continuous-time LSTM. Lastly, [21] considered the use of latent factors in HP models to represent dependencies among instances that influence reciprocity over time. But the work focused on modeling static factors of homophily and reciprocity in social networks and not the evolution of factors over time.

Perhaps the closest works to our model are [8] and [16]. In [8], the authors proposed a Dirichlet Hawkes Processes (DHP) model that combines the CRP with HP in a

unified framework, where the cluster assignment in CRP is driven by the intensities of HP. [16] further developed this in their Hierarchical Dirichlet Hawkes Processes (HDHP) model by replacing the CRP with a CRFP that is capable of modeling steaming data for multiple users. However, there are several major distinctions compared to our IBHP: 1) In both the DHP and HDHP models, events are triggered by single factors, while in our IBHP, multiple latent triggering factors are introduced; 2) the form of the triggering kernels do not depend on history events, and in contrast, our IBHP model is more flexible to be able to adopt non-additive triggering rules to learn different perspectives of the observed data. We will compare our model to [8] and [16] next.

## 6 EXPERIMENTS

We compare IBHP with three methods from the previous section: the vanilla Hawkes process (HP), the Dirichlet Hawkes (DHP; [8]), and the Hierarchical Dirichlet Hawkes (HDHP; [16]). We evaluate the models on both synthetic and real-world data.

### 6.1 SYNTHETIC DATASETS

The purpose of our synthetic-data experiments is twofold: 1) to understand the identifiability of our model and the accuracy of our SMC algorithm when the true data generation process satisfies the model assumptions, and 2) to understand the effects of misspecification.

Our setup is as follows. The Hawkes process base rate is  $\lambda_0 = 2$ . For the basis kernels, we use:  $\gamma_1(\delta) = e^{-\delta/0.3}$ ,  $\gamma_2(\delta) = 2e^{-\delta/0.2}$ ,  $\gamma_3(\delta) = 3e^{-\delta/0.1}$ .  $\gamma_1$  has the smallest jump but also the largest time-scale; at the other extreme,  $\gamma_3$  has the largest jump with a fast decay-parameter.  $\gamma_2$  might be used to model ‘regular’ events, while  $\gamma_1$  and  $\gamma_3$  are for non-urgent and urgent ones respectively. We construct the dictionary  $\mathcal{S}$  from the top 1000 words from the NIPS dataset [2], and the document lengths are set to  $D = 20$ . The hyperparameters, which are not to be estimated, are set as  $\mathbf{w}_0 = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ ,  $\mathbf{v}_0 = (\frac{1}{1000}, \dots, \frac{1}{1000})$ . We generate  $N = 1000$  observations with this setup, and use the first 80% of the dataset for training, and the last 20% for testing. For each SMC iteration, we use 10 particles, and report averages and error bars based on 10 runs with different random seeds.

**A. Parameter learning and prediction.** Experiments A1 and A2 shown in Table 1 are the parameter estimates and the log-likelihoods over training and test datasets. Our model outperforms other models in terms of predictive log-likelihood. This demonstrates two points. Firstly, our SMC algorithm is able to accurately recover the underlying model parameters. Furthermore, estimating parameters for the misspecified models on this dataset is fair, since they have the same interpretation. Thus for

instance, our results tell us that fitting a Hawkes model that does not include multiple triggering factors results in a significant overestimation of the base rate  $\lambda_0$ : a result that one might have expected.

A1. Parameter Estimation			
Parameter Values	$\lambda_0$	$\{\beta_i\}$	$\{\pi_i\}$
	2	1,2,3	0.3,0.2,0.1
<b>IBHP</b>	<b>1.8</b>	<b>0.92, 1.63, 2.71</b>	<b>0.33, 0.18, 0.09</b>
HDHP	3.3	0.77, 4.56, 6.11	3.75, 3.20, 2.94
DHP	2.9	0.83, 5.72, 5.83	1.21, 1.58, 1.28
HP	5.4	2.25, 4.38, 3.01	0.73, 2.54, 3.56
A2. Log-likelihoods			
	Training		Test
<b>IBHP</b>	<b>318.52</b>		<b>47.68</b>
HDHP	192.74		12.23
DHP	201.96		11.78
HP	81.68		6.18
B. Learn Latent State Variables ( $K = 5, 10, 20$ )			
	Jaccard(K)		1 - Hellinger(V)
<b>IBHP</b>	<b>0.83, 0.81, 0.77</b>		<b>0.79, 0.73, 0.68</b>
HDHP	0.56, 0.40, 0.35		0.51, 0.44, 0.29
DHP	0.61, 0.42, 0.38		0.64, 0.41, 0.36

Table 1: Model comparison over the synthetic datasets.

**B. Learn latent state variables.** Table 1 part B focuses on learning the latent state variables. Now, rather than generating data from our nonparametric model, we fix  $K = 5, 10, 20$  in the data-generating process, and then compare these with our nonparametric estimates using two metrics: the Jaccard Index to compare the *binary* matrices  $\mathbf{C}$  and the Hellinger distance for  $\mathbf{V}$ . A first complication is that these matrices need not have the same number of columns, and so for each comparison, we pad the smaller matrix with zero-columns to facilitate comparison. A bigger challenge is a ‘label-switching’ issue that arises since column permutations do not effect the quality of the estimates. To overcome this, after matching dimensions, we greedily match columns, and then compute scores. We point out that padding with zeros favors the alternative methods, since their solutions have many zeros; nevertheless, our model still gives the best Jaccard scores as well as Hellinger distances (we actually report *complementary* Hellinger distances (viz. one minus the actual distance), so that large numbers imply better performance for both statistics. As before, our results demonstrate the insufficiency of the alternate models and justifies the need for multiple factors.

**C. The effects of base rate and basis kernels.** The base rate  $\lambda_0$ , together with the evolving kernels, control the dynamics of latent factors. In Table 2 part C, we see that increasing  $\lambda_0$  increases the average number of factors per observation increases—more strongly violating the *single* factor assumption of competing methods. This observation is also accompanied by a widening performance gap between our model and the alternatives.

	$\lambda_0$	Topics	Jaccard	Hellinger	Test
IBHP	4	9.01	0.79	0.75	50.21
	8	12.28	0.72	0.69	68.37
	16	28.33	0.64	0.61	72.07
HDHP	4		0.32	0.40	43.78
	8	1	0.28	0.38	51.06
	16		0.31	0.26	50.79
DHP	4		0.29	0.37	41.67
	8	1	0.33	0.31	49.18
	16		0.27	0.28	52.33

Table 2: Effects of model specifications.

**D. The effects of triggering rule.** In Equation 11, the event rate depends on the rates of the underlying factors in an additive manner. We can allow more flexible triggering rules by allowing richer interactions among factor dynamics. For example, we define a “double-sharing” triggering rule as follows: *trigger a jump in the rate function only when two or more factors are shared with a previous observation.* Thus Equation 11 becomes:

$$\lambda(t_n) = \sum_{\kappa_{nk} \neq 0} \left[ \sum_{i=1}^{n-1} \frac{\kappa_{ik}(t_n, t_i)}{\|\kappa_i\|_0} + \phi \left( \frac{\kappa_{ik}(t_n, t_n)}{\|\kappa_n\|_0} \right) \right] \quad (13)$$

where  $\phi = 0$  if the rule is not triggered—there is no “jump”, otherwise  $\phi = \mathbf{w}_k^T \beta / \|\kappa_n\|_0$ —there is a “jump”. We sketch this out in Figure 3.

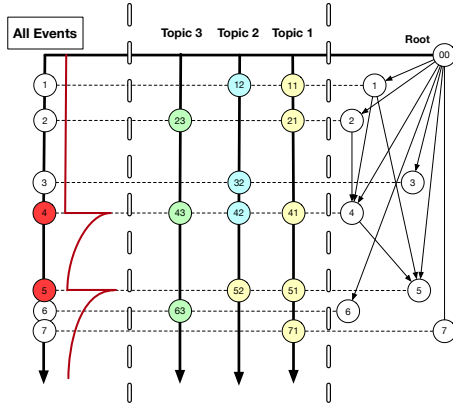


Figure 3: IBHP with “double-sharing” rule. Obs. 2 does not trigger a “jump” because no previous observations share more than two factors with it. However, obs. 4 triggers *two* jumps because it shares two factors with obs. 1 (factor 1 and 2), and two with obs. 2 (factor 1 and 3).

Incorporating such nonlinearities result in dynamics that are significantly different from the additive setup: this is evidenced in Table 3, where the simpler additive version the IBHP now has a degraded score. There are numerous variations to our simple “double-sharing” rule that are relevant across a variety of situations.

	Additive Model	Double-sharing Model
IBHP	15.38 $\pm 3.82$	20.82 $\pm 3.23$
HDHP	8.97 $\pm 4.07$	12.36 $\pm 3.18$
DHP	8.26 $\pm 3.19$	10.17 $\pm 3.20$
HP	4.98 $\pm 3.61$	5.04 $\pm 3.22$

Table 3: Model comparison with “double-sharing” data.

## 6.2 REAL DATASETS

The purpose of our real data experiments is threefold: 1) to verify that the multiple triggering factors in IBHP are indeed relevant to real applications, 2) to demonstrate that our SMC inference algorithm is scalable for real-world datasets, and 3) to use our IBHP model to present meaningful findings, both quantitative and qualitative. We consider four different datasets: **Facebook Dataset**. This data contains Facebook message communications among 20,603 individuals. We pick the top 10 most connected individuals (based on the number of friends), and add in their one-hop and two-hop friends. This results in a total of 376 individuals. **NIPS Dataset [2]**. The Kaggle NIPS dataset contains the title, authors, abstracts, and extracted text for all 7241 NIPS papers from the first 1987 conference to the current 2017 conference. This dataset is different in that it contains rich message information; however the number of time-points is just 30. **Santa Barbara Corpus Dataset [3]**. This is a standard dataset used for applications involving Hawkes processes. We use conversation #33, a lively family discussion which centers around a disagreement that an individual, Jennifer, is having with her mother, Lisbeth. **Enron Email Dataset[1]**. The Enron dataset contains about half a million email messages communicated among about 150 senior managers of the Enron corporation. We pick the longest thread of emails.

For each experiment, we use the first 80% of the dataset as training set, the next 10% as validation set, and the last 10% as test set. We train our model on training sets with different hyperparameters, then pick the best one based on their performances on the validation set, and use this model to report performances on the test set. The reported values are based on ten runs with different random number seeds. The dictionary  $\mathcal{S}$  is all the unique words in the dataset; the document length  $D_n$  is counted from each observed text  $\mathcal{T}_n$ ; and we use the three ( $L = 3$ ) exponential basis kernels defined in Equation 5.

**A. Predictive log-likelihood.** The log-likelihoods in Table 4 show that for three of four datasets, our model outperforms the alternatives. The performance gaps exhibit a range of values. On the NIPS dataset, our model shows a massive improvement over the competition, while there is no significant improvement for the Enron dataset. The numbers in parentheses, giving the average number of

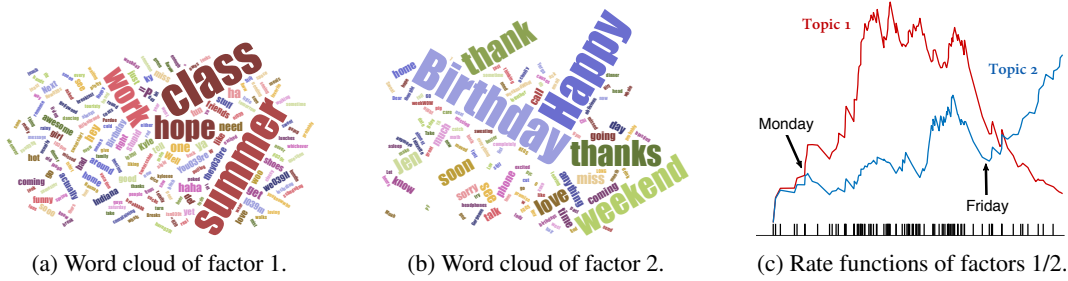


Figure 4: **FB dataset**. Topic dynamics.

topics associated with each message, provides a partial explanation. For the Enron dataset, this number is just two, suggesting that there is limited benefit from modeling multiple factors, and that the simpler HDHP model may be more appropriate. For the NIPS dataset, this number is about 10, explaining the gap in performance.

FB Dataset (average # factors = 4.19)			
	Training	Validation	Test
<b>IBHP</b>	<b>1822</b> $\pm 96$	<b>219</b> $\pm 10$	<b>277</b> $\pm 11$
HDHP	1083 $\pm 88$	123 $\pm 10$	133 $\pm 10$
DHP	1058 $\pm 90$	144 $\pm 9$	200 $\pm 14$
HP	782 $\pm 75$	62 $\pm 7$	69 $\pm 7$
NIPS Dataset (average # factors = 10.21)			
	Training	Validation	Test
<b>IBHP</b>	<b>8378</b> $\pm 172$	<b>913</b> $\pm 23$	<b>1012</b> $\pm 28$
HDHP	3229 $\pm 169$	216 $\pm 12$	191 $\pm 11$
DHP	2018 $\pm 164$	203 $\pm 10$	202 $\pm 10$
HP	390 $\pm 48$	49 $\pm 8$	40 $\pm 7$
SB Dataset (average # factors = 6.52)			
	Training	Validation	Test
<b>IBHP</b>	<b>520</b> $\pm 62$	<b>187</b> $\pm 12$	<b>137</b> $\pm 9$
HDHP	132 $\pm 9$	32 $\pm 6$	34 $\pm 6$
DHP	169 $\pm 10$	51 $\pm 7$	78 $\pm 9$
HP	96 $\pm 10$	15 $\pm 4$	23 $\pm 4$
Enron Dataset (average # factors = 2.17)			
	Training	Validation	Test
<b>IBHP</b>	<b>2602</b> $\pm 101$	<b>313</b> $\pm 12$	<b>381</b> $\pm 12$
HDHP	2322 $\pm 117$	203 $\pm 10$	392 $\pm 11$
DHP	2639 $\pm 118$	268 $\pm 11$	339 $\pm 12$
HP	729 $\pm 92$	28 $\pm 5$	19 $\pm 5$

Table 4: Model comparisons over the real datasets.

**B. Latent structure vs. dynamics.** The rich structure of the NIPS dataset is balanced by its simple temporal structure just with 30 time points. This raises the question: how much of our models performance is due to the latent structure incorporated into our modeling framework, and how much is due to temporal dynamics of this structure. To study this more carefully, we shuffle the publication years (documents published in the same year remain together, however), thus destroying temporal information. Table 5 shows that this incurs a relatively small loss now, suggesting that most of the performance gains observed in Table 4 are due to the latent factors. However, removing temporal information still incurs enough of a hit in performance to justify our methodology.

<i>Test Log-likelihoods on the NIPS Dataset</i>			
	Original	Shuffled	Relative Loss
<b>IBHP</b>	<b>1012.08</b>	<b>914.76</b>	<b>-9.62%</b>
HDHP	191.29	88.19	-53.90%
DHP	201.73	79.05	-60.81%
HP	40.17	18.22	-54.64%

Table 5: Model comparison on the shuffled NIPS dataset.

**C. Discovering popular topics and words.** One of the immediate benefits of our IBHP is that it returns the factor rate matrix  $\mathbf{K}$  and the word-distribution matrix  $\mathbf{V}$ , providing a rich summary of popular topics and words. Figure 5 shows, in the NIPS dataset, the most popular three topics at the end of the training dataset time span. The lists of words suggest that the first topic is related to kernel methods, the second to deep learning, and the third to Bayesian methods. The intensity of the colors indicates popularities. Our model suggests that topic 2, which hypothetically is related to deep learning, has been increasingly more popular in the NIPS community.

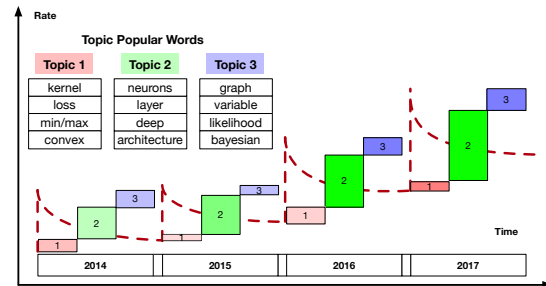


Figure 5: **NIPS dataset**. Popular topics and words.

**D. Learning factor dynamics.** Unlike the IBP, the IBHP matrix not only carries binary “present/missing” information, but also real-valued kernel weights  $\kappa_{ik}$ , which reveal the temporal dynamics of the factors. Figure 4 shows two most popular factors from the FB dataset. The first relates to school life, and the second to off-class activities. To confirm this, we plot the average of the estimated rate functions across four similar one week periods in Figure 4. The patterns of the two factor rates are quite different: The first factor is active after Monday, and peaks in the middle of the week, before cooling down near the weekend. The second factor, however, climbs steadily and becomes more excited during the weekend.

**E. Inferring dependencies and causalities.** According to Equation 11, the rate after an event depends on earlier events that share factors with it. Figure 6 provides a detailed view of IBHP on the SB dataset under the usual additive rule. We also apply the “double-sharing” rule to the dataset. In Figure 7, we see several consequences: 1) the rate functions are not triggered until the 6<sup>th</sup> observation under the double-sharing rule, 2) the IBHP matrices are different, and 3) the inferred factors are different. Further investigation shows the first red circle corresponds to the observation with text “I am mean to you all the time!” and the last red circle to “What time is it?”—one to heat up the process and one to cool it down. This suggests that adopting different triggering rules may allow us to capture different aspects of the data, which in our SB double-sharing case, bookends an active family discussion.

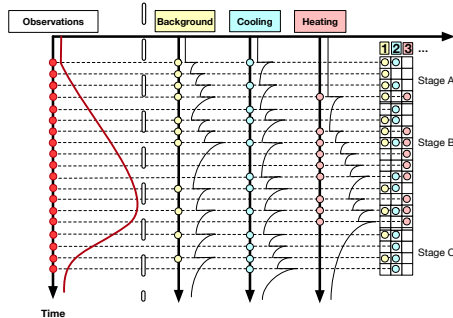


Figure 6: **SB Dataset.** Additive rule. Every observation creates a jump of the rate function. Topics can be interpreted as background, cooling, and heating activities.

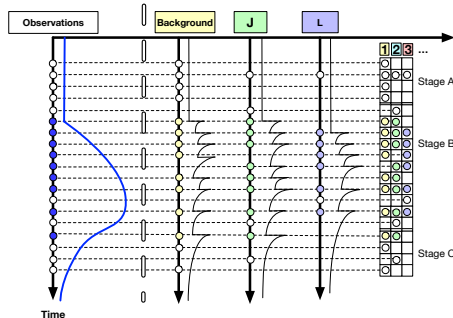


Figure 7: **SB Dataset with double-sharing.** White circles represent observations that do not trigger the rule. Topics can be interpreted as background activities, and those of Jennifer and Lisbeth.

**F. Predict future event times.** In Table 4, we report the log-likelihoods on the test datasets for each a model. To evaluate the predictive ability of our model in more depth, we use it for a different predictive task: predict the time of the next event in windows of increasing sizes, and for each case, report the absolute different from the observed data. Table 6 shows that, as the size of predictions increases, the mean absolute error increases, as well as

the standard error: as the predictions becomes harder, the predictions becomes inaccurate and unreliable. Nonetheless, our model outperforms competing models according to this metric as well.

	Prediction Window Size		
	$pws = 1$	$pws = 5$	$pws = 10$
IBHP	$0.61 \pm 0.11$	$0.97 \pm 0.18$	$1.37 \pm 0.28$
HDHP	$0.82 \pm 0.13$	$1.24 \pm 0.20$	$2.18 \pm 0.33$
DHP	$0.87 \pm 0.10$	$1.19 \pm 0.16$	$2.21 \pm 0.29$
HP	$0.92 \pm 0.17$	$2.06 \pm 0.23$	$3.56 \pm 0.31$

Table 6: **FB Dataset.** Predicting future event times.

**G. Predicting future topics and words.** Our last experiment is concerned with the prediction of the latent state variables. The dotted line in Figure 8 represents the end of the training phase, where we have obtained the latent factor rate matrix  $\mathbf{K}$  and the latent factor word distribution matrix  $\mathbf{V}$ . To the right of the dotted line, we show the projected rate function, along with the first three predictions and their predicted top words. Our model suggests that, for the NIPS dataset, topic 2 is taking over topic 3 and may become dominant in the next few events.

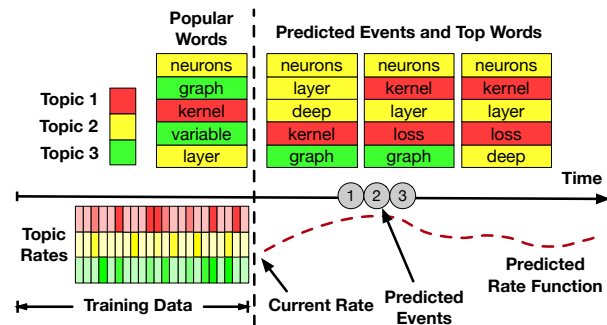


Figure 8: **NIPS dataset.** Predicted events.

## 7 CONCLUSION

In this paper, we proposed the Indian Buffet Hawkes Process (IBHP)—a novel Bayesian nonparametric stochastic point process model for learning multiple latent triggering factors of streaming document/message data. Our approach establishes the synergy between Indian Buffet Processes (IBP) and Hawkes processes (HP): on the one hand, we use the IBP to add multiple triggering factors to the HP, which helps to better model dynamics and improves interpretation, and on the other hand, the temporal information from the HP is embedded into the IBP to drive the latent factor estimation, which expands its capability to model evolution of factors.

## Acknowledgment

This research is supported by NSF under contract numbers IIS-1546488 and IIS-1618690.

## References

- [1] Enron Email Data Set. <https://www.cs.cmu.edu/~./enron/>.
- [2] NIPS Dataset. <http://www.kaggle.com/>.
- [3] SB Dataset. [www.linguistics.ucsb.edu/research/santa-barbara-corpus](http://www.linguistics.ucsb.edu/research/santa-barbara-corpus).
- [4] Charles Blundell, Jeff Beck, and Katherine A Heller. Modelling reciprocating relationships with hawkes processes. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2600–2608, 2012.
- [5] Daryl J Daley and David Vere-Jones. *An introduction to the theory of point processes: volume I: Elementary Theory and Methods*. Springer Science & Business Media, 2003.
- [6] Arnaud Doucet, Nando De Freitas, and Neil Gordon. An introduction to sequential monte carlo methods. In *Sequential Monte Carlo methods in practice*, pages 3–14. Springer, 2001.
- [7] Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, pages 1555–1564. ACM, 2016.
- [8] Nan Du, Mehrdad Farajtabar, Amr Ahmed, Alexander J Smola, and Le Song. Dirichlet-hawkes processes with applications to clustering continuous-time document streams. In *Proceedings of the 21th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, pages 219–228. ACM, 2015.
- [9] Thomas L Griffiths and Zoubin Ghahramani. The indian buffet process: An introduction and review. *Journal of Machine Learning Research (JMLR)*, 12(Apr):1185–1224, 2011.
- [10] Alan G Hawkes. Point spectra of some mutually exciting point processes. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 438–443, 1971.
- [11] Alan G Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.
- [12] Alan G Hawkes and David Oakes. A cluster process representation of a self-exciting process. *Journal of Applied Probability*, 11(3):493–503, 1974.
- [13] Peng Lin, Ting Guo, Yang Wang, Fang Chen, et al. Infinite hidden semi-markov modulated interaction point process. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3900–3908, 2016.
- [14] Peng Lin, Bang Zhang, Ting Guo, Yang Wang, and Fang Chen. Interaction point processes via infinite branching model. In *Association for the Advancement of Artificial Intelligence (AAAI)*, pages 1853–1859, 2016.
- [15] Thomas Josef Liniger. *Multivariate hawkes processes*. PhD thesis, ETH Zurich, 2009.
- [16] Charalampos Mavroforakis, Isabel Valera, and Manuel Gomez-Rodriguez. Modeling the dynamics of learning activity on the web. In *Proceedings of the 26th International Conference on World Wide Web (WWW)*, pages 1421–1430, 2017.
- [17] Hongyuan Mei and Jason M Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. In *Advances in Neural Information Processing Systems (NIPS)*, pages 6757–6767, 2017.
- [18] Xi Tan, Syed AZ Naqvi, Yuan (Alan) Qi, Katherine A Heller, and Vinayak Rao. Content-based modeling of reciprocal relationships using hawkes and gaussian processes. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2016.
- [19] Xi Tan, Vinayak Rao, and Jennifer Neville. Nested crp with hawkes-gaussian processes. In *Artificial Intelligence and Statistics (AISTATS)*, 2018.
- [20] Frank Wood and Thomas L Griffiths. Particle filtering for nonparametric bayesian matrix factorization. In *Advances in neural information processing systems*, pages 1513–1520, 2007.
- [21] Jiasen Yang, Vinayak Rao, and Jennifer Neville. Decoupling homophily and reciprocity with latent space network models. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2017.
- [22] Lingjiong Zhu. *Nonlinear Hawkes Processes*. PhD thesis, New York University, 2013.



---

# Battle of Bandits

---

**Aadirupa Saha**

aadirupa@iisc.ac.in

Dept. of Computer Science and Automation  
Indian Institute of Science, Bangalore 560012

**Aditya Gopalan**

aditya@iisc.ac.in

Dept. of Electrical Communication Engineering  
Indian Institute of Science, Bangalore 560012

## Abstract

We introduce *Battling-Bandits* – an online learning framework where given a set of  $n$  arms, the learner needs to select a subset of  $k \geq 2$  arms in each round and subsequently observes a stochastic feedback indicating the winner of the round. This framework generalizes the standard *Dueling-Bandit* framework which applies to several practical scenarios such as medical treatment preferences, recommender systems, search engine optimization etc., where it is easier and more effective to collect feedback for multiple options simultaneously. We develop a novel class of *pairwise-subset choice model*, for modelling the subset-wise winner feedback and propose three algorithms - *Battling-Doubler*, *Battling-MultiSBM* and *Battling-Duel*: While the first two are designed for a special class of *linear-link* based choice models, the third one applies to a much general class of *pairwise-subset choice models* with *Condorcet winner*. We also analyzed their regret guarantees and show the optimality of *Battling-Duel* proving a matching regret lower bound of  $\Omega(n \log T)$ , which (perhaps surprisingly) shows that the flexibility of playing size- $k$  subsets does not really help to gather information faster than the corresponding dueling case ( $k = 2$ ), at least for the current subsetwise feedback choice model. The efficacy of our algorithms are demonstrated through extensive experimental evaluations on a variety of synthetic and real world datasets.

## 1 INTRODUCTION

The problem of *Dueling-Bandits* has recently gained much attention in the machine learning community [22, 4, 24, 25, 23, 16]. Dueling bandits is an online learn-

ing framework, generalizing multi-armed bandits [5], in which learning proceeds in rounds: At each round the learner selects a pair of arm and observes a stochastic feedback of the winner of the comparison (duel) between the selected arms. Several algorithms have been proposed for this problem which are designed to learn to play the best arm as often as possible over time [22, 4, 24, 25, 23, 16]. These algorithms are tailor-made to work well under specific assumptions on the underlying pairwise comparison model that generates the stochastic feedback and under specific definition of the winner of a set of arms [16].

In this work, we introduce a natural generalization of the dueling bandits problem where given a set of  $n$  items (bandit arms), the learner’s objective at each round is to choose a subset of  $k \geq 2$  arms (unlike selecting just *two* arms as in case of dueling bandits), upon which the winner of the ‘battle’ among these  $k$  selected items is revealed by the environment as stochastic feedback. The goal of the learner is to identify an appropriately defined ‘best’ item in the process and play it often as possible.

We term this as the problem of *Battling-Bandits* as at each round, essentially a subset of  $k$  items are competing against each other unlike a pairwise duel as in the case of *Dueling-Bandits*. Such settings occur naturally in many application domains where it is practically easier for customers or patients to give a *single* feedback for a set of options (products or medical treatments), click on one link from set of search engine outcomes etc., as opposed to comparing only two options at a time. To the best of our knowledge this is the first work to generalize the pairwise feedback model of dueling bandits to a subsetwise model in an online regret minimization setup.

## Related Work

The most related work to the current problem setup is [17], where also a fixed set of arms is chosen in each round. However, the key difference lies in the feedback structure. While in [17] the feedback is a pairwise prefer-

ence matrix consisting outcomes of one or more chosen pairs (maximum of  $\binom{n}{2}$  pairs), in our setting we only observe a single index of the winning arm. In [8], the authors also consider an extension of the dueling bandits framework where multiple arms are chosen in each round. We differ from their setup since we allow to choose only a fixed  $k$ -set of arms at each round, whereas [8] allows a variable number of arm selection. Moreover their work does not have any theoretical guarantees while we provide regret guarantees for our algorithms. Another work in the similar essence is DCM-bandits [12], where a list of  $k$  distinct items are offered at each round and the users choose one or more from it scanning the list from top to bottom. Their learning objective differs substantially from ours since the DCM feedback is based on a fairly different cascading feedback model. Moreover, their regret objective demands to find the set of best  $k$  items as opposed to finding a unique best item as in our case.

Another related body of literature is dynamic assortment optimization where the objective is offer a subset of a fixed set of items to the customers in order to maximize the expected revenue. The demand of any item depends on the substitution behavior of the customers that is captured mathematically by a choice model specifying the probability of a consumer selecting a particular item from any offered set. The problem has been studied under different choice models – e.g. multinomial logit [19], mallows and mixture of mallows [9], markov chain based choice models [10], single transition model [15], general discrete choice models [7] etc. A related bandit setting has also been studied as the MNL-Bandits problem in [2] where the learner selects a fixed set of  $k$  arms in each iteration. However, the feedback is observed from a multinomial logit model (MNL) which is different from the subset choice model we considered here. Moreover their setting takes item prices into account due to which the notion of the ‘best item’ is different from ours, i.e. the *Condorcet winner*. Thus our current problem setting can not be reduced to theirs and vice-versa.

## Proposed Work

The main challenge of *Battling-Bandits* lies in keeping track of the subset choice probabilities, i.e. the probability of an item winning in a given subset of  $k$  items, which could be potentially of size  $O(kn^k)$  as our objective is to find the “best” (Condorcet winner) item in the hindsight, we must allow repetitions of items within a offered set, which actually results in  $n^k$  possible number of subsets and each subset may give rise to atmost  $k$  choice probabilities depending on number of distinct items in the subset. Thus without any further structural or parametric assumptions on the feedback choice model, the problem becomes computationally intractable.

We thus introduce the *pairwise-subset choice model* for the purpose which is based on a pairwise preference model with Condorcet winner (Section 2) and propose three different algorithms (Section 3): The first two – *Battling-Doubler* and *Battling-MultiSBM*, are inspired by the Doubler and MultiSBM algorithms of [4] which works under a special class of *pairwise-subset choice model*, viz. *linear-subset choice model*, which naturally generalizes the linear-link based dueling feedback model of [4]. Both the algorithms are based on a novel reduction of the *battling bandit* problem to classical *multiarmed bandit* (MAB) [5]. Note that, although they apply to a special subclass of choice models, their regret guarantees hold for a richer class of arm sets, e.g. the regret of *Battling-Doubler* holds for any general class of (even infinitely many!) structured arms, whereas *Battling-MultiSBM* applies to any finite set of unstructured arms.

Our third algorithm, *Battling-duel*, works for the most general class of *pairwise subset choice models*, which is built on the novel idea of reducing *battling bandits* to the *dueling* case by using a dueling bandit algorithm as a black box, e.g. Relative-UCB [24] or Double-Thompson Sampling [21] which are guaranteed to work optimally (with  $O(n \log T)$  regret guarantee) under any pairwise preference based feedback model with Condorcet winner.

**Contributions.** The specific contributions of this paper can be summarized as follows:

1. We develop a novel class of subsetwise feedback model, called *pairwise-subset choice model*, which is based on a pairwise preference model with Condorcet winner that models the winning probability of an item in a battle in terms of its pairwise winning probabilities over others. We further analyse a special class of the above model, namely *linear-subset choice model* which generalizes the linear-link based dueling feedback model of [4] (Section 2).
2. We propose three algorithms for the problem of *Battling-Bandits* and analyze the regret guarantees of each under a natural notion of regret with respect to the Condorcet item (see Section 2.2). In particular, we show that the regret for the first two algorithms, *Battling-Doubler* and *Battling-MultiSBM*, scales as  $O(nk \ln^2(T))$  and  $O(nK(\ln(T) + n \ln(n) + n \ln \ln(T)))$  respectively, under *linear-subset choice model*. The regret of our third algorithm *Battling-Duel* holds under the general class of *pairwise-subset choice model* that scales as  $O(n \ln T)$  (Section 3).
3. We also prove a lower bound of  $\Omega(n \ln(T))$  for *Battling-Bandits* under *pairwise-subset choice model* which shows that the regret of *Battling-Duel* algorithm matches the lower bound (upto constant factor),

thereby making it the optimal possible algorithm for the current problem setup (Section 4). An interesting and perhaps surprising point to note here is that our regret bounds are independent of the subset size  $k$ , which implies the flexibility of playing larger subsets does not really help to gather information faster than the corresponding dueling case ( $k = 2$ ), atleast with the current setting of the battling problem.

4. Our extensive simulation based experiments justifies the derived theoretical guarantees of our proposed algorithms. We also compare our algorithms to *Self-Sparring* algorithm of [17], which is the only existing work applicable to our setting and show the superior performance of our algorithms on both synthetic and real word data sets (Section 5).

**Organization:** In Section 2, we describe the problem setup and introduce our notions of regret. Section 3 describes our three proposed algorithms along with theoretical regret guarantees. In Section 4, we derive the lower bound for *Battling-Bandits* problem. Section 5 presents our experimental evaluations and finally we conclude with remarks and directions for future work in Section 6.

## 2 PROBLEM SETUP

We proposed the problem of *Battling-Bandit* (or in short BB) as a natural generalization of the well-studied *Dueling-Bandit* (DB) problem in the bandit literature: Given a set of  $n \geq 2$  items (equivalently, bandit arms) denoted by  $[n] = \{1, 2, \dots, n\}$ , at each round  $t \in \mathbb{N}$ , the learner’s task is to build a multiset of  $k \geq 2$  items from  $[n]$ . The environment then picks a ‘winner’ – one of the  $k$  items from the chosen set – according to a subset choice model, unknown to the learner, and reveals the winner’s identity to the learner. We denote by  $S_t \subseteq [n]$  the multiset of  $k$  items chosen by the learner, i.e.,  $S_t \equiv (S_t(1), \dots, S_t(k)) \in [n]^k$ , and  $i_t^* \in [k]$  to be the index of the winning item in  $S_t$ , at iteration  $t$ . Each selection of  $k$  items also carries with it a cost or regret. The aim of the learner is to choose sets of items to minimize the total cumulative regret over a time horizon  $T$ .

From a different point of view, the setting of receiving the winner information of the subset  $S_t$  at each round  $t$  can be seen as a game between  $k$  players. Each player is associated with an index  $i \in [k]$  and chooses an arm from  $[n]$ , thus specifying the multiset  $S_t$ . The winning player is the index of the winning item revealed to the learner at time  $t$  – the winner of the battle among  $k$  players. Hence we named it as the problem of *Battling-Bandit* (BB). We next describe the rule of winner selection in a given battle.

### 2.1 Subset Choice Models

Given a fixed set of items (context), choice modeling defines the decision probability of preferring an individual or set of items through stochastic models. In the present case, we use subset choice models to specify the winning probability of an item in a given set. We first introduce a broad class of subset choice models, called *pairwise-subset choice models*, extending the notions from pairwise preference models for the dueling bandit ( $k = 2$ ) problem.

**Pairwise-subset choice model.** We define a class of subset choice models based on any pairwise preference matrix  $\mathbf{Q} \in [0, 1]^{n \times n}$ , where  $Q_{a,b}$  denotes the probability of arm  $a$  beating  $b$ , for any  $a, b \in [n]$ . Clearly,  $Q_{a,b} + Q_{b,a} = 1$ . Now given a set  $S \subseteq [n]$  of  $k$  items with  $S \equiv (a_1, \dots, a_k) \in [n]^k$  and any  $i \in [k]$ , we define the probability of  $i^{\text{th}}$  index gets selected as the winner as:

$$P(i|S) = \sum_{j=1, j \neq i}^k \frac{2Q_{a_i, a_j}}{k(k-1)} \quad \forall i \in [k]. \quad (1)$$

It can be easily checked that the formula above defines a valid probability distribution over the indices  $i \in [k]$ . We remark that since  $S$  is a multiset, the arm corresponding to the winning index is not necessarily unique; as an extreme example, in the multiset of  $k$  items  $(a_1, a_2, \dots, a_k)$ , we might have  $a_i = a \in [n]$ ,  $\forall i \in [k]$ , in which case each index  $i \in [k]$  wins with probability  $1/k$ .

Note that when  $k = 2$  (the dueling bandit case), for any  $S = (a, b)$ , we have  $P(i|S) = Q_{a_i, a_j}$ , where  $i, j \in [2], i \neq j$  and  $a_1 = a$  and  $a_2 = b$ ; which defines the pairwise probability of item  $a$  winning over item  $b$  in a pairwise duel. The following result provides an alternative interpretation of the *pairwise-subset choice model* in terms of the average probability that the item in question wins in a randomly chosen duel:

**Lemma 1.** *Let  $S \equiv (a_1, \dots, a_k) \in [n]^k$  be a multiset of  $k$  arms from  $[n]$ . Suppose  $U$  and  $V$  are two items (indices) chosen uniformly at random without replacement from  $[k]$ , and  $W \in [2]$  is drawn as the winning index according to the pairwise preference model  $\mathbf{Q}$  over the set  $(a_U, a_V)$ . Let  $X = U$  if  $W = 1$  and  $X = V$  if  $W = 2$ . Then, for each  $i \in [k]$ ,  $\mathbf{P}(i|S)$  in (1) is the probability that  $X = i$ .*

**Remark 1.** Note that if a *Condorcet winner* [16]  $a^* \in [n]$  exists with respect to the preference matrix  $\mathbf{Q}$ , i.e.  $\exists a^* \in [n]$ , such that  $Q_{a^*, j} > \frac{1}{2}$ ,  $\forall j \in [n] \setminus \{a^*\}$ , then it is easy to verify that for any (multi)set  $S \subseteq [n]$ ,  $P(i|S) > P(j|S)$  whenever  $a_i = a^*$  and  $a_j \in [n] \setminus \{a^*\}$ ,  $\forall i, j \in [k], i \neq j$ . Our objective is to identify this ‘best’ arm  $a^*$  and play it as often as possible; as spelt out in the definition of our regret (Section 2.2).

We now define an utility score based subset choice model as a special class of *pairwise-subset choice models*.

**Linear-subset choice model.** Let us assume that each arm  $a \in [n]$  is associated with an unknown utility score  $\theta_a \in [0, 1]$ . Then given a multiset of  $k$  items  $S \equiv (a_1, \dots, a_k) \in [n]^k$ , the probability that its  $i^{\text{th}}$  index gets selected as the winner with probability

$$\begin{aligned} \mathbf{P}(i|S) &= \frac{\sum_{j=1, j \neq i}^k (\theta_{a_i} - \theta_{a_j} + 1)}{k(k-1)} \\ &= \frac{1}{k} + \frac{\sum_{j=1, j \neq i}^k (\theta_{a_i} - \theta_{a_j})}{k(k-1)}, \forall i \in [k]. \end{aligned} \quad (2)$$

We call this the *linear-subset choice model* since the model is can be seen as a special case of *pairwise-subset choice model* when the underlying pairwise preference model  $\mathbf{Q}^\theta$  is linear, i.e.  $\Pr(a \text{ beats } b) = Q_{a,b}^\theta = \frac{(\theta_a - \theta_b + 1)}{2}$ ,  $a, b \in [n]$ . Note that this model generalizes the linear-link based pairwise feedback model of [4] at  $k = 2$ , as for any  $S = (a, b)$ , the probability  $\Pr(a \text{ beats } b) = (\theta_a - \theta_b + 1)/2$  becomes exactly equal to that of [4] used for modeling the dueling bandit feedback.

**Remark 2.** The *linear-subset choice model* satisfy a natural monotonicity property: For any set  $S \equiv (a_1, \dots, a_k)$  and  $i, j \in [k]$ ,  $\theta_{a_i} > \theta_{a_j} \Rightarrow P(i|S) > P(j|S)$ , thus the element with highest  $\theta$  score is most likely to get selected as the winner of set  $S$ . In other words, an ordering over  $\theta$  values, induces an ordering over the arms as well.

There also exist other notions subset choice models in the literature, e.g., one popular class among them is the *random utility based models* (RUM) [6] as described below:

**RUM based Choice Models.** One of the most popularly studied class of choice models are *Random Utility Models (RUM)*. RUM assumes an underlying ground truth of utility score  $\theta_i \in \mathbb{R}$  for each item  $i \in [n]$ , and assigns a conditional distribution  $\mathcal{D}_i(\cdot|\theta_i)$  for scoring item  $i$ . So given  $S \subseteq [n]$ , one first draws a random utility score  $X_i \sim \mathcal{D}_i(x_i|\theta_i)$   $x_i \in \mathbb{R}$ , for each item  $i \in S$ , and selects  $i$  with probability of  $X_i$  being the maximum among all the scores of items in  $S$ , i.e.  $i \sim \mathbf{P}(i|S) = \Pr(X_i > X_j \forall j \in S \setminus \{i\})$ ,  $\forall i \in S$

One widely used example of RUM is the *Multinomial-Logit (MNL)* or famously called *Plackett-Luce model (PL)* where  $\mathcal{D}_i$ 's are independent Gumbel distributions [6], i.e.  $\mathcal{D}_i(x_i|\theta_i) = e^{-(x_i - \theta_i)} e^{-e^{-(x_i - \theta_i)}}$ . In this case it can be shown that  $\mathbf{P}(i|S) = \frac{e^{\theta_{a_i}}}{\sum_{j \in S} e^{\theta_{a_j}}}$ ,  $\forall i \in S$ .

Similarly, alternative family of discrete choice models can be obtained assuming different distributions over the utility scores  $X_i$ , e.g. when  $(X_1, \dots, X_n) \sim \mathcal{N}(\boldsymbol{\theta}, \boldsymbol{\Sigma})$  are jointly normal with mean  $\boldsymbol{\theta} = (\theta_1, \dots, \theta_n)$  and covariance  $\boldsymbol{\Sigma}$ , above reduces to *Multinomial Probit Model (MNP)*, although unlike MNL, choice probabilities  $\mathbf{P}(i|S)$  for MNP do not have a closed formed solution [20].

## 2.2 Measuring performance – Regret

We compare the performance of the learner's strategy with respect to a 'best' arm of the choice model. As defined before, for *pairwise-subset choice models*, the most natural candidate for the 'best' arm is the Condorcet winner  $a^* \in [n]$ , i.e.  $Q_{a^*,a} > \frac{1}{2}$ ,  $\forall a \in [n] \setminus \{a^*\}$ , assuming  $\mathbf{Q}$  contains a Condorcet arm. Then an intuitive way to define the regret of *Battling-Bandit* is by extending the notion of dueling bandit regret [24, 22, 4] as follows:

$$R_T^{BB} = \sum_{t=1}^T \left( \frac{\sum_{a \in S_t} (Q_{a^*,a} - \frac{1}{2})}{k} \right), \quad (3)$$

Consequently, the aim of the learner is to play sets  $S_t$  at times  $t = 1, 2, \dots$  to keep the regret as *low* as possible which in fact corresponds to playing  $a^*$  as many times as possible in  $S_t$ , at any round  $t$ . Clearly only if the learner plays the set  $S_t = (a_1, a_2, \dots, a_k)$  such that  $a_i = a^* \forall i \in [k]$ , the corresponding regret incurred at round  $t$  is 0.

Note that, for *linear-subset choice models*, the 'best' arm is  $a^* = \operatorname{argmax}_{a \in [n]} \theta_a$ , i.e., an arm having the highest utility score, as that happens to be the Condorcet winner of the underlying pairwise preference model  $\mathbf{Q}^\theta$ . Thus using (3), we can similarly define the regret  $R_T^{BB}$  in this case as well with  $Q_{a^*,a}^\theta = \frac{(\theta_{a^*} - \theta_a + 1)}{2}$ ,  $\forall a \in [n]$ .

## 3 PROPOSED ALGORITHMS

In this section we describe three algorithms for the *Battling-Bandit* problem. The first two algorithms, *Battling-Doubler* and *Battling-MultiSBM*, respectively generalize the two algorithms for utility based dueling bandits (UBDB) Doubler and MultiSBM, proposed by Ailon et al. [4], which essentially address the problem by using classical multi-armed bandit (MAB) algorithm as an underlying black box. Our third algorithm, *Battling-Duel* is based on dueling matches that uses black-box instances of a dueling bandit algorithms for the purpose.

The main advantage of *Battling-Doubler* is that it works even with an infinite set of arms, although its regret guarantee is off by an extra multiplicative factor of  $\ln T$ . On the other hand, *Battling-MultiSBM* guarantees  $O(nk \ln T)$  regret for any finite set of  $n$  arms. However both these algorithms are tailored for *linear-subset choice model*, unlike our third algorithm, *Battling-duel*, which in contrast applies to the general class of *pairwise-subset choice models* (Section 2) and is shown to perform optimally with a regret guarantee of  $O(n \log T)$  as long as the it uses an optimal dueling bandit algorithm as the black box.

Before describing our proposed algorithms, it is worth describing the black-box algorithms used to design them.

**SBM:** We call a black box algorithm for the classical

MAB problem<sup>1</sup> as a single bandit machine (SBM). Any SBM instance  $\mathcal{S}$  supports three operations: Reset, Advance and Feedback.  $\text{Reset}(\mathcal{S})$  initializes the instance  $\mathcal{S}$ .  $\text{Advance}(\mathcal{S})$  suggests which arm to play next and  $\text{Feedback}(\mathcal{S}, r)$  feedbacks a reward  $r \in [0, 1]$  to  $\mathcal{S}$ .

**Definition 2. ( $\alpha$ -robust SBM) [4]** Consider a SBM instance  $\mathcal{S}$  with  $n$  arms. For any sub-optimal arm  $x \in [n]$ , let  $T_x$  be the number of times  $x$  is played by  $\mathcal{S}$  in  $T$  rounds. The SBM  $\mathcal{S}$  is said to be  $\alpha$ -robust if  $\forall s \geq 4\alpha\Delta_x^{-2} \ln T$ , it holds that  $\mathbf{P}[T_x > s] < \frac{2}{\alpha}(s/2)^{-\alpha}$ , where  $\Delta_x$  denotes the gap between the expected reward of the best arm and that of arm  $x$  in the underlying MAB instance.

**DBM:** Similar to SBM, we call a black box algorithm for the dueling bandit problem as dueling bandit machine (DBM). A DBM also supports the same three operations as that of a SBM instance, with the only difference being that a DBM instance  $\mathcal{D}$ , outputs *two* arms  $x, y \in [n]$  on the  $\text{Advance}(\mathcal{D})$  operation instead of one. We refer  $x$  as the right arm and  $y$  the left arm. Also, in this case, the feedback  $r$  upon  $\text{Feedback}(\mathcal{S}, r)$  is a preference relation between  $x$  and  $y$  defined as  $r = \mathbf{1}(y \text{ beats } x)$ . We now describe our main algorithms and their regret guarantees. Proofs of all the theorems are deferred to the Appendix.

### 3.1 Battling-Doubler

The first algorithm, *Battling-Doubler*, maintains a single SBM instance  $\mathcal{S}$ . The total time horizon  $T$  is divided into exponentially growing epochs, and a MAB game is played within each epoch using  $\mathcal{S}$ . Specifically, at any epoch, the algorithm plays the first  $(k-1)$  arms uniformly from the multiset of arms  $\mathcal{L}$  selected by  $\mathcal{S}$  in the previous epoch, the  $k^{\text{th}}$  arm is played adaptively according to suggestion of the SBM  $\mathcal{S}$  upon which  $\mathcal{S}$  receives a binary reward based on the defeat or victory of the  $k^{\text{th}}$  arm it suggested. Algorithm 1 describes *Battling-Doubler* formally.

**Remark 3.** Note that when  $[n]$  is finite, in order to save the memory overhead of maintaining the multiset  $\mathcal{L}$  (line 14), a more elegant approach can be to instead maintain a probability distribution  $\mathbf{p}^t \in \Delta_n$  over the  $n$  arms, where  $p_a^t \in [0, 1]$  denotes the fraction of times arm  $a \in [n]$  was played as the  $k^{\text{th}}$  arm of  $\mathcal{S}_{t-1}$  at round  $(t-1)$ , and sample  $a_1^t, a_2^t, \dots, a_{k-1}^t$  according to  $\mathbf{p}^t$  (in line 7).

**Theorem 3. Battling-Doubler Regret for general arm sets.** Assume that the SBM  $\mathcal{S}$  used by *Battling-Doubler* has expected regret no more than  $c \ln^\beta(t)$  at the end of  $t \in \mathbb{N}$  rounds, where  $c > 0, \beta > 0$  are constants independent

<sup>1</sup>Given a fixed set of  $n$  arms, each associated to a reward distribution with their expectation bounded in the range  $[0, 1]$ , the classical MAB defines the problem of identifying the best arm with highest expected reward by actively selecting one arm at each round sequentially and receiving a feedback from its underlying reward distribution in an online fashion [5].

---

#### Algorithm 1 Battling-Doubler

---

```

1: Initialize:  $\mathcal{S} \leftarrow$  an SBM over set of  $[n]$  arms
2:    $\mathcal{L} \leftarrow [n]$ 
3:    $\ell \leftarrow 1, t \leftarrow 1$ 
4: while true do
5:   reset( $\mathcal{S}$ )
6:   for  $j = 1, 2, 3 \dots 2^\ell$  do
7:     Select  $a_1^t, a_2^t, \dots, a_{k-1}^t$  uniformly from  $\mathcal{L}$ 
8:      $a_k^t \leftarrow \text{Advance}(\mathcal{S})$ 
9:     Play  $S_t = (a_1^t, a_2^t, \dots, a_k^t)$ 
10:    Receive winner  $i_t^* \in [k]$ 
11:    Feedback( $\mathcal{S}, \mathbf{1}(i_t^* = k)$ )
12:     $t \leftarrow t + 1$ 
13:   end for
14:    $\mathcal{L} \leftarrow$  the multiset of arms played as  $a_k^t$  in epoch  $\ell$ 
15:    $\ell \leftarrow \ell + 1$ 
16: end while

```

---

of  $t$ . Then, under the linear-subset choice model, the expected regret of *Battling-Doubler* at the end of  $T$  rounds is at most  $2c \frac{k\beta}{\beta+1} \ln^{\beta+1}(T)$ .

**Corollary 4. Battling-Doubler Regret for finite set of arms.** Assume the SBM  $\mathcal{S}$  used in *Battling-Doubler* is the Upper Confidence Bound (UCB) algorithm [5] and suppose the underlying feedback model used for the *Battling-Bandit* problem is linear-subset choice model with parameter  $\theta \in [0, 1]^n$ , such that  $\theta_1 > \max_{i=2}^n \theta_i$ . Then the expected regret of *Battling-Doubler* is  $O(kH \log^2 T)$ , where  $H := \sum_{i=2}^n \frac{1}{\Delta_i}$ , and  $\Delta_i = \theta_1 - \theta_i \forall i \in [n]$ .

Note that, the above regret guarantee becomes trivial if the gap parameter  $H$  is large. Instead, we can also derive the a gap-independent regret bound as follows:

**Corollary 5. Battling-Doubler Regret (Gap-independent regret bound)** Assume that the SBM  $\mathcal{S}$  in *Battling-Doubler* is the Upper Confidence Bound (UCB) algorithm [5]. Then under any linear-subset choice model, the expected regret of *Battling-Doubler* is at most  $O(k\sqrt{nT \log^3 T})$ .

### 3.2 The Battling-MultiSBM algorithm

Unlike *Battling-Doubler*, *Battling-MultiSBM* simultaneously maintains  $n$  independent SBMs  $\mathcal{S}_a, \forall a \in [n]$ . At each round  $t$ , the first  $(k-1)$  arms are played according to the last  $(k-1)$  arms of round  $(t-1)$  and the  $k^{\text{th}}$  arm is played according to the suggestion of SBM  $\mathcal{S}_{a_k^{t-1}}$  which corresponds to the  $k^{\text{th}}$  arm played at round  $t-1$ . As before, a binary reward is fed back to  $\mathcal{S}_{a_k^{t-1}}$  based on whether the arm it suggested at round  $t$  wins or not. *Battling-MultiSBM* is formally described in Algorithm 2.

**Theorem 6. Battling-MultiSBM Regret with finite arms.** Suppose all the SBMs used

---

**Algorithm 2 Battling-MultiSBM**

---

- 1: **Initialize:** For each arm  $a \in [n]$ ,  $\mathcal{S}_a \leftarrow$  new SBM over set of arms  $[n]$ .  $\text{Reset}(\mathcal{S}_a)$ .
  - 2:       Select  $a_2^0, a_3^0, \dots, a_k^0$  uniformly from  $[n]$
  - 3: **for**  $t = 1, 2, \dots, T$  **do**
  - 4:    $a_j^t = a_{j+1}^{t-1}, \forall j \in [k-1]$
  - 5:    $a_k^t \leftarrow \text{advance}(\mathcal{S}_{a_{k-1}^t})$
  - 6:   Play  $S_t = (a_1^t, a_2^t, \dots, a_k^t)$
  - 7:   Receive winner  $i_t^* \in [k]$
  - 8:   Feedback  $(\mathcal{S}_{a_{k-1}^t}, \mathbf{1}(i_t^* = k))$
  - 9: **end for**
- 

in *Battling-MultiSBM* are  $\alpha$ -robust, where  $\alpha = \max\{3, 2 + \frac{\ln K}{\ln \ln T}\}$ . Also assume  $\theta_1 > \max_{i=2}^n \theta_i$ ,  $\Delta_i = (\theta_1 - \theta_i)$ ,  $\forall i \in [n]$  and  $H := \sum_{i=2}^n \frac{1}{\Delta_i}$ . Then, under the linear-subset choice model with parameter  $\theta \in [0, 1]^n$ , the regret of *Battling-MultiSBM* is  $O\left(kH\alpha \left(\ln T + n \ln n + n \ln \ln T + 2 \sum_{i=2}^n \ln \frac{1}{\theta_1 - \theta_i}\right)\right)$ .

Note that the above bound is essentially of  $O(nk \log T)$ , since  $H = O(n)$  given a fixed instance of *linear-subset choice model*  $\theta$ . Similar *Battling-Doubler*, here also we can derive a gap-independent regret bound as follows:

**Corollary 7. Battling-MultiSBM Regret (Gap independent regret bound).** *If the SBMs used in Battling-MultiSBM are  $\alpha$ -robust,  $\alpha = \max\{3, 2 + \frac{\ln K}{\ln \ln T}\}$ , then under any linear subset choice model, the regret of Battling-MultiSBM is  $O\left(k\sqrt{nT}\alpha \left(\sqrt{\ln T} + n \frac{(\ln n + \ln \ln T)}{\sqrt{\ln T}}\right)\right)$ .*

### 3.3 Battling-Duel

Our third algorithm *Battling-Duel*, is a simple general algorithm for *Battling-Bandits* that uses a good (low-regret) dueling bandit algorithm as its black-box and works under any *pairwise-subset choice model*. *Battling-Duel* maintains an instance of a dueling bandit algorithm (DBM)  $\mathcal{D}$ , at each round  $t$ , the algorithm receives two arms  $x_t, y_t \in [n]$  from  $\mathcal{D}$ , and plays the multiset  $S_t = (x_t, x_t, \dots, x_t, y_t, y_t, \dots, y_t)$  of  $k$  arms by replicating  $x_t$  and  $y_t$  equal number of times on an average. More precisely,  $x_t$  is replicated for either  $\lfloor k/2 \rfloor$  or  $\lceil k/2 \rceil$  number of times with equal probability of  $\frac{1}{2}$  and the rest half of  $S_t$  is filled with  $y_t$ . Upon playing  $S_t$ , once the identity of the battling winner is revealed,  $\mathcal{D}$  receives a corresponding dueling feedback depending on if its  $x_t$  or  $y_t$ . The formal description is given in Algorithm 3.

The following result shows an exact equivalence between the regret of *Battling-Duel*  $R_T^{BB}(BD)$  and that of its underlying dueling bandit algorithm  $R_T^{DB}(\mathcal{D})$ .

**Theorem 8. Battling-Duel Regret.** *Under any pairwise-subset choice model with preference matrix  $\mathbf{Q}$ , the regret*

---

**Algorithm 3 Battling-Duel**

---

- 1: **Initialize:**  $\mathcal{D} \leftarrow$  new dueling bandit algorithm over set of  $[n]$  arms
  - 2: **for**  $t = 1, 2, \dots$  **do**
  - 3:    $\{x_t, y_t\} \leftarrow \text{Advance}(\mathcal{D})$
  - 4:    $S_t = (x_t, \dots, x_t, y_t, \dots, y_t)$ , where  $x_t$  and  $y_t$  are respectively replicated for  $\lfloor k/2 \rfloor$  and  $\lceil k/2 \rceil$  or  $\lceil k/2 \rceil$  and  $\lfloor k/2 \rfloor$  times, each with probability  $\frac{1}{2}$ .
  - 5:   Receive winner  $i_t^* \in [k]$
  - 6:   Feedback:  $(\mathcal{D}, \mathbf{1}(S_t(i_t^*) = y_t))$
  - 7: **end for**
- 

incurred by *Battling-Duel (BD)* in  $T$  rounds is

$$R_T^{BB}(BD) = \kappa R_T^{DB}(\mathcal{D}),$$

where  $\kappa = \frac{2(k-1)}{k}$  if  $k$  is even, or  $\kappa = \frac{2k}{k+1}$  otherwise.  $R_T^{DB}(\mathcal{D})$  is the regret incurred by  $\mathcal{D}$  in  $T$  rounds, i.e.  $R_T^{DB}(\mathcal{D}) = \sum_{t=1}^T \frac{(Q'_{a^*, x_t} - \frac{1}{2}) + (Q'_{a^*, y_t} - \frac{1}{2})}{2}$  as per the standard definition of regret for any dueling bandit algorithm  $\mathcal{D}$  [24, 22] under  $\mathbf{Q}'$  (as also obtained from (3) with  $k = 2$ ),  $\mathbf{Q}'$  being the pairwise preference model perceived by  $\mathcal{D}$  in Algorithm 3, such that  $Q'_{x_t, y_t} := \mathbf{P}(i_t^* = x_t)$ , for any choices of  $(x_t, y_t)$ , at any round  $t$ .

Using  $\mathcal{D}$  as the state-of-the-art RUCB algorithm [24], gives the following regret guarantee for *Battling-Doubler*:

**Corollary 9. Battling-Duel Regret with RUCB.** *Assume that the DBM  $\mathcal{D}$  in Battling-Duel is RUCB [24], then under any pairwise-subset choice models with preference matrix  $\mathbf{Q}$ , the regret of Battling-Duel is*

$$\kappa \left( \tilde{C} + \sum_{i \in [a] \setminus \{a^*\}} \frac{2\alpha(\Delta_i + 4\Delta_{\max})}{\Delta_i^2} \ln T \right), \quad (4)$$

where  $\tilde{C}$  is a problem instance (i.e.  $\mathbf{Q}$ ) dependent constant, independent of the time horizon  $T$ ,  $\Delta_i = \left(Q_{a^*, i} - \frac{1}{2}\right)$ ,  $\Delta_{\max} = \max_{i \in [n]} \Delta_i$ ,  $\forall i \in [n]$ ,  $\kappa = \frac{2(k-1)}{k}$  if  $k$  is even, or  $\kappa = \frac{2k}{k+1}$  otherwise.

Note that Corollary 9 essentially gives an  $O(n \log T)$  regret guarantee for *Battling-Duel* since the first term of (4) is constant given a fixed  $\mathbf{Q}$ , whereas the second term scales as  $\log T$  for each  $(n-1)$  suboptimal arms. Clearly *Battling-Duel* performs the best in terms of dependency of its regret guarantee on  $n$ ,  $k$  and  $T$ , among all three of our proposed algorithms. We next establish a matching regret lower bound of  $\Omega(n \log T)$  for the problem, which essentially proves the optimality of *Battling-Duel*.

## 4 REGRET LOWER BOUND

In this section, we derive an  $\Omega(n \ln T)$  regret lower bound (Theorem 8) for the problem of *Battling-Bandit* under any

*pairwise-subset choice model*. Our proof involves reduction of an instance of the *Dueling-Bandit (DB)* problem to an instance of the *Battling-Bandit (BB)* problem and solve the former using an algorithm designed for the later. More specifically, we first prove the following key result:

**Theorem 10** (Reducing *Dueling-Bandit* to *Battling-Bandit*). *There exists a reduction from the Dueling-Bandits problem to Battling-Bandits, which preserves expected regret under any pairwise-subset choice model.*

*Proof.* Consider that we have an algorithm  $\mathcal{A}_{BB}$  for the BB problem and our goal is to construct a DB algorithm  $\mathcal{A}_{DB}$  using this. One intuitive way to do this is: At any round  $t$ , first play  $\mathcal{A}_{BB}$  to generate the set  $S_t$  of  $k$  arms, randomly sample two indices  $i_t, j_t \in [k]$  from the set  $[k]$ , play  $a_{i_t}, a_{j_t}$  respectively as the left and right arm of DB, receive the winner  $w_t$  of the duel ( $a_{i_t}, a_{j_t}$ ) from the DB environment and feedback a winning index  $i_t^*$  to  $\mathcal{A}_{BB}$  accordingly as the winner of the  $S_t$  battle. The resulting algorithm  $\mathcal{A}_{DB}$  is as summarized in Algorithm 4.

---

**Algorithm 4**  $\mathcal{A}_{DB}$ : Reducing DB to BB

---

- 1: **for**  $t = 1, 2, \dots$  **do**
  - 2:    $S_t \leftarrow$  Multiset of arms played by  $\mathcal{A}_{BB}$  at round  $t$
  - 3:   Draw  $i_t, j_t \sim \text{Unif}[k]$  (without replacement)
  - 4:   Play  $(a_{i_t}, a_{j_t})$
  - 5:   Receive feedback  $w_t = \mathbf{1}\{a_{i_t} \text{ beats } a_{j_t}\}$
  - 6:   Return  $i_t^* = i_t w_t + j_t (1 - w_t) \in \{i_t, j_t\}$  to  $\mathcal{A}_{BB}$  as winning index to  $\mathcal{A}_{BB}$
  - 7: **end for**
- 

The crucial observation is that if the DB environment actually simulates the winner from an underlying (unknown) preference matrix  $\mathbf{Q}$ , then internally  $\mathcal{A}_{BB}$  sees a world where subset choice probabilities are given by  $\mathbf{P}(i|S) = \frac{\sum_{j=1, j \neq i}^k 2Q_{a_i, a_j}}{k(k-1)}$ , due to Lemma 1. Thus at each round  $t$ , the average instantaneous regret of  $\mathcal{A}_{DB}$  is:

$$\begin{aligned} & \mathbf{E}_{i_t, j_t \sim [k], i_t \neq j_t} [r_t(\mathcal{A}_{DB})] \\ &= \mathbf{E}_{i_t, j_t \sim [k], i_t \neq j_t} \left[ \frac{(Q_{a^*, a_{i_t}} - \frac{1}{2}) + (Q_{a^*, a_{j_t}} - \frac{1}{2})}{2} \right] \\ &= \frac{1}{k(k-1)} \sum_{i=1}^k 2(k-1) \left[ \frac{(Q_{a^*, i} - \frac{1}{2})}{2} \right] \\ &= \sum_{i=1}^k \left[ \frac{(Q_{a^*, i} - \frac{1}{2})}{k} \right] = r_t(\mathcal{A}_{BB}). \end{aligned}$$

where the second equality follows since the expectation is taken over the random draw of two indices  $i_t, j_t$  from  $[k]$  without replacement,  $a^* \in [n]$  being the Condorcet arm of  $\mathbf{Q}$  and  $r_t(\mathcal{A}_{BB})$  denotes the instantaneous

regret of  $\mathcal{A}_{BB}$  at round  $t$ , as defined in Section 2.2. Thus we get  $\mathbf{E}[R_T(\mathcal{A}_{DB})] = \mathbf{E} \left[ \sum_{t=1}^T r_t(\mathcal{A}_{DB}) \right] = \sum_{t=1}^T r_t(\mathcal{A}_{BB}) = R_T(\mathcal{A}_{BB})$ , proving the claim.  $\square$

**Corollary 11.** *Given any algorithm  $\mathcal{A}_{BB}$  for Battling-Bandits (BB) under pairwise-subset choice model associated to a preference matrix  $\mathbf{Q}$  with Condorcet winner, there exists a problem instance of BB such that*

$$\liminf_{T \rightarrow \infty} \frac{\mathbf{E}[R_T(\mathcal{A}_{BB})]}{\ln T} \geq \sum_{i \in [n] \setminus \{a^*\}} \min_{j \in B_i} \frac{\Delta_{ij}}{\text{kl}(Q_{i,j}, \frac{1}{2})},$$

where  $\Delta_{ij} = \frac{(Q_{a^*, i} - \frac{1}{2}) + (Q_{a^*, j} - \frac{1}{2})}{2}$ ,  $B_i = \{j \mid Q_{i,j} < \frac{1}{2}\}$ , and  $\text{kl}(p, q) = p \log \frac{p}{q} + (1-p) \log \frac{1-p}{1-q}$  denotes the *kl-divergence* between two Bernoulli distributions with parameters  $p$  and  $q$ .

**Remark 4.** Note that Corollary 11 implies that the asymptotic regret lower bound is  $\Omega(n \log T)$  since  $\sum_{i \in [n] \setminus \{a^*\}} \min_{j \in B_i} \frac{\Delta_{ij}}{\text{kl}(Q_{i,j}, \frac{1}{2})}$  essentially involves a sum over  $(n-1)$  terms, each being a constant for a fixed  $\mathbf{Q}$ , thus making it  $\Omega(n)$ . This therefore concludes that the regret guarantee of *Battling-Duel* (Theorem 8) is indeed optimal when used with a ‘good’ dueling bandit algorithm of  $O(n \log T)$  regret guarantee (Corollary 9).

**Remark 5.** The optimal regret guarantee of  $O(n \log T)$  of *Battling-Bandits* with *pairwise-subset choice model* is independent of the subset size  $k$ , which essentially clarifies the *tradeoff of learning rate with subset size  $k$*  — even with the flexibility of playing larger  $k$ -sized sets ( $k \geq 2$ ) does not help in faster information aggregation than the corresponding dueling setup ( $k = 2$ ) — which might appear counter intuitive but is justified as information theoretically the winner information of a  $k$ -set does not reveal any additional information over that in a 2-set.

## 5 EXPERIMENTS

We now present empirical evaluations for our proposed algorithms on different synthetic and real world datasets and also compare them with the *Self-Sparring* algorithm of [17], which is the only existing work applicable to our framework. In all our experimental results, our proposed algorithm *Battling-Duel* outperforms the rest, rightfully justifying the optimality of its regret guarantees as discussed in Remark 4. A detailed discussion is given below:

**Algorithms.** We compared the performances of the following 5 algorithms: **1. BD-RUCB:** *Battling-Duel* (Section 3.3) with *RUCB* [24] as the DBM  $\mathcal{D}$ . **2. BD-TS:** *Battling-Duel* (Section 3.3) with *Double-Thompson Sampling* [21] as the DBM  $\mathcal{D}$ . **3. B-Dblr:** *Battling-Doubler* (Section 3.1) with *UCB* [5] as the SBM  $\mathcal{S}$ . **4. B-Msbm:** *Battling-MultiSBM* (Section 3.2) with *UCB* [5] as the

SBM  $S$ . **5. SS-TS:** *Self-Sparring* algorithm [17] with Thompson Sampling [3]. This algorithm closely resembles to the Sparring algorithm of [4] that maintains a single copy of SBM (a MAB algorithm), and at each round  $t$ , it queries the SBM  $k$  times to produce a  $k$ -sized battling set  $S_t$ . To the best of our knowledge no other existing work applies to the setup of *Battling-Bandits*.

**Experimental Setup and Performance Measures** We plot regret of each of the 5 algorithms for different real world and synthetic datasets, as describe in Section 5.1 and 5.2. In all the experiments the time horizon is fixed to  $T = 5000$  (with few exceptions if the regret plot do not converge within 5000 time iterations) and the experiments are run for different item sizes  $n$  and subset sizes  $k$  as specified in the corresponding experiments. The measure of performances in all the plots is the total regret  $R_T^{BB}$  in  $T$  round as defined in (3). All results are reported as average across 50 runs along with the standard deviations.

### 5.1 Experiments on Synthetic Datasets

For synthetic experiments with *linear-subset choice model* (Section 2), we use the following four different utility score vectors  $\theta$ : 1. *arith* 2. *geom* 3. *g1* and 4. *g3*.

Both *arith* and *geom* has  $n = 8$  items, with item 1 being the ‘best’ (Condorcet) item of highest score, i.e.  $\theta_1 > \max_{i=2}^8 \theta_i$ ; the rest of the  $\theta_i$ s are in an arithmetic or geometric progression respectively, as their names suggest. The two score vectors are described in Table 2.

arith	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1
geom	0.8	0.7	0.512	0.374	0.274	0.2	0.147	0.108

Table 1: Parameters for *linear-subset choice model*

The next two utility score vectors has  $n = 15$  items in each. Similarly as before, item 1 is the Condorcet winner here as well, with  $\theta_1 > \max_{i=2}^{15} \theta_i$ . More specifically for *g1*, the individual score vectors are of the form:  $\theta_i = 0.8$ , if  $i = 1$  and  $\theta_i = 0.2$ ,  $\forall i \in [15] \setminus \{1\}$ . For *g3*, the individual score vectors are of the form:  $\theta_i = 0.8$ , if  $i = 1$ ,  $\theta_i = 0.7$ ,  $\forall i \in [8] \setminus \{1\}$  and  $\theta_i = 0.6$ , otherwise

Clearly, *g3* is a harder model (for learning the Condorcet item), than *g1* as in the former case, the gap between the items scores are very close to each other and the best and the second best item is only 0.1 distance apart, whereas gap is 0.6 for every suboptimal items in the later case. The fact is reflected in our experimental results as well.

**Results on linear-subset choice model.** Figure 1 shows the comparative regret performances of the 5 algorithms, for *Battling-Bandits* with *linear-subset choice model* on 4 different utility score vectors as described above. We set  $k = 4$  for *arith* and *geom* and  $k = 8$  for the rest two.

The results clearly shows the superiority of *Battling-Duel* compared to the rest. In fact, BD-TS performs slightly better than BD-RUCB as Thompson sampling based algorithms are known to perform empirically well compared to UCB based algorithms (in spite of both having a similar  $O(n \log T)$  regret guarantee), although it comes at the cost of a higher performance variability as evident from our plots. SS-TS being a Thompson Sampling based algorithm, it exhibits a very high variability too.

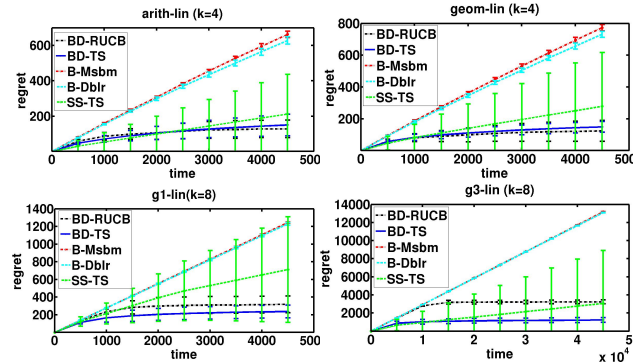


Figure 1: Averaged regret over time on synthetic datasets (on *linear-subset choice model*)

**Results on MNL choice model.** We also run the above experiment for the same 4 utility scores 1. *arith* 2. *geom* 3. *g1* and 4. *g3* on *Multinomial Logit (MNL)* choice model (as describes in Section 2). Similarly as before, even in this case the two *Battling-Duel* algorithms, BD-RUCB and BD-TS, perform the best among all 5. As argued before, *g3* being the ‘hardest instance to learn’, for both *linear* and MNL choice models, we had to run the algorithms for comparatively larger number of iterations until convergence. The results are shown in Figure 2.

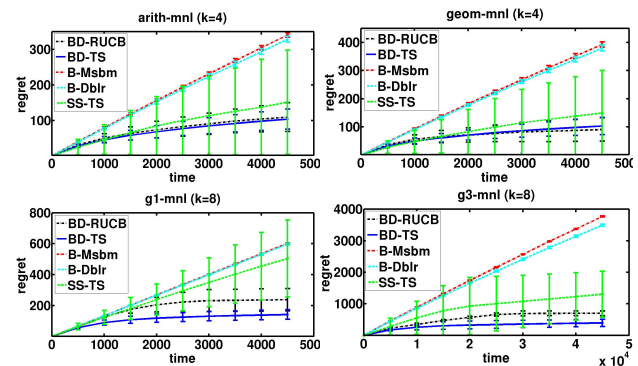


Figure 2: Averaged regret over time on synthetic datasets (on *multinomial logit (MNL) model*)

**Results on Pairwise-subset choice model.** We finally run experiments for the general *pairwise-subset choice model* on two synthetic pairwise preference matrices: *arxiv-pref* and *arith-pref* with  $n = 6$  and  $n = 8$  respectively. See Appendix E.2 for the details of the datasets.



We run the experiments for  $k = 4$  for both the datasets. As before, the two *Battling-Duel* algorithms excel the rest in this case as well, as follows from Figure 3.

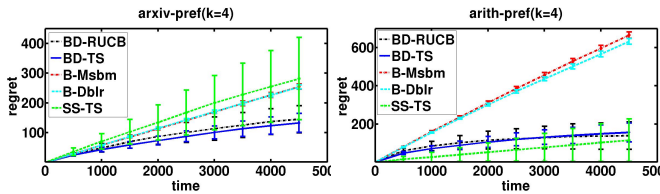


Figure 3: Averaged regret over time on synthetic datasets (on *pairwise-subset choice model*)

## 5.2 Experiments on Real Datasets

We also evaluated our method on four real-world preference learning datasets: 1. *Car* [1] 2. *Hurdy* [16] 3. *Tennis* [16] and 4. *Sushi* [11]. Each of the dataset contains pairwise preferences of a given set of  $n$  items, where  $n = 10$  for both *Car* and *Hurdy*, and it is respectively 8 and 16 for *Tennis* and *Sushi*. All the preference matrices contain a Condorcet winner (as required as per our problem setup in Section 2). We set  $k = 6$  for both *Hurdy* and *Tennis* and respectively 4 and 10 for *Car* and *Sushi*. The description of the datasets along with data extraction procedure and the actual preference matrices are given in Appendix E.

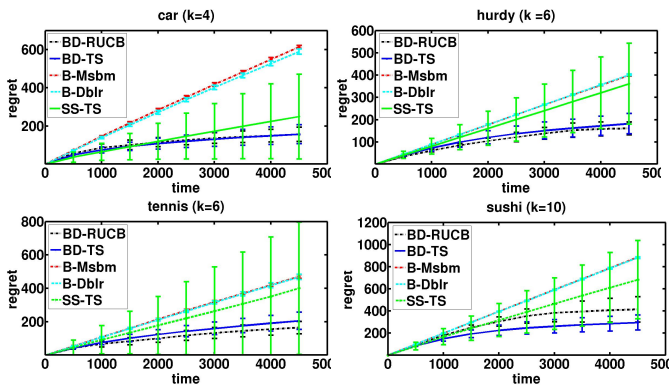


Figure 4: Averaged regret over time on real datasets (on *pairwise-subset choice model*)

**Results.** Figure 4 shows the comparative regret performances of the 5 algorithms used. As expected, BD-TS turns out to be the best algorithm for most of the cases, with BD-RUCB following it closely, whereas B-BMsbm and B-Dblr performs poorly in comparison, rightfully justifying their suboptimal regret guarantees (Theorem 3 and 6). SS-TS shows a very high variability as usual and performs worse than both BD-RUCB and BD-TS.

## 5.3 Effect of varying subset size $k$

We also analyze the scaling of the regret performances our optimal algorithm *Battling-Duel* with increasing  $k$ .

We use BD-RUCB for the purpose on two score vectors 1.  $g1$  and 2.  $g1-big$  with varying  $k$ , keeping  $n$  fixed to 15 and 50 respectively. Here  $g1-big$  is just a larger version of  $g1$  utility score with  $n = 50$  items, such that  $\theta_1 = 0.8$  and  $\theta_i = 0.2, \forall i \in [50] \setminus \{1\}$  (see Appendix E for details). The results are shown in Figure 5, which clearly reflects that the learning rate of *Battling-Duel* does not scale with  $k$ , justifying that its regret guarantee is indeed independent of the subset size  $k$  (Theorem 8).

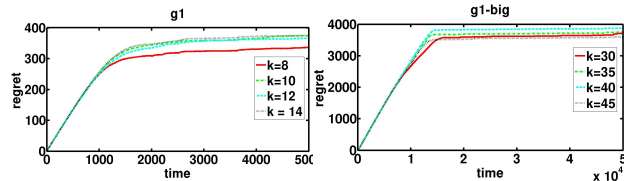


Figure 5: Averaged regret over time with varying  $k$  and fixed  $n$  (on *linear-subset choice model*)

## 6 CONCLUSION AND FUTURE WORK

We introduce the problem of *Battling-Bandit* – generalization of the well-studied *Dueling-Bandit* problem, where the objective is to find the ‘best’ arm by successively playing a subset of  $k$  arms from a pool of  $n$  arms and subsequently receiving the winner feedback in an online fashion. For this we develop a novel  $k$ -wise feedback model, viz. *pairwise-subset choice model* and propose three algorithms along with their regret bound guarantees. We also show a matching regret lower bound of  $\Omega(n \log T)$  proving the optimality of our algorithms.

Our proposed framework of *Battling-Bandits* opens up a set of new directions to explore – with different choices of feedback models, regret objectives, or even applying this to new settings like revenue maximization, contextual or adversarial bandits etc. One very interesting point noted here is that the optimal regret guarantee is independent of the subset size  $k \geq 2$ , which implies the flexibility of playing larger subsets does not really help to gather information faster than the corresponding dueling case ( $k = 2$ ), atleast with the current *pairwise-subset choice feedback model*. It will be interesting to study the tradeoff of the subset size on the regret (learning rate to identify the ‘best’ arm) for different subset choice models, e.g. MNL, MNP etc. Lastly, it would also be useful to analyze other dueling bandit algorithms, e.g. Sparring [4], especially for large set of structured arms and their implications in solving *Battling-Bandit* with different settings.

## ACKNOWLEDGEMENTS

Thanks to Arun Rajkumar (Conduent Labs, India) for all the useful discussions and the anonymous reviewers for their insightful comments. This work is supported by Qualcomm, ACM-India and IARCS travel grant.

## References

- [1] Ehsan Abbasnejad, Scott Sanner, Edwin V Bonilla, Pascal Poupart, et al. Learning community-based preferences via dirichlet process mixtures of gaussian processes. In *IJCAI*, pages 1213–1219, 2013.
- [2] Shipra Agrawal, Vashist Avadhanula, Vineet Goyal, and Assaf Zeevi. A near-optimal exploration-exploitation approach for assortment selection. In *Proceedings of the 2016 ACM Conference on Economics and Computation*. ACM, 2016.
- [3] Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on Learning Theory*, 2012.
- [4] Nir Ailon, Zohar Shay Karnin, and Thorsten Joachims. Reducing dueling bandits to cardinal bandits. In *ICML*, volume 32, pages 856–864, 2014.
- [5] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.
- [6] Hossein Azari, David Parks, and Lirong Xia. Random utility theory for social choice. In *Advances in Neural Information Processing Systems*, pages 126–134, 2012.
- [7] Gerardo Berbeglia and Gwenaël Joret. Assortment optimisation under a general discrete choice model: A tight analysis of revenue-ordered assortments. *arXiv preprint arXiv:1606.01371*, 2016.
- [8] Brian Brost, Yevgeny Seldin, Ingemar J. Cox, and Christina Lioma. Multi-dueling bandits and their application to online ranker evaluation. *CoRR*, abs/1608.06253, 2016.
- [9] Antoine Désir, Vineet Goyal, Srikanth Jagabathula, and Danny Segev. Assortment optimization under the mallows model. In *Advances in Neural Information Processing Systems*, pages 4700–4708, 2016.
- [10] Antoine Désir, Vineet Goyal, Danny Segev, and Chun Ye. Capacity constrained assortment optimization under the markov chain based choice model. *Operations Research*, 2016.
- [11] Toshihiro Kamishima and Shotaro Akaho. Efficient clustering for orders. In *Mining Complex Data*. Springer, 2009.
- [12] Sumeet Katariya, Branislav Kveton, Csaba Szepesvari, and Zheng Wen. Dcm bandits: Learning to rank with multiple clicks. In *International Conference on Machine Learning*, pages 1215–1224, 2016.
- [13] Junpei Komiyama, Junya Honda, Hisashi Kashima, and Hiroshi Nakagawa. Regret lower bound and optimal algorithm in dueling bandit problem. In *COLT*, pages 1141–1154, 2015.
- [14] Rémi Munos et al. From bandits to monte-carlo tree search: The optimistic principle applied to optimization and planning. *Foundations and Trends® in Machine Learning*, 7(1):1–129, 2014.
- [15] Kameng Nip, Zhenbo Wang, and Zizhuo Wang. Assortment optimization under a single transition model. 2017.
- [16] Siddhartha Y Ramamohan, Arun Rajkumar, and Shivani Agarwal. Dueling bandits: Beyond condorcet winners to general tournament solutions. In *Advances in Neural Information Processing Systems*, pages 1253–1261, 2016.
- [17] Yanan Sui, Vincent Zhuang, Joel W Burdick, and Yisong Yue. Multi-dueling bandits with dependent arms. *arXiv preprint arXiv:1705.00253*, 2017.
- [18] Csaba Szepesvari and Tor Lattimore. *Bandit Algorithms*, 2016. <http://banditalgs.com/2016/09/18/the-upper-confidence-bound-algorithm/>.
- [19] Kalyan Talluri and Garrett Van Ryzin. Revenue management under a general discrete choice model of consumer behavior. *Management Science*, 2004.
- [20] Ondrej Vojacek, Iva Pecakova, et al. Comparison of discrete choice models for economic environmental research. *Prague Economic Papers*, 2010.
- [21] Huasen Wu and Xin Liu. Double thompson sampling for dueling bandits. In *Advances in Neural Information Processing Systems*, 2016.
- [22] Yisong Yue, Josef Broder, Robert Kleinberg, and Thorsten Joachims. The k-armed dueling bandits problem. *Journal of Computer and System Sciences*, 78(5):1538–1556, 2012.
- [23] Masrour Zoghi, Zohar S Karnin, Shimon Whiteson, and Maarten De Rijke. Copeland dueling bandits. In *Advances in Neural Information Processing Systems*, pages 307–315, 2015.
- [24] Masrour Zoghi, Shimon Whiteson, Remi Munos, Maarten de Rijke, et al. Relative upper confidence bound for the k-armed dueling bandit problem. In *JMLR Workshop and Conference Proceedings*, number 32, pages 10–18. JMLR, 2014.
- [25] Masrour Zoghi, Shimon A Whiteson, Maarten De Rijke, and Remi Munos. Relative confidence sampling for efficient on-line ranker evaluation. In *Proceedings of the 7th ACM international conference on Web search and data mining*, pages 73–82. ACM, 2014.

---

# Adaptive Stochastic Dual Coordinate Ascent for Conditional Random Fields

---

**Rémi Le Priol**  
MILA and DIRO  
Université de Montréal, Canada

**Alexandre Piché**  
MILA and DIRO  
Université de Montréal, Canada

**Simon Lacoste-Julien**  
MILA and DIRO  
Université de Montréal, Canada

## Abstract

This work investigates the training of conditional random fields (CRFs) via the stochastic dual coordinate ascent (SDCA) algorithm of [Shalev-Shwartz and Zhang \(2016\)](#). SDCA enjoys a linear convergence rate and a strong empirical performance for binary classification problems. However, it has never been used to train CRFs. Yet it benefits from an “exact” line search with a single marginalization oracle call, unlike previous approaches. In this paper, we adapt SDCA to train CRFs, and we enhance it with an adaptive non-uniform sampling strategy based on block duality gaps. We perform experiments on four standard sequence prediction tasks. SDCA demonstrates performances on par with the state of the art, and improves over it on three of the four datasets, which have in common the use of sparse features.

## 1 INTRODUCTION

The conditional random field (CRF) model ([Lafferty et al., 2001](#)) is a common tool in natural language processing and computer vision for structured prediction. The optimization of this model is notoriously challenging. [Schmidt et al. \(2015\)](#) describes a practical implementation of the stochastic average gradient (SAG) algorithm ([Roux et al., 2012](#)) for CRFs and proposes a non-uniform sampling scheme that boosts performance. This algorithm (SAG-NUS) is currently the state of the art for CRFs optimization and we refer to [Schmidt et al. \(2015\)](#) for a detailed review of competing methods.

Deterministic (batch) methods such as L-BFGS ([Sha and Pereira, 2003](#); [Wallach, 2002](#)) have linear convergence rate but the cost per iteration is large. On the other hand, the online exponentiated gradient method (OEG) ([Collins et al., 2008](#)) and SAG are both members of a family of

algorithms with cheap stochastic updates and linear convergence rates, and they have both been applied to the training of CRFs. They are called variance reduced algorithms, because their common point is to use memory to reduce the variance of the stochastic update direction as they get closer from the optimum. [Johnson and Zhang \(2013\)](#) coined the name stochastic variance reduced gradient (SVRG) and [Defazio et al. \(2014\)](#) unified the family.

The stochastic dual coordinate ascent (SDCA) algorithm proposed by [Shalev-Shwartz and Zhang \(2013b, 2016\)](#) is a member of this family that has not yet been applied to CRFs. It is closely related to OEG in that it also does block-coordinate ascent on the dual objective. Yet an interesting advantage of SDCA over OEG (and SAG) is that the form of its update makes it possible to perform an “exact” line search with only *one* call to the *marginalization oracle*, i.e. the computation of the marginal probabilities for the CRF. This is in contrast to both SAG and OEG where each step size change requires a new call to the marginalization oracle. We thus propose in this paper to investigate the performance of SDCA for training CRFs.

**Contributions.** We adapt the multiclass variant of SDCA to the CRF setting by considering the marginal probabilities over the cliques of the graphical model. We provide a novel interpretation of SDCA as a relaxed fixed point update and highlights the block separability of the duality gap. We propose to enhance SDCA with an adaptive non-uniform sampling strategy based on the block gaps, and analyze its theoretical convergence improvement over uniform sampling. We compare the state-of-the-art methods on four prediction tasks with a sequence structure. SDCA with uniform sampling performs comparably with OEG and SAG. When SDCA is enhanced with the adaptive sampling strategy, it outperforms its competitors in terms of number of parameters updates on three of the tasks. These three tasks are all about natural language with handcrafted sparse features. We hypothesize that the efficiency of the dual methods can be related to the sparsity of these features.

**Related work.** Our proposed gap sampling strategy is similar to the one from [Osokin et al. \(2016\)](#) in the context of SDCA applied to the structured SVM objective, which reduces to the block-coordinate Frank-Wolfe (BCFW) algorithm ([Lacoste-Julien et al., 2013](#)). [Dünner et al. \(2017\)](#) recently analyzed a general adaptive sampling scheme for approximate block coordinate ascent that generalizes SDCA. Their proposed sampling scheme (which basically chooses the biggest gap) was motivated in the different context of mixed GPU and CPU computations, which does not apply to our setting. Our proposed practical strategy takes in consideration the staleness of the gaps and is more robust in our experimental setting. [Csiba et al. \(2015\)](#) proposes an adaptive sampling scheme for SDCA for binary classification which unfortunately cannot be generalized to the CRF setting due to an intractable computation. Closely related to our work is [Perekrestenko et al. \(2017\)](#) who analyzed several adaptive sampling strategies for a generalization of the primal-dual SDCA setup, including our proposed gap sampling scheme. However their analysis was focused on the single coordinate descent method (e.g. binary SDCA) and on sublinear convergence results obtained when strong convexity is not assumed. We cover instead the block-coordinate approach relevant to CRFs, and one of our notable results is to show that the linear convergence rate for gap sampling **dominates** the one for uniform sampling, in contrast to what happens in the sublinear regime studied by [Perekrestenko et al. \(2017\)](#).

**Outline.** We review the optimization problem for CRFs as well as provide novel insights on the primal-dual optimization structure in Section 2. We present SDCA for CRFs in Section 3 and discuss important implementation aspects in Section 4. We present and analyze various adaptive sampling schemes for SDCA in Section 5. We provide experiments in Section 6 and discuss the implications in Section 7.

## 2 CONDITIONAL RANDOM FIELDS

In this section, we review the CRF model and its associated primal and dual optimization problems. We then derive some interesting properties which motivate several optimization algorithms.

### 2.1 DEFINITION

A CRF models the conditional probability of a structured output  $y \in \mathcal{Y}$  (e.g. a sequence) given an input  $x \in \mathcal{X}$  with a Markov random field that uses an exponential family parameterization with sufficient statistics  $F(x, y) \in \mathbb{R}^d$  and parameters  $w \in \mathbb{R}^d$ :  $p(y|x; w) \propto \exp(w^\top F(x, y))$ . The feature vector  $F$  decomposes as a

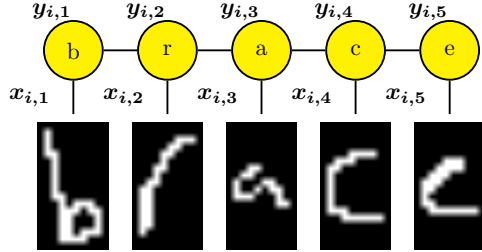


Figure 1: Example of graphical model for the optical character recognition (OCR) task. We want to exploit the structure of the word to predict that  $y_{i,5}$  is an "e" and not a "c". This can be done by working on the pairs  $y_{i,\{t,t+1\}} = (y_{i,t}, y_{i,t+1})$ , the cliques of that model.

sum over the cliques  $C \in \mathcal{C}$  of the graphical model for  $y$ :  $F(x, y) = \sum_C F_C(x, y_C)$ , where  $y_C$  denotes the subset of coordinates of  $y$  selected by the indices from the set  $C$ . See Figure 1 for an illustration.

### 2.2 PRIMAL PROBLEM

We have a data set  $(x_i, y_i)_{i \in [1, n]}$  of  $n$  i.i.d. input and structured output pairs. The parameter is learned by minimizing the  $\ell_2$ -regularized negative log-likelihood:

$$\min_{w \in \mathbb{R}^d} \frac{\lambda}{2} \|w\|_2^2 + \frac{1}{n} \sum_{i=1}^n -\log(p(y_i|x_i; w)). \quad (1)$$

We now rewrite it using the notation for the SDCA setup for multi-class classification from [Shalev-Shwartz and Zhang \(2016\)](#). Denote  $M_i = |\mathcal{Y}_i|$  the number of labelings for sequence  $i$ . Denote  $A_i$  the  $d \times M_i$  matrix whose columns are the *corrected features*  $\{\psi_i(y) := F(x_i, y) - F(x_i, y^*)\}_{y \in \mathcal{Y}_i}$ . Denote also  $\phi_i(s) := \log(\sum_{y \in \mathcal{Y}_i} \exp(s_y))$  the log-partition function for the scores  $s \in \mathbb{R}^{M_i}$ . The negative log-likelihood can be written  $-\log(p(y_i|x_i; w)) = \phi_i(-A_i^\top w)$ . The primal objective function to minimize over  $w \in \mathbb{R}^d$  thus becomes:

$$\mathcal{P}(w) := \frac{\lambda}{2} \|w\|_2^2 + \frac{1}{n} \sum_{i=1}^n \phi_i(-A_i^\top w). \quad (2)$$

### 2.3 DUAL FORMULATION

The above minimization problem (2) has an equivalent *Fenchel convex dual* problem ([Lebanon and Lafferty, 2002](#)). Denote  $\Delta_M$  the probability simplex over  $M$  elements. Denote  $\alpha_i \in \Delta_{M_i}$  the set of dual variables for a given  $x_i$ . The dual problem handles directly the probability of the labels for the training set. The dual objective to maximize over the choice of  $\alpha = (\alpha_1, \dots, \alpha_n) \in$

$\Delta_{|\mathcal{Y}_1|} \times \dots \times \Delta_{|\mathcal{Y}_n|}$  is:

$$\mathcal{D}(\boldsymbol{\alpha}) := -\frac{\lambda}{2} \left\| \frac{1}{n\lambda} \sum_i A_i \alpha_i \right\|^2 + \frac{1}{n} \sum_{i=1}^n H(\alpha_i), \quad (3)$$

where  $H(\alpha_i) := -\sum_{y \in \mathcal{Y}_i} \alpha_i(y) \log(\alpha_i(y))$  is the entropy of the probability distribution  $\alpha_i$ . The negative entropy appears as the convex conjugate of the softmax:  $-H = \phi^*$ .

## 2.4 OPTIMALITY CONDITION

We define the *conjugate weight* function  $\hat{w}$  as follows:

$$\begin{aligned} \hat{w}(\boldsymbol{\alpha}) &:= \frac{1}{n\lambda} \sum_i A_i \alpha_i = \frac{1}{\lambda n} \sum_{i=1}^n \mathbb{E}_{y \sim \alpha_i} [\psi_i(y)] \\ &= \frac{1}{\lambda} \left( \frac{1}{n} \sum_{i=1}^n F(x_i, y_i) - \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{y \sim \alpha_i} [F(x_i, y)] \right). \end{aligned}$$

It is the difference between the average of the ground truth features, and the average of the expected features for the dual variable, up to a factor  $\frac{1}{\lambda}$ . We can show that  $\hat{w}(\boldsymbol{\alpha}^*) = \mathbf{w}^*$  where  $\mathbf{w}^*$  and  $\boldsymbol{\alpha}^*$  are respectively the optimal primal parameters and the optimal dual parameters.

We can also define the *conjugate* probabilities  $\hat{\alpha}_i$  as follows:

$$\forall i, \quad \hat{\alpha}_i(\mathbf{w}) := \nabla_s \phi_i(-A_i^\top \mathbf{w}) = p(\cdot | x_i; \mathbf{w}). \quad (4)$$

We get another optimality condition  $\hat{\alpha}(\mathbf{w}^*) = \boldsymbol{\alpha}^*$ . These two optimality conditions can be deduced directly from the structure of the duality gaps.

## 2.5 DUALITY GAPS

Note that  $\mathcal{P}(\mathbf{w}) \geq \mathcal{D}(\boldsymbol{\alpha})$  is always true, with equality at the optimum. The *duality gap* is defined by:

$$g(\mathbf{w}, \boldsymbol{\alpha}) = \mathcal{P}(\mathbf{w}) - \mathcal{D}(\boldsymbol{\alpha}). \quad (5)$$

Note that we can rewrite the primal gradient as following:

$$\nabla \mathcal{P}(\mathbf{w}) = \lambda(\mathbf{w} - \hat{w} \circ \hat{\alpha}(\mathbf{w})). \quad (6)$$

One can verify that:

$$g(\mathbf{w}, \hat{\alpha}(\mathbf{w})) = \frac{\lambda}{2} \|\mathbf{w} - \hat{w}(\hat{\alpha}(\mathbf{w}))\|^2 \quad (7)$$

$$= \frac{1}{2\lambda} \|\nabla \mathcal{P}(\mathbf{w})\|^2. \quad (8)$$

This structure of the gap for the primal weights and its conjugate dual probabilities have an equivalent in the dual. Denote the Fenchel duality gap of  $\phi_i$  for the scores  $s_i = -A_i^\top \mathbf{w}$  and probabilities  $\alpha_i$ :

$$F_i(s_i, \alpha_i) := \phi_i(s_i) + \phi_i^*(\alpha_i) + s_i^\top \alpha_i \geq 0. \quad (9)$$

The positivity comes from the definition of convex conjugates. The gap is zero when  $s_i$  and  $\alpha_i$  are conjugate variables for  $\phi_i$ , e.g.  $\alpha_i = \nabla \phi_i(s_i)$ . For any smooth loss  $\phi_i$ , the duality gap between  $\hat{w}(\boldsymbol{\alpha})$  and  $\boldsymbol{\alpha}$  decomposes as a sum of Fenchel gaps (Shalev-Shwartz and Zhang, 2013a):

$$g(\hat{w}(\boldsymbol{\alpha}), \boldsymbol{\alpha}) = \frac{1}{n} \sum_i F(-A_i^\top \hat{w}(\boldsymbol{\alpha}), \alpha_i). \quad (10)$$

The log-sum-exp and the entropy are a special pair of conjugates. Their Fenchel duality gap is also equal to the Bregman divergence generated by  $\phi_i^* = -H$ , the Kullback-Leibler divergence:  $F_i(s_i, \alpha_i) = D_{KL}(\alpha_i \| \nabla \phi_i(s_i))$ . Writing this for the same pair of conjugate variables yields:

$$g(\hat{w}(\boldsymbol{\alpha}), \boldsymbol{\alpha}) = \frac{1}{n} \sum_i D_{KL}(\alpha_i \| \hat{\alpha}_i(\hat{w}(\boldsymbol{\alpha}))). \quad (11)$$

The duality gaps (7) and (11) are typically used to monitor the optimization. In Appendix D, we explain how one can transfer a convergence guarantee on the primal or dual suboptimality to a convergence guarantee on the duality gap.<sup>1</sup> Moreover, the block-separability of gaps from (11) can motivate an adaptive sampling scheme, as we describe in Section 5.

## 2.6 INTERPRETATION

The primal formulation chooses a  $\mathbf{w}$  of small norm so as to maximise the conditional probability of observing the labels. Conversely, the dual formulation chooses conditional probabilities of the labels so as to minimize the  $\ell_2$  distance between the expected features and empirical expectation of the ground truth features. The optimal distribution would be the empirical distribution, if not for the entropic regularization that favors more uniform probabilities. This is the regularized version of the classical duality between maximum-likelihood and maximum-entropy for exponential families.

The optimality conditions show that the solution of the primal Problem (2) is also a *fixed point* for the function  $\hat{w} \circ \hat{\alpha}$ . Because of the gradient form (6), the gradient descent update can also be written as a *relaxed* fixed point update:

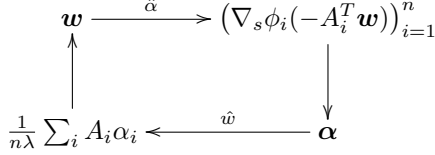
$$\mathbf{w}^+ = \mathbf{w} - \gamma \nabla \mathcal{P}(\mathbf{w}) \quad (12)$$

$$= (1 - \gamma\lambda)\mathbf{w} + \gamma\lambda \hat{w} \circ \hat{\alpha}(\mathbf{w}). \quad (13)$$

The algorithm SDCA described in the next section also admits a relaxed fixed point update on the block  $\alpha_i$

<sup>1</sup> This implies that convergence results on the dual problem directly translates to convergence results on the primal and vice-versa; a fact apparently missed in the linear rate comparison of Schmidt et al. (2015).

(see (14)). More generally, optimization algorithms for Problem (2) can often be interpreted as a back and forth between the conjugate variables  $w$  and  $\hat{w}(\hat{\alpha}(w))$  (primal methods) or  $\alpha$  and  $\hat{\alpha}(\hat{w}(\alpha))$  (dual methods). For instance, one could interpret OEG as a relaxed fixed point iteration over the score variables  $s_i = -A_i^T w$ .



Most of the results presented in this section and in Section 5 can be transposed to other kinds of loss and regularization, under some regularity assumptions. Our focus in this paper is the application of SDCA to CRF models and thus we focused the discussion on the log-likelihood setting and the  $\ell_2$  norm, which are widely used.

### 3 PROXIMAL STOCHASTIC DUAL COORDINATE ASCENT

We first describe the SDCA in its general setting, and then describe the necessary modifications for training a CRF.

#### 3.1 GENERAL SETTING

The stochastic dual coordinate ascent algorithm (SDCA) updates one dual coordinate at a time so as to maximize the dual objective. SDCA was originally proposed for binary classification (Shalev-Shwartz and Zhang, 2013b) where each dual variable  $\alpha_i$  lives in  $\Delta_2 = [0, 1]$ . In this case, it is possible to do exact coordinate maximization of the dual objective over a single  $\alpha_i$  with standard one dimensional optimization.

In the multi-class setting however, there is no simple way to maximize the dual objective over the block  $\alpha_i \in \Delta_K$ . The algorithm with the surprising name of Proximal-SDCA<sup>2</sup>, option II (Shalev-Shwartz and Zhang, 2016) proposes a solution to this problem. It updates  $\alpha_i$  in a clever direction derived from the primal-dual relationship, which amounts to a relaxed fixed point update. See Algorithm 1.

We now describe the idea. At all time, we maintain the pair of dual and primal variables  $(\alpha, w = \hat{w}(\alpha))$ . At each step, we sample a training point  $i$ . We compute  $\beta_i = \nabla_s \phi_i(-A_i^T w) = \hat{\alpha}_i \circ \hat{w}(\alpha)$ , the next fixed point iterate. We then define the dual ascent direction by  $\delta_i := \beta_i - \alpha_i$ . Finally we update the block  $\alpha_i$  with the right step size so as to increase the dual objective  $\mathcal{D}(\alpha)$  using a relaxed fixed point update:

$$\alpha_i^+ \leftarrow \alpha_i + \gamma \delta_i = (1 - \gamma)\alpha_i + \gamma \hat{\alpha}_i \circ \hat{w}(\alpha). \quad (14)$$

<sup>2</sup>We simply call it SDCA in the rest of this paper

---

#### Algorithm 1 Prox-SDCA (option II) called SDCA here

---

```

Initialize  $\alpha_i^{(0)} \in \Delta_{M_i}, \forall i$ 
Let  $w^{(0)} = \hat{w}(\alpha^{(0)}) = \frac{1}{\lambda n} \sum_i A_i \alpha_i$ 
for  $t = 0, 1 \dots$  do
  Sample  $i$  uniformly at random in  $\{1, \dots, n\}$ 
  Let  $\beta_i := \hat{\alpha}_i(w) = \nabla_s \phi(-A_i^T w)$ 
  Let  $\delta_i = \beta_i - \alpha_i^{(t)}$  {dual ascent direction}
  Let  $v_i = \frac{1}{\lambda n} A_i \delta_i$  {primal direction}
  Solve Equation (15) to get  $\gamma^*$  {Line Search}
  Update  $\alpha_i^{(t+1)} := \alpha_i^{(t)} + \gamma^* \delta_i$ 
  Update  $w^{(t+1)} := \hat{w}(\alpha^{(t+1)}) = w^{(t)} + \gamma^* v_i$ 

```

---

The dual ascent direction is guaranteed to increase  $\mathcal{D}(\alpha)$ , unless  $\delta_i = 0$  (this actually means that the block is already optimal, see (11)). The primal weights  $w = \hat{w}(\alpha)$  are related to  $\alpha$  by a linear transformation. Define the primal direction  $v_i = \frac{1}{\lambda n} A_i \delta_i \in \mathbb{R}^d$ . One can update the weights directly:  $w^+ \leftarrow w + \gamma v_i$ .

The step size  $\gamma \in [0, 1]$  is either fixed, or found via line search. In practice the fixed step size for which convergence is guaranteed is really small. The line search is relatively cheap as we are looking at only one block:

$$\gamma^* := \arg \max_{\gamma \in [0, 1]} -\phi_i^*(\alpha_i + \gamma \delta_i) - \frac{\lambda n}{2} \|w + \gamma v_i\|^2. \quad (15)$$

Note that one can decompose the quadratic term and precompute  $\langle w, v_i \rangle$  and  $\|v_i\|^2$  to accelerate the optimisation. The bottleneck remains the computation of  $\phi_i^*$  (and its derivatives).

#### 3.2 ADAPTATION TO CRF

In the CRF setting, the dual variable  $\alpha_i$  is exponentially large in the input size  $x_i$ . For a sequence  $x_i$  of length  $T$  where each node can take up to  $K$  values, the number of possible labels is  $|\mathcal{Y}_i| = K^T$ . It might not even fit in memory. Instead, the standard approach used in OEG and SAG is to consider the marginal probabilities  $(\mu_C)_{C \in \mathcal{C}}$  on the cliques of the graphical model. Similarly, we replace  $\alpha$  by  $\mu = (\mu_1, \dots, \mu_n)$ , where  $\mu_i \in \prod_C \Delta_C$  is the concatenation of all the clique marginal vectors for the sample  $i$ . For the same sequence  $x_i$ , this reduces the memory cost to  $K^2(T - 1)$  for the pair marginals. We denote  $m_i = \sum_C |\mathcal{Y}_{i,C}|$  this new memory fingerprint. For a sequence long enough, we have  $m_i \ll M_i$ . The associated weight vector can still be expressed as function of  $\mu$  thanks to the separability of the features:

$$\hat{w}(\mu) = \frac{1}{\lambda n} \sum_i \sum_C \mathbb{E}_{\mu_{i,C}} [\psi_{i,C}] = \frac{1}{\lambda n} \sum_i B_i \mu_i, \quad (16)$$

where  $B_i = (\psi_{i,C}(y_C))_{C, y_C} \in \mathbb{R}^{d \times m_i}$  is the horizontal concatenation of the cliques feature vectors.

---

**Algorithm 2** SDCA for CRF
 

---

Initialize  $\mu_i^{(0)} \in \prod_C \Delta_C$  consistently  $\forall i$  {use (21)}  
 Set  $\mathbf{w}^{(0)} := \hat{\mathbf{w}}(\boldsymbol{\mu}^{(0)}) = \frac{1}{\lambda n} \sum_i B_i \mu_i^{(0)}$  {See (16)}  
 (Optional) Let  $g_i = 100, \forall i$   
**for**  $t = 0, 1 \dots$  **do**  
   Sample  $i$  uniformly at random in  $\{1, \dots, n\}$   
   (Alternatively) Sample  $i$  proportionally to  $g_i$   
   Let  $\nu_{i,C}(y_C) := p(y_C | x_i; \mathbf{w}^{(t)}, \forall C \in \mathcal{C}$  {**oracle**}  
   (Optional) Let  $g_i = \tilde{D}(\mu_i || \nu_i)$  {duality gap (19)}  
   Let  $\delta_i = \nu_i - \mu_i^{(t)}$  {ascent direction}  
   Let  $\mathbf{v}_i = \frac{1}{\lambda n} \hat{\mathbf{w}}(\delta_i)$  {primal direction}  
   Solve Equation (20) to get  $\gamma^*$  {Line Search}  
   Update  $\mu_i^{(t+1)} := \mu_i^{(t)} + \gamma^* \delta_i$   
   Update  $\mathbf{w}^{(t+1)} := \hat{\mathbf{w}}(\boldsymbol{\mu}^{(t+1)}) = \mathbf{w}^{(t)} + \gamma^* \mathbf{v}_i$

---

Now, assume that the graph has a *junction tree* structure  $T = (\mathcal{C}, \mathcal{S})$  (Koller and Friedman, 2009, Def. 10.3), where  $\mathcal{C}$  is the set of maximal cliques and  $\mathcal{S}$  the set of separators. We can then run message passing on the junction tree to infer the new marginals given weights  $\mathbf{w}$ :  $\hat{\mu}_i(\mathbf{w}) = p(y_C = \cdot | x_i; \mathbf{w})$ . We can also now recover the joint probability  $\alpha_i(y)$  as a function of its marginals  $\mu_{i,C}$  (Koller and Friedman, 2009, Def. 10.6):

$$\alpha_i(y) = \frac{\prod_{C \in \mathcal{C}} \mu_{i,C}(y_C)}{\prod_{S \in \mathcal{S}} \mu_{i,S}(y_S)}. \quad (17)$$

Equation (17) in turn allows us to compute the entropy and the divergences of the joints, using only the marginals. Let  $\mu_i$  and  $\nu_i$  be the marginals of respectively  $\alpha_i$  and  $\beta_i$ , then the entropy and the Kullback-Leibler divergence are given by:

$$\tilde{H}(\mu_i) := H(\alpha_i) = \sum_C H(\mu_{i,C}) - \sum_S H(\mu_{i,S}) \quad (18)$$

and

$$\begin{aligned} \tilde{D}(\mu_i || \nu_i) &:= D_{KL}(\alpha_i || \beta_i) \\ &= \sum_C D_{KL}(\mu_{i,C} || \nu_{i,C}) - \sum_S D_{KL}(\mu_{i,S} || \nu_{i,S}). \end{aligned} \quad (19)$$

With this expression of the entropy (18), we can compute the dual objective, and thus perform the line search:

$$\gamma^* = \arg \max_{\gamma \in [0,1]} \tilde{H}(\mu_i^{(t)} + \gamma \delta_i) - \frac{\lambda n}{2} \|\mathbf{w}^{(t)} + \gamma \mathbf{v}_i\|^2. \quad (20)$$

With the Kullback-Leibler divergence (19), we can compute efficiently the individual duality gaps from (11). Algorithm 2 describes this variation of SDCA, with as an option a non-uniform sampling strategy defined in Section 5.3.

## 4 IMPLEMENTATION

We provide in Appendix A a discussion of various important implementation aspects summarized here.

1. The initialization of dual methods for CRFs can significantly influence their performance. As explained in Appendix A, we use:

$$\boldsymbol{\alpha}^{(0)} := \varepsilon \mathbf{u} + (1 - \varepsilon) \boldsymbol{\delta}, \quad (21)$$

where  $\mathbf{u}$  is the uniform distribution on each block,  $\boldsymbol{\delta}$  is a unit mass on each ground truth label and  $\varepsilon$  is a small number.

2. Storing the dual variable may be expensive and one should allocate a decent amount of memory.
3. The line search requires computing the entropy of the marginals. This is costly and we used Newton-Raphson algorithm to minimize the number of iterations. This in turn requires storing the logarithm of the dual variable.

## 5 ADAPTIVE SAMPLING FOR SDCA

Recently, there has been a lot of attention on non-uniform sampling for stochastic methods. The general goal is to sample more often points which are harder to classify and can bring more progress on the objective. These methods are said to be *adaptive* when the sampling probability changes during the optimization. SDCA itself has had several adaptive schemes proposed. In the following, we attempt to explain and relate these methods, and suggest new schemes that work well on our problem.

### 5.1 ASCENT LEMMA

We start by restating the ascent lemma from Equation (25) in Shalev-Shwartz and Zhang (2013a). This lemma inspires and supports all the strategies.

**Ascent after sampling  $i$ :** At iteration  $t$ , if we sample  $i$  and take a step of size  $\gamma_i \in [0, 1]$ , we can lower bound the resulting dual improvement:

$$\begin{aligned} &n(\mathcal{D}(\boldsymbol{\alpha}^+) - \mathcal{D}(\boldsymbol{\alpha})) \\ &\geq \gamma_i \underbrace{\left[ \phi(-A_i^T \mathbf{w}) + \phi^*(\alpha_i) + \mathbf{w}^T A_i \alpha_i \right]}_{\text{Fenchel gap} =: g_i} \\ &\quad + \gamma_i \left( \frac{(1 - \gamma_i)}{2} - \frac{\gamma_i R_i}{2\lambda n} \right) \|\beta_i - \alpha_i\|_1^2 \end{aligned} \quad (22)$$

where  $R_i := \|A_i\|_{1 \rightarrow 2}^2 = \max_{y \in \mathcal{Y}_i} \|\psi_i(y)\|_2^2$  is the squared radius of the corrected features for sample  $i$ .

Note that compared to the original text, we used the fact that the regularizer is the  $\ell_2$  norm and the loss is 1-smooth

with respect to the  $\ell_\infty$  norm. We define  $R := \max_i R_i$ ,  $\bar{R} := \frac{1}{n} \sum_i R_i$  and  $\bar{g} := \frac{1}{n} \sum_i g_i$  the true duality gap (see (9)-(10)). We also introduce  $L_i := \lambda + \frac{R_i}{n}$  an upper bound on the smoothness of loss  $i$  plus regularizer for the  $\ell_2$  norm. We recall from Section 2.5 that  $g_i = D_{KL}(\alpha_i || \beta_i)$  (11). We give the name *residual* to  $d_i := \|\beta_i - \alpha_i\|_1^2$ .

This lemma is derived with standard assumptions and inequalities on the smoothness of the loss and the strong convexity of the regularizer. The first term of the lower bound is the ascent guarantee while the other term gives condition on the step-size to ensure progress. We refer the reader to the original paper for more details.

To get the expected progress (conditioned on the past) after sampling with probability  $p$ , we simply need to take the sum of the inequality above after multiplying both sides by  $p_i$ . Our goal is to maximize this lower bound by choosing the right probability  $p$  and step sizes  $\gamma$ . To be able to conclude the proof with the original method, we also want some constants time the duality gap  $\bar{g}$  to appear in the lower bound – the gap is lower bounded by the dual suboptimality and thus this constant will give the linear rate of convergence. The lemma can then transpose this result from the dual sub-optimality to the duality gap as described in Appendix D. From there on there are two general approaches: importance sampling and duality gap sampling.

## 5.2 IMPORTANCE AND RESIDUAL SAMPLING

With the importance sampling approach, the goal is to set the step-size and the probability so that they cancel each other out:  $\gamma_i = \frac{\gamma}{p_i}$ . One then get an unbiased estimate of the true duality gap from (11) as the first term of the upper bound. What is left is maximizing the second term with respect to  $p$ . This is the approach proposed by [Zhao and Zhang \(2015\)](#) (Importance Sampling, left term below) and generalized by [Csiba et al. \(2015\)](#) (Residual sampling, a.k.a. AdaSDCA for binary classification, right term):

$$p_i \propto L_i \quad \text{or} \quad p_i \propto d_i \sqrt{L_i}. \quad (23)$$

These sampling schemes somehow allow to maximize the second term of (22). Intuitively, they replace a dependency on  $R$  in the convergence rate by a dependency on  $\bar{R}$ . They can give good results on binary and multi-class logistic regression. There are a few issues though.

- One needs an accurate estimate of the  $L_i$ .
- Importance sampling is not adaptive.
- In the CRF setting, the residual is  $d_i = \|\beta_i - \alpha_i\|_1^2$ . It is the squared  $\ell^1$  norm of a vector of exponential size. We are not aware of any trick to compute it efficiently.

## 5.3 GAP SAMPLING

To make sure that the second term is positive, the original proof of uniform SDCA sets  $\gamma_i = \gamma = (1 + \frac{R}{\lambda n})^{-1}$  to obtain:

$$n\mathbb{E}_p[D(\alpha^+) - D(\alpha)] \geq \gamma \sum_i p_i g_i. \quad (24)$$

Assuming a full knowledge of the duality gaps  $g_i$ , the optimal decision is to sample the point with maximum duality gap. This was done by [Dünner et al. \(2017\)](#) in the context of multi-class classification on a pair CPU-GPU. While the GPU computes the update, the CPU updates as many duality gaps as possible. This lead to impressive acceleration over massive datasets.

However, this is not our current setting. We know and update only one gap at a time (for efficiency). Because of staleness of the gaps, our experiments with this method did not even converge for the most part (see Section 6.3). We need a more robust method.

We take inspiration from what was done by [Osokin et al. \(2016\)](#) to improve the Block-Coordinate Frank-Wolfe (BCFW) algorithm ([Lacoste-Julien et al., 2013](#)). We propose to bias sampling towards examples whose duality gaps are large:  $p_i \propto g_i$ . If we know all the duality gaps, the expected improvement reads:

$$n\mathbb{E}_p[D(\alpha^+) - D(\alpha)] \geq \chi(\mathbf{g})^2 \gamma \bar{g}, \quad (25)$$

where  $\chi(\mathbf{g}) = \sqrt{\frac{\frac{1}{n} \sum_i g_i^2}{\bar{g}^2}} \in [1, \sqrt{n}]$  is the non-uniformity of the duality gaps, as defined in [Osokin et al. \(2016, Section 3.1\)](#). The value  $\chi(\mathbf{g})^2 \gamma$  is the value that will appear in the linear convergence rate of this method. It means that the convergence rate for gap sampling **dominates** the one for uniform sampling. This is different from what was observed for BCFW where they could not prove dominance in general.

In practice we use stale estimates of the gaps and there are no convergence guarantees. We discuss more this issue in section 6.3.

We also explored a combination of gap sampling and importance sampling. We could get similar convergence rate where a trade-off appeared between the mean smoothness and the non-uniformity. We detail these considerations as a technical report in Appendix F for the interested reader.

## 6 EXPERIMENTS

We conducted these experiments to answer three questions: (1) How does the line search influence SDCA? (2) How do the non-uniform sampling schemes compare with each other? and (3) How does SDCA compare with SAG and OEG on sequence prediction?



Table 1: Dataset summary.  $d$  is the dimension of  $w$ .  $n$  is the number of data points (sequences).  $N$  is the number of nodes (e.g. sum of sequences length).  $K$  is the number of possible labels for each node.  $A$  is the number of attributes (see Appendix B).  $a$  is the maximum number of attributes extracted from one node. Mem. is the memory required by the pairwise marginals stored as float 64. The pairwise marginals dominate the memory cost.

Dataset	OCR	CONLL	NER	POS
$d$	4,082	$1.6 \times 10^6$	$2.8 \times 10^6$	$8.6 \times 10^6$
$n$	6,202	8,936	15,806	38,219
$N$	52,827	$2.1 \times 10^5$	$2 \times 10^5$	$9.1 \times 10^5$
$K$	26	22	9	45
$A$	128	74,658	$3.1 \times 10^5$	$1.9 \times 10^5$
$a$	128	19	20	13
Mem.(GiB)	0.2	0.7	0.1	13

## 6.1 EXPERIMENTAL SETTING

We applied the experimental setup outlined by Schmidt et al. (2015). We implemented SDCA to train a classifier on four CRF training tasks: (1) the optical character recognition (OCR) dataset (Taskar et al., 2004), (2) the CoNLL-2000 shallow parse chunking dataset (CONLL), (3) the CoNLL-2002 Dutch named-entity recognition dataset (NER), and (4) a part-of-speech (POS) tagging task using the Penn Treebank Wall Street Journal data. Additional details regarding these datasets are provided in Table 1. Note that the tasks (2), (3), (4) are about language understanding. They use sparse features (the ratio  $a/A$  from the table is small). The sparsest data set is NER. Note that POS is considerably larger than other datasets. All experiments are performed with a regularization factor  $\lambda = 1/n$ . We used our own implementation<sup>3</sup> of SDCA coded in plain Python and Numpy (Walt et al., 2011). In most plots we report the logarithm base 10 of the primal sub-optimality. We got the optimum by running L-BFGS a large number of iterations.

## 6.2 EFFECT OF THE LINE SEARCH

We implemented the safe bounded Newton-Raphson method from Press et al. (1992, Section 9.4) on the derivative of the line search function. A natural question to ask is : how precise should the line search be? The stopping criterion for this algorithm is the size of the last step taken so there is no proper precision parameter. We refer to this stopping criterion for the line search as the sub-precision of SDCA.

<sup>3</sup>The code to reproduce our experiments is available at: <https://remileprieol.github.io/research/sdca4crf.html>.

We discovered experimentally that the convergence of SDCA is mostly independent of the sub-precision. On all datasets, if we ask 0.01 sub-precision or less, SDCA converges with the same rate. An explanation is that the accuracy of the optimization arises from iterates  $\alpha$  and  $\hat{\alpha}(\hat{w}(\alpha))$  getting closer to each other in the simplex with each iteration.

Reaching 0.01 or 0.001 takes on average 2 iterations. Each iteration of Newton’s method require the computation of the first and second derivative of the line search objective (20). In the following we report results with sub-precision 0.001 to be on the safe side. These 2 iterations were taking about 30% of the algorithms running time for each dataset.<sup>4</sup>

We also performed experiments with only one step of the Newton update. The convergence was not affected on OCR, CONLL and POS, but convergence failed on NER (see Figure 8 of Appendix E). This phenomenon could be related to sparsity.

## 6.3 COMPARISON OF SAMPLING SCHEMES

We compare the performance of four sampling strategies with 20% of uniform sampling against the full Uniform approach, on the OCR dataset (see results in Figure 2):

- *Importance*: sample proportionally to the smoothness constants  $L_i = \lambda + \frac{R_i}{n}$ . We report how we evaluated the radii  $R_i$  in Appendix C.
- *Gap*: sample proportionally to our current estimate of the duality gaps.<sup>5</sup>
- *Gap  $\times$  importance*: sample proportionally to the product of the gap and smoothness constants.
- *Max*: sample deterministically the variable with the largest recorded gap (Dünner et al., 2017).

As discussed in Section 5.3, Max sampling is not robust enough to the staleness of the gap estimates and fails to converge here. We also observe that Importance performs worse than Uniform, and that Gap  $\times$  Importance performs worse than Gap. This indicates that the smoothness upper bounds we estimated are not informative of the difficulty of optimizing a point for SDCA. Overall, Gap sampling gives the best performance and this is what we use in the following experiments.

The ratio of uniform sampling is here to mitigate the fact that we sample proportionally to stale gaps. This is

<sup>4</sup> We also tried initializing the line search with 0.5 or with the previous step size. There was no significant difference.

<sup>5</sup> For the gap approaches, we initialize the gap estimates with large values (100) so as to perform a pass over the whole dataset before starting to sample proportionally to the stale estimates.

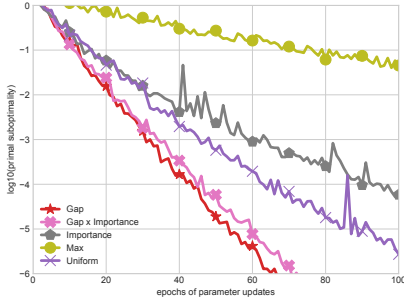


Figure 2: Performance of competing sampling schemes on the OCR dataset with 80% of non-uniformity. Sampling proportionally to the gap gives the best performance.

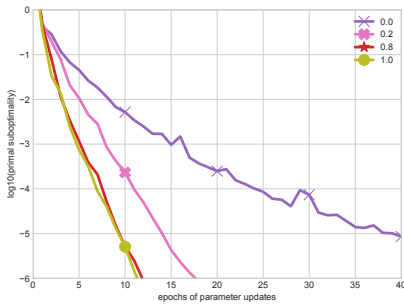


Figure 3: SDCA with Gap sampling applied on NER with various fractions of non-uniform sampling, as indicated by the number in the legend. Increasing the fraction only improves the performance, up to a certain point.

the strategy adopted by SAG-NUS (Schmidt et al., 2015) which samples uniformly half of the time. Another strategy used by Osokin et al. (2016) is to update all the duality gaps at once every 10 epochs or so. Our experiments indicate that these strategies are not needed for SDCA-GAP. Increasing the ratio of non-uniformity up to 1 only improves the performance on all datasets, though after 0.8 the improvements are marginal, as illustrated by Figure 3 for the NER dataset.

In fact, the estimate of the total gap maintained by SDCA is somewhat accurate, as illustrated for different datasets in Figure 9 of Appendix E. Empirically, it always remains within a factor 2 of the true duality gap. This accuracy is a good news because one can use this estimate of the duality gap as a stopping criterion for the whole algorithm. Once it reaches a certain precision threshold, one just has to perform one last batch update to check the real value. This is similar in spirit to SAG, which uses the norm of its estimate of the true gradient as a stopping criterion. Both are duality gaps estimators (see Equation (7)).

## 6.4 COMPARISON AGAINST SAG AND OEG

We downloaded the code for OEG and SAG-NUS as implemented by Schmidt et al. (2015) from the SAG4CRF project page.<sup>6</sup> We used our own implementation of SDCA with a line search sub-precision of 0.001. We provide the comparison in Figure 4 according to two different measures of complexity which are implementation independent.

**Oracle calls.** Schmidt et al. (2015) compared the algorithms on the basis of the number of oracle calls. We report these on OCR and NER in Figures 4a and 4d. Results on the other datasets are in Figure 6 in Appendix E. This metric was suitable for the methods they compared. Both OEG and SAG-NUS use a line search where they call an oracle on each step. SDCA does not need the oracle to perform its line search. However the oracle is message passing on a junction tree. It has a cost proportional to the size of the marginals. Each iteration of the line search require computing the entropy of these marginals, or their derivatives. These costs are roughly the same. Comparing the number of oracle calls for each method is thus unfairly advantaging SDCA by hiding the cost of its line search. It becomes a relevant comparison when a marginalization oracle becomes much more expensive than approximating the entropy (see the discussion in Section 7). When this cost is hidden, SDCA-GAP is on par with SAG-NUS\* on OCR and it is much faster on the sparse datasets.

**Parameter updates.** To give a different perspective, we report the log of the sub-optimality against the number of parameter updates in Figures 4b, 4c, 4e and 4f. This removes the additional cost of the line search for all methods.<sup>7</sup>

We observe that uniform SDCA and OEG need roughly the same number of parameters update on all four datasets. When we add the adaptive gap sampling, SDCA outperforms OEG by a margin. On OCR, SDCA and SDCA-GAP do not perform as well as SAG-NUS. On the three other datasets, SDCA-GAP needs less iterations. In fact, the more sparse the dataset, the less iterations are needed.

This is likely explained by SDCA’s ability to almost perfectly optimize each block separately due to its line search method. More specifically, as the datasets become sparser, the prediction between data points becomes less and less correlated (i.e. the label distribution for two points that share no attributes will not influence each other directly through their primal weights). In settings where no points

<sup>6</sup><https://www.cs.ubc.ca/~schmidtm/Software/SAG4CRF.html>

<sup>7</sup> This is a penalty for SAG-NUS\* which enforces a line-search skipping strategy.

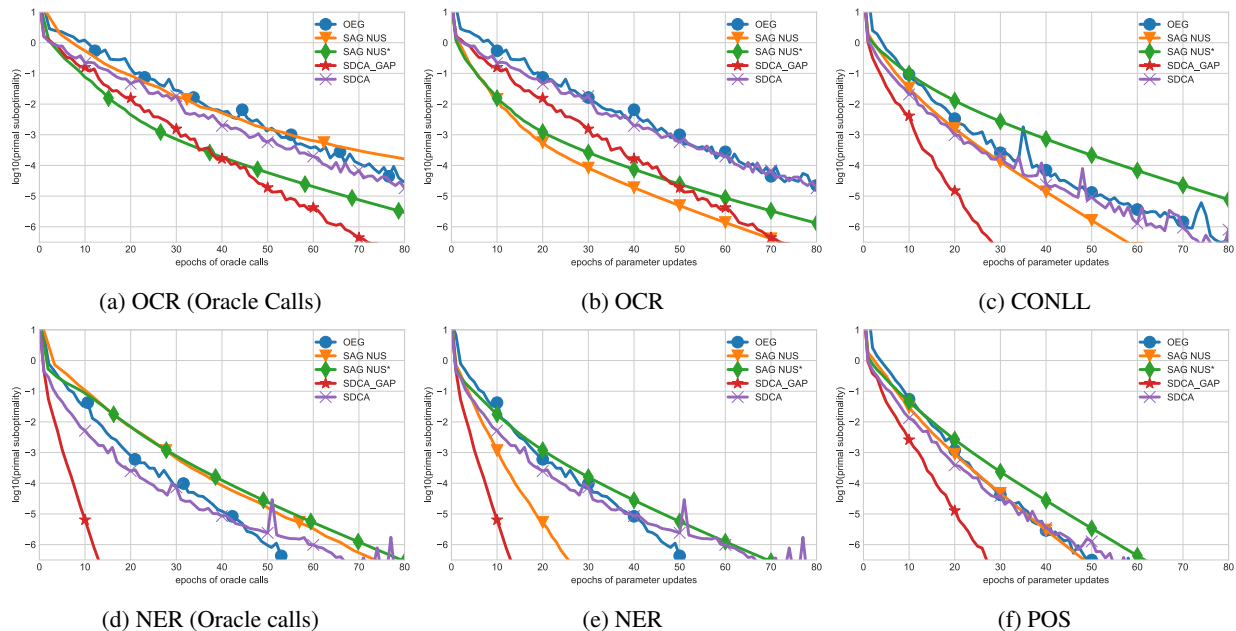


Figure 4: Primal sub-optimality as a function of the number of oracle calls (left) or parameters updates (center and right). SDCA refers to uniform sampling. SDCA-GAP refers to sampling Gap sampling 80% of the time. SAG-NUS performs a line search at every iteration. SAG-NUS\* implements a line-search skipping strategy. It appears worse than SAG-NUS when we look at the number of updates, which hides the cost of the line search.

share any attributes (completely sparse), all methods optimize each point independently. SDCA may perform very well thanks to its precise line search.

In terms of test error, SDCA is on par with SAG, and a bit better than OEG. All methods reach maximum accuracy after a few epochs. We report the evolution of the test error in Figure 7 of Appendix E.

Comparing the number of parameters updates also has a disadvantage. It penalizes methods with line search skipping strategies like OEG and SAG. The running time is highly implementation dependent and providing a fair comparison is non-trivial. We focused on implementation independent comparisons. SDCA, SAG and OEG have many common operations: the oracle, the computation of the scores and the primal direction. The fact that the line search took only 30% of SDCA’s runtime indicates that the conclusion drawn from the number of updates may hold for other metrics.

## 7 DISCUSSION

In this work, we investigated using SDCA for training CRFs for the first time. The observed empirical convergence per parameter update was similar for standard SDCA and OEG. However, SDCA can be enhanced with an adaptive sampling scheme, consistently accelerating

its convergence and also yielding faster convergence than SAG with non-uniform sampling on datasets with sparse features. It would be natural to also implement a gap sampling scheme for OEG, though several quantities needed for the computation are not readily available in standard OEG and would yield higher overhead in actual implementation. We leave finding a more efficient implementation of a gap sampling scheme for OEG as an interesting research direction.

A key feature of SDCA is to only require one marginalization oracle per line-search. This could become advantageous over SAG or OEG when the marginalization oracle becomes much more expensive than evaluating the entropy function from the marginals. Examples for this scenario include: when a parallel implementation is used for the entropy computation; or when the marginalization oracle uses an iterative approximate inference algorithms such as TRW BP whereas an approximation of the entropy is direct from the marginals (Krishnan et al., 2015). Investigating these scenarios with full timing comparison (which is implementation dependent) is a further interesting direction of future work.

We also note that acceleration schemes have been proposed for both SAG and SDCA (Lin et al., 2015; Shalev-Shwartz and Zhang, 2016), though they have not been tested yet for training CRFs.

## Acknowledgments

We are thankful to Thomas Schweizer for his numerous software engineering advices. We thank Gauthier Gidel and Akram Erraqabi who started this project. We are indebted to Ahmed Touati for his involvement during the first phase of the project. Alexandre Piché was supported by the Open Philanthropy Project. This work was partially supported by the NSERC Discovery Grant RGPIN-2017-06936.

## References

- M. Collins, A. Globerson, T. Koo, X. Carreras, and P. L. Bartlett. Exponentiated gradient algorithms for conditional random fields and max-margin Markov networks. *Journal of Machine Learning Research*, 2008.
- D. Csiba, Z. Qu, and P. Richtárik. Stochastic dual coordinate ascent with adaptive probabilities. In *ICML*, 2015.
- A. Defazio, F. Bach, and S. Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *NIPS*, 2014.
- C. Düner, T. Parnell, and M. Jaggi. Efficient use of limited-memory accelerators for linear learning on heterogeneous systems. In *NIPS*, 2017.
- R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *NIPS*, 2013.
- D. Koller and N. Friedman. *Probabilistic Graphical Models*. The MIT Press, 2009.
- R. G. Krishnan, S. Lacoste-Julien, and D. Sontag. Barrier Frank-Wolfe for marginal inference. In *NIPS*, 2015.
- S. Lacoste-Julien, M. Jaggi, M. Schmidt, and P. Pletscher. Block-coordinate Frank-Wolfe optimization for structural SVMs. In *ICML*, 2013.
- J. Lafferty, A. McCallum, and F. C. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.
- G. Lebanon and J. D. Lafferty. Boosting and maximum likelihood for exponential models. In *NIPS*, 2002.
- H. Lin, J. Mairal, and Z. Harchaoui. A universal catalyst for first-order optimization. In *NIPS*, 2015.
- Y. Nesterov. *Introductory Lectures on Convex Optimization*. Applied Optimization. Springer US, 2004.
- A. Osokin, J.-B. Alayrac, I. Lukasewitz, P. Dokania, and S. Lacoste-Julien. Minding the gaps for block Frank-Wolfe optimization of structured SVMs. In *ICML*, 2016.
- D. Perekrestenko, V. Cevher, and M. Jaggi. Faster coordinate descent via adaptive importance sampling. In *AISTATS*, 2017.
- W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical recipes in C: the art of scientific computing*. Cambridge University Press, Cambridge, 2nd edition, 1992.
- N. L. Roux, M. Schmidt, and F. R. Bach. A stochastic gradient method with an exponential convergence rate for finite training sets. In *NIPS*, 2012.
- M. Schmidt, R. Babanezhad, M. Ahmed, A. Defazio, A. Clifton, and A. Sarkar. Non-uniform stochastic average gradient method for training conditional random fields. In *AISTATS*, 2015.
- F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *NAACL*, 2003.
- S. Shalev-Shwartz and T. Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. *arXiv:1309.2375*, 2013a.
- S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14, 2013b.
- S. Shalev-Shwartz and T. Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. *Mathematical Programming*, 2016.
- B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *NIPS*, 2004.
- H. Wallach. Efficient training of conditional random fields. Master's thesis, University of Edinburgh, 2002.
- S. v. d. Walt, S. C. Colbert, and G. Varoquaux. The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 2011.
- P. Zhao and T. Zhang. Stochastic optimization with importance sampling for regularized loss minimization. In *ICML*, 2015.

---

# Adaptive Stratified Sampling for Precision-Recall Estimation

---

**Ashish Sabharwal**

Allen Institute for Artificial Intelligence  
ashishs@allenai.org

**Yexiang Xue**

Purdue University  
yexiang@cs.purdue.edu

## Abstract

We propose a new algorithm for computing a constant-factor approximation of precision-recall (PR) curves for massive noisy datasets produced by generative models. Assessing validity of items in such datasets requires human annotation, which is costly and must be minimized. Our algorithm, ADASTRAT, is the first data-aware method for this task. It chooses the next point to query on the PR curve adaptively, based on previous observations. It then selects specific items to annotate using stratified sampling. Under a mild monotonicity assumption, ADASTRAT outputs a guaranteed approximation of the underlying precision function, while using a number of annotations that scales very slowly with  $N$ , the dataset size. For example, when the minimum precision is bounded by a constant, it issues only  $\log \log N$  precision queries. In general, it has a regret of no more than  $\log \log N$  w.r.t. an oracle that issues queries at data-dependent (unknown) optimal points. On a scaled-up NLP dataset of 3.5M items, ADASTRAT achieves a remarkably close approximation of the true precision function using only 18 precision queries, 13x fewer than best previous approaches.

## 1 INTRODUCTION

Generative machine learning models can produce massive amounts of noisy data. To be fruitfully used as a standalone resource for human consumption or in downstream applications, a practitioner must understand the quality of such data. This is often done with a precision-recall or PR curve, which characterizes how data quality degrades as the model’s confidence in the validity of each

item reduces. While a PR curve can be easily created for discriminative models by using pre-annotated held-out data, doing so for generative models is not straightforward. The latter is particularly challenging when human judgment or an expensive simulation is required to assess the validity or quality of generated data items.

Consider, for example, a creative deep learning system that can generate a million poems about a given topic (Ghazvininejad et al., 2016) or a natural language system that has produced over a hundred million English paraphrase pairs (Ganitkevitch et al., 2013; Pavlick et al., 2015). How does one go about assessing the quality of such generated data or of the models behind them?

A key bottleneck is annotation: Despite substantial advances in crowdsourcing technology, our ability to annotate novel data at a reasonable cost is far outpaced by increasingly sophisticated models that generate data at an even quicker pace. Computing the exact precision of a dataset of  $N$  items requires annotating the validity of every item, making exact computation infeasible for all but the smallest datasets. Conventional random sampling methods can achieve a constant-factor approximation of the PR curve with  $\Theta(\sqrt{N \log N})$  valid/invalid annotations, but this, as Sabharwal and Sedghi (2017) argued, is also impractical in the modern era of big data. They proposed a logarithmic stratified sampling algorithm, henceforth referred to as LOGSTRAT, that can do so using only  $O(\log N \log \log N)$  annotations,<sup>1</sup> as long as the underlying precision function satisfies a *weak monotonicity* property. They also proposed PAULA, which achieves this with  $O(\Delta \log N)$  annotations, but requires a stronger notion of local monotonicity akin to concavity. This stronger monotonicity is characterized by a parameter  $\Delta$ , which is difficult to estimate from data.

Both of these algorithms *query* the precision function at a set  $S$  of geometrically spaced points (thus  $|S| = \log N$ ),

---

<sup>1</sup>These logarithms are w.r.t. base  $1 + \epsilon$ , the guaranteed approximation factor. The bounds thus scale roughly as  $1/\epsilon$ .

and interpolate between them. They, however, suffer from a limitation that  $S$  is chosen in a *data oblivious* way—it depends only on  $N$  and the desired approximation ratio, independent of the actual data. While  $\log N$  queries are sufficient, they might be overkill, e.g., in the extreme case when the precision function is a constant.

We present a new algorithm, called ADASTRAT for adaptive stratified sampling, that *adaptively chooses what to query next based on current observations of the data*. It provides a guaranteed approximation under the same weak monotonicity condition as LOGSTRAT, without the stronger condition needed by PAULA.

The main novelty is the following: Given any  $k$  points observed on a PR curve, we show how to precisely characterize the “envelope” (Figure 1) of all possible PR curves that pass through these  $k$  points (Theorem 2). This envelope can be maintained efficiently as more points are observed. This leads to a natural bisection-style algorithm, which iterates until the “height” of the envelope (i.e., the maximum gap between its upper and lower boundaries) falls within the desired approximation ratio. The approximate curve ADASTRAT outputs is the geometric mean of the resulting upper and lower envelopes, which are *non-linear*, in line with the fact that a linear interpolation isn’t appropriate in the precision-recall space (Davis and Goadrich, 2006).

ADASTRAT is surprisingly powerful both in theory and in practice. Formally, besides the initial few data points that each of these algorithm annotates, ADASTRAT uses  $O(K \log K)$  annotations chosen via adaptive stratified sampling (Theorem 8) if it ends up querying  $K$  points before meeting the stopping condition. The data determines how large  $K$  is. When the precision function decays very rapidly or very slowly,  $K$  can be as small as 2. Indeed, in two extreme cases, ADASTRAT queries only the first and last points of the PR curve and accurately interpolates everything in-between. When the minimum precision is bounded by a constant (e.g., 0.5 or 0.3) as in most practical cases,  $K$  scales as  $\log \log N$  (Corollary 1). In the worst case,  $K$  is  $\log N$  (Theorem 4), matching the asymptotic bound for LOGSTRAT.

We perform a *regret analysis* of ADASTRAT, showing (Theorem 6) that it never needs more than roughly  $\log \log N$  times more queries than an “optimal” oracle algorithm that may use *a priori* knowledge of the shape of the precision function to decide which points to query.

Using the envelope view, we also provide a *matching lower bound*: every algorithm that operates by querying the precision function at some subset of points and guarantees a constant-factor approximation, must query  $\Omega(\log N)$  points in the worst case (Theorem 7).

From a practical perspective, we evaluate various algorithms on scaled-up versions of the fully-annotated PPDB dataset used by Sabharwal and Sedghi (2017). On the PPDB-36K dataset with 35,615 items, we find that ADASTRAT *queries only 18 points* of the precision function, a 4.3x reduction from the 78 points queried by both LOGSTRAT and PAULA. Its strength is further highlighted by larger datasets, such as PPDB-100x, a 100x larger fully-annotated randomized variant that we created with a similar PR curve as the original. Here, despite the 100-fold increase in dataset size, ADASTRAT continues to query only 18 points, 13x fewer than the 234 needed by LOGSTRAT and PAULA. ADASTRAT uses mere 24K annotations,<sup>2</sup> a tiny fraction of the 3.5M items in this expanded dataset, while still yielding an impressive practical approximation (Figure 4).

## 1.1 RELATED WORK

Despite the importance of evaluating the precision-recall tradeoff of generative machine learning models, much research has been devoted to computing summary statistics (average precision AP, discounted cumulative gain DCG, etc.). Various results provide confidence intervals around estimated statistics (Carterette et al., 2006; Yilmaz et al., 2008; Aslam et al., 2006; Yilmaz and Aslam, 2006; Schnabel et al., 2016), often using different sampling approaches equipped with variance reduction techniques. Kanoulas (2015) provides a survey of relevant quality evaluation approaches in information retrieval.

In contrast to these efforts, we focus on characterizing the full precision recall curve at scale (over millions of items) and with provable guarantees. This task is considerably more challenging than computing summary statistics, an evidence of which is that these statistics can often be easily “read off” if one has computed the entire curve.

Relatively little research effort has been devoted to capturing an entire precision curve. In the area of vision, Welinder et al. (2013) propose semi-supervised performance evaluation, which is a generative model to capture a classifier’s confidence scores. Unlike their use of a parametric model that makes certain assumptions about the curve, ours is a model-free approach relying only on a (weak form of) monotonicity.

Our setup is closest to that of Sabharwal and Sedghi (2017). Different from their approach, we propose to access the precision-recall curve in a data-aware, *adaptive* fashion. This, as we show, greatly reduces the sample complexity. Further, we do not make the strong monotonicity assumption needed for their strongest algorithm.

<sup>2</sup>The conventional method needs 284K annotations and LOGSTRAT needs 54K.

## 2 PRELIMINARIES

Consider the ranked output  $T = (t_1, t_2, \dots, t_N)$  of an algorithm  $\mathcal{A}$ , where each  $t_i$  comes from some universe  $U$  (e.g., all documents on the Web, all paraphrase pairs, all subject-verb-object triples, etc.). Each item  $u \in U$  is associated with an unknown true label  $v(u) \in \{0, 1\}$  that captures the semantics of some underlying task (e.g., whether a document is relevant to a query, whether a pair of phrases is a linguistic paraphrase, whether a triple denotes a true fact, etc.). We assume access to a noisy estimator, e.g., a crowd-sourced annotation,  $\tilde{v}(u)$  of  $v(u)$  that equals  $1 - v(u)$  with probability  $\eta < 1/2$ , and equals  $v(u)$  otherwise. The **precision function of  $\mathcal{A}$** ,  $p : [N] \rightarrow [0, 1]$ , maps each rank  $r \in [N]$  to the fraction of the top  $r$  items in  $T$  that are positive, i.e., labeled as 1:

$$p(r) = \frac{1}{r} \sum_{i=1}^r v(t_i) \quad (1)$$

where we omit  $\mathcal{A}$  from the notation for brevity.

Precision functions are widely used in machine learning. In fact, they are the building blocks of many statistical metrics. For example, a commonly used metric, precision-at- $k$ , which measures the quality of the top- $k$  ranked items, is exactly  $p(k)$ . As a second example, precision-recall curves can be built from precision functions. To see this, suppose a classifier outputs and ranks items based on its belief that each item is positive.  $v(t_i)$  is an indicator variable, that is 1 if and only if the item ranked at the  $i$ -th place is positive. The classifier draws a line and classifies the top  $k$  items as positive examples. The precision of such a decision is  $1/k \sum_{i=1}^k v(t_i)$ , which is exactly  $p(k)$ , while the recall is  $\sum_{i=1}^k v(t_i) / \sum_{i=1}^N v(t_i)$ , which is  $p(k)/p(N)$ . Other metrics, such as Gain@ $k$ , accuracy, F1, true positive rate (TPR), false positive rate (FPR), Receiver Operating Characteristic (ROC) curve, average precision (AP), specificity, sensitivity, etc, can all be computed from  $p$ . Surveys by Fawcett (2006), Davis and Goadrich (2006), and Majnik and Bosnic (2013) provide more examples.

Given  $T$ , indirect access to  $\tilde{v}$ , and  $\epsilon \in (0, 1]$ , our goal is to compute a pointwise  $(1 + \epsilon)$ -approximation  $\tilde{p}$  of  $p$ . We assume accessing each  $\tilde{v}(t_i)$  is costly, e.g., needs human annotation. Therefore, we would like to compute  $\tilde{p}$  efficiently in terms of the number of evaluations of  $\tilde{v}$ .

### 2.1 POINT ESTIMATES: RANDOM SAMPLING

A simple way to obtain an estimate  $\tilde{p}(r)$  of  $p(r)$  for a fixed rank  $r$ , which we refer to as a *point estimate* at  $r$ , is via random sampling: Sample (with repetition) a set of indices  $J$  independently and uniformly

from  $\{1, 2, \dots, r\}$ , obtain a noisy estimate  $\tilde{v}(t_j)$  for each  $v(t_j)$ , and compute the empirical average  $\tilde{p}(r) = \frac{1}{z} \sum_{j \in J} \tilde{v}(t_j)$  where  $z = |J|$ . Then, assuming  $p \geq 1/3$ , the expected value of  $\tilde{p}(r)$  is within a factor of  $1 + \eta$  of  $p(r)$  (see Appendix). One can apply tail inequalities such as the two-sided Hoeffding bound (Hoeffding, 1963) to compute how tight the estimate is. For any  $\epsilon > \eta$ , to obtain a  $(1 + \epsilon)$ -approximation of  $p(r)$  with a confidence of  $1 - \delta$  (e.g., a 95% confidence would mean  $\delta = 0.05$ ), it suffices to have  $z$  samples where:

$$z \geq \frac{(1 + \eta)^2}{2(\epsilon - \eta)^2 p(r)^2} \ln \frac{2}{\delta}. \quad (2)$$

Details are deferred to the Appendix. When  $\eta = 0$ , this simplifies to the bound of Sabharwal and Sedghi (2017).

### 2.2 WEAK MONOTONICITY

Being the average of  $r$  0-1 numbers,  $p(r)$  necessarily fluctuates up and down as  $r$  increases. Nevertheless, we assume that  $T = (t_1, t_2, \dots, t_N)$  is a ranked output of an algorithm  $\mathcal{A}$ , where the true  $v(t_i)$  in the beginning are more likely to be 1. In other words, one expects  $p(r)$  to broadly decrease with increasing  $r$ . This property is captured by the following weak monotonicity notion introduced by Sabharwal and Sedghi (2017), for which we use a slightly different notation:

**Definition 1** (Weak Monotonicity). *Let  $m, \tilde{r} \in \mathbb{N}^+$ . Then  $p$  is  $(\tilde{r}, m)$ -weak monotone if for all  $r_1 \geq \tilde{r}$  and  $r_2 \geq r_1 + m$ , we have  $p(r_1) \geq p(r_2)$ .*

Weak monotonicity guarantees that, after the first  $\tilde{r}$  points, precision is non-increasing for points ranked at least  $m$  apart. Under this property, Sabharwal and Sedghi (2017) showed that it is sufficient to compute precision at only logarithmically many points in order to guarantee a tight approximation of the entire PR curve, which is reflect by their algorithm LOGSTRAT. They also relied on a stronger monotonicity assumption for their strongest algorithm, which we do *not* assume here.

**Theorem 1** (LOGSTRAT (Sabharwal and Sedghi, 2017)). *Let  $T, v, p, \tilde{r}, m$  be as above. Let  $\epsilon \in (0, 1], \delta > 0, p_{\min}$  be the minimum value of  $p$ , and  $\beta > 1$ . Let  $\ell = \lceil \log_{1+\epsilon} \tilde{r} \rceil$  and  $L = \lfloor \log_{1+\epsilon} N \rfloor$ . If  $m \leq \lfloor \epsilon(1 + \epsilon)^\ell - 1 \rfloor$  and  $p$  is  $(\tilde{r}, m)$ -monotone, then with probability at least  $1 - \delta$ , the output of LOGSTRAT on input  $(T, v, \epsilon, \tilde{r}, \delta, p_{\min}, \beta)$  is a  $\beta(1 + \epsilon)$ -approximation of  $p(r)$ . Further, LOGSTRAT queries  $p$  at  $L - \ell$  points, and uses annotation of the first (roughly)  $\tilde{r}$  points and of  $\frac{\epsilon(L-\ell)}{2(\beta-1)^2(1+\epsilon)p_{\min}^2} \ln \frac{L-\ell}{\delta/2}$  points chosen randomly via stratified sampling.*

Note that this result assumes the noiseless setting,  $\eta = 0$ . Note also that since  $L = \Theta(\log N)$ , LOGSTRAT re-

quires querying  $p$  at  $\Theta(\log N)$  points and annotating  $\Theta(\log N \log \log N)$  data points. Our goal is to improve upon this by adaptively deciding where to query (and which points to annotate), and when to stop.

### 3 CHARACTERIZING PRECISION FUNCTIONS THROUGH $k$ POINTS

What could a precision function possibly look like if we know values of it at  $k$  points? We answer this question by providing a precise characterization of all precision functions passing through  $k$  given points, under the assumption of weak monotonicity. First, we characterize a tight upper bound  $ub(v; y, p(y))$  and a lower bound  $lb(v; y, p(y))$  for every point  $p(v)$  at the precision function if we know the value of a single point  $p(y)$ . We call the space between  $ub$  and  $lb$  an *envelope* induced by the value of  $p(y)$ , because any  $p(v)$  must be sandwiched between  $lb(v; y, p(y))$  and  $ub(v; y, p(y))$ . These bounds are formally defined next, and illustrated in Figure 1.

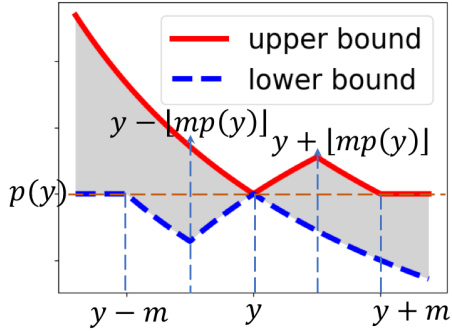


Figure 1: A graphical illustration of the upper bound  $ub$  (red line) and the lower bound  $lb$  (dashed blue line) induced by one point  $p(y)$ . The envelope is shaded.

**Definition 2.** Let  $p$  be a precision function whose value  $p(y)$  is known at a point  $y$ . Define  $ub$  and  $lb$ , each parameterized by  $y, p(y)$ , and (implicitly) by  $m$ , as:

$$ub(v; y, p(y)) = \begin{cases} p(y)y/v & \text{if } v \leq y \\ (p(y)y + v - y)/v & \text{if } y < v \leq y + \lfloor mp(y) \rfloor \\ (p(y)y + \lfloor mp(y) \rfloor)/v & \text{if } y + \lfloor mp(y) \rfloor < v \leq y + m \\ p(y) & \text{if } v > y + m \end{cases}$$

$$lb(v; y, p(y)) = \begin{cases} p(y), & \text{if } v < y - m, \\ (p(y)y - \lfloor mp(y) \rfloor)/v, & \text{if } y - m \leq v < y - \lfloor mp(y) \rfloor \\ (p(y)y + v - y)/v, & \text{if } y - \lfloor mp(y) \rfloor \leq v < y, \\ p(y)y/v, & \text{if } v \geq y. \end{cases}$$

Our characterization is summarized by the following theorem, whose proof is left to the appendix:

**Theorem 2.** Let  $p$  be any  $(\tilde{r}, m)$ -monotonic precision function. Then, for any  $v, y > \tilde{r}$ , we have:

$$lb(v; y, p(y)) \leq p(v) \leq ub(v; y, p(y))$$

Further,  $ub$  and  $lb$  are tight—each corresponds to a valid precision function whose value at  $y$  is  $p(y)$ .

This single point envelope characterization easily extends to the case where the values of  $p$  at are known at  $k$  points,  $p(y_1), p(y_2), \dots, p(y_k)$ . The envelope here is the intersection of the  $k$  single point envelopes:

$$ub(v) = \min_{j=1}^k ub(v; y_j, p(y_j)) \quad (3)$$

$$lb(v) = \max_{j=1}^k lb(v; y_j, p(y_j)) \quad (4)$$

Finally, we define the *height* of the envelope induced by  $ub$  and  $lb$  as the maximum over  $i$  of  $ub(i)/lb(i)$ .

### 4 The ADASTRAT ALGORITHM

Armed with the notion of an envelope characterizing all precision functions that could possibly pass through  $k$  observed points, we describe ADASTRAT (Algorithm 1). The idea is to *query*  $p$  near the beginning and the end, compute the envelope induced by these two observations, and continue making further queries in the middle and tightening the envelope until its height is within (the square of) the desired approximation ratio. The algorithm then outputs the geometric mean of the (non-linear) upper and lower bounds of the final envelope.

As before,  $T = (t_1, t_2, \dots, t_N)$  are the ranked data items with (unknown) true binary labels  $v(t_i)$  and precision function  $p$ . We assume access to an  $\eta$ -noisy estimator  $\tilde{v}$  of  $v$  and an oracle  $\text{QUERY}(i, T, \tilde{v})$  that returns a guaranteed  $\beta$ -approximation of the true precision  $p(i)$  at a given point  $i$ , for some  $\beta \geq 1 + \eta$ . Given  $\epsilon, \delta > 0$ , our goal is to obtain a  $\beta(1 + \epsilon)$ -approximation of the entire  $p$  with confidence at least  $1 - \delta$ . For  $m, \tilde{r} \in \mathbb{N}^+$ , we assume  $p$  is  $(\tilde{r}, m)$ -weak monotonic. For brevity, we define:

$$\tilde{l} = \max \left\{ \left\lceil \frac{(1 + \epsilon)^2 m}{2\epsilon + \epsilon^2} \right\rceil, \tilde{r} \right\}.$$

We first discuss a simple case, where  $\text{QUERY}(i, T, \tilde{v})$  returns the exact value of  $p(i)$ , i.e.,  $\beta = 1$  (and thus  $\eta = 0$ ). We will extend our result to the case where  $\beta > 1$  later. In Algorithm (1), we maintain the envelope of possible precision functions represented by the upper bound  $\tilde{ub}(i)$  and the lower bound  $\tilde{lb}(i)$ . We update these bounds in function  $\text{UPDATEUL}$  as we get access to the values of the precision function at different locations.  $\text{UPDATEUL}$



---

**Algorithm 1:** ADASTRAT( $T, \tilde{l}, \tilde{v}, \epsilon$ ): Adaptive Stratified Sampling for Approximating the Precision Function.

---

```

for  $i = 1, 2, \dots, N$  do  $\tilde{ub}(i) \leftarrow 1; \tilde{lb}(i) \leftarrow 0$ 
for  $i = 1, 2, \dots, \tilde{l}$  do
   $v(t_i) \leftarrow \text{ACCESS}(i, T)$ 
   $\tilde{p}(i) \leftarrow \frac{1}{i} \sum_{j=1}^i v(t_j)$ 
   $\tilde{ub}, \tilde{lb} \leftarrow \text{UPDATEUL}(i, \tilde{p}(i), \tilde{ub}, \tilde{lb})$ 
 $\tilde{p}(N) \leftarrow \text{QUERY}(N, T, \tilde{v})$ 
 $\tilde{ub}, \tilde{lb} \leftarrow \text{UPDATEUL}(N, \tilde{p}(N), \tilde{ub}, \tilde{lb})$ 
 $\tilde{p}(\tilde{l}+1), \dots, \tilde{p}(N-1) \leftarrow \text{PR}(\tilde{l}, N, \tilde{p}(\tilde{l}), \tilde{p}(N), \tilde{ub}, \tilde{lb})$ 
return  $\tilde{p}(1), \dots, \tilde{p}(N)$ 

```

```

Function PR( $l, r, \tilde{p}(l), \tilde{p}(r), \tilde{ub}, \tilde{lb}$ )
  if  $\max_{i \in \{l, \dots, r\}} \frac{\tilde{ub}(i)}{\tilde{lb}(i)} \leq (1 + \epsilon)^2$  or  $\frac{r}{l} \leq (1 + \epsilon)^2$ 
    then
      // stopping condition met
      for  $i \in \{l+1, \dots, r-1\}$  do
         $\tilde{p}(i) \leftarrow \sqrt{\tilde{ub}(i) \tilde{lb}(i)}$ 
    else
       $c \leftarrow \text{round}(\sqrt{lr})$  // bisect the interval
       $\tilde{p}(c) \leftarrow \text{QUERY}(c, T, \tilde{v})$  // query mid-point
       $\tilde{ub}, \tilde{lb} \leftarrow \text{UPDATEUL}(c, \tilde{p}(c), \tilde{ub}, \tilde{lb})$ 
       $\tilde{p}(l+1), \dots, \tilde{p}(c-1) \leftarrow \text{PR}(l, c, \tilde{p}(l), \tilde{p}(c), \tilde{ub}, \tilde{lb})$ 
       $\tilde{p}(c+1), \dots, \tilde{p}(r-1) \leftarrow \text{PR}(c, r, \tilde{p}(c), \tilde{p}(r), \tilde{ub}, \tilde{lb})$ 
    return  $\tilde{p}(l+1), \dots, \tilde{p}(r-1)$ 

```

```

Function UPDATEUL( $y, \tilde{p}(y), \tilde{ub}, \tilde{lb}$ ):
  for  $i = 1, \dots, N$  do
     $\tilde{ub}(i) \leftarrow \min\{\tilde{ub}(i), \text{ub}(\tilde{v}; y, \tilde{p}(y))\}$ 
     $\tilde{lb}(i) \leftarrow \max\{\tilde{lb}(i), \text{lb}(\tilde{v}; y, \tilde{p}(y))\}$ 
  return  $\tilde{ub}, \tilde{lb}$ 

```

---

intersects the old envelope with a new pointwise upper and lower bound, just as in Equation (3,4). We compute the exact values of  $p(1), \dots, p(\tilde{l})$  by accessing the values of  $v(t_1), \dots, v(t_{\tilde{l}})$  directly. Here, ACCESS( $i, T$ ) returns the exact value of  $v(t_i)$ .<sup>3</sup>

Function PR returns  $\tilde{p}(l+1), \dots, \tilde{p}(r-1)$ , which form an  $(1 + \epsilon)$ -approximation to the true values  $p(l+1), \dots, p(r-1)$ . In function PR, first the algorithm checks the height of the envelope between  $\tilde{p}(l)$  and  $\tilde{p}(r)$ . If the height is less than  $(1 + \epsilon)^2$ , then the algorithm stops,

<sup>3</sup>For simplicity, we assume ACCESS uses  $v$  instead of  $\tilde{v}$ . Under the noisy setting where ACCESS uses  $\tilde{v}$ , the results can be extended by averaging multiple calls to ACCESS.

returning the geometric mean of  $\tilde{ub}$  and  $\tilde{lb}$ . When  $\beta = 1$ , the second stopping condition  $r/l \leq (1 + \epsilon)^2$  is redundant, because for any  $l, r$ , such that  $\tilde{l} \leq l < r < (1 + \epsilon)^2 l$ , we must have  $p(r) \geq p(l)l/r \geq p(l)/(1 + \epsilon)^2$  and  $p(l) \geq p(r)(r - m)/l \geq p(r)/(1 + \epsilon)^2$ , due to Theorem 2. In other words, if condition  $r/l < (1 + \epsilon)^2$  is met, then the height of the envelope has already dropped below  $(1 + \epsilon)^2$ . If the function does not stop, there is at least one point  $i$  between  $l$  and  $r$ , where  $\tilde{ub}(i)/\tilde{lb}(i)$  exceeds  $(1 + \epsilon)^2$ . In this case, we query the function value at a middle point  $c = \text{round}(\sqrt{lr})$ , and recursively call PR on intervals  $(\tilde{p}(l), \dots, \tilde{p}(c))$  and  $(\tilde{p}(c), \dots, \tilde{p}(r))$ .

When  $\beta > 1$ , we stop first when the estimated boundaries  $\tilde{ub}$  and  $\tilde{lb}$  are within  $(1 + \epsilon)^2$ . In this case, we know that true values of  $p$  lie in the range between  $\beta \tilde{ub}$  and  $\tilde{lb}/\beta$ , which are at most  $\beta^2(1 + \epsilon)^2$  apart. It is easy to see that  $\tilde{p} = \sqrt{\tilde{ub} \tilde{lb}}$  provides a  $\beta(1 + \epsilon)$  approximation to any curve in this range, which includes  $p$ . We also stop when  $r/l \leq (1 + \epsilon)^2$ . In this case, we know that the actual height of the envelope (distance between the true boundaries  $ub$  and  $lb$ ) is bounded by  $(1 + \epsilon)^2$  (due to the same reason as why  $r/l \leq (1 + \epsilon)^2$  is redundant when  $\beta = 1$ ). Since all point estimations are at most off by  $\beta$ ,  $\tilde{ub}$  is at most  $\beta ub$  and  $\tilde{lb}$  is at least  $lb/\beta$ . Therefore,  $\tilde{p} = \sqrt{\tilde{ub} \tilde{lb}}$  is a  $\beta(1 + \epsilon)$  approximation. Putting this all together, we have the following theorem:

**Theorem 3.** *Let  $T, \tilde{v}, p, m, \tilde{r}, \tilde{l}, \beta$ , and  $\epsilon$  be as defined before. If the precision function  $p$  is  $(\tilde{r}, m)$ -weak monotonic and QUERY( $i, T, \tilde{v}$ ) is a  $\beta$ -approximation of  $p(i)$  for all  $i$ , then the output of ADASTRAT (Algorithm 1) on input  $(T, \tilde{l}, \tilde{v}, \epsilon)$  is a pointwise  $\beta(1 + \epsilon)$ -approximation of the true precision values  $p(1), \dots, p(N)$ .*

### Sufficient Conditions for Stopping

To understand the complexity of ADASTRAT in terms of the number of calls to QUERY, we analyze the stopping condition of PR, namely, whether the height of the envelope is within than  $(1 + \epsilon)^2$ . We provide two sufficient conditions for stopping. The two lemmas below follow by writing down the pointwise envelopes induced by  $\tilde{p}(l)$  and  $\tilde{p}(r)$  and making use of the fact that  $r > l \geq \tilde{l}$ .

**Lemma 1.** *Under weak monotonicity, if  $\tilde{p}(l)/\tilde{p}(r) \leq (1 + \epsilon)^2$ , the height of the envelope<sup>4</sup> is bounded by  $(1 + \epsilon)^2$ .*

**Lemma 2.** *Under weak monotonicity, if  $(\tilde{p}(r)^r)/(\tilde{p}(l)^l) \leq (1 + \epsilon)^2$ , the height of the envelope is bounded by  $(1 + \epsilon)^2$ .*

The sufficient stopping conditions captured by Lemmas 1 and 2 are two interesting cases of early stopping, in

<sup>4</sup>defined by substituting  $\tilde{p}$  into (3) and (4).

contrast to LOGSTRAT, where  $O(\log_{1+\epsilon} N)$  queries are needed regardless of the shape of the precision function.

Lemma 1 captures the case where  $p$  does not drop too much from  $l$  to  $r$ . This corresponds to the density of  $v(t_i)$  that are 1 staying almost the same for all entries in the range from  $l$  to  $r$ . Notice that the density almost always cannot increase, because of weak monotonicity.

Lemma 2 captures the other extreme, where  $v(t_i)$  is almost always zero for the entries in the range from  $l$  to  $r$ . In this case, the precision function drops at its fastest rate. Our algorithm is able to capture these two cases, stopping early, thereby preventing unnecessary queries.

### Upper Bound on the Number of Query Calls

The above stopping conditions imply that ADASTRAT never makes more calls to QUERY than LOGSTRAT does. Specifically, deferring a proof to the Appendix:

**Theorem 4.** *Under the conditions of Theorem 3, the number of calls to QUERY is at most  $\log_{1+\epsilon}(N/\tilde{l})$ .*

## 4.1 REGRET BOUNDS

Consider an “optimal” algorithm that is guaranteed to produce a  $(1 + \epsilon)$ -approximation of all weak monotonic precision function with as few accesses to QUERY as possible. If this algorithm knew the shape of  $p$  a priori, it could clearly be very smart about where it queries  $p$  in order to generate a guaranteed approximation. The *regret* of any algorithm, then, is defined as how many (multiplicatively) more accesses to QUERY it needs, compared to this optimal algorithm who knows all. We prove that ADASTRAT has a regret of no more than  $\log_2 \log_{1+\epsilon} N$ .

We start by exploring how such an “optimal” algorithm might behave. Suppose it has access to the maximum and minimum precision values,  $p_{\max}$  and  $p_{\min}$ , as well as to  $q_1, \dots, q_K$ , where  $K = \left\lceil \log_{1+\epsilon} \frac{p_{\max}}{p_{\min}} \right\rceil$  and  $q_i$  is the *first* location where  $p$  falls below  $p_{\max}/(1 + \epsilon)^{i-1}$ . Then, as we show next, it suffice for the “optimal” algorithm to make only  $K$  queries, namely to  $p(q_1), \dots, p(q_K)$ , to guarantee a  $(1 + \epsilon)$ -approximation:

**Lemma 3.** *Let  $q_1, \dots, q_K$  be as defined above. Let  $\tilde{p}(j) = p(q_i)$  whenever  $j \in \{q_i, \dots, q_{i+1}\}$ . Then  $\tilde{p}$  is a  $(1 + \epsilon)$ -approximation of  $p$  in the range  $[\tilde{l}', N]$ .*

Lemma 3 guarantees that the “optimal” algorithm does not make too many queries when the precision function decays slowly, i.e.,  $p_{\max}/p_{\min}$  is small. In the other extreme, where the precision function decays in its fastest possible way,  $p(r)r$  stays almost as a constant. In this case, we can prove that the “optimal” algorithm does not make much more queries beyond the ratio of the maximal and minimal values of  $p(r)r$ . Specifically, suppose

the “optimal” algorithm has access to  $s_1, \dots, s_P$ , where  $s_j$  is the first location that function  $p(r)r$  goes above  $p(\tilde{l})\tilde{l}(1+\epsilon)^{j-1}$ . Then  $P = \left\lceil \log_{1+\epsilon} \frac{p(N)N}{p(\tilde{l})\tilde{l}} \right\rceil$ . We can also prove that it suffices for the “optimal” algorithm to query the above  $P$  points to obtain a  $(1 + \epsilon)$ -approximation:

**Lemma 4.** *Let  $s_1, \dots, s_P$  be as defined above. Let  $\tilde{p}(j) = p(s_i)s_i/j$  whenever  $j \in \{s_i, \dots, s_{i+1}\}$ . Then  $\tilde{p}$  is a  $(1 + \epsilon)$ -approximation of  $p$  in the range  $[\tilde{l}, N]$ .*

Proofs of these two lemmas may be found in Appendix B. Putting these together gives a bound on OPT, the number of times the optimal algorithm calls QUERY:

**Theorem 5.** *Under the conditions of Theorem 3,*

$$\text{OPT} \leq \left\lceil \log_{1+\epsilon} \min \left( \frac{p_{\max}}{p_{\min}}, \frac{p(N)N}{p(\tilde{l})\tilde{l}} \right) \right\rceil.$$

Now we state our main regret bound:

**Theorem 6.** *Under the conditions of Theorem 3, ADASTRAT calls QUERY no more than  $(\text{OPT} + 1)(1 + \log_2 \log_{1+\epsilon} N) + 1$  times.*

This says that the number of QUERY calls made by ADASTRAT is roughly  $O(\text{OPT} \cdot \log_2 \log_{1+\epsilon} N)$ . The high level idea to prove Theorem 6 is as follows. Suppose  $r_1, \dots, r_{\text{OPT}}$  are the actual query points of the optimal algorithm. Because ADASTRAT uses a binary search, i.e., it always splits an interval at its geometric middle point. Then it takes ADASTRAT roughly  $O(\log_2 \log_{1+\epsilon} N)$  splits to “locate” one query point  $r_i$  of the optimal algorithm (more precisely, find a point that is sufficiently close to  $r_i$  that guarantees the approximation bound). Hence, the total number of queries of ADASTRAT is bounded by OPT times  $\log_2 \log_{1+\epsilon} N$ . Our actual proof to Theorem 6 is based on walking through the actual calling map of the function PR, where each node in this map represents an actual interval  $(p(l), p(r))$  that PR called. We leave this proof to Appendix B.

Combining Theorems 5 and 6, we immediately obtain the following worst case upper bound for ADASTRAT.

**Corollary 1.** *Under the conditions of Theorem 3, ADASTRAT calls QUERY no more than*

$$O \left( \log_{1+\epsilon} \min \left\{ \frac{p_{\max}}{p_{\min}}, \frac{p(N)N}{p(\tilde{l})\tilde{l}} \right\} \cdot \log_2 \log_{1+\epsilon} N \right)$$

*times.*

Thus, ADASTRAT makes very few queries when  $p$  is flat or decays very fast. In general, when  $p_{\min}$  may be treated as a constant bounded away from zero (e.g., 0.5 or 0.3, as is the case in many practical applications), this shows that ADASTRAT scales essentially as  $\log \log N$ .

## 4.2 ASYMPTOTIC LOWER BOUND

What is the minimum number of calls to the QUERY function needed in order to guarantee an  $(1 + \epsilon)$ -approximation to  $p$ ? We provide a worst-case lower bound, confirming that ADASTRAT is asymptotically optimal in terms of the number of queries.

**Theorem 7.** *Let  $\mathcal{A}$  be any algorithm that accesses the precision function only via the QUERY oracle and, for any  $(\tilde{r}, m)$ -weak monotonic precision function, outputs a curve that  $(1 + \epsilon)$ -approximates it. For any  $\epsilon' > \epsilon$ ,  $\mathcal{A}$  must make at least  $\Omega(\log_{1+\epsilon'} N)$  accesses to QUERY.*

The high level idea of the proof to Theorem 7 is as follows: let  $J \approx \log_{1+\epsilon'} N$ . We carefully construct a family of  $2^J$  valid precision functions  $F = \{f_0, f_1, \dots, f_{2^J-1}\}$  such that, for any two functions  $f_i$  and  $f_j$ , there exists at least one point  $y_{i,j}$ , such that  $f_i(y_{i,j})$  and  $f_j(y_{i,j})$  are separated by more than  $(1 + \epsilon)^2$  (i.e., either  $f_i(y_{i,j}) > (1 + \epsilon)^2 f_j(y_{i,j})$  or  $f_j(y_{i,j}) > (1 + \epsilon)^2 f_i(y_{i,j})$ ). We call this point  $y_{i,j}$  a *separating point* between  $f_i$  and  $f_j$ .

Now suppose algorithm  $\mathcal{A}$  can output a  $(1 + \epsilon)$ -approximation to any given precision function. Starting with an unknown function  $f \in F$ , we can use  $\mathcal{A}$  to identify  $f$ . To do so, we run  $\mathcal{A}$  to obtain a  $(1 + \epsilon)$ -approximate curve  $\tilde{f}$  and examine its values at all separating points. Because  $\tilde{f}$  is a  $(1 + \epsilon)$ -approximation and the distance between two functions at a separating point is more than  $(1 + \epsilon)^2$ , we can unambiguously determine the correct  $f$ . Appendix B includes a detailed construction of the function family  $F$  following this high-level idea.

## 4.3 STRATIFIED SAMPLING FOR QUERY

Suppose ADASTRAT ends up calling QUERY on the  $K$  points  $r_1 < r_2 \dots < r_K$  (generally not in this order) before terminating. By design,  $r_1 > \tilde{l}$  and  $r_K = N$ . Let  $\delta > 0$  and  $\beta > 1 + \eta \geq 1$ . We would like QUERY to provide a  $\beta$ -approximation of  $p(r_i)$  for all  $i \in \{1, \dots, K\}$  with an overall (cumulative) confidence of at least  $1 - \delta$ . To achieve this, QUERY proceeds similarly to LOGSTRAT but with  $\tilde{v}$  rather than  $v$ : it uses as an estimate of  $p(r_i)$  the empirical average of  $\eta$ -noisy estimates  $\tilde{v}(t_j)$  of true labels  $v(t_j)$  for  $s$  uniform random samples  $j$  drawn independently from  $[1, r_i]$ , where:

$$s = \left\lceil \frac{(1 + \eta)^2}{2(\beta - 1 - \eta)^2 p_{\min}^2} \ln \frac{2K}{\delta} \right\rceil \quad (5)$$

Here  $p_{\min}$  is an estimate of (a lower bound on) the minimum value of  $p$  for the given data.<sup>5</sup>

<sup>5</sup>Domain knowledge about the data might allow using a small constant, such as 0.3, for  $p_{\min}$ . Alternatively, one can use an estimate of  $p(N)$  obtained via an *adaptive concen-*

Of course, we don't know  $K$  *a priori*; we will address this shortly. It follows from the Hoeffding bound, Eq. (2), that such an empirical average provides a  $\beta$ -approximation of  $p(r_i)$  with confidence at least  $1 - \delta/K$ . Applying the union bound over all  $i$ , ADASTRAT has overall confidence at least  $1 - \delta$  in its estimates being correct simultaneously at all  $K$  points  $r_1, \dots, r_K$ .

As in LOGSTRAT, since we rely only on the union bound, the samples obtained for  $r_1$  can be (partially) reused as samples for all  $r_i > r_1$ . The amount of reuse is determined by what we will refer to as the *sample density* of an interval in  $\{1, \dots, N\}$ , defined as the ratio of the number of samples in this interval to the size of the interval. Clearly, in order to have  $s$  uniform samples available for  $r_i$ , we must have a sample density of at least  $s/r_i$  in the interval  $[1, r_i]$ . We would like to achieve this while minimizing the total number of samples.

It can be verified that the following *stratified sampling strategy*, henceforth referred to as  $\mathcal{S}$ , results in the minimum overall number of samples while ensuring that the sample density in  $[1, r_i]$  is at least  $s/r_i$ :

$$\begin{aligned} &\text{draw } \left\lceil \frac{(r_1 - \tilde{l})s}{r_1} \right\rceil \text{ samples in } [\tilde{l} + 1, r_1] \\ &\text{draw } \left\lceil \frac{(r_i - r_{i-1})s}{r_i} \right\rceil \text{ samples in } [r_{i-1} + 1, r_i] \text{ for } i > 1 \end{aligned}$$

In LOGSTRAT, the  $K$  points are visited in increasing order, simplifying the implementation of  $\mathcal{S}$  in practice. Further,  $K$  is known *a priori* to be  $\log_{1+\epsilon} N/\tilde{l}$  and  $r_i$  by design equals  $r_{i-1}(1 + \epsilon)$ . This makes it easy to compute the total number of evaluations of  $v$  needed, which sums up to  $\tilde{l} + \frac{\epsilon}{1+\epsilon} s \log_{1+\epsilon} N/\tilde{l}$ , in line with Theorem 1.

The adaptive nature of ADASTRAT makes both the implementation of  $\mathcal{S}$  and a similar calculation challenging. Nevertheless, the following result holds:

**Theorem 8.** *Under the conditions of Theorem 3, for any  $\delta > 0$  and  $\beta > 1 + \eta \geq 1$ , QUERY can be implemented using a stratified sampling strategy such that ADASTRAT provides a  $\beta(1 + \epsilon)$ -approximation of the precision function  $p$  with a confidence of at least  $1 - \delta$  using  $\tilde{l}$  evaluations of the true label  $v$  and:*

$$\left( \left\lceil \frac{r_1 - \tilde{l}}{r_1} \right\rceil + \sum_{i=2}^K \left\lceil \frac{r_i - r_{i-1}}{r_i} \right\rceil \right) \cdot s$$

evaluations of the noisy estimate  $\tilde{v}$ , where  $s = \left\lceil \frac{(1 + \eta)^2}{2(\beta - 1 - \eta)^2 p_{\min}^2} \ln \frac{2K}{\delta} \right\rceil$  and  $r_1, r_2, \dots, r_K$  are the points where ADASTRAT calls QUERY.

*tration inequality*, such as Corollary 1 of Zhao et al. (2016), which provides a dynamic stopping condition to decide how many samples are sufficient, and guarantees that this number,  $z_{\min}$ , is upper bounded by a generalization of Hoeffding's bound with an additional log log term:  $1.8z_{\min}(\gamma^2 p(N)^2 - 0.6 \log(\log_{1.1} z_{\min} + 1)) \leq \ln(12/\delta)$ , where  $1 + \gamma = \frac{\beta}{1 + \eta}$ .

We note that this quantity is bounded above by  $Ks$ , which scales as  $O(K \log K)$ , considering other parameters as constants. This simplified expression is equivalent to *not* reusing samples at all, and thus quite loose in practice. Even so, for datasets requiring  $K \ll \log N$ , this is substantially smaller than the equivalent  $O(\log N \log \log N)$  expression for LOGSTRAT.

Unlike LOGSTRAT, there are two hurdles to implementing a stratified sampling strategy that supports the number of annotations claimed in Theorem 8:  $K$  is unknown in the beginning and ADASTRAT does not visit  $r_1, \dots, r_K$  in increasing order. Let  $r'_1, \dots, r'_K$  be the order in which ADASTRAT actually queries the  $K$  points. To address the first hurdle (unknown  $K$ ), we follow an **iterative deepening approach** and simply begin by assuming  $K = 1$  when querying  $r'_1$ . When ADASTRAT decides to make the next query at  $r'_2$ , we set  $K = 2$ , go back to  $r'_1$  to obtain correspondingly more annotations for it, and then obtain samples for  $r'_2$  based on  $K = 2$ . This process continues, slowly incrementing  $K$  and obtaining more samples at previously queried points to make up for the difference. Since the samples are drawn independently, this yields the same outcome as if we had known the true value of  $K$  in advance and obtained the corresponding number of samples for each  $r'_i$  in a single shot.

To address the second hurdle (queries not in increasing order), we **adapt stratified sampling** as follows. For simplicity of exposition, we assume here that  $K$  is known at the start. When querying  $r'_1$ , we use stratification similar to LOGSTRAT and obtain  $s(r'_1 - \bar{l})/r'_1$  fresh samples in the range  $[\bar{l} + 1, r'_1]$ , inducing a sample density  $s/r'_1$  in this range. When querying  $r'_2$ , there are two possibilities. If  $r'_2 > r'_1$ , then again we obtain fresh samples with density  $s/r'_2$  in the range  $[r'_1 + 1, r'_2]$ , similar to LOGSTRAT. If, on the other hand,  $r'_2 < r'_1$ , we obtain fresh samples instead in the range  $[\bar{l} + 1, r'_2]$  to increase the sample density here from  $s/r'_1$  to  $s/r'_2$ . This process continues with each new query, whose effect is to raise the sample density between the immediately lower queried point and the current point. It can be verified that this process ends with the sample density underlying the expression in Theorem 8, namely  $s/r_i$  in the range  $[r_{i-1} + 1, r_i]$ .

## 5 EXPERIMENTS

For an empirical evaluation, we consider the fully-annotated subset of PPDB 2.0 (Ganitkevitch et al., 2013) used by Sabharwal and Sedghi (2017), henceforth referred to as PPDB-36K. It contains  $N = 35,615$  English language paraphrase pairs for each of which Pavlick et al. (2015) provide a correctness confidence score (thus inducing an overall ranking) obtained using a machine learning algorithm, as well as crowdsourced annotations

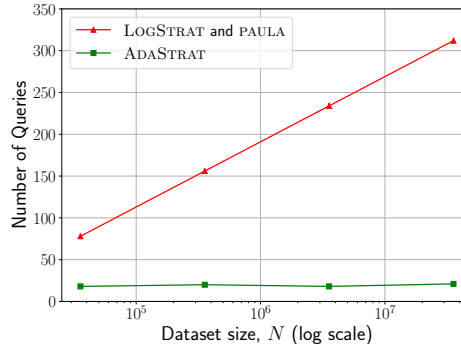


Figure 2: Number of queries of the precision function as dataset size increases. ADASTRAT uses only 21 queries even for PPDB-36K-1000x with 35M items.

of the validity of each pair as a valid paraphrase, on a 5-point scale (1-5). A pair  $t_i$  receiving an average human judgment of at least 3 is considered correct, i.e.,  $v(t_i) = 1$  for such  $i$ , and 0 otherwise. For a direct comparison with prior work, we experiment with noiseless access to  $v$ , i.e.,  $\eta = 0$  and  $\tilde{v} = v$ .

Given the fully-annotated nature of PPDB-36K, the true precision function for it can be easily calculated and used to assess the performance of algorithms such as ADASTRAT. One drawback of this dataset, however, is its relatively small size. To alleviate this while still retaining the property of having a fully-annotated yet realistic dataset, we consider *scaled up variants* of PPDB-36k, created as follows. Using a sliding window of size  $\Delta = 100$ , we compute the running average  $q(i)$  of  $v(t_i)$  for  $1 \leq i \leq N$ , using smaller sliding windows as appropriate when  $i < \Delta$  or  $i > N - \Delta$ . For a scaling factor  $s \in \{10, 100, 1000\}$ , for each  $1 \leq i \leq N$ , we draw  $s$  independent random samples from the Bernoulli distribution with parameter  $q(i)$ . This results in 3 datasets, PPDB-36K-10x, PPDB-36K-100x, and PPDB-36K-1000x, that are 10, 100, and 1000 times larger than PPDB-36K, resp.

### 5.1 SCALING: QUERIES AND ANNOTATIONS

Our first experiment evaluates the number of queries (of the precision function) used by various algorithms, as well as the total number of annotations, as the dataset size is varied from 36K to 35M. We use the following parameters throughout:  $\epsilon = 0.03, \delta = 0.05, \beta = 1.05$ .<sup>6</sup>

Figure 2 shows in a semi-log plot the number of queries needed, as the dataset size grows.<sup>7</sup> As expected, both LOGSTRAT and PAULA use the same number of queries, which starts with 78 for PPDB-36K and grows propor-

<sup>6</sup>Code and data available at <http://allenai.org>.

<sup>7</sup>The exact number of queries is reported in Appendix A.

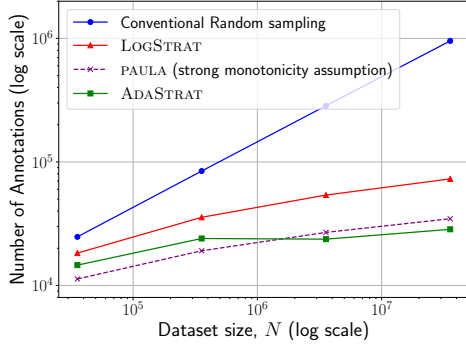


Figure 3: Number of annotations needed by various algorithms as dataset size increases. ADASTRAT needs substantially fewer annotations than competing algorithms that also do not assume strong monotonicity.

tional to  $\log N$ , reaching 312 queries for PPDB-36K-1000x. In contrast, ADASTRAT uses only 18 queries, even for the largest dataset with over 35M items. This aligns with the intuition that the adaptive nature of ADASTRAT allows it to be driven more by the “shape” of the precision function, rather than by the raw data size.

Figure 3 illustrates in a log-log plot the total number of samples used by each method, as the dataset size grows; again, exact numbers may be found in Appendix A. The conventional random sampling baseline asymptotically scales as  $\Theta(\sqrt{N \log N})$ . In line with this, the corresponding blue curve has a slope of roughly 0.5, reaching close to 1M required annotations for PPDB-36K-1000x. LOGSTRAT (red curve) is substantially more practical, growing from 18K annotations to 73K. PAULA (dashed purple line) needs the fewest annotations for the two smaller datasets, but relies the assumption of strong local monotonicity, which is difficult to verify in practice. Finally, ADASTRAT (green line) uses the fewest number of annotations (24K and 28K, resp.) for the two larger datasets. Further, among algorithms that do not rely on strong monotonicity, ADASTRAT has 20%-61% higher annotation efficiency than LOGSTRAT and 41%-97% higher than conventional random sampling.

## 5.2 APPROXIMATION QUALITY

The top plot in Figure 4 shows the approximate precision function  $\tilde{p}$  (green curve) produced by ADASTRAT for PPDB-36K-1000x. Despite querying only 18 points (marked with small red squares) along the true curve, ADASTRAT is able to obtain a remarkably good approximation of the entire true precision function (shown in black, and often occluded by the green curve).

Both LOGSTRAT and PAULA (bottom plot, red) also obtain a similarly tight approximation, except towards the

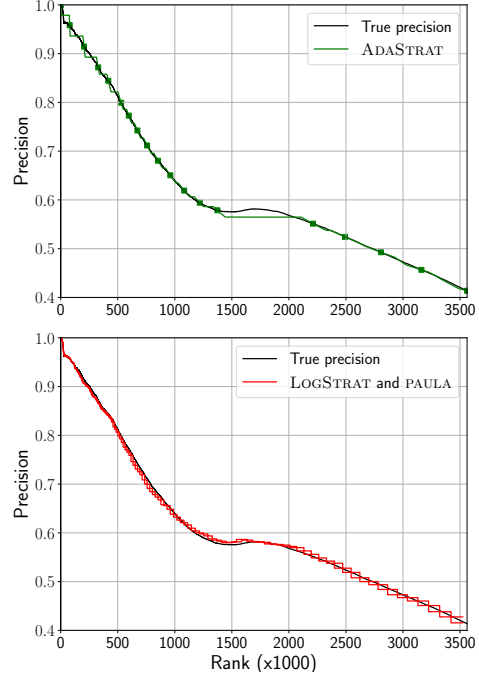


Figure 4: Precision function approximations generated by ADASTRAT (top) and LOGSTRAT and PAULA (bottom), for PPDB-36K-1000x. 18 green markers (top) and 234 corners of red boxes (bottom) are the points queried by the corresponding algorithm.

right end of the curve. Importantly, however, they do so by querying the true precision at 234 points, visually identifiable as the “corners” of the little red boxes. In particular, because of the geometrically spaced nature of the points they query, there is an enormous number of queries in the left part of the curve, which, as illustrated by ADASTRAT’s more spaced-out query points, is unnecessary. This demonstrates the strength of ADASTRAT in exploiting data observations to be smart about where and how often to query the true precision function.

## 6 CONCLUSION

We proposed ADASTRAT, a data-aware algorithm for computing the precision function of massive noisy datasets, with a constant-factor approximation guarantee. ADASTRAT intelligently chooses precision points to query. Under a mild monotonicity assumption, it outputs a guaranteed curve with minimal queries made to the PR curve, scaling very slowly with  $N$ , the number of items. ADASTRAT’s regret w.r.t. an oracle is bounded by  $\log \log N$ . We also provide a matching asymptotic lower bound in terms of the number of queries. On an NLP dataset of 3.5M items, ADASTRAT achieves a close approximation with merely 18 precision queries.

## References

- J. A. Aslam, V. Pavlu, and E. Yilmaz. A statistical method for system evaluation using incomplete judgments. In *SIGIR*, 2006.
- B. Carterette, J. Allan, and R. K. Sitaraman. Minimal test collections for retrieval evaluation. In *SIGIR*, 2006.
- J. Davis and M. Goadrich. The relationship between precision-recall and ROC curves. In *ICML*, 2006.
- T. Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.
- J. Ganitkevitch, B. V. Durme, and C. Callison-Burch. PPDB: The paraphrase database. In *HLT-NAACL*, 2013.
- M. Ghazvininejad, X. Shi, Y. Choi, and K. Knight. Generating topical poetry. In *EMNLP*, 2016.
- W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.
- E. Kanoulas. A short survey on online and offline methods for search quality evaluation. In *RuSSIR*, 2015.
- M. Majnik and Z. Bosnic. Roc analysis of classifiers in machine learning: A survey. *Intell. Data Anal.*, 17: 531–558, 2013.
- E. Pavlick, P. Rastogi, J. Ganitkevitch, B. Van Durme, and C. Callison-Burch. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *ACL*, 2015.
- A. Sabharwal and H. Sedghi. How good are my predictions? efficiently approximating precision-recall curves for massive datasets. In *UAI*, 2017.
- T. Schnabel, A. Swaminathan, P. I. Frazier, and T. Joachims. Unbiased comparative evaluation of ranking functions. In *ICTIR*, 2016.
- P. Welinder, M. Welling, and P. Perona. A lazy man’s approach to benchmarking: Semisupervised classifier evaluation and recalibration. In *CVPR*, 2013.
- E. Yilmaz and J. A. Aslam. Estimating average precision with incomplete and imperfect judgments. In *CIKM*, 2006.
- E. Yilmaz, E. Kanoulas, and J. A. Aslam. A simple and efficient sampling method for estimating ap and ndcg. In *SIGIR*, 2008.
- S. Zhao, E. Zhou, A. Sabharwal, and S. Ermon. Adaptive concentration inequalities for sequential decision problems. In *NIPS*, 2016.

---

# Fast Kernel Approximations for Latent Force Models and Convolved Multiple-Output Gaussian processes

---

**Cristian Guarnizo**

Faculty of Engineering  
Universidad Tecnológica de Pereira  
Pereira, Colombia, 660003

**Mauricio A. Álvarez**

Department of Computer Science  
The University of Sheffield  
Sheffield, UK, S1 4DP

## Abstract

A latent force model is a Gaussian process with a covariance function inspired by a differential operator. Such covariance function is obtained by performing convolution integrals between Green's functions associated to the differential operators, and covariance functions associated to latent functions. In the classical formulation of latent force models, the covariance functions are obtained analytically by solving a double integral, leading to expressions that involve numerical solutions of different types of error functions. In consequence, the covariance matrix calculation is considerably expensive, because it requires the evaluation of one or more of these error functions. In this paper, we use random Fourier features to approximate the solution of these double integrals obtaining simpler analytical expressions for such covariance functions. We show experimental results using ordinary differential operators and provide an extension to build general kernel functions for convolved multiple output Gaussian processes.

## 1 INTRODUCTION

Latent force models (LFMs) [Álvarez et al., 2009] are a type of multiple-output Gaussian processes (GPs) where the covariance function has been derived from physical models. In particular, LFMs assume that each output  $\{f_d(t)\}_{d=1}^D$  can be expressed as the convolution integral of a latent function  $u(t)$ , and a Green's function  $G_d(t)$  associated to a linear dynamical system, one per output,  $f_d(t) = \int_0^t G_d(t - \tau)u(\tau)d\tau$ . Such representation for  $f_d(t)$  introduces a dependency between outputs  $f_d(t)$  and  $f_{d'}(t)$ . For example, if we assume that  $u(t)$  follows a Gaussian process prior with zero mean function and covariance  $k(t, t')$ , due to the linearity of the

integral transform,  $f_d(t)$  and  $f_{d'}(t)$  are jointly Gaussian with a cross-covariance function given as  $k_{f_d, f_{d'}}(t, t') = \int_0^t G_d(t - \tau) \int_0^{t'} G_{d'}(t' - \tau')k(\tau, \tau')d\tau'd\tau$ .

LFMs have been used for uncovering the dynamics of transcription factors in a gene network [Gao et al., 2008], for extrapolating human motion from motion capture data [Álvarez et al., 2013], for segmenting motor primitives in humanoid robotics [Álvarez et al., 2011], for modeling the thermal properties of buildings [Ghosh and et al., 2015], among several other applications for which prior knowledge of a mechanistic model can be coded in the covariance function of a GP. By including physics in the covariance function of a GP, we grant extrapolation abilities to an otherwise interpolation only-model.

In a classical latent force model, the covariance of the latent function  $k(t, t')$  follows an Exponentiated Quadratic (EQ) form, leading to analytical solutions for the cross-covariances  $k_{f_d, f_{d'}}(t, t')$ . However, these solutions are computationally expensive since they involve calculating functions that can only be obtained by numerical methods. For example, using the second order LFM introduced in Álvarez et al. [2009], involves computing the error function  $\text{erf}(\cdot)$  with a complex argument or the Faddeeva function, that require the evaluation of numerical integrals that are expensive to compute.

In this work, we use random Fourier features (RFF) [Rahimi and Recht, 2008] to reduce the mathematical complexity of the expressions involved in the covariance functions of the LFM. In particular, we approximate the calculation of the EQ kernel, with a representation that involves its probability density via the Bochner's theorem. Such representation for the covariance of  $k(\tau, \tau')$  transforms the double integral for  $k_{f_d, f_{d'}}(t, t')$  into two separate integrals that can easily be solved using the Laplace or Fourier transforms. Once the inner integrals are solved (the integrals that depend on  $\tau$  and  $\tau'$ ), the remaining integral is solved using a Monte Carlo approximation with  $S$  samples. The quality of the approximation of the

cross-covariances  $k_{f_d, f_{d'}}(t, t')$  will depend, then, on the number of samples  $S$  used. Additionally, by representing the latent force model kernel using a sum of basis functions, we are able to reduce the computational complexity of inverting the  $ND \times ND$  kernel matrix obtained from the multiple outputs, assuming that each output has  $N$  data observations.

Following a similar procedure, we also introduce a random Fourier feature approximation for the more general convolved multiple output Gaussian process kernel, a model that can be used for multiple-output with no particular known dynamics.

## 2 LATENT FORCE MODELS

Latent force models are Gaussian processes for multiple outputs with the characteristic that their covariance function involves ordinary or partial differential equations. In particular, LFMs assume that each output  $\{f_d(t)\}_{d=1}^D$  can be described using

$$\mathcal{D}_d\{f_d(t)\} = u(t),$$

where  $\mathcal{D}_d$  is the differential operator associated to a linear ordinary differential equation (ODE) or a linear partial differential equation (PDE), and  $u(t)$  is the excitation function. LFMs assume that  $u(t)$  is unknown and place a Gaussian process prior over it. The solution for  $f_d(t)$  follows as

$$f_d(t) = \int_0^t G_d(t - \tau)u(\tau)d\tau, \quad (1)$$

where  $G_d(\cdot)$  corresponds to the Green's function associated to the differential operator  $\mathcal{D}_d$ . The latent force or function  $u(t)$  is unobserved, and follows a Gaussian process prior with zero mean function, and covariance function given by  $k(t, t')$ . Since  $u(t)$  is being transformed by a linear operator,  $f_d(t)$  also follows a Gaussian process with covariance function  $k_{f_d, f_d}(t, t')$ . Furthermore, since all  $f_d(t)$  have a common input  $u(t)$ , it is also possible to compute a cross-covariance function between  $f_d(t)$ , and  $f_{d'}(t')$ ,  $k_{f_d, f_{d'}}(t, t')$ .

Equation (1) can be extended to include additional latent functions with different characteristics, leading to express each output as

$$f_d(t) = \sum_{q=1}^Q S_{d,q} \int_0^t G_d(t - \tau)u_q(\tau)d\tau,$$

where there are  $Q$  latent functions or forces  $\{u_q(t)\}_{q=1}^Q$ , and  $S_{d,q}$  is a sensitivity parameter that accounts for the influence of force  $u_q(t)$  over output  $d$ . Assuming the

independence of these latent forces and that they all follow Gaussian process priors with covariance functions  $k_q(t, t')$ , it is possible to compute the cross-covariance functions  $k_{f_d, f_{d'}}(t, t')$ ,  $\forall d, d' = 1 \dots, D$ . The following general expression can be used to build the covariance  $k_{f_d, f_{d'}}(t, t')$  of a LFM

$$\sum_{q=1}^Q S_{d,q} S_{d',q} \int_0^t G_d(t - \tau) \int_0^{t'} G_{d'}(t' - \tau') \times k_q(\tau, \tau') d\tau' d\tau. \quad (2)$$

Depending on the form for the covariance function for  $k_q(t, t')$ , it is possible to find a closed-form expression for  $k_{f_d, f_{d'}}(t, t')$ . A common option for  $k_q(\tau, \tau')$  is the Exponentiated Quadratic form

$$k_q(\tau, \tau') = \exp\left[-\frac{(\tau - \tau')^2}{\ell_q^2}\right],$$

where  $\ell_q$  is known as the length-scale parameter.

LFMs have mostly being used for multiple output regression. In this case, the observed output  $d$ ,  $y_d(t)$ , is assumed to follow a Gaussian likelihood,  $y_d(t) = f_d(t) + \epsilon_d$ , where  $\epsilon_d \sim \mathcal{N}(0, \sigma_d^2)$ .

## 3 FEATURE EXPANSIONS FOR KERNELS DERIVED FROM LATENT FORCE MODELS

In order to scale kernel machines, Rahimi and Recht [2008] introduced the idea of random Fourier features to approximate a kernel function using inner products between basis functions. Parameters of these basis functions are sampled from a distribution associated to the kernel function. We are particularly interested in the approximation for the EQ kernel, which has been commonly used in LFMs. The idea is to replace the EQ kernel that is usually assumed for  $k_q(\tau, \tau')$  by providing a random Fourier feature representation for it via the Bochner's theorem,

$$k_q(\tau, \tau') = e^{-\frac{(\tau - \tau')^2}{\ell_q^2}} = \int p(\lambda) e^{j(\tau - \tau')\lambda} d\lambda, \quad (3)$$

where  $p(\lambda) = \mathcal{N}(\lambda|0, \frac{2}{\ell_q^2})$ . A key insight from Rahimi and Recht [2008] was to use a finite approximation for  $k_q(\tau, \tau')$  by using Monte Carlo sampling to solve the above integral over  $\lambda$ ,

$$\begin{aligned} k_q(\tau, \tau') &\approx \frac{1}{S} \sum_{s=1}^S e^{j\lambda_s \tau} e^{-j\lambda_s \tau'}, \\ &= \frac{1}{S} \sum_{s=1}^S v(\tau, \lambda_s) v^*(\tau, \lambda_s), \end{aligned}$$



where  $S$  is the number of Monte Carlo samples,  $v(\tau, \lambda_s)$  is a basis function with parameter  $\lambda_s$ ,  $v^*(\tau, \lambda_s)$  is the complex conjugate of  $v(\tau, \lambda_s)$ , and  $\lambda_s \sim p(\lambda)$ . Since the kernel function is a real function, the real part of the product  $v(\tau, \lambda_s)v^*(\tau, \lambda_s)$  is used instead.

Using the expression for  $k_q(\tau, \tau')$  in Eq. (3) inside the expression for the cross-covariance function for the LFM,  $k_{f_d f_{d'}}(t, t')$ , we get

$$\sum_{q=1}^Q S_{d,q} S_{d',q} \int_0^t G_d(t-\tau) \int_0^{t'} G_{d'}(t'-\tau') \times \int p(\lambda) e^{j(\tau-\tau')\lambda} d\lambda d\tau d\tau'.$$

Organizing the above expression we obtain

$$\sum_{q=1}^Q S_{d,q} S_{d',q} \int p(\lambda) v_d(t, \theta_d \lambda) v_{d'}^*(t', \theta_{d'} \lambda) d\lambda, \quad (4)$$

with

$$v_d(t, \theta_d, \lambda) = \int_0^t G_d(t-\tau) e^{j\lambda\tau} d\tau,$$

where  $\theta_d$  makes reference to the parameters of the Green's function  $G_d(\cdot)$ . Also,  $v_{d'}^*(t', \theta_{d'} \lambda)$  is the complex conjugate for  $v_{d'}(t', \theta_{d'} \lambda)$ . The integrals over  $t$  and  $t'$  above can be solved using the Laplace transform  $\mathcal{L}\{\cdot\}$

$$\begin{aligned} v_d(t, \theta_d, \lambda) &= \mathcal{L}^{-1} \mathcal{L} \left\{ \int_0^t G_d(t-\tau) e^{j\lambda\tau} d\tau \right\} \\ &= \mathcal{L}^{-1} \left\{ \mathcal{G}_d(s) \mathcal{L} \{ e^{j\lambda\tau} \} \right\}, \end{aligned}$$

where  $\mathcal{G}_d(s)$  is the Laplace transform for  $G_d(t)$ . The operator  $\mathcal{L}^{-1}\{\cdot\}$  refers to the inverse Laplace transform. Furthermore, notice that when  $G_{d'}(\cdot)$  is a real function, we can compute  $v_{d'}^*(t', \theta_{d'} \lambda) = v_{d'}(t', \theta_{d'} \lambda)$ .

Similarly to Rahimi and Recht [2008], we use Monte Carlo sampling to approximate the integral over  $\lambda$  in Eq. (4), leading to

$$\sum_{q=1}^Q \frac{S_{d,q} S_{d',q}}{S} \left[ \sum_{s=1}^S v_d(t, \theta_d, \lambda_s) v_{d'}^*(t', \theta_{d'} \lambda_s) \right],$$

where  $\lambda_s \sim p(\lambda)$ .

The steps to compute a RFF approximation of the LFM kernel are

1. Compute  $v_d(t, \theta_d, \lambda) = \int_0^t G_d(t-\tau) e^{j\lambda\tau} d\tau$  using the Laplace transform.

2. Compute the RFF approximation for the LFM covariance function  $k_{f_d f_{d'}}(t, t')$  using

$$\sum_{q=1}^Q \frac{S_{d,q} S_{d',q}}{S} \left[ \sum_{s=1}^S v_d(t, \theta_d, \lambda_s) v_{d'}^*(t', \theta_{d'} \lambda_s) \right],$$

where  $\lambda_s \sim p(\lambda)$ . The distribution we use to sample from,  $p(\lambda)$ , depends on the kernel assumed for the latent forces  $u_q(t)$ .

Interestingly,  $v_d(t, \theta_d, \lambda)$  represents the response of the dynamical system to the excitation  $e^{j\lambda t}$  up to time  $t$ . We will occasionally refer to this random feature as a *random Fourier response feature* (RFRF).

In different applications of LFMs, we need to perform inference over the latent forces  $u_q(t)$ . Inference over  $u_q(t)$  requires the evaluation of the cross-covariance functions  $k_{f_d, u_q}(t, t')$ . Such cross-covariances are also important in schemes that reduce computational complexity in convolved multiple output Gaussian processes, where the underlying process  $u_q(t)$  evaluated at a discrete set of input locations serve the purpose of *inducing variables* [Álvarez et al., 2010, Álvarez and Lawrence, 2011]. The approximation of  $k_{f_d, u_q}(t, t')$  using RFFs is given by

$$k_{f_d, u_q}(t, t') = \frac{1}{S} \sum_{s=1}^S v_d(t, \theta_d, \lambda_s) e^{-j\lambda_s t'}.$$

## 4 HYPERPARAMETER SELECTION AND COMPUTATIONAL COMPLEXITY

Let us assume, we are given observations  $\{\mathbf{y}, \mathbf{X}\} = \{\mathbf{y}_d, \mathbf{X}_d\}_{d=1}^D$  (each  $\mathbf{y}_d \in \mathbb{R}^N$  and  $\mathbf{X}_d \in \mathbb{R}^{N \times p}$ ), and we want to learn the hyperparameters of the kernel function,  $\{\{\theta_d, \sigma_d^2\}_{d=1}^D, \{\ell_q\}_{q=1}^Q\}$ , that allow us to explain  $\mathbf{y}$ . With that in mind, the hyperparameters can be learned from the log-marginal likelihood [Rasmussen and Williams, 2006]

$$\begin{aligned} \log p(\mathbf{y}|\mathbf{X}) &= -\frac{ND}{2} \log(2\pi) - \frac{1}{2} \mathbf{y}^\top (\mathbf{K}_{f,f} + \mathbf{\Sigma})^{-1} \mathbf{y} \\ &\quad - \frac{1}{2} \log |\mathbf{K}_{f,f} + \mathbf{\Sigma}|, \end{aligned} \quad (5)$$

where  $\mathbf{\Sigma}$  is a diagonal matrix containing the variances of the noise level per output, and  $\mathbf{K}_{f,f} \in \mathbb{R}^{ND \times ND}$  is a block-wise matrix with blocks calculated using (2). As it is usual, we can use a gradient-based optimization procedure to estimate the hyperparameters that maximize the log-marginal likelihood leading to the infamous computational complexity of  $\mathcal{O}(D^3 N^3)$ .

However, notice that by the elegance of the RFF representation, the covariance matrix can instead be approximated

as  $\mathbf{K}_{\mathbf{f},\mathbf{f}} = \mathbb{R} \{ \Phi \Phi^H \}$ , where  $\Phi \in \mathbb{C}^{ND \times QS}$  has entries  $v_d(t, \theta_d, \lambda_s)$ , and  $\Phi^H$  is the conjugate transpose of  $\Phi$ . Furthermore, the covariance matrix can be re-written as  $\mathbf{K}_{\mathbf{f},\mathbf{f}} = \Phi_c \Phi_c^T$ , with  $\Phi_c = [\mathbb{R}\{\Phi\} \ \text{I}\{\Phi\}] \in \mathbb{C}^{ND \times 2QS}$ . Using the matrix inversion and determinant lemmas, we express the log-marginal likelihood as

$$\begin{aligned} \log p(\mathbf{y}|\mathbf{X}) &= -\frac{1}{2} \log |\Sigma| - \frac{1}{2} (\mathbf{y}^T \Sigma^{-1} \mathbf{y} - \boldsymbol{\alpha}^T \mathbf{A}^{-1} \boldsymbol{\alpha}) \\ &\quad - \frac{1}{2} \log |\mathbf{A}| - \frac{ND}{2} \log(2\pi), \end{aligned} \quad (6)$$

with  $\mathbf{A} = \mathbf{I} + \Phi_c^T \Sigma^{-1} \Phi_c$  and  $\boldsymbol{\alpha} = \Phi_c^T \Sigma^{-1} \mathbf{y}$ , effectively reducing computational complexity from  $\mathcal{O}(D^3 N^3)$  to  $\mathcal{O}(DNQ^2 S^2)$ , which is now linear with respect to the data size.

Alternatively, one could couple the computation of the kernel functions  $k_{f_d, f_{d'}}(t, t')$  and  $k_{f_d, u_q}(t, t')$  through random Fourier response features, with (i) any of the different computationally efficient approximations for optimizing the log-marginal likelihood in convolved multiple-output Gaussian process [Álvarez and Lawrence, 2011], or (ii) a lower bound on the log-marginal likelihood through a variational approximation [Álvarez et al., 2010]. Both styles of approximations require the specification of  $K$  inducing variables.

## 5 FAST KERNEL BUILDING FROM ORDINARY DIFFERENTIAL EQUATIONS

Let us assume we are interested in analyzing an ODE of order  $P$  given as

$$\mathcal{D}_d^{(P)} \{f_d(t)\} = \sum_{q=1}^Q S_{d,q} u_q(t),$$

where the differential operator  $\mathcal{D}_d^{(P)}$  is defined as

$$\mathcal{D}_d^{(P)} = a_0 \frac{d^P}{dt^P} + a_1 \frac{d^{P-1}}{dt^{P-1}} + \dots + a_{P-1} \frac{d}{dt} + a_P.$$

The Laplace transform of the Green's function  $G_d(t)$  for the above ODE can be found as

$$\begin{aligned} \mathcal{G}_d(s) &= \frac{1}{a_0} \frac{1}{s^P + \frac{a_1}{a_0} s^{P-1} + \dots + \frac{a_P}{a_0}} \quad (7) \\ &= \frac{1}{a_0} \frac{1}{(s - s_1)(s - s_2) \dots (s - s_P)}, \end{aligned}$$

where the  $s_i$ 's represent the roots of the polynomial given in the denominator of (7). Additionally, the Laplace transform for  $\mathcal{L}\{e^{j\lambda\tau}\} = \frac{1}{s - j\lambda}$ . We can use a partial-fraction expansion for  $\mathcal{G}_d(s)$ , and then apply the inverse

Laplace transform over the product  $\mathcal{G}_d(s) \mathcal{L}\{e^{j\lambda\tau}\}$  to find  $v_d(t, \theta_d, \lambda)$ .

Interestingly, if all the roots  $s_1, \dots, s_P$  are distinct real or distinct complex, and  $s_{P+1} = j\lambda$  (the additional root obtained from  $\mathcal{L}\{e^{j\lambda\tau}\}$ ), the random Fourier response feature  $v_d(t, \theta_d, \lambda)$  can be expressed as

$$\frac{1}{a_0} \mathcal{L}^{-1} \left\{ \sum_{p=1}^{P+1} \frac{A_p}{(s - s_p)} \right\} = \frac{1}{a_0} \sum_{p=1}^{P+1} A_p e^{s_p t},$$

where each coefficient  $A_p$  is calculated as

$$A_p = \frac{1}{\prod_{\forall i \neq p} (s_p - s_i)}, \quad (8)$$

and, as before,  $s_{P+1} = j\lambda$ .

Next, we show some examples of the expressions obtained for the random Fourier response features associated to the ODE of first and second orders. Besides, for all ODE experiments the hyperparameters are learned using the variational approach described in Álvarez et al. [2010] and they were carried out using a single core of an AMD FX-8350 @ 4.0 GHz. We also include measures of the time required to evaluate the objective function and its gradients to compare the time cost induced by the evaluation of the different covariance functions. Code to replicate the following experiments is available at [github.com/cdguarnizo/kff\\_lfm](https://github.com/cdguarnizo/kff_lfm).

### 5.1 FIRST-ORDER MODEL (ODE1)

For the first-order ODE we have the following equation

$$\mathcal{D}_d^{(1)} \{f_d(t)\} = \frac{df_d(t)}{dt} + \gamma_d f_d(t) = \sum_{q=1}^Q u_q(t),$$

from which the Laplace transform is given by  $\mathcal{G}_d(s) = \frac{1}{s + \gamma_d}$ . We then have  $s_1 = -\gamma_d$ , and  $s_2 = j\lambda$ . The random Fourier response feature for the  $d$ -th output function of a first-order ODE is obtained as

$$\begin{aligned} v_d^{(1)}(t, \theta_d, \lambda) &= A_1 e^{s_1 t} + A_2 e^{s_2 t} \\ &= -\frac{e^{-\gamma_d t}}{\gamma_d + j\lambda} + \frac{e^{j\lambda t}}{\gamma_d + j\lambda} \\ &= \frac{e^{j\lambda t} - e^{-\gamma_d t}}{\gamma_d + j\lambda}. \end{aligned}$$

Next, we compare the performance of the first order ODE described in Gao et al. [2008] with the kernel obtained by using the above random Fourier response feature for interpolation of Air temperature.

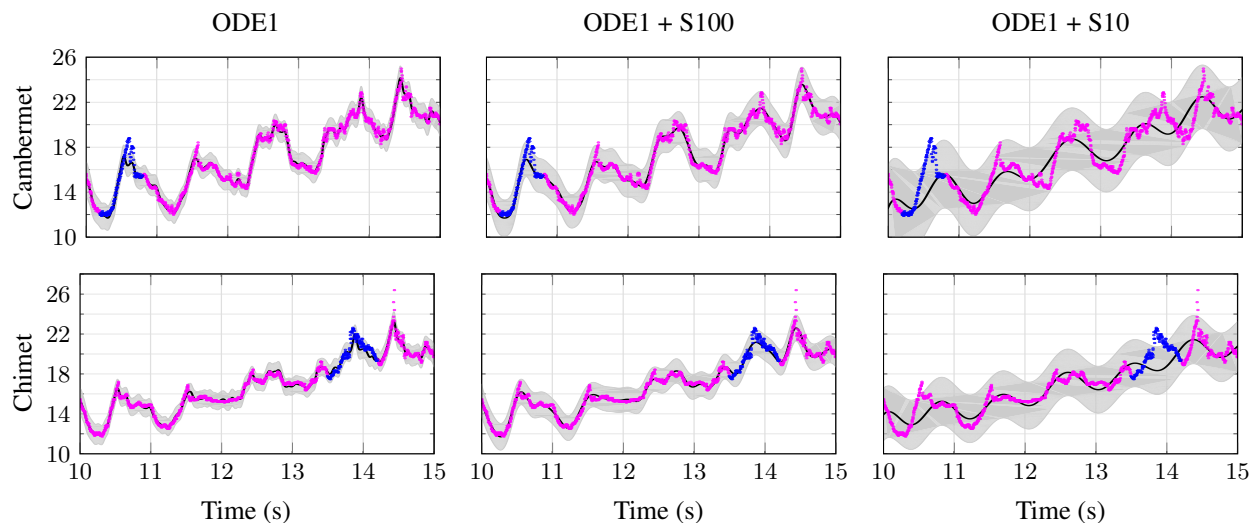


Figure 1: Comparison of the predictive GPs, for the air temperature experiment, using the standard LFM (first column) and the RFRF approximation for  $S = 100$  (second column) and  $S = 10$  samples (third column). Training data is represented using red dots and Test data using blue dots. The black line in the mean over the predictive GP function, and the shaded region denotes two times the standard deviation.

**Air temperature** Here, we consider the problem of modeling and predicting air temperature time series from a network sensor located at the south coast of England. The dataset consists of temperature measurements at four locations known as Bramblemet, Sotonmet, Cambermet and Chimet.<sup>1</sup> The air temperatures are measured during the period from July 10 to July 15, 2013. Specifically, we adopt the same experiment (train and test data) used in Nguyen and Bonilla [2014] and described in Tab. 1. The variational approach is configured with 200 inducing variables, six latent forces and the maximum number of iterations for the optimization procedure is set to 500.

Table 1: Number of training and test data-points considered on the air temperature experiment.

#	Name	Training	Test
1	Bramblemet	1425	0
2	Cambermet	1268	173
3	Chimet	1235	201
4	Sotonmet	1097	0

Table 2 reports the predictive performance using the covariance functions build from the LFM and the proposed RFRF. Note that for a low number of samples  $S$ , the proposed approach presents the worst performance. This is because the more samples we use the better the mean of predictive GP is able to fit the coarse behavior from the observed data, as shown in figure 1. Interestingly, the

<sup>1</sup>Weather data can be found in <http://www.bramblemet.co.uk>.

RFRF starts to outperform the standard one, using only 50 or 100 samples with about half of the time required by the original covariance function.

Table 2: Results on air temperature data.

Kernel	Cambermet		Chimet		Time [s]
	NMSE	NLPD	NMSE	NLPD	
ODE1+S10	0.74	3.26	0.58	1.53	1.89
ODE1+S20	0.45	1.95	0.93	1.75	2.09
ODE1+S50	<b>0.08</b>	<b>1.10</b>	0.21	1.08	2.68
ODE1+S100	0.12	1.18	<b>0.12</b>	<b>0.82</b>	3.93
ODE1	0.11	1.37	0.19	0.99	6.28

## 5.2 SECOND-ORDER MODEL (ODE2)

As a second example of a random Fourier feature representation of a LFM, we use a second-order ordinary differential operator  $\mathcal{D}_d^{(2)}\{\cdot\}$  that represents, e.g., a mass-spring-damper system. The second-order operator is given as

$$\mathcal{D}_d^{(2)} = m_d \frac{d^2}{dt^2} + c_d \frac{d}{dt} + b_d,$$

where  $m_d$ ,  $c_d$  and  $b_d$  are the mass, damper and spring constants, respectively. From the above equation, we obtain the Laplace transform of the Green's function as

$$\mathcal{G}_d(s) = \frac{1}{m_d s^2 + \frac{c_d}{m_d} s + \frac{b_d}{m_d}}.$$

Following the procedure described above, it can be shown that the random Fourier response feature for the  $d$ -th out-

put is given by

$$v_d^{(2)}(t, \theta_d, \lambda) = \frac{1}{m_d} \left[ A_1 e^{s_1 t} + A_2 e^{s_2 t} + A_3 e^{s_3 t} \right],$$

where

$$s_1, s_2 = -\frac{c_d}{2m_d} \pm \sqrt{\frac{c_d^2}{4m_d^2} - \frac{b_d}{m_d}},$$

are the roots of the polynomial obtained from the second-order ODE, and  $s_3 = j\lambda$  corresponds to the root induced by the excitation  $e^{j\lambda t}$ . Note that the coefficients  $A_1$  and  $A_2$  were calculated using (8). Furthermore, if  $c_d^2 > 4m_d b_d$  then the roots  $s_1$  and  $s_2$  are real, and the model’s response is known as “overdamped”. When  $c_d^2 < 4m_d b_d$  the roots are a pair of complex conjugates, and the response is known as “underdamped”.

Figure 2 shows the covariance matrices for a two-output LFM using the standard expression for the covariance function in Álvarez et al. [2009], and the kernel obtained by using the random Fourier response features for the ODE2,  $v_d^{(2)}(t, \theta_d, \lambda)$ , based on  $S = 100$  samples. In this example, we consider that the first output follows an overdamped response, while the second output has an underdamped response. Additionally, the input times comprises 100 values in the range from 0s to 3s for each output. Just to have a quantitative measure of the approximation obtained by the RFRF approach, the Frobenius norm between the covariance matrices shown in figure 2 is 239.1. However, for  $S = 10^5$  samples, the Frobenius norm is 5.8, which states that we are able to reduce the approximation error by the cost of increasing the number of samples. Note that the covariance values are similar,

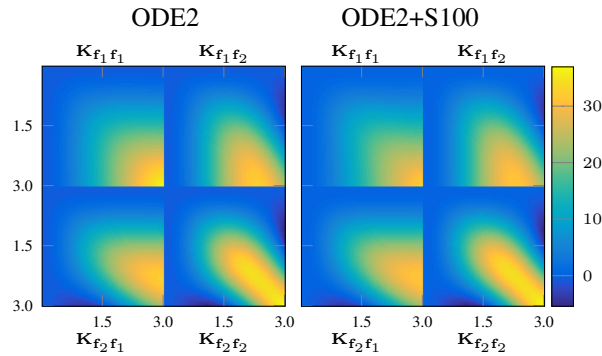


Figure 2: Comparison of the covariance matrix evaluation using the standard LFM and the RFRF.

indicating that the correlation between the outputs and within each output is preserved and well approximated by the inner products of the random features  $v_d^{(2)}(t, \theta_d, \lambda)$ .

For the following experiments, we consider two motion capture (MOCAP) datasets,<sup>2</sup> which consist of measured joint angles from different types of motions. Additionally, the variational approach is configured with 25 inducing variables, six latent forces and the maximum number of iterations set to 500.

**MOCAP - Golf swing** In this experiment, we consider the movement “Golf swing” performed by subject 64 motion 01. From the 62 available channels, we selected 56 each having 448 samples, except for two outputs where 81 consecutive samples were considered for testing purposes. The complete dataset for training consists of 24926 data-points.

Table 3: Results for Golf Swing dataset.

Kernel	root-Ypos		lowerback-Yrot		Time [s]
	NMSE	NLPD	NMSE	NLPD	
ODE2+S10	0.39	-2.23	0.98	2.69	2.20
ODE2+S20	0.24	-2.35	1.49	4.30	3.02
ODE2+S50	0.17	-2.39	<b>0.27</b>	<b>1.17</b>	4.59
ODE2+S100	0.12	<b>-2.45</b>	0.32	1.34	9.31
ODE2	<b>0.11</b>	-2.39	3.19	7.26	28.96

Table 3 reports the predictive performance using the covariance functions built from the LFM and the proposed RFRF. In this experiment, the RFRF approximations fit better the testing data for output “lowerback-Yrot”, as shown in figure 3. In contrast, output “root-Ypos” testing data is best fitted by the standard LFM. In summary, the models learned using 50 and 100 samples not only performed better than the standard LFM, but also their cost time is reduced by a fraction of three and six, respectively.

**MOCAP - Walk** For this experiment, we consider the movement “walk” from subject 02 motion 01. From the 62 available channels, we selected 48 each having 343 samples, except for 121 and 105 consecutive samples of two outputs that were considered for testing purposes. The complete dataset for training consists of 16238 data-points.

Table 4 reports the predictive performance for the testing data used in “walk” experiment. Output “lowerback-Yrot” missing data is best fitted by the standard LFM. However, the testing data for output “lradius-Xrot” is best fitted by the proposed RFRF approach, as shown in figure 3. Interestingly, for this experiment, the observed data are smooth, which can be fitted with adequate accuracy using 10 or 20 samples using the RFRF approach.

<sup>2</sup>MOCAP datasets are available at <http://mocap.cs.cmu.edu/>.

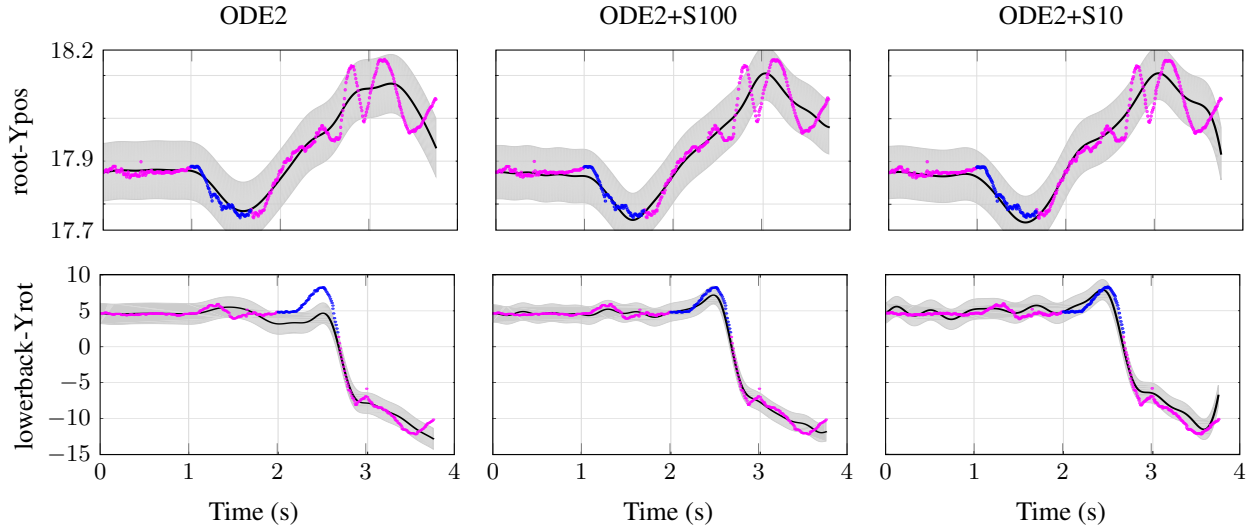


Figure 3: Comparison of the predictive GPs, for the Golf swing experiment, using the standard LFM (first column) and the RFRF approximation for  $S = 100$  (second column) and  $S = 10$  samples (third column). Training data is represented using red dots and Test data using blue dots. The black line in the mean over the predictive GP function, and the shaded region denotes two times the standard deviation.

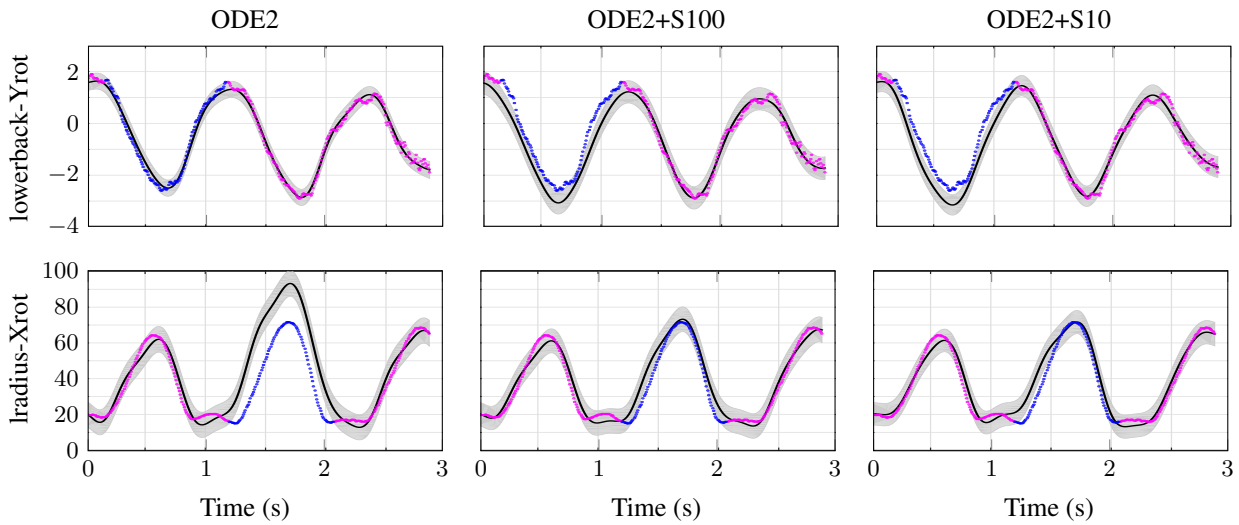


Figure 4: Comparison of the predictive GPs obtained for the the motion “Walk” using the standard LFM (first column) and the RFF approximation for  $S = 100$  (third column) and  $S = 10$  samples (third column). Training data is represented using red dots and Test data using blue dots. The black line in the mean over the predictive GP function, and the shaded region denotes two times the standard deviation.

We remark that the evaluation of the covariance function ODE2 is the most expensive one because it requires the evaluation of the Faddeeva function. Hence, the computation time per iteration is reduced using the inner product of  $v_d^{(2)}(t, \theta_d, \lambda)$ .

## 6 RANDOM FOURIER FEATURES FOR CONVOLVED MULTIPLE OUTPUT GAUSSIAN PROCESSES

Convolution processes can be used to build kernels for vector-valued functions, as reviewed in Álvarez and Lawrence [2011]. Following similar expressions to the ones in section 3, an output  $f_d(\mathbf{x})$ , with  $\mathbf{x} \in \mathbb{R}^p$ , can be modeled as a convolution integral of general smooth-

Table 4: Results for Walk Dataset.

Kernel	lowerback-Yrot		Iradius-Xrot		Time [s]
	NMSE	NLPD	NMSE	NLPD	
ODE2+S10	0.21	5.05	0.12	1.06	1.45
ODE2+S20	0.22	2.09	0.49	<b>0.87</b>	2.04
ODE2+S50	0.22	4.77	0.19	5.28	3.24
ODE2+S100	0.18	3.35	<b>0.09</b>	3.86	6.09
ODE2	<b>0.02</b>	<b>-0.10</b>	0.99	19.63	19.67

ing kernels  $\{G_{d,q}^i(\cdot)\}_{d=1,q=1,i=1}^{D,Q,R_q}$ , and latent processes  $\{u_q^i(\mathbf{x})\}_{q=1,i=1}^{Q,R_q}$

$$f_d(\mathbf{x}) = \sum_{q=1}^Q \sum_{i=1}^{R_q} \int_{\mathcal{X}} G_{d,q}^i(\mathbf{x} - \mathbf{z}) u_q^i(\mathbf{z}) d\mathbf{z},$$

where, according to Álvarez and Lawrence [2011], the variable  $R_q$  makes reference to the number of latent functions  $u_q$  that share the same covariance function  $k_q(x, x')$ , although are sampled independently. Granted that the  $u_q^i(\mathbf{x})$  are independent GPs with zero mean and covariance functions  $\text{cov}[u_q^i(\mathbf{x}), u_{q'}^j(\mathbf{x}')] = k_q(\mathbf{x}, \mathbf{x}') \delta_{q,q'} \delta_{i,j}$ , where  $\delta_{q,q'}$  and  $\delta_{i,j}$  are Kronecker deltas, the cross-covariance between  $f_d(\mathbf{x})$ , and  $f_{d'}(\mathbf{x}')$ ,  $k_{f_d, f_{d'}}(\mathbf{x}, \mathbf{x}')$ , follows a familiar form

$$\sum_{q=1}^Q \sum_{i=1}^{R_q} \int_{\mathcal{X}} G_{d,q}^i(\mathbf{x} - \mathbf{z}) \int_{\mathcal{X}} G_{d',q}^i(\mathbf{x}' - \mathbf{z}') k_q(\mathbf{z}, \mathbf{z}') d\mathbf{z} d\mathbf{z}'.$$

This covariance function subsumes several other covariance functions proposed in the literature for multiple output GPs, including the linear model of coregionalization [Álvarez and Lawrence, 2011].

A general purpose expression for  $k_{f_d, f_{d'}}(\mathbf{x}, \mathbf{x}')$  can be obtained by assuming that both  $G_{d,q}^i(\cdot)$  and  $k_q(\cdot, \cdot)$  follow Gaussian forms. The cross-covariance  $k_{f_d, f_{d'}}(\mathbf{x}, \mathbf{x}')$  would then also follow a Gaussian form after solving the double integration for  $\mathcal{X} = \mathbb{R}^p$ . The authors in Álvarez and Lawrence [2011] provided a closed-form expression for  $k_{f_d, f_{d'}}(\mathbf{x}, \mathbf{x}')$  for this case, when  $R_q = 1$ .

We can also use random Fourier features for  $k_q(\cdot, \cdot)$  in the expression above. For the Gaussian case, since the integrations are over  $\mathbb{R}^p$ , we use a Fourier transform instead of a Laplace transform as it was the case for the LFM. Let us assume that both  $G_{d,q}(\cdot)$  and  $k_q(\cdot, \cdot)$  follow Gaussian forms,

$$G_{d,q}(\boldsymbol{\tau}) = \exp \left[ -\frac{P_d}{2} \boldsymbol{\tau}^\top \boldsymbol{\tau} \right],$$

$$k_q(\mathbf{z}, \mathbf{z}') = \exp \left[ -\frac{1}{\ell_q^2} (\mathbf{z} - \mathbf{z}')^\top (\mathbf{z} - \mathbf{z}') \right],$$

where  $P_d$  is the inverse-width associated to the smoothing kernel for output  $d$ , and  $\ell_q$  is the length-scale for the kernel of the latent function. The cross-covariance  $k_{f_d, f_{d'}}(\mathbf{x}, \mathbf{x}')$  follows as

$$\sum_{q=1}^Q S_{d,q} S_{d',q'} \int_{\mathcal{X}} \int_{\mathcal{X}} \exp \left[ -\frac{P_d}{2} (\mathbf{x} - \mathbf{z})^\top (\mathbf{x} - \mathbf{z}) \right] \times \exp \left[ -\frac{P_{d'}}{2} (\mathbf{x}' - \mathbf{z}')^\top (\mathbf{x}' - \mathbf{z}') \right] k_q(\mathbf{z}, \mathbf{z}') d\mathbf{z} d\mathbf{z}'.$$

Using again the Bochner's theorem for  $k_q(\mathbf{z}, \mathbf{z}')$ ,

$$k_q(\mathbf{z}, \mathbf{z}') = \int p(\boldsymbol{\lambda}) \exp(j\boldsymbol{\lambda}^\top (\mathbf{z} - \mathbf{z}')) d\boldsymbol{\lambda}.$$

Placing this form for  $k_q(\mathbf{z}, \mathbf{z}')$  inside the expression for  $k_{f_d, f_{d'}}(\mathbf{x}, \mathbf{x}')$ , and solving the integral over  $\boldsymbol{\lambda}$  using Monte Carlo, we get that  $k_{f_d, f_{d'}}(\mathbf{x}, \mathbf{x}')$  follows

$$\sum_{q=1}^Q \frac{S_{d,q} S_{d',q'}}{S} \phi_d^\top(\mathbf{x}, P_d, \boldsymbol{\Lambda}_q) \phi_{d'}^*(\mathbf{x}', P_{d'}, \boldsymbol{\Lambda}_q),$$

where

$$\phi_d(\mathbf{x}, P_d, \boldsymbol{\Lambda}_q) = \exp \left[ -\frac{1}{2P_d} \mathbf{b}_q + j\boldsymbol{\Lambda}_q \mathbf{x} \right],$$

with  $\mathbf{b}_q = \sum_j (\boldsymbol{\Lambda}_q \odot \boldsymbol{\Lambda}_q)_{i,j} \in \mathbb{R}^{S \times 1}$ , being  $\odot$  the Hadamard product, and  $\boldsymbol{\Lambda}_q = \frac{1}{\ell_q} \mathbf{Z} \in \mathbb{R}^{S \times p}$ , where the entries of the matrix  $\mathbf{Z}$  are sampled from  $\mathcal{N}(0, 1)$ . Hyperparameters  $\theta_d$  and  $\ell_q$  can be estimated using similar procedures to the ones described in section 4.

**SARCOS** As an illustration of the use of the kernel above, we performed an experiment on a subset of the SARCOS dataset described in the book by Rasmussen and Williams [2006].<sup>3</sup> We use a subset of the data in the file `sarcos_inv.mat`. In particular, we randomly select 10000 data observations that include two outputs, corresponding to the first two joint torques, and the first seven inputs, corresponding to the joint positions. We then randomly select 1000 observations for the second output as the test data. We use the remaining 19000 for training, this is, for hyperparameter optimization. We compare the performance between the kernel proposed in Álvarez and Lawrence [2011] (CMOC) and the kernel obtained using the random Fourier response features for different values of  $S$ . For the CMOC we optimize the marginal likelihood as in Eq. (5), whereas for the RFRF, we use the marginal likelihood as in Eq. (6). Table 5 reports the NMSE and NLPD for the 1000 test observations for the second output. These experiments were carried out using a single core of an Intel Xeon E5-2630v3 @ 2.4 GHz.

<sup>3</sup>Available at <http://www.gaussianprocess.org/gpml/data/>

Table 5: Results for the Sarcos Experiment.

Kernel	NMSE	NLPD	Time [s]
RFF+GG+S50	0.34	3.58	10.14
RFF+GG+S100	0.30	3.52	18.47
RFF+GG+S200	0.26	3.44	38.55
RFF+GG+S500	0.24	3.41	64.62
RFF+GG+S1000	0.22	3.36	85.00
CMOC	<b>0.19</b>	<b>3.21</b>	353.00

We notice that the performance of the approximation increases with  $S$ , and approaches the performance of CMOC, keeping the computation time per iteration to a fraction of the original one. As it was also expected, in higher dimensions, we need a larger number of random features to approach the performance of the CMOC.

## 7 RELATED WORK

Random Fourier features have been used in the literature for Gaussian processes before. For example, in Bonilla et al. [2016], the authors use RFFs in order to propose a multi-task GP model that circumvents the scalability problem of the GPs. Their model for the multiple outputs uses an affine transformation of the random features, whereas we use a non-instantaneous transformation via the Green’s functions. Also in Yang et al. [2015], the authors use a faster approximation of random Fourier features via the FastFood kernels [Le et al., 2013], for approximating the kernel functions of a GP. Their method is not used for multiple outputs, nor does include dynamical systems.

Latent force models have been also studied using a state-space formulation [Hartikainen and Särkkä, 2011] and in that line of research, low-rank approximations for computing features have also been introduced [Solin and Särkkä, 2014]. Specifically, this work approximates the covariance function using the Laplace operator eigenvalues and eigenfunctions. This formulation has been used in Svensson et al. [2016] to approximate the GP priors that are placed over the functions that transform the state vector in the update state and observation equations. Thus, it has not been considered to approximate the GP model of the excitation function.

Brault et al. [2016] directly build random Fourier features for vector-valued kernels using an operator-valued version of Bochner’s theorem. The construction is applied to the decomposable kernel, the curl-free kernel and the div-free kernel. In our construction, rather than starting with a fixed form for the operator-valued kernel, we use a general mechanism used to build valid operator kernel functions and apply linear operators over the random Fourier features defined for single output kernels.

## 8 CONCLUSIONS AND FUTURE WORK

We have shown in this paper how to use random Fourier features for easing the computation of the kernel functions associated to LFM. As a by-product, we have also reduced the computational complexity of working in multiple-output GPs from  $\mathcal{O}(D^3N^3)$  to  $\mathcal{O}(DNQ^2S^2)$ . We showed experiments over datasets of different sizes for which results with LFM are slow to compute. Our random Fourier response features reduce computational time without compromising performance. Also, notice that by having decoupled the solution of the convolution integrals from the particular form for the kernel of the latent functions, we now can easily build kernels for latent force models with different kernel functions in the GPs of the latent functions, just by changing the distribution  $p(\lambda)$  from which we sample from.

These novel representations of latent force models open the path for different types of future work: the application of random Fourier response features for building more efficient versions of sequential LFM [Álvarez et al., 2011] and hierarchical LFM [Honkela and et al., 2010]; the use of physically inspired Fourier features in other Gaussian process models, particularly, deep models [Cutajar et al., 2017]; the use of more efficient sampling techniques for obtaining the Fourier features, e.g. Quasi-Monte Carlo sampling [Avron et al., 2016]. With a more efficient way to compute kernels for multiple-outputs, we can also use more expensive model selection approaches, for example, those based on automatic composition of kernel functions [Duvenaud and et al., 2013], for building more complex covariance functions, e.g. combinations of first order models and second order models, as sums of kernels or as products of kernels. For the case of convolved multiple outputs GPs where the input dimension is greater than three (compared to typical LFMs), the computation of dense Gaussian matrices can be replaced by the product between Hadamard matrices and diagonal Gaussian matrices, which are faster to compute [Le et al., 2013].

### Acknowledgments

CG would like to thank to Convocatoria 567 from Administrative Department of Science, Technology and Innovation of Colombia (COLCIENCIAS) for the support. MAA has been financed by the Engineering and Physical Research Council (EPSRC) Research Project EP/N014162/1.

### References

M. A. Álvarez and N. D. Lawrence. Computationally Efficient Convolved Multiple Output Gaussian Processes.

- Journal of Machine Learning Research*, 12:1425–1466, 2011.
- M. A. Álvarez, D. Luengo, and N. D. Lawrence. Latent Force Models. In David van Dyk and M. Welling, editors, *Proceedings of AISTATS 2009*, pages 9–16, Clearwater Beach, Florida, 16-18 April 2009. JMLR W&CP 5.
- M. A. Álvarez, D. Luengo, M. Titsias, and N. D. Lawrence. Efficient Multioutput Gaussian Processes through Variational Inducing Kernels. In Y.-W. Teh and M. Titterton, editors, *Proceedings of AISTATS 2010*, volume 9 of *Proceedings of Machine Learning Research*, pages 25–32, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010.
- M. A. Álvarez, J. Peters, B. Schölkopf, and N. D. Lawrence. Switched Latent Force Models for Movement Segmentation. In J. Shawe-Taylor, R. Zemel, C. Williams, and J. Lafferty, editors, *Advances in Neural Information Processing Systems 24*, pages 55–63. MIT, 2011.
- M. A. Álvarez, D. Luengo, and N. D. Lawrence. Linear Latent Force Models using Gaussian Processes. *IEEE TPAMI*, 35(11):2693–2705, 2013.
- H. Avron, V. Sindhwani, J. Yang, and M. W. Mahoney. Quasi-Monte Carlo Feature Maps for Shift-Invariant Kernels. *Journal of Machine Learning Research*, 17(120):1–38, 2016.
- E. Bonilla, D. Steinberg, and A. Reid. Extended and Unscented Kitchen Sinks. In M. F. Balcan and K. Q. Weinberger, editors, *Proceedings of ICML 2016*, volume 48 of *Proceedings of Machine Learning Research*, pages 1651–1659, New York, New York, USA, 20–22 Jun 2016.
- R. Brault, M. Heinonen, and F. Buc. Random Fourier Features For Operator-Valued Kernels. In Robert J. Durrant and Kee-Eung Kim, editors, *Proceedings of The 8th Asian Conference on Machine Learning*, volume 63 of *Proceedings of Machine Learning Research*, pages 110–125, 2016.
- K. Cutajar, E. V. Bonilla, P. Michiardi, and M. Filippone. Random Feature Expansions for Deep Gaussian Processes. In Doina Precup and Yee Whye Teh, editors, *Proceedings of ICML 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 884–893, International Convention Centre, Sydney, Australia, 06–11 Aug 2017.
- D. Duvenaud and et al. Structure Discovery in Non-parametric Regression through Compositional Kernel Search. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of ICML 2013*, volume 28 of *Proceedings of Machine Learning Research*, pages 1166–1174, Atlanta, Georgia, USA, 17–19 Jun 2013.
- P. Gao, A. Honkela, M. Rattray, and N. D. Lawrence. Gaussian process modelling of latent chemical species: applications to inferring transcription factor activities. *Bioinformatics*, 24(16):i70–i75, 2008.
- S. Ghosh and et al. Modeling the thermal dynamics of buildings: A latent-force- model-based approach. *ACM Trans. Intell. Syst. Technol.*, 6(1):7:1–7:27, March 2015.
- J. Hartikainen and S. Särkkä. Sequential Inference for Latent Force Models. In *Proceedings of UAI 2011*, pages 311–318, 2011.
- A. Honkela and et al. Model-based method for transcription factor target identification with limited data. *Proc. Natl. Acad. Sci.*, 107(17):7793–7798, 2010.
- Q. Le, T. Sarlós, and A. Smola. Fastfood: Approximating Kernel Expansions in Loglinear Time. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of ICML 2013*, volume 28 of *Proceedings of Machine Learning Research*, pages 244–252, Atlanta, Georgia, USA, 17–19 Jun 2013.
- Trung V. Nguyen and Edwin V. Bonilla. Collaborative Multi-output Gaussian Processes. In *Proceedings of UAI 2014, Quebec City, Quebec, Canada, July 23-27, 2014*, pages 643–652, 2014.
- A. Rahimi and B. Recht. Random features for large-scale kernel machines. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1177–1184. Curran Associates, Inc., 2008.
- C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006. ISBN 0-262-18253-X.
- Arno Solin and Simo Särkkä. Hilbert Space Methods for Reduced-Rank Gaussian Process Regression. <https://arxiv.org/pdf/1401.5508.pdf>, 2014.
- Andreas Svensson, Arno Solin, Simo Särkkä, and Thomas Schön. Computationally Efficient Bayesian Learning of Gaussian Process State Space Models. In Arthur Gretton and Christian C. Robert, editors, *Proceedings of AISTATS 2016*, volume 51 of *Proceedings of Machine Learning Research*, pages 213–221, Cadiz, Spain, 09–11 May 2016. PMLR.
- Z. Yang, A. Wilson, A. Smola, and Le Song. À la Carte – Learning Fast Kernels. In G. Lebanon and S. V. N. Vishwanathan, editors, *Proceedings of AISTATS 2015*, volume 38 of *Proceedings of Machine Learning Research*, pages 1098–1106, San Diego, California, USA, 09–12 May 2015.



---

# Fast Policy Learning through Imitation and Reinforcement

---

**Ching-An Cheng**  
Georgia Tech  
Atlanta, GA 30332

**Xinyan Yan**  
Georgia Tech  
Atlanta, GA 30332

**Nolan Wagener**  
Georgia Tech  
Atlanta, GA 30332

**Byron Boots**  
Georgia Tech  
Atlanta, GA 30332

## Abstract

Imitation learning (IL) consists of a set of tools that leverage expert demonstrations to quickly learn policies. However, if the expert is suboptimal, IL can yield policies with inferior performance compared to reinforcement learning (RL). In this paper, we aim to provide an algorithm that combines the best aspects of RL and IL. We accomplish this by formulating several popular RL and IL algorithms in a common mirror descent framework, showing that these algorithms can be viewed as a variation on a single approach. We then propose LOKI, a strategy for policy learning that first performs a small but random number of IL iterations before switching to a policy gradient RL method. We show that if the switching time is properly randomized, LOKI can learn to outperform a suboptimal expert and converge faster than running policy gradient from scratch. Finally, we evaluate the performance of LOKI experimentally in several simulated environments.

## 1 INTRODUCTION

Reinforcement learning (RL) has emerged as a promising technique to tackle complex sequential decision problems. When empowered with deep neural networks, RL has demonstrated impressive performance in a range of synthetic domains (Mnih et al., 2013; Silver et al., 2017). However, one of the major drawbacks of RL is the enormous number of interactions required to learn a policy. This can lead to prohibitive cost and slow convergence when applied to real-world problems, such as those found in robotics (Pan et al., 2017).

Imitation learning (IL) has been proposed as an alternate strategy for faster policy learning that works by

leveraging additional information provided through expert demonstrations (Pomerleau, 1989; Schaal, 1999). However, despite significant recent breakthroughs in our understanding of imitation learning (Ross et al., 2011; Cheng and Boots, 2018), the performance of IL is still highly dependent on the quality of the expert policy. When only a suboptimal expert is available, policies learned with standard IL can be inferior to the policies learned by tackling the RL problem directly with approaches such as policy gradients.

Several recent attempts have endeavored to combine RL and IL (Ross and Bagnell, 2014; Chang et al., 2015; Nair et al., 2017; Rajeswaran et al., 2017; Sun et al., 2018). These approaches incorporate the cost information of the RL problem into the imitation process, so the learned policy can *both* improve faster than their RL-counterparts and outperform the suboptimal expert policy. Despite reports of improved empirical performance, the theoretical understanding of these combined algorithms are still fairly limited (Rajeswaran et al., 2017; Sun et al., 2018). Furthermore, some of these algorithms have requirements that can be difficult to satisfy in practice, such as state resetting (Ross and Bagnell, 2014; Chang et al., 2015).

In this paper, we aim to provide an algorithm that combines the best aspects of RL and IL. We accomplish this by first formulating first-order RL and IL algorithms in a common mirror descent framework, and show that these algorithms can be viewed as a single approach that only differs in the choice of first-order oracle. On the basis of this new insight, we address the difficulty of combining IL and RL with a simple, randomized algorithm, named LOKI (Locally Optimal search after  $K$ -step Imitation). As its name suggests, LOKI operates in two phases: picking  $K$  randomly, it first performs  $K$  steps of online IL and then improves the policy with a policy gradient method afterwards. Compared with previous methods that aim to combine RL and IL, LOKI is extremely straightforward to implement. Furthermore, it

has stronger theoretical guarantees: by properly randomizing  $K$ , LOKI performs as if directly running policy gradient steps with the expert policy as the initial condition. Thus, not only can LOKI improve faster than common RL methods, but it can also significantly outperform a suboptimal expert. This is in contrast to previous methods, such as AGGREGATE (Ross and Bagnell, 2014), which generally cannot learn a policy that is better than a one-step improvement over the expert policy. In addition to these theoretical contributions, we validate the performance of LOKI in multiple simulated environments. The empirical results corroborate our theoretical findings.

## 2 PROBLEM DEFINITION

We consider solving discrete-time  $\gamma$ -discounted infinite-horizon RL problems.<sup>1</sup> Let  $\mathbb{S}$  and  $\mathbb{A}$  be the state and the action spaces, and let  $\Pi$  be the policy class. The objective is to find a policy  $\pi \in \Pi$  that minimizes an accumulated cost  $J(\pi)$  defined as

$$\min_{\pi \in \Pi} J(\pi), \quad J(\pi) := \mathbb{E}_{\rho_\pi} [\sum_{t=0}^{\infty} \gamma^t c(s_t, a_t)], \quad (1)$$

in which  $s_t \in \mathbb{S}$ ,  $a_t \in \mathbb{A}$ ,  $c$  is the instantaneous cost, and  $\rho_\pi$  denotes the distribution of trajectories  $(s_0, a_0, s_1, \dots)$  generated by running the stationary policy  $\pi$  starting from  $s_0 \sim p_0(s_0)$ .

We denote  $Q_\pi(s, a)$  as the Q-function under policy  $\pi$  and  $V_\pi(s) = \mathbb{E}_{a \sim \pi_s} [Q_\pi(s, a)]$  as the associated value function, where  $\pi_s$  denotes the action distribution given state  $s$ . In addition, we denote  $d_{\pi,t}(s)$  as the state distribution at time  $t$  generated by running the policy  $\pi$  for the first  $t$  steps, and we define a joint distribution  $d_\pi(s, t) = (1 - \gamma)d_{\pi,t}(s)\gamma^t$  which has support  $\mathcal{S} \times [0, \infty)$ . Note that, while we use the notation  $\mathbb{E}_{a \sim \pi}$ , the policy class  $\Pi$  can be either deterministic or stochastic.

We generally will not deal with the objective function in (1) directly. Instead, we consider a surrogate problem

$$\min_{\pi \in \Pi} \mathbb{E}_{s,t \sim d_\pi} \mathbb{E}_{a \sim \pi_s} [A_{\pi'}(s, a)], \quad (2)$$

where  $A_{\pi'} = Q_{\pi'} - V_{\pi'}$  is the (dis)advantage function with respect to some fixed reference policy  $\pi'$ . For compactness of writing, we will often omit the random variable in expectation; e.g., the objective function in (2) will be written as  $\mathbb{E}_{d_\pi} \mathbb{E}_\pi [A_{\pi'}]$  for the remainder of paper.

By the performance difference lemma below, it is easy to see that solving (2) is equivalent to solving (1).

**Lemma 1.** (Kakade and Langford, 2002) *Let  $\pi$  and  $\pi'$  be two policies and  $A_{\pi'}(s, a) = Q_{\pi'}(s, a) - V_{\pi'}(s)$  be*

<sup>1</sup>LOKI can be easily adapted to finite-horizon problems.

the (dis)advantage function with respect to running  $\pi'$ . Then it holds that

$$J(\pi) = J(\pi') + \frac{1}{1 - \gamma} \mathbb{E}_{d_\pi} \mathbb{E}_\pi [A_{\pi'}]. \quad (3)$$

## 3 FIRST-ORDER RL AND IL

We formulate both first-order RL and IL methods within a single mirror descent framework (Nemirovski et al., 2009), which includes common update rules (Sutton et al., 2000; Kakade, 2002; Peters and Schaal, 2008; Peters et al., 2010; Rawlik et al., 2012; Silver et al., 2014; Schulman et al., 2015b; Ross et al., 2011; Sun et al., 2017). We show that policy updates based on RL and IL mainly differ in first-order stochastic oracles used, as summarized in Table 1.

### 3.1 MIRROR DESCENT

We begin by defining the iterative rule to update policies. We assume that the learner’s policy  $\pi$  is parametrized by some  $\theta \in \Theta$ , where  $\Theta$  is a closed and convex set, and that the learner has access to a family of strictly convex functions  $\mathcal{R}$ .

To update the policy, in the  $n$ th iteration, the learner receives a vector  $g_n$  from a first-order oracle, picks  $R_n \in \mathcal{R}$ , and then performs a mirror descent step:

$$\theta_{n+1} = P_{n,g_n}(\theta_n) \quad (4)$$

where  $P_{n,g_n}$  is a prox-map defined as

$$P_{n,g_n}(\theta_n) := \arg \min_{\theta \in \Theta} \langle g_n, \theta \rangle + \frac{1}{\eta_n} D_{R_n}(\theta || \theta_n). \quad (5)$$

$\eta_n$  is the step size, and  $D_{R_n}$  is the Bregman divergence associated with  $R_n$  (Bregman, 1967):  $D_{R_n}(\theta || \theta_n) := R_n(\theta) - R_n(\theta_n) - \langle \nabla R_n(\theta_n), \theta - \theta_n \rangle$ .

By choosing proper  $R_n$ , the mirror descent framework in (4) covers most RL and IL algorithms. Common choices of  $R_n$  include negative entropy (Peters et al., 2010; Rawlik et al., 2012),  $\frac{1}{2} \|\theta\|_2^2$  (Sutton et al., 2000; Silver et al., 2014), and  $\frac{1}{2} \theta^\top F(\theta_n) \theta$  with  $F(\theta_n)$  as the Fisher information matrix (Kakade, 2002; Peters and Schaal, 2008; Schulman et al., 2015a).

### 3.2 FIRST-ORDER ORACLES

While both first-order RL and IL methods can be viewed as performing mirror descent, they differ in the choice of the first-order oracle that returns the update direction  $g_n$ . Here we show the vector  $g_n$  of both approaches can be derived as a stochastic approximation of the (partial) derivative of  $\mathbb{E}_{d_\pi} \mathbb{E}_\pi [A_{\pi'}]$  with respect to policy  $\pi$ , but with a different reference policy  $\pi'$ .

Table 1: Comparison of First-Order Oracles

Method	First-Order Oracle
POLICY GRADIENT (Section 3.2.1)	$\mathbb{E}_{d_{\pi_n}} (\nabla_{\theta} \mathbb{E}_{\pi}) [A_{\pi_n}]$
DAGGERED (Section 3.2.2)	$\mathbb{E}_{d_{\pi_n}} (\nabla_{\theta} \mathbb{E}_{\pi}) [\mathbb{E}_{\pi^*} [d]]$
AGGREGATED (Section 3.2.2)	$\mathbb{E}_{d_{\pi_n}} (\nabla_{\theta} \mathbb{E}_{\pi}) [A_{\pi^*}]$
SOLS (Section 6)	$\mathbb{E}_{d_{\pi_n}} (\nabla_{\theta} \mathbb{E}_{\pi}) [(1 - \lambda)A_{\pi_n} + \lambda A_{\pi^*}]$
THOR (Section 6)	$\mathbb{E}_{d_{\pi_n}} (\nabla_{\theta} \mathbb{E}_{\pi}) [A_{\pi_n, t}^{H, \pi^*}]$

### 3.2.1 Policy Gradients

A standard approach to RL is to treat (1) as a stochastic nonconvex optimization problem. In this case,  $g_n$  in mirror descent (4) is an estimate of the policy gradient  $\nabla_{\theta} J(\pi)$  (Williams, 1992; Sutton et al., 2000).

To compute the policy gradient in the  $n$ th iteration, we set the current policy  $\pi_n$  as the reference policy in (3) (i.e.  $\pi' = \pi_n$ ), which is treated as constant in  $\theta$  in the following policy gradient computation. Because  $\mathbb{E}_{\pi_n} [A_{\pi_n}] = \mathbb{E}_{\pi_n} [Q_{\pi_n}] - V_{\pi_n} = 0$ , using (3), the policy gradient can be written as<sup>2</sup>

$$\begin{aligned}
& (1 - \gamma) \nabla_{\theta} J(\pi) |_{\pi=\pi_n} \\
&= \nabla_{\theta} \mathbb{E}_{d_{\pi}} \mathbb{E}_{\pi} [A_{\pi_n}] |_{\pi=\pi_n} \\
&= (\nabla_{\theta} \mathbb{E}_{d_{\pi}}) [0] + \mathbb{E}_{d_{\pi}} (\nabla_{\theta} \mathbb{E}_{\pi}) [A_{\pi_n}] |_{\pi=\pi_n} \\
&= \mathbb{E}_{d_{\pi}} (\nabla_{\theta} \mathbb{E}_{\pi}) [A_{\pi_n}] |_{\pi=\pi_n} \tag{6}
\end{aligned}$$

The above expression is unique up to a change of baselines:  $(\nabla_{\theta} \mathbb{E}_{\pi}) [A_{\pi_n}]$  is equivalent to  $(\nabla_{\theta} \mathbb{E}_{\pi}) [A_{\pi_n} + b]$ , because  $(\nabla_{\theta} \mathbb{E}_{\pi}) [b(s)] = \nabla_{\theta} b(s) = 0$ , where  $b : \mathbb{S} \rightarrow \mathbb{R}$  is also called a control variate (Greensmith et al., 2004).

The exact formulation of  $(\nabla_{\theta} \mathbb{E}_{\pi}) [A_{\pi_n}]$  depends on whether the policy  $\pi$  is stochastic or deterministic. For stochastic policies,<sup>3</sup> we can compute it with the likelihood-ratio method and write

$$(\nabla_{\theta} \mathbb{E}_{\pi}) [A_{\pi_n}] = \mathbb{E}_{\pi} [A_{\pi_n} \nabla_{\theta} \log \pi] \tag{7}$$

For deterministic policies, we replace the expectation as evaluation (as it is the expectation over a Dirac delta function, i.e.  $a = \pi(s)$ ) and use the chain rule:

$$(\nabla_{\theta} \mathbb{E}_{\pi}) [A_{\pi_n}] = \nabla_{\theta} A_{\pi_n}(s, \pi) = \nabla_{\theta} \pi \nabla_a A_{\pi_n} \tag{8}$$

Substituting (7) or (8) back into (6), we get the equation for stochastic policy gradient (Sutton et al., 2000) or deterministic policy gradient (Silver et al., 2014). Note that the above equations require the exact knowledge, or

<sup>2</sup>We assume the cost is sufficiently regular so that the order of differentiation and expectation can exchange.

<sup>3</sup>A similar equation holds for reparametrization (Grathwohl et al., 2017).

an unbiased estimate, of  $A_{\pi}$ . In practice, these terms are further approximated using function approximators, leading to biased gradient estimators (Konda and Tsitsiklis, 2000; Schulman et al., 2015b; Mnih et al., 2016).

### 3.2.2 Imitation Gradients

An alternate strategy to RL is IL. In particular, we consider *online* IL, which interleaves data collection and policy updates to overcome the covariate shift problem of traditional batch IL (Ross et al., 2011). Online IL assumes that a (possibly suboptimal) expert policy  $\pi^*$  is available as a black-box oracle, from which demonstrations  $a^* \sim \pi_s^*$  can be queried for any given state  $s \in \mathbb{S}$ . Due to this requirement, the expert policy in online IL is often an *algorithm* (rather than a human demonstrator), which is hard-coded or based on additional computational resources, such as trajectory optimization (Pan et al., 2017). The goal of IL is to learn a policy that can perform similar to, or better than, the expert policy.

Rather than solving the stochastic nonconvex optimization directly, online IL solves an online learning problem with per-round cost in the  $n$ th iteration defined as

$$l_n(\pi) = \mathbb{E}_{d_{\pi_n}} \mathbb{E}_{\pi} [\tilde{c}] \tag{9}$$

where  $\tilde{c} : \mathbb{S} \times \mathbb{A} \rightarrow \mathbb{R}$  is a surrogate loss satisfying the following condition: For all  $s \in \mathbb{S}$  and  $\pi \in \Pi$ , there exists a constant  $C_{\pi^*} > 0$  such that

$$C_{\pi^*} \mathbb{E}_{\pi} [\tilde{c}] \geq \mathbb{E}_{\pi} [A_{\pi^*}]. \tag{10}$$

By Lemma 1, this implies  $J(\pi_n) \leq J(\pi^*) + \frac{C_{\pi^*}}{1-\gamma} l_n(\pi_n)$ . Namely, in the  $n$ th iteration, online IL attempts to minimize an online upper-bound of  $J(\pi_n)$ .

DAGGER (Ross et al., 2011) chooses  $\tilde{c}$  to be a strongly convex function  $\tilde{c}(s, a) = \mathbb{E}_{a^* \sim \pi_s^*} [d(a, a^*)]$  that penalizes the difference between the learner’s policy and the expert’s policy, where  $d$  is some metric of space  $\mathbb{A}$  (e.g., for a continuous action space Pan et al. (2017) choose  $d(a, a^*) = \|a - a^*\|^2$ ). More directly, AGGREGATE simply chooses  $\tilde{c}(s, a) = A_{\pi^*}(s, a)$  (Ross and Bagnell, 2014); in this case, the policy learned with online IL can potentially outperform the expert policy.

First-order online IL methods operate by updating policies with mirror descent (4) with  $g_n$  as an estimate of

$$\nabla_{\theta} l_n(\pi_n) = \mathbb{E}_{d_{\pi_n}} (\nabla_{\theta} \mathbb{E}_{\pi} [\tilde{c}])|_{\pi=\pi_n} \quad (11)$$

Similar to policy gradients, the implementation of (11) can be executed using either (7) or (8) (and with a control variate). One particular case of (11), with  $\tilde{c} = A_{\pi^*}$ , is known as AGGREGATED (Sun et al., 2017),

$$\nabla_{\theta} l_n(\pi_n) = \mathbb{E}_{d_{\pi_n}} (\nabla_{\theta} \mathbb{E}_{\pi} [A_{\pi^*}])|_{\pi=\pi_n}. \quad (12)$$

Similarly, we can turn DAGGER into a first-order method, which we call DAGGERED, by using  $g_n$  as an estimate of the first-order oracle

$$\nabla_{\theta} l_n(\pi_n) = \mathbb{E}_{d_{\pi_n}} (\nabla_{\theta} \mathbb{E}_{\pi} \mathbb{E}_{\pi^*} [d]). \quad (13)$$

A comparison is summarized in Table 1.

## 4 THEORETICAL COMPARISON

With the first-order oracles defined, we now compare the performance and properties of performing mirror descent with policy gradient or imitation gradient. We will see that while both approaches share the same update rule in (4), the generated policies have different behaviors: using policy gradient generates a *monotonically* improving policy sequence, whereas using imitation gradient generates a policy sequence that improves *on average*. Although the techniques used in this section are not completely new in the optimization literature, we specialize the results to compare performance and to motivate LOKI in the next section. The proofs of this section are included in Appendix B for completeness.

### 4.1 POLICY GRADIENTS

We analyze the performance of policy gradients with standard techniques from nonconvex analysis.

**Proposition 1.** *Let  $J$  be  $\beta$ -smooth and  $R_n$  be  $\alpha_n$ -strongly convex with respect to norm  $\|\cdot\|$ . Assume  $\mathbb{E}[g_n] = \nabla_{\theta} J(\pi_n)$ . For  $\eta_n \leq \frac{2\alpha_n}{\beta}$ , it satisfies*

$$\begin{aligned} \mathbb{E}[J(\pi_{n+1})] &\leq J(\pi_0) + \mathbb{E} \left[ \sum_{n=1}^N \frac{2\eta_n}{\alpha_n} \|\nabla_{\theta} J(\pi_n) - g_n\|_*^2 \right] \\ &\quad + \frac{1}{2} \mathbb{E} \left[ \sum_{n=1}^N \left( -\alpha_n \eta_n + \frac{\beta \eta_n^2}{2} \right) \|\hat{\nabla}_{\theta} J(\pi_n)\|^2 \right] \end{aligned}$$

where the expectation is due to randomness of sampling  $g_n$ , and  $\hat{\nabla}_{\theta} J(\pi_n) := \frac{1}{\eta_n} (\theta_n - P_{n, \nabla_{\theta} J(\pi_n)}(\theta_n))$  is a gradient surrogate.

Proposition 1 shows that monotonic improvement can be made under proper smoothness assumptions if the step

size is small and noise is comparably small with the gradient size. However, the final policy’s performance is sensitive to the initial condition  $J(\pi_0)$ , which can be poor for a randomly initialized policy.

Proposition 1 also suggests that the size of the gradient  $\|\hat{\nabla}_{\theta} J(\pi_n)\|^2$  does not converge to zero on average. Instead, it converges to a size proportional to the sampling noise of policy gradient estimates due to the linear dependency of  $\frac{2\eta_n}{\alpha_n} \|\nabla_{\theta} J(\pi_n) - g_n\|_*^2$  on  $\eta_n$ . This phenomenon is also mentioned by Ghadimi et al. (2016). We note that this pessimistic result is because the prox-map (5) is non-linear in  $g_n$  for general  $R_n$  and  $\Theta$ . However, when  $R_n$  is quadratic and  $\Theta$  is unconstrained, the convergence of  $\|\hat{\nabla}_{\theta} J(\pi_n)\|^2$  to zero on average can be guaranteed (see Appendix B.1 for a discussion).

### 4.2 IMITATION GRADIENTS

While applying mirror descent with a policy gradient can generate a monotonically improving policy sequence, applying the same algorithm with an imitation gradient yields a different behavior. The result is summarized below, which is a restatement of (Ross and Bagnell, 2014, Theorem 2.1), but is specialized for mirror descent.

**Proposition 2.** *Assume  $l_n$  is  $\sigma$ -strongly convex with respect to  $R_n$ .<sup>4</sup> Assume  $\mathbb{E}[g_n] = \nabla_{\theta} l_n(\pi_n)$  and  $\|g_n\|_* \leq G < \infty$  almost surely. For  $\eta_n = \frac{1}{\hat{\sigma} n}$  with  $\hat{\sigma} \leq \sigma$ , it holds*

$$\frac{1}{N} \mathbb{E} \left[ \sum_{n=1}^N J(\pi_n) \right] \leq J(\pi^*) + \frac{C_{\pi^*}}{1-\gamma} (\epsilon_{\text{class}} + \epsilon_{\text{regret}})$$

where the expectation is due to randomness of sampling  $g_n$ ,  $\epsilon_{\text{class}} = \sup_{\{\pi_n\}} \inf_{\pi \in \Pi} \frac{1}{N} \sum_{n=1}^N l_n(\pi)$  and  $\epsilon_{\text{regret}} = \frac{G^2(\log N + 1)}{2\hat{\sigma} N}$ .

Proposition 2 is based on the assumption that  $l_n$  is strongly convex, which can be verified for certain problems (Cheng and Boots, 2018). Consequently, Proposition 2 shows that the performance of the policy sequence on average can converge close to the expert’s performance  $J(\pi^*)$ , with additional error that is proportional to  $\epsilon_{\text{class}}$  and  $\epsilon_{\text{regret}}$ .

$\epsilon_{\text{regret}}$  is an upper bound of the average regret, which is less than  $\tilde{O}(\frac{1}{N})$  for a large enough step size.<sup>5</sup> This characteristic is in contrast to policy gradient, which requires small enough step sizes to guarantee local improvement.

$\epsilon_{\text{class}}$  measures the expressiveness of the policy class  $\Pi$ . It can be *negative* if there is a policy in  $\Pi$  that outper-

<sup>4</sup>A function  $f$  is said to be  $\sigma$ -strongly convex with respect to  $R$  on a set  $\mathcal{K}$  if for all  $x, y \in \mathcal{K}$ ,  $f(x) \geq f(y) + \langle \nabla f(y), x - y \rangle + \sigma D_R(x||y)$ .

<sup>5</sup>The step size should be large enough to guarantee  $\tilde{O}(\frac{1}{N})$  convergence, where  $\tilde{O}$  denotes Big-O but omitting log dependency. However, it should be bounded since  $\epsilon_{\text{regret}} = \Theta(\frac{1}{\hat{\sigma}})$ .

---

**Algorithm 1** LOKI

---

**Parameters:**  $d, N_m, N_M$ **Input:**  $\pi^*$ 

- 1: Sample  $K$  with probability in (15).
  - 2: **for**  $t = 1 \dots K$  **do** # Imitation Phase
  - 3:   Collect data  $\mathcal{D}_n$  by executing  $\pi_n$
  - 4:   Query  $g_n$  from (11) using  $\pi^*$
  - 5:   Update  $\pi_n$  by mirror descent (5) with  $g_n$
  - 6:   Update advantage function estimate  $\hat{A}_{\pi_n}$  by  $\mathcal{D}_n$
  - 7: **end for**
  - 8: **for**  $t = K + 1 \dots$  **do** # Reinforcement Phase
  - 9:   Collect data  $\mathcal{D}_n$  by executing  $\pi_n$ .
  - 10:   Query  $g_n$  from (6) f using  $\hat{A}_{\pi_n}$
  - 11:   Update  $\pi_n$  by mirror descent (5) with  $g_n$
  - 12:   Update advantage function estimate  $\hat{A}_{\pi_n}$  by  $\mathcal{D}_n$
  - 13: **end for**
- 

forms the expert policy  $\pi^*$  in terms of  $\tilde{c}$ . However, since online IL attempts to minimize an online upper bound of the accumulated cost through a surrogate loss  $\tilde{c}$ , the policy learned with imitation gradients in general cannot be better than performing one-step policy improvement from the expert policy (Ross and Bagnell, 2014; Cheng and Boots, 2018). Therefore, when the expert is suboptimal, the reduction from nonconvex optimization to online convex optimization can lead to suboptimal policies.

Finally, we note that updating policies with imitation gradients does not necessarily generate a monotonically improving policy sequence, even for deterministic problems; whether the policy improves monotonically is completely problem dependent (Cheng and Boots, 2018). Without going into details, we can see this by comparing policy gradient in (6) and the special case of imitation gradient in (12). By Lemma 3, we see that

$$\begin{aligned} & \mathbb{E}_{d_{\pi_n}} (\nabla_{\theta} \mathbb{E}_{\pi}) [A_{\pi_n}] \\ &= (\nabla_{\theta} \mathbb{E}_{d_{\pi}}) \mathbb{E}_{\pi_n} [A_{\pi^*}] + \mathbb{E}_{d_{\pi_n}} (\nabla_{\theta} \mathbb{E}_{\pi}) [A_{\pi^*}]. \end{aligned}$$

Therefore, even with  $\tilde{c} = A_{\pi^*}$ , the negative of the direction in (12) is not necessarily a descent direction; namely applying (12) to update the policy is not guaranteed to improve the policy performance locally.

## 5 IMITATE-THEN-REINFORCE

To combine the benefits from RL and IL, we propose a simple randomized algorithm LOKI: first perform  $K$  steps of mirror descent with imitation gradient and then switch to policy gradient for the rest of the steps. Despite the algorithm’s simplicity, we show that, when  $K$  is appropriately randomized, running LOKI has similar performance to performing policy gradient steps directly from the expert policy.

## 5.1 ALGORITHM: LOKI

The algorithm LOKI is summarized in Algorithm 1. The algorithm is composed of two phases: an imitation phase and a reinforcement phase. In addition to learning rates, LOKI receives three hyperparameters ( $d, N_m, N_M$ ) which determine the probability of random switching at time  $K$ . As shown in the next section, these three hyperparameters can be selected fairly simply.

**Imitation Phase** Before learning, LOKI first randomly samples a number  $K \in [N_m, N_M]$  according to the prescribed probability distribution (15). Then it performs  $K$  steps of mirror descent with imitation gradient. In our implementation, we set the per-round loss as<sup>6</sup>

$$l_n(\pi) = \mathbb{E}_{d_{\pi_n}} [\text{KL}(\pi^* || \pi)], \quad (14)$$

which is the KL-divergence between the two policies. It can be easily shown that a proper constant  $C^*$  exists satisfying the requirement in (10) (Gibbs and Su, 2002). While using (14) does not guarantee learning a policy that outperforms the expert due to  $\epsilon_{\text{class}} \geq 0$ , with another reinforcement phase available, the imitation phase of LOKI is only designed to quickly bring the initial policy closer to the expert policy. Compared with choosing  $\tilde{c} = A_{\pi^*}$  as in AGGREGATED, one benefit of choosing  $\text{KL}(\pi^* || \pi)$  (or its variants, e.g.  $\mathbb{E}_{a \sim \pi} \mathbb{E}_{a^* \sim \pi^*} [\|a - a^*\|^2]$ ) is that it does not require learning a value function estimator. In addition, the imitation gradient can be calculated through *reparametrization* instead of a likelihood-ratio (Tucker et al., 2017), as now  $l_n$  is presented as a differentiable function. Consequently, the sampling variance of imitation gradient can be significantly reduced, for example, by using multiple samples of  $a \sim \pi_n$  (with a single query from the expert policy) and then performing averaging.

**Reinforcement Phase** After the imitation phase, LOKI switches to the reinforcement phase. At this point, the policy  $\pi_K$  is much closer to the expert policy than the initial policy  $\pi_0$ . In addition, an estimate of  $A_{\pi_K}$  is also available. Because the learner’s policies were applied to collect data in the previous *online* imitation phase,  $A_{\pi_n}$  can already be updated accordingly, for example, by minimizing TD error. Compared with other warm-start techniques, LOKI can learn *both* the policy and the advantage estimator in the imitation phase.

## 5.2 ANALYSIS

We now present the theoretical properties of LOKI. The analysis is composed of two steps. First, we show the

<sup>6</sup>While (14) does not conform with the classical choice  $\mathbb{E}_{\pi}[\tilde{c}]$  in (9), a bound similar to (10) can be derived.

performance of  $J(\pi_K)$  in Theorem 1, a generalization of Proposition 2 to consider the effects of non-uniform random sampling. Next, combining Theorem 1 and Proposition 1, we show the performance of LOKI in Theorem 2. The proofs are given in Appendix C.

**Theorem 1.** *Let  $d \geq 0$ ,  $N_m \geq 1$ , and  $N_M \geq 2N_m$ . Let  $K \in [N_m, N_M]$  be a discrete random variable such that*

$$P(K = n) = n^d / \sum_{m=N_m}^{N_M} m^d. \quad (15)$$

*Suppose  $l_n$  is  $\sigma$ -strongly convex with respect to  $R_n$ ,  $\mathbb{E}[g_n] = \nabla_{\theta} l_n(\pi_n)$ , and  $\|g_n\|_* \leq G < \infty$  almost surely. Let  $\{\pi_n\}$  be generated by running mirror descent with step size  $\eta_n = n^d / \hat{\sigma} \sum_{m=1}^n m^d$ . For  $\hat{\sigma} \leq \sigma$ , it holds that*

$$\mathbb{E}[J(\pi_K)] \leq J(\pi^*) + \Delta,$$

*where the expectation is due to sampling  $K$  and  $g_n$ ,  $\Delta = \frac{C_{\pi^*}}{1-\gamma} (\epsilon_{\text{class}}^w + 2^{-d} \hat{\sigma} D_{\mathcal{R}} + G^2 C_{N_M} / \hat{\sigma} N_M)$ ,  $D_{\mathcal{R}} = \sup_{R \in \mathcal{R}} \sup_{\pi, \pi' \in \Pi} D_R(\pi' || \pi)$ ,  $\epsilon_{\text{class}}^w := \sup_{\{w_n\}, \{\pi_n\}} \inf_{\pi \in \Pi} \frac{\sum_{n=1}^N w_n l_n(\pi)}{\sum_{n=1}^N w_n}$ , and*

$$C_{N_M} = \begin{cases} \log(N_M) + 1, & \text{if } d = 0 \\ \frac{8d}{3} \exp\left(\frac{d}{N_M}\right), & \text{if } d \geq 1 \end{cases}$$

Suppose  $N_M \gg d$ . Theorem 1 says that the performance of  $J(\pi_K)$  in expectation converges to  $J(\pi^*)$  in a rate of  $\tilde{O}(d/N_M)$  when a proper step size is selected. In addition to the convergence rate, we notice that the performance gap between  $J(\pi^*)$  and  $J(\pi_K)$  is bounded by  $O(\epsilon_{\text{class}}^w + 2^{-d} D_{\mathcal{R}})$ .  $\epsilon_{\text{class}}^w$  is a weighted version of the expressiveness measure of policy class  $\Pi$  in Proposition 2, which can be made small if  $\Pi$  is rich enough with respect to the *suboptimal* expert policy.  $D_{\mathcal{R}}$  measures the size of the decision space with respect to the class of regularization functions  $\mathcal{R}$  that the learner uses in mirror descent. The dependency on  $D_{\mathcal{R}}$  is because Theorem 1 performs a suffix random sampling with  $N_m > 0$ . While the presence of  $D_{\mathcal{R}}$  increases the gap, its influence can easily be made small with a slightly large  $d$  due to the factor  $2^{-d}$ .

In summary, due to the sublinear convergence rate of IL,  $N_M$  does not need to be large (say less than 100) as long as  $N_M \gg d$ ; on the other hand, due to the  $2^d$  factor,  $d$  is also small (say less than 5) as long as it is large enough to cancel out the effects of  $D_{\mathcal{R}}$ . Finally, we note that, like Proposition 2, Theorem 1 encourages using larger step sizes, which can further boost the convergence of the policy in the imitation phase of LOKI.

Given Proposition 1 and Theorem 1, now it is fairly easy to understand the performance of LOKI.

**Theorem 2.** *Running LOKI holds that*

$$\begin{aligned} \mathbb{E}[J(\pi_N)] &\leq J(\pi^*) + \Delta \\ &+ \mathbb{E} \left[ \sum_{n=K+1}^N \frac{2\eta_n}{\alpha_n} \|\nabla_{\theta} J(\pi_n) - g_n\|_*^2 \right] \\ &+ \frac{1}{2} \mathbb{E} \left[ \sum_{n=K+1}^N \left( -\alpha_n \eta_n + \frac{\beta \eta_n^2}{2} \right) \|\hat{\nabla}_{\theta} J(\pi_n)\|^2 \right], \end{aligned}$$

*where the expectation is due to sampling  $g_n$  and  $K$ .*

Firstly, Theorem 2 shows that  $\pi_N$  can perform better than the expert policy  $\pi^*$ , and, in fact, it converges to a locally optimal policy on average under the same assumption as in Proposition 1. Compare with to running policy gradient steps directly from the expert policy, running LOKI introduces an additional gap  $O(\Delta + K \|\hat{\nabla}_{\theta} J(\pi)\|^2)$ . However, as discussed previously,  $\Delta$  and  $K \leq N_M \ll N$  are reasonably small, for usual  $N$  in RL. Therefore, performing LOKI almost has the same effect as using the expert policy as the initial condition, which is the best we can hope for when having access to an expert policy.

We can also compare LOKI with performing usual policy gradient updates from a randomly initialized policy. The performance difference can be easily shown as  $O(J(\pi^*) - J(\pi_0) + \Delta + K \|\hat{\nabla}_{\theta} J(\pi)\|^2)$ . Therefore, if performing  $K$  steps of policy gradient from  $\pi_0$  gives a policy with performance worse than  $J(\pi^*) + \Delta$ , then LOKI is favorable.

## 6 RELATED WORK

We compare LOKI with some recent attempts to incorporate the loss information  $c$  of RL into IL so that it can learn a policy that outperforms the expert policy. As discussed in Section 4, when  $\tilde{c} = A_{\pi^*}$ , AGGREGATE(D) can potentially learn a policy that is better than the expert policy (Ross and Bagnell, 2014; Sun et al., 2017). However, implementing AGGREGATE(D) exactly as suggested by theory can be difficult and inefficient in practice. On the one hand, while  $A_{\pi^*}$  can be learned off-policy using samples collected by running the expert policy, usually the estimator quality is unsatisfactory due to covariate shift. On the other hand, if  $A_{\pi^*}$  is learned on-policy, it requires restarting the system from any state, or requires performing  $\frac{1}{1-\gamma}$ -times more iterations to achieve the same convergence rate as other choices of  $\tilde{c}$  such as  $\text{KL}(\pi^* || \pi)$  in LOKI; both of which are impractical for usual RL problems.

Recently, Sun et al. (2018) proposed THOR (Truncated HORizon policy search) which solves a truncated RL problem with the expert's value function as the terminal loss to alleviate the strong dependency of AGGREGATED on the quality of  $A_{\pi^*}$ . Their algorithm uses an  $H$ -step truncated advantage function

defined as  $A_{\pi_n, t}^{H, \pi^*} = \mathbb{E}_{\rho_{\pi_n}} [\sum_{\tau=t}^{t+H-1} \gamma^{\tau-t} c(s_\tau, a_\tau) + \gamma^H V_{\pi^*}(s_{t+H}) - V_{\pi^*}(s_t)]$ . While empirically the authors show that the learned policy can improve over the expert policy, the theoretical properties of THOR remain somewhat unclear.<sup>7</sup> In addition, THOR is more convoluted to implement and relies on multiple advantage function estimators. By contrast, LOKI has clearer theoretical properties, while being straightforward to implement with off-the-shelf learning algorithms.

Finally, we compare LOKI with LOLS (Locally Optimal Learning to Search), proposed by Chang et al. (2015). LOLS is an online IL algorithm which sets  $\tilde{c} = Q_{\hat{\pi}_n^\lambda}$ , where  $\lambda \in [0, 1]$  and  $\hat{\pi}_n^\lambda$  is a mixed policy that at each time step chooses to run the current policy  $\pi_n$  with probability  $1 - \lambda$  and the expert policy  $\pi^*$  with probability  $\lambda$ . Like AGGREGATED, LOLS suffers from the impractical requirement of estimating  $Q_{\hat{\pi}_n^\lambda}$ , which relies on the state resetting assumption.

Here we show that such difficulty can be addressed by using the mirror descent framework with  $g_n$  as an estimate of  $\nabla_{\theta} l_n^\lambda(\pi_n)$ , where  $l_n^\lambda(\pi) := \mathbb{E}_{d_{\pi_n}} \mathbb{E}_{\pi} [(1 - \lambda)A_{\pi_n} + \lambda A_{\pi^*}]$ . That is, the first-order oracle is simply a convex combination of policy gradient and AGGREGATED gradient. We call such linear combination SLOLS (simple LOLS) and we show it has the same performance guarantee as LOLS.

**Theorem 3.** *Under the same assumption in Proposition 2, running SLOLS generates a policy sequence, with randomness due to sampling  $g_n$ , satisfying*

$$\frac{1}{N} \mathbb{E} \left[ \sum_{n=1}^N J(\pi_n) - ((1 - \lambda)J_{\pi_n}^* + \lambda J(\pi^*)) \right] \leq \frac{\epsilon_{\text{class}}^\lambda + \epsilon_{\text{regret}}^\lambda}{1 - \gamma}$$

where  $(1 - \gamma)J_{\pi_n}^* = \min_{\pi \in \Pi} \mathbb{E}_{d_{\pi_n}} \mathbb{E}_{\pi} [Q_{\pi_n}] =: \mathbb{E}_{d_{\pi_n}} [V_{\pi_n}^*]$  and  $\epsilon_{\text{class}}^\lambda = \min_{\pi \in \Pi} \frac{1}{N} (\sum_{n=1}^N \mathbb{E}_{d_{\pi_n}} \mathbb{E}_{\pi} [(1 - \lambda)Q_{\pi_n} + \lambda Q_{\pi^*}]) - \frac{1}{N} (\sum_{n=1}^N \mathbb{E}_{d_{\pi_n}} [(1 - \lambda)V_{\pi_n}^* + \lambda V_{\pi^*}^*])$ .

In fact, the performance in Theorem 3 is actually a lower bound of Theorem 3 in (Chang et al., 2015).<sup>8</sup> Theorem 3 says that on average  $\pi_n$  has performance between the expert policy  $J(\pi^*)$  and the intermediate cost  $J_{\pi_n}^*$ , as long as  $\epsilon_{\text{class}}^\lambda$  is small (i.e., there exists a single policy in  $\Pi$  that is better than the expert policy or the local improvement from any policy in  $\Pi$ ). However, due to the presence of  $\epsilon_{\text{class}}^\lambda$ , despite  $J_{\pi_n}^* \leq J(\pi_n)$ , it is not guaranteed that  $J_{\pi_n}^* \leq J(\pi^*)$ . As in Chang et al. (2015), either LOLS or SLOLS can necessarily perform on average better than the expert policy  $\pi^*$ . Finally, we note that recently both Nair et al. (2017) and Rajeswaran et al.

<sup>7</sup>The algorithm actually implemented by Sun et al. (2018) does not solve precisely the same problem analyzed in theory.

<sup>8</sup>The main difference is due to technicalities. In Chang et al. (2015),  $\epsilon_{\text{class}}^\lambda$  is compared with a time-varying policy.

(2017) propose a scheme similar to SLOLS, but with the AGGREGATE(D) gradient computed using offline batch data collected by the expert policy. However, there is no theoretical analysis of this algorithm’s performance.

## 7 EXPERIMENTS

We evaluate LOKI on several robotic control tasks from OpenAI Gym (Brockman et al., 2016) with the DART physics engine (Lee et al., 2018)<sup>9</sup> and compare it with several baselines: TRPO (Schulman et al., 2015a), TRPO from expert, DAGGERED (the first-order version of DAGGER (Ross et al., 2011) in (13)), SLOLS (Section 6), and THOR (Sun et al., 2018).

### 7.1 TASKS

We consider the following tasks. In all tasks, the discount factor of the RL problem is set to  $\gamma = 0.99$ . The details of each task are specified in Table A in Appendix A.

**Inverted Pendulum** This is a classic control problem, and its goal is to swing up an pendulum and to keep it balanced in a upright posture. The difficulty of this task is that the pendulum cannot be swung up directly due to a torque limit.

**Locomotion** The goal of these tasks (Hopper, 2D Walker, and 3D Walker) is to control a walker to move forward as quickly as possible without falling down. In Hopper, the walker is a monopod, which is subjected to significant contact discontinuities, whereas the walkers in the other tasks are bipeds. In 2D Walker, the agent is constrained to a plane to simplify balancing.

**Robot Manipulator** In the Reacher task, a 5-DOF (degrees-of-freedom) arm is controlled to reach a random target position in 3D space. The reward consists of the negative distance to the target point from the finger tip plus a control magnitude penalty. The actions correspond to the torques applied to the 5 joints.

### 7.2 ALGORITHMS

We compare five algorithms (LOKI, TRPO, DAGGERED, THOR, SLOLS) and the idealistic setup of performing policy gradient steps directly from the expert policy (Ideal). To facilitate a fair comparison, all the algorithms are implemented based on a publicly available TRPO implementation (Dhariwal et al., 2017). Furthermore, they share the same parameters except for those

<sup>9</sup>The environments are defined in DartEnv, hosted at <https://github.com/DartEnv>.

that are unique to each algorithm as listed in Table A in Appendix A. The experimental results averaged across 25 random seeds are reported in Section 7.3.

**Policy and Value Networks** Feed-forward neural networks are used to construct the policy networks and the value networks in all the tasks (both have two hidden layers and 32 tanh units per layer). We consider Gaussian stochastic policies, i.e. for any state  $s \in \mathbb{S}$ ,  $\pi_s(a)$  is Gaussian distributed. The mean of the Gaussian  $\pi_s(a)$ , as a function of state, is modeled by the policy network, and the covariance matrix of Gaussian is restricted to be diagonal and independent of state. The policy networks and the value function networks are initialized randomly, except for the ideal setup (TRPO from expert), which is initialized as the expert.

**Expert Policy** The same sub-optimal expert is used by all algorithms (LOKI, DAGGERED, SLOLS, and THOR). It is obtained by running TRPO and stopping it before convergence. The estimate of the expert value function  $V_{\pi^*}$  (required by SLOLS and THOR) is learned by minimizing the sum of squared TD(0) error on a large separately collected set of demonstrations of this expert. The final explained variance for all the tasks is more than 0.97 (see Appendix A).

**First-Order Oracles** The on-policy advantage  $A_{\pi_n}$  in the first-order oracles for TRPO, SLOLS, and LOKI (in the reinforcement phase) is implemented using an on-policy value function estimator and Generalized Advantage Estimator (GAE) (Schulman et al., 2015b). For DAGGERED and the imitation phase of LOKI, the first-order oracle is calculated using (14). For SLOLS, we use the estimate  $A_{\pi^*}(s_t, a_t) \approx c(s_t, a_t) + \gamma \hat{V}_{\pi^*}(s_{t+1}) - \hat{V}_{\pi^*}(s_t)$ . And for THOR,  $A_{\pi_n, t}^{H, \pi^*}$  of the truncated-horizon problem is approximated by Monte-Carlo samples with an on-policy value function baseline estimated by regressing on these Monte-Carlo samples. Therefore, for *all* methods, an on-policy component is used in constructing the first-order oracle. The exponential weighting in GAE is 0.98; the mixing coefficient  $\lambda$  in SLOLS is 0.5;  $N_M$  in LOKI is reported in Table A in Appendix A, and  $N_m = \lfloor \frac{1}{2} N_M \rfloor$ , and  $d = 3$ .

**Mirror Descent** After receiving an update direction  $g_n$  from the first-order oracle, a KL-divergence-based trust region is specified. This is equivalent to setting the strictly convex function  $R_n$  in mirror descent to  $\frac{1}{2} \theta^\top F(\theta_n) \theta$  and choosing a proper learning rate. In our experiments, a larger KL-divergence limit (0.1) is selected for imitation gradient (14) (in DAGGERED and in the imitation phase of LOKI), and a smaller one (0.01) is set for all other algorithms. This decision follows

the guideline provided by the theoretical analysis in Section 3.2.2 and is because of the low variance in calculating the gradient of (14). Empirically, we observe using the larger KL-divergence limit with policy gradient led to high variance and instability.

### 7.3 EXPERIMENTAL RESULTS

We report the performance of these algorithms on various tasks in Figure 1. The performance is measured by the accumulated *rewards*, which are directly provided by OpenAI Gym.

We first establish the performance of two baselines, which represent standard RL (TRPO) and standard IL (DAGGERED). TRPO is able to achieve considerable and almost monotonic improvement from a randomly initialized policy. DAGGERED reaches the performance of the suboptimal policy in a relatively very small number of iterations, e.g. 15 iterations in 2D Walker, in which the suboptimal policy to imitate is TRPO at iteration 100. However, it fails to outperform the suboptimal expert.

Then, we evaluate the proposed algorithm LOKI and Ideal, the performance of which we wish to achieve in theory. LOKI consistently enjoys the best of both TRPO and DAGGERED: it improves as fast as DAGGERED at the beginning, keeps improving, and then finally matches the performance of Ideal after transitioning into the reinforcement phase. Interestingly, the on-policy value function learned, though not used, in the imitation phase helps LOKI transition from imitation phase to reinforcement phase smoothly.

Lastly, we compare LOKI to the two other baselines (SLOLS and THOR) that combine RL and IL. LOKI outperforms these two baselines by a considerably large margin in Hopper, 2D Walker, and 3D Walker; but surprisingly, the performance of SLOLS and THOR are inferior even to TRPO on these tasks. The main reason is that the first-order oracles of both methods is based on an estimated expert value function  $\hat{V}_{\pi^*}$ . As  $\hat{V}_{\pi^*}$  is only regressed on the data collected by running the expert policy, large covariate shift error could happen if the dimension of the state and action spaces are high, or if the uncontrolled system is complex or unstable. For example, in the low-dimensional Pendulum task and the simple Reacher task, the expert value function can generalize better. As a result, in these two cases, SLOLS and THOR achieve super-expert performance. However, in more complex tasks, where the effects of covariate shift amplifies exponentially with the dimension of the state space, THOR and SLOLS start to suffer from the inaccuracy of  $\hat{V}_{\pi^*}$ , as illustrated in the 2D Walker and 3D Walker tasks.



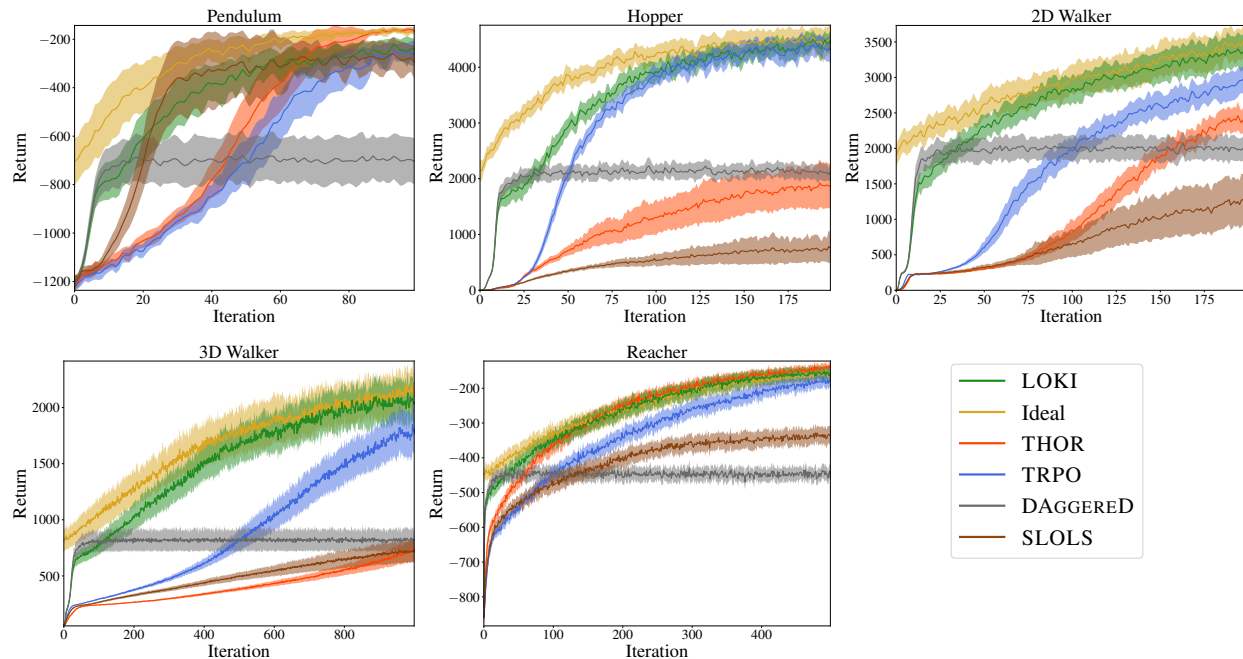


Figure 1: Learning curves. Shaded regions correspond to  $\pm \frac{1}{2}$ -standard deviation.

## 8 CONCLUSION

We present a simple, elegant algorithm, LOKI, that combines the best properties of RL and IL. Theoretically, we show that, by randomizing the switching time, LOKI can perform as if running policy gradient steps directly from the expert policy. Empirically, LOKI demonstrates superior performance compared with the expert policy and more complicated algorithms that attempt to combine RL and IL.

### Acknowledgements

This work was supported in part by NSF NRI Award 1637758, NSF CAREER Award 1750483, and NSF Graduate Research Fellowship under Grant No. 2015207631.

### References

Bregman, L. M. (1967). The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7(3):200–217.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. (2016). Openai gym.

Chang, K.-W., Krishnamurthy, A., Agarwal, A.,

Daume III, H., and Langford, J. (2015). Learning to search better than your teacher.

Cheng, C.-A. and Boots, B. (2018). Convergence of value aggregation for imitation learning. In *International Conference on Artificial Intelligence and Statistics*, pages 1801–1809.

Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., and Wu, Y. (2017). Openai baselines.

Ghadimi, S., Lan, G., and Zhang, H. (2016). Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Mathematical Programming*, 155(1-2):267–305.

Gibbs, A. L. and Su, F. E. (2002). On choosing and bounding probability metrics. *International Statistical Review*, 70(3):419–435.

Grathwohl, W., Choi, D., Wu, Y., Roeder, G., and Duvenaud, D. (2017). Backpropagation through the void: Optimizing control variates for black-box gradient estimation. *arXiv preprint arXiv:1711.00123*.

Greensmith, E., Bartlett, P. L., and Baxter, J. (2004). Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5(Nov):1471–1530.

Juditsky, A., Nemirovski, A., et al. (2011). First order methods for nonsmooth convex large-scale optimiza-

- tion, i: general purpose methods. *Optimization for Machine Learning*, pages 121–148.
- Kakade, S. and Langford, J. (2002). Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*, volume 2, pages 267–274.
- Kakade, S. M. (2002). A natural policy gradient. In *Advances in Neural Information Processing Systems*, pages 1531–1538.
- Konda, V. R. and Tsitsiklis, J. N. (2000). Actor-critic algorithms. In *Advances in Neural Information Processing Systems*, pages 1008–1014.
- Lee, J., Grey, M. X., Ha, S., Kunz, T., Jain, S., Ye, Y., Srinivasa, S. S., Stilman, M., and Liu, C. K. (2018). DART: Dynamic animation and robotics toolkit. *The Journal of Open Source Software*, 3(22):500.
- Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937.
- Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.
- Nair, A., McGrew, B., Andrychowicz, M., Zaremba, W., and Abbeel, P. (2017). Overcoming exploration in reinforcement learning with demonstrations. *arXiv preprint arXiv:1709.10089*.
- Nemirovski, A., Juditsky, A., Lan, G., and Shapiro, A. (2009). Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19(4):1574–1609.
- Pan, Y., Cheng, C.-A., Saigol, K., Lee, K., Yan, X., Theodorou, E., and Boots, B. (2017). Agile off-road autonomous driving using end-to-end deep imitation learning. *arXiv preprint arXiv:1709.07174*.
- Peters, J., Mülling, K., and Altun, Y. (2010). Relative entropy policy search. In *AAAI*, pages 1607–1612. Atlanta.
- Peters, J. and Schaal, S. (2008). Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190.
- Pomerleau, D. A. (1989). *Alvinn: An autonomous land vehicle in a neural network*. In *Advances in Neural Information Processing Systems*, pages 305–313.
- Rajeswaran, A., Kumar, V., Gupta, A., Schulman, J., Todorov, E., and Levine, S. (2017). Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*.
- Rawlik, K., Toussaint, M., and Vijayakumar, S. (2012). On stochastic optimal control and reinforcement learning by approximate inference. In *Robotics: Science and Systems*, volume 13, pages 3052–3056.
- Ross, S. and Bagnell, J. A. (2014). Reinforcement and imitation learning via interactive no-regret learning. *arXiv preprint arXiv:1406.5979*.
- Ross, S., Gordon, G., and Bagnell, D. (2011). A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics*, pages 627–635.
- Schaal, S. (1999). Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, 3(6):233–242.
- Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015a). Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897.
- Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. (2015b). High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*.
- Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. (2017). Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*.
- Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. (2014). Deterministic policy gradient algorithms. In *International Conference on Machine Learning*.
- Sun, W., Bagnell, J. A., and Boots, B. (2018). Truncated horizon policy search: Deep combination of reinforcement and imitation. *International Conference on Learning Representations*.
- Sun, W., Venkatraman, A., Gordon, G. J., Boots, B., and Bagnell, J. A. (2017). Deeply aggregated: Differentiable imitation learning for sequential prediction. *arXiv preprint arXiv:1703.01030*.
- Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, pages 1057–1063.
- Tucker, G., Mnih, A., Maddison, C. J., Lawson, J., and Sohl-Dickstein, J. (2017). Rebar: Low-variance, unbiased gradient estimates for discrete latent variable models. In *Advances in Neural Information Processing Systems*, pages 2624–2633.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Reinforcement Learning*, pages 5–32. Springer.

---

# Hyperspherical Variational Auto-Encoders

---

Tim R. Davidson\* Luca Falorsi\* Nicola De Cao\* Thomas Kipf Jakub M. Tomczak

University of Amsterdam

## Abstract

The Variational Auto-Encoder (VAE) is one of the most used unsupervised machine learning models. But although the default choice of a Gaussian distribution for both the prior and posterior represents a mathematically convenient distribution often leading to competitive results, we show that this parameterization fails to model data with a latent hyperspherical structure. To address this issue we propose using a von Mises-Fisher (vMF) distribution instead, leading to a hyperspherical latent space. Through a series of experiments we show how such a hyperspherical VAE, or  $S$ -VAE, is more suitable for capturing data with a hyperspherical latent structure, while outperforming a normal,  $\mathcal{N}$ -VAE, in low dimensions on other data types.

## 1 INTRODUCTION

First introduced by Kingma and Welling (2013); Rezende et al. (2014), the Variational Auto-Encoder (VAE) is an unsupervised generative model that presents a principled fashion for performing variational inference using an auto-encoding architecture. Applying the non-centered parameterization of the variational posterior (Kingma and Welling, 2014), further simplifies sampling and allows to reduce bias in calculating gradients for training. Although the default choice of a Gaussian prior is mathematically convenient, we can show through a simple example that in some cases it breaks the assumption of an *uninformative* prior leading to unstable results. Imagine a dataset on the circle  $\mathcal{Z} \subset \mathcal{S}^1$ , that is subsequently embedded in  $\mathbb{R}^N$  using a transformation  $f$  to obtain  $f : \mathcal{Z} \rightarrow \mathcal{X} \subset \mathbb{R}^N$ . Given two hidden units, an autoencoder quickly discovers

the latent circle, while a normal VAE becomes highly unstable. This is to be expected as a Gaussian prior is concentrated around the origin, while the KL-divergence tries to reconcile the differences between  $\mathcal{S}^1$  and  $\mathbb{R}^2$ . A more detailed discussion of this ‘manifold mismatch’ problem will follow in subsection 2.3.

The fact that some data types like *directional data* are better explained through spherical representations is long known and well-documented (Mardia, 1975; Fisher et al., 1987), with examples spanning from protein structure, to observed wind directions. Moreover, for many modern problems such as text analysis or image classification, data is often first normalized in a preprocessing step to focus on the directional distribution. Yet, few machine learning methods explicitly account for the intrinsically spherical nature of some data in the modeling process. In this paper, we propose to use the *von Mises-Fisher* (vMF) distribution as an alternative to the Gaussian distribution. This replacement leads to a hyperspherical latent space as opposed to a hyperplanar one, where the Uniform distribution on the hypersphere is conveniently recovered as a special case of the vMF. Hence this approach allows for a truly uninformative prior, and has a clear advantage in the case of data with a hyperspherical interpretation. This was previously attempted by Hasnat et al. (2017), but crucially they do not learn the concentration parameter around the mean,  $\kappa$ .

In order to enable training of the concentration parameter, we extend the *reparameterization trick* for rejection sampling as recently outlined in Naesseth et al. (2017) to allow for  $n$  additional transformations. We then combine this with the rejection sampling procedure proposed by Ulrich (1984) to efficiently reparameterize the VAE <sup>1</sup>.

We demonstrate the utility of replacing the normal distribution with the von Mises-Fisher distribution for generating latent representations by conducting a range of experiments in three distinct settings. First, we show that

---

\* Equal contribution.

<sup>1</sup> <https://github.com/nicola-decao/s-vae>

our  $\mathcal{S}$ -VAEs outperform VAEs with the Gaussian variational posterior ( $\mathcal{N}$ -VAEs) in recovering a hyperspherical latent structure. Second, we conduct a thorough comparison with  $\mathcal{N}$ -VAEs on the MNIST dataset through an unsupervised learning task and a semi-supervised learning scenario. Finally, we show that  $\mathcal{S}$ -VAEs can significantly improve link prediction performance on citation network datasets in combination with a *Variational Graph Auto-Encoder* (VGAE) (Kipf and Welling, 2016).

## 2 VARIATIONAL AUTO-ENCODERS

### 2.1 FORMULATION

In the VAE setting, we have a latent variable model for data, where  $\mathbf{z} \in \mathbb{R}^M$  denotes latent variables,  $\mathbf{x}$  is a vector of  $D$  observed variables, and  $p_\phi(\mathbf{x}, \mathbf{z})$  is a parameterized model of the joint distribution. Our objective is to optimize the log-likelihood of the data,  $\log \int p_\phi(\mathbf{x}, \mathbf{z}) d\mathbf{z}$ . When  $p_\phi(\mathbf{x}, \mathbf{z})$  is parameterized by a neural network, marginalizing over the latent variables is generally intractable. One way of solving this issue is to maximize the Evidence Lower Bound (ELBO)

$$\log \int p_\phi(\mathbf{x}, \mathbf{z}) d\mathbf{z} \geq \mathbb{E}_{q(\mathbf{z})}[\log p_\phi(\mathbf{x}|\mathbf{z})] - KL(q(\mathbf{z})||p(\mathbf{z})), \quad (1)$$

where  $q(\mathbf{z})$  is the approximate posterior distribution, belonging to a family  $\mathcal{Q}$ . The bound is tight if  $q(\mathbf{z}) = p(\mathbf{z}|\mathbf{x})$ , meaning  $q(\mathbf{z})$  is optimized to approximate the true posterior. While in theory  $q(\mathbf{z})$  should be optimized for every data point  $\mathbf{x}$ , to make inference more scalable to larger datasets the VAE setting introduces an inference network  $q_\psi(\mathbf{z}|\mathbf{x}; \theta)$  parameterized by a neural network that outputs a probability distribution for each data point  $\mathbf{x}$ . The final objective is therefore to maximize

$$\mathcal{L}(\phi, \psi) = \mathbb{E}_{q_\psi(\mathbf{z}|\mathbf{x}; \theta)}[\log p_\phi(\mathbf{x}|\mathbf{z})] - KL(q_\psi(\mathbf{z}|\mathbf{x}; \theta)||p(\mathbf{z})), \quad (2)$$

In the original VAE both the prior and the posterior are defined as normal distributions. We can further efficiently approximate the ELBO by Monte Carlo estimates, using the *reparameterization trick* (Kingma and Welling, 2013; Rezende et al., 2014). This is done by expressing a sample of  $\mathbf{z} \sim q_\psi(\mathbf{z}|\mathbf{x}; \theta)$ , as  $\mathbf{z} = h(\theta, \varepsilon, \mathbf{x})$ , where  $h$  is a reparameterization transformation and  $\varepsilon \sim s(\varepsilon)$  is some noise random variable independent from  $\theta$ .

### 2.2 THE LIMITATIONS OF A GAUSSIAN DISTRIBUTION PRIOR

**Low dimensions: origin gravity** In low dimensions, the Gaussian density presents a concentrated probability

mass around the origin, encouraging points to cluster in the center. This is particularly problematic when the data is divided into multiple clusters. Although an ideal latent space should separate clusters for each class, the normal prior will encourage all the cluster centers towards the origin. An ideal prior would only stimulate the variance of the posterior without forcing its mean to be close to the center. A prior satisfying these properties is a uniform over the entire space. Such a uniform prior, however, is not well defined on the hyperplane.

**High dimensions: soap bubble effect** It is a well-known phenomenon that the standard Gaussian distribution in high dimensions tends to resemble a uniform distribution on the surface of a hypersphere, with the vast majority of its mass concentrated on the hyperspherical shell. Hence it would appear interesting to compare the behavior of a Gaussian approximate posterior with an approximate posterior already naturally defined on the hypersphere. This is also motivated from a theoretical point of view, since the Gaussian definition is based on the  $L_2$  norm that suffers from the curse of dimensionality.

### 2.3 BEYOND THE HYPERPLANE

Once we let go of the hyperplanar assumption, the possibility of a uniform prior on the hypersphere opens up. Mirroring our discussion in the previous subsection, such a prior would exhibit no pull towards the origin allowing clusters of data to evenly spread over the surface with no directional bias. Additionally, in higher dimensions, the cosine similarity is a more meaningful distance measure than the Euclidean norm.

**Manifold mapping** In general, exploring VAE models that allow a mapping to distributions in a latent space not homeomorphic to  $\mathbb{R}^D$  is of fundamental interest. Consider data lying in a small  $M$ -dimensional manifold  $\mathcal{M}$ , embedded in a much higher dimensional space  $\mathcal{X} = \mathbb{R}^N$ . For most real data, this manifold will likely not be homeomorphic to  $\mathbb{R}^M$ . An encoder can be considered as a smooth map  $enc : \mathcal{X} \rightarrow \mathcal{Z} = \mathbb{R}^D$  from the original space to  $\mathcal{Z}$ . The restriction of the encoder to  $\mathcal{M}$ ,  $enc|_{\mathcal{M}} : \mathcal{M} \rightarrow \mathcal{Z}$  will also be a smooth mapping. However since  $\mathcal{M}$  is not homeomorphic to  $\mathcal{Z}$  if  $D \leq M$ , then  $enc|_{\mathcal{M}}$  cannot be a homeomorphism. That is, there exists no invertible and globally continuous mapping between the coordinates of  $\mathcal{M}$  and the ones of  $\mathcal{Z}$ . Conversely if  $D > M$  then  $\mathcal{M}$  can be smoothly embedded in  $\mathcal{Z}$  for  $D$  sufficiently large <sup>2</sup>, such that  $enc|_{\mathcal{M}} : \mathcal{M} \rightarrow enc|_{\mathcal{M}}(\mathcal{M}) =: emb(\mathcal{M}) \subset \mathcal{Z}$  is a homeomorphism and  $emb(\mathcal{M})$  denotes the embedding of

<sup>2</sup>By the Whitney embedding theorem any smooth real  $M$ -dimensional manifold can be smoothly embedded in  $\mathbb{R}^{2M}$

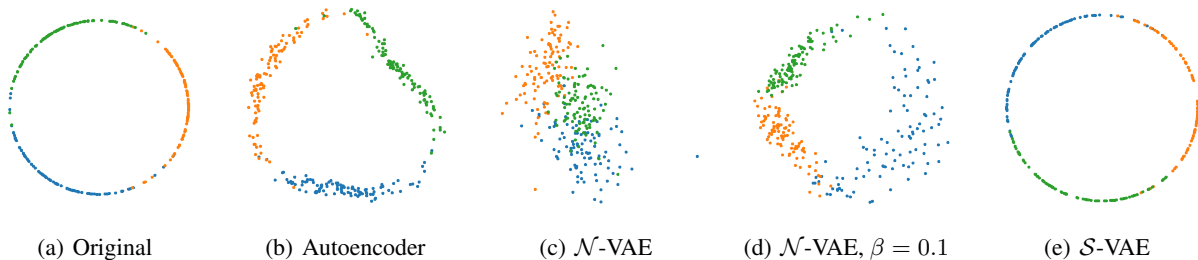


Figure 1: Plots of the original latent space (a) and learned latent space representations in different settings, where  $\beta$  is a re-scaling factor for weighting the KL divergence. (Best viewed in color)

$\mathcal{M}$ . Yet, since  $D > M$ , when taking random points in the latent space they will most likely *not* be in  $emb(\mathcal{M})$  resulting in a poorly reconstructed sample.

The VAE tries to solve this problem by forcing  $\mathcal{M}$  to be mapped into an approximate posterior distribution that has support in the entire  $\mathcal{Z}$ . Clearly, this approach is bound to fail since the two spaces have a fundamentally different structure. This can likely produce two behaviors: first, the VAE could just smooth the original embedding  $emb(\mathcal{M})$  leaving most of the latent space empty, leading to bad samples. Second, if we increase the KL term the encoder will be pushed to occupy all the latent space, but this will create instability and discontinuity, affecting the convergence of the model. To validate our intuition we performed a small proof of concept experiment using  $\mathcal{M} = \mathcal{S}^1$ , which is visualized in Figure 1. Note that as expected the auto-encoder in Figure 1(b) mostly recovers the original latent space of Figure 1(a) as there are no distributional restrictions. In Figure 1(c) we clearly observe for the  $\mathcal{N}$ -VAE that points collapse around the origin due to the KL, which is much less pronounced in Figure 1(d) when its contribution is scaled down. Lastly, the  $\mathcal{S}$ -VAE almost perfectly recovers the original circular latent space. The observed behavior confirms our intuition.

To solve this problem the best option would be to directly specify a  $\mathcal{Z}$  homeomorphic to  $\mathcal{M}$  and distributions on  $\mathcal{M}$ . However, for real data discovering the structure of  $\mathcal{M}$  will often be a difficult inference task. Nevertheless, we believe this shows investigating VAE architectures that map to posterior distributions defined on manifolds different than the Euclidean space is a topic worth exploring.

### 3 REPLACING GAUSSIAN WITH VON MISES-FISHER

#### 3.1 VON MISES-FISHER DISTRIBUTION

The *von Mises-Fisher* (vMF) distribution is often seen

as the Normal Gaussian distribution on a hypersphere. Analogous to a Gaussian, it is parameterized by  $\mu \in \mathbb{R}^m$  indicating the mean direction, and  $\kappa \in \mathbb{R}_{\geq 0}$  the concentration around  $\mu$ . For the special case of  $\kappa = 0$ , the vMF represents a Uniform distribution. The probability density function of the vMF distribution for a random unit vector  $\mathbf{z} \in \mathbb{R}^m$  (or  $\mathbf{z} \in \mathcal{S}^{m-1}$ ) is then defined as

$$q(\mathbf{z}|\mu, \kappa) = \mathcal{C}_m(\kappa) \exp(\kappa \mu^T \mathbf{z}), \quad (3)$$

$$\mathcal{C}_m(\kappa) = \frac{\kappa^{m/2-1}}{(2\pi)^{m/2} \mathcal{I}_{m/2-1}(\kappa)}, \quad (4)$$

where  $\|\mu\|^2 = 1$ ,  $\mathcal{C}_m(\kappa)$  is the normalizing constant, and  $\mathcal{I}_v$  denotes the modified Bessel function of the first kind at order  $v$ .

#### 3.2 KL DIVERGENCE

As previously emphasized, one of the main advantages of using the vMF distribution as an approximate posterior is that we are able to place a uniform prior on the latent space. The KL divergence term  $KL(\text{vMF}(\mu, \kappa) || U(\mathcal{S}^{m-1}))$  to be optimized is:

$$\kappa \frac{\mathcal{I}_{m/2}(k)}{\mathcal{I}_{m/2-1}(k)} + \log \mathcal{C}_m(\kappa) - \log \left( \frac{2(\pi^{m/2})}{\Gamma(m/2)} \right)^{-1}, \quad (5)$$

see Appendix B for complete derivation. Notice that since the KL term does not depend on  $\mu$ , this parameter is only optimized in the reconstruction term. The above expression cannot be handled by automatic differentiation packages because of the modified Bessel function in  $\mathcal{C}_m(\kappa)$ . Thus, to optimize this term we derive the gradient with respect to the concentration parameter:

$$\nabla_{\kappa} KL(\text{vMF}(\mu, \kappa) || U(\mathcal{S}^{m-1})) = \frac{1}{2} k \left( \frac{\mathcal{I}_{m/2+1}(k)}{\mathcal{I}_{m/2-1}(k)} + \frac{\mathcal{I}_{m/2}(k) (\mathcal{I}_{m/2-2}(k) + \mathcal{I}_{m/2}(k))}{\mathcal{I}_{m/2-1}(k)^2} + 1 \right), \quad (6)$$

---

**Algorithm 1** vMF sampling

---

**Input:** dimension  $m$ , mean  $\mu$ , concentration  $\kappa$   
sample  $\mathbf{v} \sim U(\mathcal{S}^{m-2})$   
sample  $\omega \sim g(\omega|\kappa, m) \propto \exp(\omega\kappa)(1 - \omega^2)^{\frac{1}{2}(m-3)}$   
{acceptance-rejection sampling}  
 $\mathbf{z}' \leftarrow (\omega; (\sqrt{1 - \omega^2})\mathbf{v}^\top)^\top$   
 $U \leftarrow \text{Householder}(\mathbf{e}_1, \mu)$  {Householder transform}  
**Return:**  $U\mathbf{z}'$

---

where the modified Bessel functions can be computed without numerical instabilities using the exponentially scaled modified Bessel function.

### 3.3 SAMPLING PROCEDURE

To sample from the vMF we follow the procedure of Ulrich (1984), outlined in Algorithm 1. We first sample from a vMF  $q(\mathbf{z}|\mathbf{e}_1, \kappa)$  with modal vector  $\mathbf{e}_1 = (1, 0, \dots, 0)$ . Since the vMF density is uniform in all the  $m - 2$  dimensional sub-hyperspheres  $\{\mathbf{x} \in \mathcal{S}^{m-1} | \mathbf{e}_1^\top \mathbf{x} = \omega\}$ , the sampling technique reduces to sampling the value  $\omega$  from the univariate density  $g(\omega|\kappa, m) \propto \exp(\kappa\omega)(1 - \omega^2)^{(m-3)/2}$ ,  $\omega \in [-1, 1]$ , using an acceptance-rejection scheme. After getting a sample from  $q(\mathbf{z}|\mathbf{e}_1, \kappa)$  an orthogonal transformation  $U(\mu)$  is applied such that the transformed sample is distributed according to  $q(\mathbf{z}|\mu, \kappa)$ . This can be achieved using a Householder reflection such that  $U(\mu)\mathbf{e}_1 = \mu$ . A more in-depth explanation of the sampling technique can be found in Appendix A.

It is worth noting that the sampling technique does not suffer from the curse of dimensionality, as the acceptance-rejection procedure is only applied to a univariate distribution. Moreover in the case of  $\mathcal{S}^2$ , the density  $g(\omega|\kappa, 3)$  reduces to  $g(\omega|\kappa, 3) \propto \exp(k\omega)\mathbb{1}_{[-1, +1]}(\omega)$  which can be directly sampled without rejection.

### 3.4 N-TRANSFORMATION REPARAMETERIZATION TRICK

While the *reparameterization trick* is easily implementable in the normal case, unfortunately it can only be applied to a handful of distributions. However a recent technique introduced by Naesseth et al. (2017) allows to extend the reparameterization trick to the wide class of distributions that can be simulated using rejection sampling. Dropping the dependence from  $\mathbf{x}$  for simplicity, assume the approximate posterior is of the form  $g(\omega|\theta)$  and that it can be sampled by making proposals from  $r(\omega|\theta)$ . If *the proposal distribution can be reparameterized* we can still perform the reparameterization trick. Let  $\varepsilon \sim s(\varepsilon)$ , and  $\omega = h(\varepsilon, \theta)$ , a reparameterization of the proposal distribution,  $r(\omega|\theta)$ . Performing the reparameterization trick

for  $g(\omega|\theta)$  is made possible by the fundamental lemma proven in (Naesseth et al., 2017):

**Lemma 1.** *Let  $f$  be any measurable function and  $\varepsilon \sim \pi(\varepsilon|\theta) = s(\varepsilon)\frac{g(h(\varepsilon, \theta)|\theta)}{r(h(\varepsilon, \theta)|\theta)}$  the distribution of the accepted sample. Then:*

$$\begin{aligned} \mathbb{E}_{\pi(\varepsilon|\theta)}[f(h(\varepsilon, \theta))] &= \int f(h(\varepsilon, \theta))\pi(\varepsilon|\theta)d\varepsilon \\ &= \int f(\omega)g(\omega|\theta)d\omega = \mathbb{E}_{g(\omega|\theta)}[f(\omega)], \end{aligned} \quad (7)$$

Then the gradient can be taken using the log derivative trick:

$$\begin{aligned} \nabla_\theta \mathbb{E}_{g(\omega|\theta)}[f(\omega)] &= \nabla_\theta \mathbb{E}_{\pi(\varepsilon|\theta)}[f(h(\varepsilon, \theta))] = \\ &= \mathbb{E}_{\pi(\varepsilon|\theta)}[\nabla_\theta f(h(\varepsilon, \theta))] + \\ &+ \mathbb{E}_{\pi(\varepsilon|\theta)} \left[ f(h(\varepsilon, \theta)) \nabla_\theta \log \frac{g(h(\varepsilon, \theta)|\theta)}{r(h(\varepsilon, \theta)|\theta)} \right], \end{aligned} \quad (8)$$

However, in the case of the vMF a different procedure is required. After performing the transformation  $h(\varepsilon, \theta)$  and accepting/rejecting the sample, we sample *another* random variable  $\mathbf{v} \sim \pi_2(\mathbf{v})$ , and then apply a transformation  $\mathbf{z} = \mathcal{T}(h(\varepsilon, \theta), \mathbf{v}; \theta)$ , such that  $\mathbf{z} \sim q_\psi(\mathbf{z}|\theta)$  is distributed as the approximate posterior (in our case a vMF). Effectively this entails applying another reparameterization trick after the acceptance/rejection step. To still be able to perform the reparameterization we show that Lemma 1 fundamentally still holds in this case as well.

**Lemma 2.** *Let  $f$  be any measurable function and  $\varepsilon \sim \pi_1(\varepsilon|\theta) = s(\varepsilon)\frac{g(h(\varepsilon, \theta)|\theta)}{r(h(\varepsilon, \theta)|\theta)}$  the distribution of the accepted sample. Also let  $\mathbf{v} \sim \pi_2(\mathbf{v})$ , and  $\mathcal{T}$  a transformation that depends on the parameters such that if  $\mathbf{z} = \mathcal{T}(\omega, \mathbf{v}; \theta)$  with  $\omega \sim g(\omega|\theta)$ , then  $\mathbf{z} \sim q(\mathbf{z}|\theta)$ :*

$$\begin{aligned} \mathbb{E}_{(\varepsilon, \mathbf{v}) \sim \pi_1(\varepsilon|\theta)\pi_2(\mathbf{v})} [f(\mathcal{T}(h(\varepsilon, \theta), \mathbf{v}; \theta))] &= \\ \int f(\mathbf{z})q(\mathbf{z}|\theta)d\mathbf{z} &= \mathbb{E}_{q(\mathbf{z}|\theta)}[f(\mathbf{z})], \end{aligned} \quad (9)$$

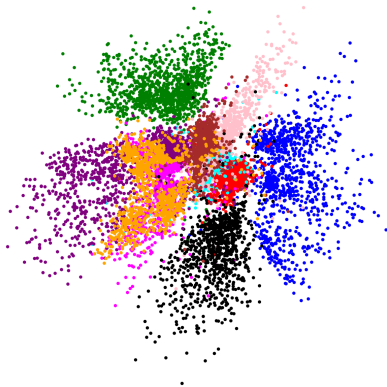
*Proof.* See Appendix C.  $\square$

With this result we are able to derive a gradient expression similarly as done in equation 8. We refer to Appendix D for a complete derivation.

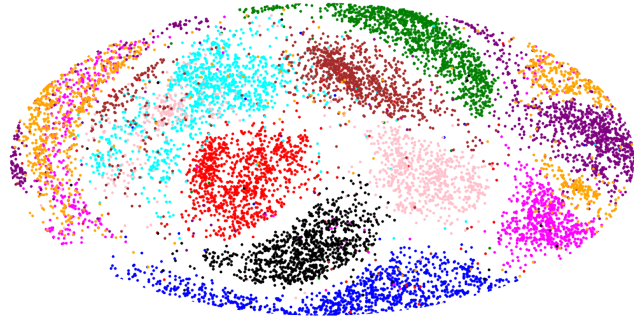
### 3.5 BEHAVIOR IN HIGH DIMENSIONS

The surface area of a hypersphere is defined as

$$S(m - 1) = r^m \frac{2(\pi^{m/2})}{\Gamma(m/2)}, \quad (10)$$



(a)  $\mathbb{R}^2$  latent space of the  $\mathcal{N}$ -VAE.



(b) Hammer projection of  $\mathcal{S}^2$  latent space of the  $\mathcal{S}$ -VAE.

Figure 2: Latent space visualization of the 10 MNIST digits in 2 dimensions of both  $\mathcal{N}$ -VAE (left) and  $\mathcal{S}$ -VAE (right). (Best viewed in color)

where  $m$  is the dimensionality and  $r$  the radius. Notice that  $S(m-1) \rightarrow 0$ , as  $m \rightarrow \infty$ . However, even for  $m > 20$  we observe a *vanishing surface problem* (see Figure 6 in Appendix E). This could thus lead to unstable behavior of hyperspherical models in high dimensions.

## 4 RELATED WORK

**Extending the VAE** The majority of VAE extensions focus on increasing the flexibility of the approximate posterior. This is usually achieved through *normalizing flows* (Rezende and Mohamed, 2015), a class of invertible transformations applied sequentially to an initial reparameterizable density  $q_0(\mathbf{z}_0)$ , allowing for more complex posteriors. Normalizing flows can be considered orthogonal to our approach. While allowing for a more flexible posterior, they do not modify the standard normal prior assumption. In (Gemici et al., 2016) a first attempt is made to extend normalizing flows to Riemannian manifolds. However, as the method relies on the existence of a diffeomorphism between  $\mathbb{R}^N$  and  $\mathcal{S}^N$ , it is unsuited for hyperspheres.

One approach to obtain a more flexible prior is to use a simple mixture of Gaussians (MoG) prior (Dilokthanakul et al., 2016). The recently introduced VampPrior model (Tomczak and Welling, 2018) outlines several advantages over the MoG and instead tries to learn a more flexible prior by expressing it as a mixture of approximate posteriors. A non-parametric prior is proposed in Nalisnick and Smyth (2017), utilizing a truncated stick-breaking process. Opposite to these approaches, we aim at using a non-informative prior to simplify the inference.

The closest approach to ours is a VAE with a vMF distribution in the latent space used for a sentence generation

task by (Guu et al., 2018). While formally this approach is cast as a variational approach, the proposed model does not reparameterize and learn the concentration parameter  $\kappa$ , treating it as a constant value that remains the same for every approximate posterior instead. Critically, as indicated in Equation 5, the KL divergence term only depends on  $\kappa$  therefore leaving  $\kappa$  constant means never explicitly optimizing the KL divergence term in the loss. The method then only optimizes the reconstruction error by adding vMF noise to the encoder output in the latent space to still allow generation. Moreover, using a fixed global  $\kappa$  for *all* the approximate posteriors severely limits the flexibility and the expressiveness of the model.

**Non-Euclidean Latent Space** In Liu and Zhu (2018), a general model to perform Bayesian inference in Riemannian Manifolds is proposed. Following other Stein-related approaches, the method does not explicitly define a posterior density but approximates it with a number of particles. Despite its generality and flexibility, it requires the choice of a kernel on the manifold and multiple particles to have a good approximation of the posterior distribution. The former is not necessarily straightforward, while the latter quickly becomes computationally unfeasible.

Another approach by Nickel and Kiela (2017), capitalizes on the hierarchical structure present in some data types. By learning the embeddings for a graph in a non-euclidean negative curvature hyperbolic space, they show this topology has clear advantages over embedding these objects in a Euclidean space. Although they did not use a VAE-based approach, that is, they did not build a probabilistic generative model of the data interpreting the embeddings as latent variables, this approach shows the merit of explicitly adjusting the choice of latent topology to the data used.



Table 1: Summary of results (mean and standard-deviation over 10 runs) of unsupervised model on MNIST. RE and KL correspond respectively to the reconstruction and the KL part of the ELBO. Best results are highlighted only if they passed a student t-test with  $p < 0.01$ .

Method	$\mathcal{N}$ -VAE				$\mathcal{S}$ -VAE			
	LL	$\mathcal{L}[q]$	RE	KL	LL	$\mathcal{L}[q]$	RE	KL
$d = 2$	-135.73 $\pm$ .83	-137.08 $\pm$ .83	-129.84 $\pm$ .91	7.24 $\pm$ .11	<b>-132.50</b> $\pm$ .73	-133.72 $\pm$ .85	-126.43 $\pm$ .91	7.28 $\pm$ .14
$d = 5$	-110.21 $\pm$ .21	-112.98 $\pm$ .21	-100.16 $\pm$ .22	12.82 $\pm$ .11	<b>-108.43</b> $\pm$ .09	-111.19 $\pm$ .08	-97.84 $\pm$ .13	13.35 $\pm$ .06
$d = 10$	-93.84 $\pm$ .30	-98.36 $\pm$ .30	-78.93 $\pm$ .30	19.44 $\pm$ .14	<b>-93.16</b> $\pm$ .31	-97.70 $\pm$ .32	-77.03 $\pm$ .39	20.67 $\pm$ .08
$d = 20$	-88.90 $\pm$ .26	-94.79 $\pm$ .19	-71.29 $\pm$ .45	23.50 $\pm$ .31	-89.02 $\pm$ .31	-96.15 $\pm$ .32	-67.65 $\pm$ .43	28.50 $\pm$ .22
$d = 40$	<b>-88.93</b> $\pm$ .30	-94.91 $\pm$ .18	-71.14 $\pm$ .56	23.77 $\pm$ .49	-90.87 $\pm$ .34	-101.26 $\pm$ .33	-67.75 $\pm$ .70	33.50 $\pm$ .45

**A Hyperspherical Perspective** As noted before, a distinction must be made between models dealing with the challenges of intrinsically hyperspherical data like omnidirectional video, and those attempting to exploit some latent hyperspherical manifold. A recent example of the first can be found in Cohen et al. (2018), where *spherical CNNs* are introduced. While flattening a spherical image produces unavoidable distortions, the newly defined convolutions take into account its geometrical properties.

The most general implementation of the second model type was proposed by Gopal and Yang (2014), who introduced a suite of models to improve cluster performance of high-dimensional data based on mixture of vMF distributions. They showed that reducing an object representation to its directional components increases clusterability over standard methods like  $K$ -Means or Latent Dirichlet Allocation (Blei et al., 2003).

Specific applications of the vMF can be further found ranging from computer vision, where it is used to infer structure from motion (Guan and Smith, 2017) in spherical video, or structure from texture (Wilson et al., 2014), to natural language processing, where it is utilized in text analysis (Banerjee et al., 2003, 2005) and topic modeling (Banerjee and Basu, 2007; Reisinger et al., 2010).

Additionally, modeling data by restricting it to a hypersphere provides some natural regularizing properties as noted in (Liu et al., 2017). Finally Aytekin et al. (2018) show on a variety of deep auto-encoder models that adding L2 normalization to the latent space during training, i.e. forcing the latent space on a hypersphere, improves clusterability.

## 5 EXPERIMENTS

In this section, we first perform a series of experiments to investigate the theoretical properties of the proposed  $\mathcal{S}$ -VAE compared to the  $\mathcal{N}$ -VAE. In a second experiment, we show how  $\mathcal{S}$ -VAEs can be used in semi-supervised

tasks to create a better separable latent representation to enhance classification. In the last experiment, we show that the  $\mathcal{S}$ -VAE indeed presents a promising alternative to  $\mathcal{N}$ -VAEs for data with a non-Euclidean latent representation of low dimensionality, on a link prediction task for three citation networks. All architecture and hyperparameter details are given in Appendix F.

### 5.1 RECOVERING HYPERSPHERICAL LATENT REPRESENTATIONS

In this first experiment we build on the motivation developed in Subsection 2.3, by confirming with a synthetic data example the difference in behavior of the  $\mathcal{N}$ -VAE and  $\mathcal{S}$ -VAE in recovering latent hyperspheres. We first generate samples from a mixture of three vMFs on the circle,  $\mathcal{S}^1$ , as shown in Figure 1(a), which subsequently are mapped into the higher dimensional  $\mathbb{R}^{100}$  by applying a noisy, non-linear transformation. After this, we in turn train an auto-encoder, a  $\mathcal{N}$ -VAE, and a  $\mathcal{S}$ -VAE. We further investigate the behavior of the  $\mathcal{N}$ -VAE, by training a model using a scaled down KL divergence.

**Results** The resulting latent spaces, displayed in Figure 1, clearly confirm the intuition built in Subsection 2.3. As expected, in Figure 1(b) the auto-encoder is perfectly capable to embed in low dimensions the original underlying data structure. However, most parts of the latent space are not occupied by points, critically affecting the ability to generate meaningful samples.

In the  $\mathcal{N}$ -VAE setting we observe two types of behaviours, summarized by Figures 1(c) and 1(d). In the first we observe that if the prior is too strong it will force the posterior to match the prior shape, concentrating the samples in the center. However, this prevents the  $\mathcal{N}$ -VAE to correctly represent the true shape of the data and creates instability problems for the decoder around the origin. On the contrary, if we scale down the KL term, we observe that the samples from the approximate posterior maintain

Table 2: Summary of results (mean accuracy and standard-deviation over 20 runs) of semi-supervised  $K$ -NN on MNIST. Best results are highlighted only if they passed a student t-test with  $p < 0.01$ .

Method	100		600		1000	
	$\mathcal{N}$ -VAE	$\mathcal{S}$ -VAE	$\mathcal{N}$ -VAE	$\mathcal{S}$ -VAE	$\mathcal{N}$ -VAE	$\mathcal{S}$ -VAE
$d = 2$	72.6 $\pm$ 2.1	<b>77.9</b> $\pm$ 1.6	80.8 $\pm$ 0.5	<b>84.9</b> $\pm$ 0.6	81.7 $\pm$ 0.5	<b>85.6</b> $\pm$ 0.5
$d = 5$	81.8 $\pm$ 2.0	<b>87.5</b> $\pm$ 1.0	90.9 $\pm$ 0.4	<b>92.8</b> $\pm$ 0.3	92.0 $\pm$ 0.2	<b>93.4</b> $\pm$ 0.2
$d = 10$	75.7 $\pm$ 1.8	<b>80.6</b> $\pm$ 1.3	88.4 $\pm$ 0.5	<b>91.2</b> $\pm$ 0.4	90.2 $\pm$ 0.4	<b>92.8</b> $\pm$ 0.3
$d = 20$	71.3 $\pm$ 1.9	<b>72.8</b> $\pm$ 1.6	88.3 $\pm$ 0.5	<b>89.1</b> $\pm$ 0.6	90.1 $\pm$ 0.4	<b>91.1</b> $\pm$ 0.3
$d = 40$	<b>72.3</b> $\pm$ 1.6	67.7 $\pm$ 2.3	88.0 $\pm$ 0.5	87.4 $\pm$ 0.7	90.3 $\pm$ 0.5	90.4 $\pm$ 0.4

a shape that reflects the  $\mathcal{S}^1$  structure smoothed with Gaussian noise. However, as the approximate posterior differs strongly from the prior, obtaining meaningful samples from the latent space again becomes problematic.

The  $\mathcal{S}$ -VAE on the other hand, almost perfectly recovers the original dataset structure, while the samples from the approximate posterior closely match the prior distribution. This simple experiment confirms the intuition that having a prior that matches the true latent structure of the data, is crucial in constructing a correct latent representation that preserves the ability to generate meaningful samples.

## 5.2 EVALUATION OF EXPRESSIVENESS

To compare the behavior of the  $\mathcal{N}$ -VAE and  $\mathcal{S}$ -VAE on a data set that does not have a clear hyperspherical latent structure, we evaluate both models on a reconstruction task using dynamically binarized MNIST (Salakhutdinov and Murray, 2008). We analyze the ELBO, KL, negative reconstruction error, and marginal log-likelihood (LL) for both models on the test set. The LL is estimated using importance sampling with 500 sample points (Burda et al., 2016).

**Results** Results are shown in Table 1. We first note that in terms of negative reconstruction error the  $\mathcal{S}$ -VAE outperforms the  $\mathcal{N}$ -VAE in all dimensions. Since the  $\mathcal{S}$ -VAE uses a uniform prior, the KL divergence increases more strongly with dimensionality, which results in a higher ELBO. However in terms of log-likelihood (LL) the  $\mathcal{S}$ -VAE clearly has an edge in low dimensions ( $d = 2, 5, 10$ ) and performs comparable to the  $\mathcal{N}$ -VAE in  $d = 20$ . This empirically confirms the hypothesis of Subsection 2.2, showing the positive effect of having a uniform prior in low dimensions. In the absence of any origin pull, the data is able to cluster naturally, utilizing the entire latent space which can be observed in Figure 2. Note that in Figure 2(a) all mass is concentrated around the center, since the prior mean is zero. Conversely, in Figure 2(b) all available space is evenly covered due to the uniform prior,

resulting in more separable clusters in  $\mathcal{S}^2$  compared to  $\mathbb{R}^2$ . However, as dimensionality increases, the Gaussian distribution starts to approximate a hypersphere, while its posterior becomes more expressive than the vMF due to the higher number of variance parameters. Simultaneously, as described in Subsection 3.5, the surface area of the vMF starts to collapse limiting the available space.

In Figure 7 and 8 of Appendix G, we present randomly generated samples from the  $\mathcal{N}$ -VAE and the  $\mathcal{S}$ -VAE, respectively. Moreover, in Figure 9 of Appendix G, we show 2-dimensional manifolds for the two models. Interestingly, the manifold given by the  $\mathcal{S}$ -VAE indeed results in a latent space where digits occupy the entire space and there is a sense of continuity from left to right.

## 5.3 SEMI-SUPERVISED LEARNING

Having observed the  $\mathcal{S}$ -VAE’s ability to increase clusterability of data points in the latent space, we wish to further investigate this property using a semi-supervised classification task. For this purpose we re-implemented the M1 and M1+M2 models as described in (Kingma et al., 2014), and evaluate the classification accuracy of the  $\mathcal{S}$ -VAE and the  $\mathcal{N}$ -VAE on dynamically binarized MNIST. In the M1 model, a classifier utilizes the latent features obtained using a VAE as in experiment 5.2. The M1+M2 model is constructed by stacking the M2 model on top of M1, where M2 is the result of augmenting the VAE by introducing a partially observed variable  $\mathbf{y}$ , and combining the ELBO and classification objective. This concatenated model is trained end-to-end<sup>3</sup>.

This last model also allows for a combination of the two topologies due to the presence of two distinct latent variables,  $\mathbf{z}_1$  and  $\mathbf{z}_2$ . Since in the M2 latent space the class assignment is expressed by the variable  $\mathbf{y}$ , while  $\mathbf{z}_2$  only needs to capture the style, it naturally follows that the

<sup>3</sup>It is worth noting that in the original implementation by Kingma et al. (2014) the stacked model did not converge well using end-to-end training, and used the extracted features of the M1 model as inputs for the M2 model instead.

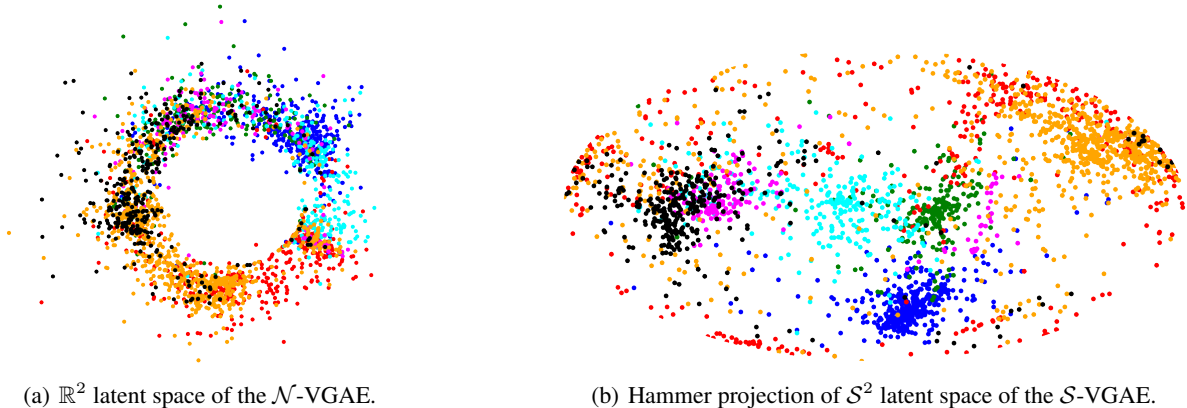


Figure 3: Latent space of unsupervised  $\mathcal{N}$ -VGAE and  $\mathcal{S}$ -VGAE models trained on Cora citation network. Colors denote documents classes which are not provided during training. (Best viewed in color)

$\mathcal{N}$ -VAE is more suited for this objective due to its higher number of variance parameters. Hence, besides comparing the  $\mathcal{S}$ -VAE against the  $\mathcal{N}$ -VAE, we additionally run experiments for the M1+M2 model by modeling  $\mathbf{z}_1, \mathbf{z}_2$  respectively with a vMF and normal distribution.

**Results** As can be seen in Table 2, for M1 the  $\mathcal{S}$ -VAE outperforms the  $\mathcal{N}$ -VAE in all dimensions up to  $d = 40$ . This result is amplified for a low number of observed labels. Note that for both models absolute performance drops as the dimensionality increases, since  $K$ -NN used as the classifier suffers from the curse of dimensionality. Besides reconfirming superiority of the  $\mathcal{S}$ -VAE in  $d < 20$ , its better performance than the  $\mathcal{N}$ -VAE for  $d = 20$  was unexpected. This indicates that although the log-likelihood might be comparable (see Table 1) for higher dimensions, the  $\mathcal{S}$ -VAE latent space better captures the cluster structure.

In the concatenated model M1+M2, we first observe in Table 3 that either the pure  $\mathcal{S}$ -VAE or the  $\mathcal{S}+\mathcal{N}$ -VAE model yields the best results, where the  $\mathcal{S}$ -VAE almost always outperforms the  $\mathcal{N}$ -VAE. Our hypothesis regarding the merit of a  $\mathcal{S}+\mathcal{N}$ -VAE model is further confirmed, as displayed by the stable, strong performance across all different dimensions. Furthermore, the clear edge in clusterability of the  $\mathcal{S}$ -VAE in low dimensional  $\mathbf{z}_1$  as already observed in Table 2, is again evident. As the dimensionality of  $\mathbf{z}_1, \mathbf{z}_2$  increases, the accuracy of the  $\mathcal{N}$ -VAE improves, reducing the performance gap with the  $\mathcal{S}$ -VAE. As previously noticed the  $\mathcal{S}$ -VAE performance drops when  $\dim \mathbf{z}_2 = 50$ , with the best result being obtained for  $\dim \mathbf{z}_1 = \dim \mathbf{z}_2 = 10$ . In fact, it is worth noting that for this setting the  $\mathcal{S}$ -VAE obtains comparable results to the original settings of (Kingma et al., 2014), while needing a considerably smaller latent space. Finally, the end-to-end trained  $\mathcal{S}+\mathcal{N}$ -VAE model is able to reach

a significantly higher classification accuracy than the original results reported by Kingma et al. (2014),  $96.7 \pm 1$ .

The M1+M2 model allows for conditional generation. Similarly to (Kingma et al., 2014), we set the latent variable  $\mathbf{z}_2$  to the value inferred from the test image by the inference network, and then varied the class label  $\mathbf{y}$ . In Figure 10 of Appendix H we notice that the model is able to disentangle the style from the class.

Table 3: Summary of results of semi-supervised model M1+M2 on MNIST.

Method		100		
$\dim \mathbf{z}_1$	$\dim \mathbf{z}_2$	$\mathcal{N}+\mathcal{N}$	$\mathcal{S}+\mathcal{S}$	$\mathcal{S}+\mathcal{N}$
5	5	90.0 $\pm$ .4	<b>94.0</b> $\pm$ .1	93.8 $\pm$ .1
	10	90.7 $\pm$ .3	94.1 $\pm$ .1	<b>94.8</b> $\pm$ .2
	50	90.7 $\pm$ .1	92.7 $\pm$ .2	<b>93.0</b> $\pm$ .1
10	5	90.7 $\pm$ .3	91.7 $\pm$ .5	<b>94.0</b> $\pm$ .4
	10	92.2 $\pm$ .1	<b>96.0</b> $\pm$ .2	<b>95.9</b> $\pm$ .3
	50	92.9 $\pm$ .4	95.1 $\pm$ .2	<b>95.7</b> $\pm$ .1
50	5	92.0 $\pm$ .2	91.7 $\pm$ .4	<b>95.8</b> $\pm$ .1
	10	93.0 $\pm$ .1	95.8 $\pm$ .1	<b>97.1</b> $\pm$ .1
	50	93.2 $\pm$ .2	94.2 $\pm$ .1	<b>97.4</b> $\pm$ .1

#### 5.4 LINK PREDICTION ON GRAPHS

In this experiment, we aim at demonstrating the ability of the  $\mathcal{S}$ -VAE to learn meaningful embeddings of nodes in a graph, showing the advantages of embedding objects in a non-Euclidean space. We test hyperspherical reparameterization on the recently introduced Variational Graph Auto-Encoder (VGAE) (Kipf and Welling, 2016), a VAE model for graph-structured data. We perform training on

a link prediction task on three popular citation network datasets (Sen et al., 2008): Cora, Citeseer and Pubmed.

Dataset statistics and further experimental details are summarized in Appendix F.3. The models are trained in an unsupervised fashion on a masked version of these datasets where some of the links have been removed. All node features are provided and efficacy is measured in terms of average precision (AP) and area under the ROC curve (AUC) on a test set of previously removed links. We use the same training, validation, and test splits as in Kipf and Welling (2016), i.e. we assign 5% of links for validation and 10% of links for testing.

Table 4: Results for link prediction in citation networks.

Method		$\mathcal{N}$ -VGAE	$\mathcal{S}$ -VGAE
Cora	AUC	92.7 $\pm$ .2	<b>94.1</b> $\pm$ .1
	AP	93.2 $\pm$ .4	<b>94.1</b> $\pm$ .3
Citeseer	AUC	90.3 $\pm$ .5	<b>94.7</b> $\pm$ .2
	AP	91.5 $\pm$ .5	<b>95.2</b> $\pm$ .2
Pubmed	AUC	<b>97.1</b> $\pm$ .0	96.0 $\pm$ .1
	AP	<b>97.1</b> $\pm$ .0	96.0 $\pm$ .1

**Results** In Table 4, we show that our model outperforms the  $\mathcal{N}$ -VGAE baseline on two out of the three datasets by a significant margin. The log-probability of a link is computed as the dot product of two embeddings. In a hypersphere, this can be interpreted as the cosine similarity between vectors. Indeed we find that the choice of a dot product scoring function for link prediction is problematic in combination with the normal distribution on the latent space. If embeddings are close to the zero-center, noise during training can have a large destabilizing effect on the angle information between two embeddings. In practice, the model finds a solution where embeddings are "pushed" away from the zero-center, as demonstrated in Figure 3(a). This counteracts the pull towards the center arising from the standard prior and can overall lead to poor modeling performance. By constraining the embeddings to the surface of a hypersphere, this effect is mitigated, and the model can find a good separation of the latent clusters, as shown in Figure 3(b).

On Pubmed, we observe that the  $\mathcal{S}$ -VAE converges to a lower score than the  $\mathcal{N}$ -VAE. The Pubmed dataset is significantly larger than Cora and Citeseer, and hence more complex. The  $\mathcal{N}$ -VAE has a larger number of variance parameters for the posterior distribution, which might have played an important role in better modeling the relationships between nodes. We further hypothesize that not all graphs are necessarily better embedded in a hyperspher-

ical space and that this depends on some fundamental topological properties of the graph. For instance, the already mentioned work from Nickel and Kiela (2017) shows that hyperbolic space is better suited for graphs with a hierarchical, tree-like structure. These considerations prefigure an interesting research direction that will be explored in future work.

## 6 CONCLUSION

With the  $\mathcal{S}$ -VAE we set an important first step in the exploration of hyperspherical latent representations for variational auto-encoders. Through various experiments, we have shown that  $\mathcal{S}$ -VAEs have a clear advantage over  $\mathcal{N}$ -VAEs for data residing on a known hyperspherical manifold, and are competitive or surpass  $\mathcal{N}$ -VAEs for data with a non-obvious hyperspherical latent representation in lower dimensions. Specifically, we demonstrated  $\mathcal{S}$ -VAEs improve separability in semi-supervised classification and that they are able to improve results on state-of-the-art link prediction models on citation graphs, by merely changing the prior and posterior distributions as a simple drop-in replacement.

We believe that the presented research paves the way for various promising areas of future work, such as exploring more flexible approximate posterior distributions through normalizing flows on the hypersphere, or hierarchical mixture models combining hyperspherical and hyperplanar space. Further research should be done in increasing the performance of  $\mathcal{S}$ -VAEs in higher dimensions; one possible solution of which could be to dynamically learn the radius of the latent hypersphere in a full Bayesian setting.

### Acknowledgements

We would like to thank Rianne van den Berg, Jonas Köhler, Pim de Haan, Taco Cohen, Marco Federici, and Max Welling for insightful discussions. T.K. is supported by SAP SE Berlin. J.M.T. was funded by the European Commission within the Marie Skłodowska-Curie Individual Fellowship (Grant No. 702666, Deep learning and Bayesian inference for medical imaging).

### References

- Aytekin, C., Ni, X., Cricri, F., and Aksu, E. (2018). Clustering and unsupervised anomaly detection with 12 normalized deep auto-encoder representations. *arXiv preprint arXiv:1802.00187*.
- Banerjee, A. and Basu, S. (2007). Topic models over text streams: A study of batch and online unsupervised learning. *ICDM*, pages 431–436.

- Banerjee, A., Dhillon, I., Ghosh, J., and Sra, S. (2003). Generative model-based clustering of directional data. *SIGKDD*, pages 19–28.
- Banerjee, A., Dhillon, I. S., Ghosh, J., and Sra, S. (2005). Clustering on the unit hypersphere using von mises-fisher distributions. *JMLP*, 6(Sep):1345–1382.
- Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *JMRL*, 3(Jan):993–1022.
- Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A., Jozefowicz, R., and Bengio, S. (2016). Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21.
- Burda, Y., Grosse, R., and Salakhutdinov, R. (2016). Importance weighted autoencoders. *ICLR*.
- Cohen, T. S., Geiger, M., Khler, J., and Welling, M. (2018). Spherical CNNs. *ICLR*.
- Dilokthanakul, N., Mediano, P. A. M., Garnelo, M., Lee, M. C. H., Salimbeni, H., Arulkumaran, K., and Shananhan, M. (2016). Deep unsupervised clustering with gaussian mixture variational autoencoders. *CoRR*, abs/1611.02648.
- Fisher, N. I., Lewis, T., and Embleton, B. J. (1987). *Statistical analysis of spherical data*. Cambridge university press.
- Gemici, M. C., Rezende, D., and Mohamed, S. (2016). Normalizing flows on riemannian manifolds. *NIPS Bayesian Deep Learning Workshop*.
- Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. *AISTATS*, pages 249–256.
- Gopal, S. and Yang, Y. (2014). Von mises-fisher clustering models. *ICML*, pages 154–162.
- Guan, H. and Smith, W. A. (2017). Structure-from-motion in spherical video using the von mises-fisher distribution. *IEEE Transactions on Image Processing*, 26(2):711–723.
- Guu, K., Hashimoto, T. B., Oren, Y., and Liang, P. (2018). Generating sentences by editing prototypes. *TACL*.
- Hasnat, M., Bohné, J., Milgram, J., Gentric, S., Chen, L., et al. (2017). von mises-fisher mixture model-based deep learning: Application to face verification. *arXiv preprint arXiv:1706.04264*.
- Kingma, D. and Welling, M. (2014). Efficient gradient-based inference through transformations between bayes nets and neural nets. *ICML*, pages 1782–1790.
- Kingma, D. P. and Ba, J. L. (2015). Adam: A method for stochastic optimization. In *ICLR*.
- Kingma, D. P., Mohamed, S., Rezende, D. J., and Welling, M. (2014). Semi-supervised learning with deep generative models. *NIPS*, pages 3581–3589.
- Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *CoRR*, abs/1312.6114.
- Kipf, T. N. and Welling, M. (2016). Variational Graph Auto-Encoders. *NIPS Bayesian Deep Learning Workshop*.
- Liu, C. and Zhu, J. (2018). Riemannian Stein Variational Gradient Descent for Bayesian Inference. *AAAI*.
- Liu, W., Zhang, Y.-M., Li, X., Yu, Z., Dai, B., Zhao, T., and Song, L. (2017). Deep hyperspherical learning. *NIPS*, pages 3953–3963.
- Mardia, K. V. (1975). Statistics of directional data. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 349–393.
- Naesseth, C., Ruiz, F., Linderman, S., and Blei, D. (2017). Reparameterization Gradients through Acceptance-Rejection Sampling Algorithms. *AISTATS*, pages 489–498.
- Nalisnick, E. and Smyth, P. (2017). Stick-breaking variational autoencoders. *ICLR*.
- Nickel, M. and Kiela, D. (2017). Poincaré embeddings for learning hierarchical representations. *NIPS*, pages 6341–6350.
- Reisinger, J., Waters, A., Silverthorn, B., and Mooney, R. J. (2010). Spherical topic models. *ICML*.
- Rezende, D. and Mohamed, S. (2015). Variational inference with normalizing flows. *ICML*, 37:1530–1538.
- Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, pages 1278–1286.
- Salakhutdinov, R. and Murray, I. (2008). On the quantitative analysis of deep belief networks. *ICML*, pages 872–879.
- Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., and Eliassi-Rad, T. (2008). Collective classification in network data. *AI magazine*, 29(3):93.
- Tomczak, J. and Welling, M. (2018). Vae with a vamp-prior. In *AISTATS*, pages 1214–1223.
- Ulrich, G. (1984). Computer generation of distributions on the m-sphere. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 33(2):158–163.
- Wilson, R. C., Hancock, E. R., Pekalska, E., and Duin, R. P. (2014). Spherical and hyperbolic embeddings of data. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2255–2269.

---

# Dissociation-Based Oblivious Bounds for Weighted Model Counting

---

**Li Chou**

The University of Texas at Dallas  
Richardson, TX 75080

**Wolfgang Gatterbauer**

Northeastern University  
Boston, MA 02115

**Vibhav Gogate**

The University of Texas at Dallas  
Richardson, TX 75080

## Abstract

We consider the weighted model counting task which includes important tasks in graphical models, such as computing the partition function and probability of evidence as special cases. We propose a novel partition-based bounding algorithm that exploits logical structure and gives rise to a set of inequalities from which upper (or lower) bounds can be derived efficiently. The bounds come with optimality guarantees under certain conditions and are oblivious in that they require only limited observations of the structure and parameters of the problem. We experimentally compare our bounds with the mini-bucket scheme (which is also oblivious) and show that our new bounds are often superior and never worse on a wide variety of benchmark networks.

## 1 INTRODUCTION

Logic and probability theory are formalisms employed for the task of automated reasoning. Logic facilitates deterministic representations and decisions, while probability theory accommodates situations where uncertainty arises. Propositional logic (Boolean satisfiability) is a prominent construct for performing deductive reasoning, particularly within a combinatorial setting. Extensive research efforts have resulted in state-of-the-art satisfiability solvers that have been successfully deployed in fields such as software/hardware model checking, planning and cybersecurity (Zhang and Malik, 2002). Graphical models (GM) have emerged as an effective scheme for modeling uncertainty. For example, Bayesian networks (Pearl, 1988) have been used in medical domains, while Markov networks are widely used in areas such as computer vision and natural language processing. However, in order to effectively

model problems in real-world domains, it is of great practical interest to solve the harder problem of developing models with the capacity to account for knowledge that is both deterministic and uncertain in a unified manner.

*Propositional model counting* is the generalization of the Boolean satisfiability problem. Extending the task of determining satisfiability, the objective is to count the number of distinct instances that result in satisfiability. This is also referred to as solution counting. Counting is a fundamental aspect to probabilistic computations (sum inference) and thus propositional model counting provides an intuitive connection between logic and uncertainty. In this paper, we address a further extension, namely the problem of *weighted model counting* (WMC). WMC allows for additional probabilistic interpretations of the variables in the model by associating a weight function either at the variable level or the clause level (Chavira and Darwiche, 2008; Gogate and Domingos, 2010; Sang et al., 2005).

It is well known that probabilistic inference in GM can be reduced to WMC (Chavira and Darwiche, 2008). The reduction has two main components: (1) encode the GM as a propositional knowledge base; and (2) leverage state-of-the-art propositional model counters to develop a WMC-based algorithm for solving the desired inference task. However, a major drawback of the aforementioned methods is that they are computationally intractable for most real-world problems. Therefore, developing fast, scalable approximate schemes is a subject of fundamental interest.

While there exists several approaches to propositional *approximate counting*, most of those are intrinsically stochastic (Ermon et al., 2013; Gogate and Dechter, 2007, 2011), and little attention has been given to deterministic methods that can bound estimates with correctness guarantees. In this paper we propose a *deterministic bounding scheme for WMC*. Our approach is partition-based (Dechter and Rish, 2003) and gives rise to a novel class of inequalities from which upper (or lower) bounds can be derived efficiently. In addition, the bounds are oblivious,

i.e. they require only limited observations of the structure and parameters of the problem, which yields fast methods.

Specifically, we extend the work of Gatterbauer and Suciú (Gatterbauer and Suciú, 2014, 2017), which is applicable to only monotone SAT formulas, to the task of WMC for arbitrary (*non-monotone*) formulas. Our method is related to the class of bounded complexity inference schemes such as mini-buckets (MB) (Dechter and Rish, 2003) and their extensions (Choi et al., 2007; Liu, 2014). MB relaxes the original problem by decomposing it into local sub-problems (by splitting/dissociating nodes) that are then solved exactly. The result is an approximate scheme that generates bounds for various inference tasks.

We make the following contributions. (1) We analyze the idea of dissociation based oblivious bounds (Gatterbauer and Suciú, 2014) using the framework of weighted model counting and extend it to the general non-monotone case; (2) we take advantage of logical structure and derive a novel set of inequalities for bounding methods that dissociate until the formula has a tree structure (namely the *i*-bound in MB is equal to 1); (3) we theoretically compare the idea of dissociation with MB and show that MB bounds are a special case of our bounds and can be quite inferior; and (4) we empirically demonstrate that dissociation based bounds are more accurate than MB on several synthetic and real-world datasets.

## 2 BACKGROUND

Let  $\mathbf{X}$ ,  $\mathbf{Y}$ , etc. be sets of propositional variables that take values (i.e., truth assignments) from the set  $\{\text{false}, \text{true}\}$  (or  $\{0, 1\}$ ). Given  $\mathbf{X} = \{X_1, \dots, X_n\}$ , let  $\Omega$  be the set of the  $2^n$  truth assignments to  $\mathbf{X}$ . Let  $\mathbf{x} = (x_1, \dots, x_n) \in \Omega$  be a truth assignment to all variables in  $\mathbf{X}$  s.t.  $X_i = x_i$ . We use the symbol ‘\*’ to denote the case when  $X_i$  can take either values, namely  $(0 \vee 1)$  or otherwise known as the *don’t care* condition. Let  $F$  be a propositional formula in conjunctive normal form (CNF) over  $\mathbf{X}$ , i.e.  $F$  is a conjunction of clauses, where a clause is a disjunction of literals, and each literal is defined as a variable  $X_i$  (positive literal, +) or its negation  $\bar{X}_i$  (negative literal, -). Let  $\mathbf{C}$  be the set of clauses of  $F$ . In this paper, we will focus on arbitrary (non-monotone) CNF.

**Definition 1.** (Monotonicity). A formula  $F$  is “monotone in variable  $X_i$ ” iff  $X_i$  appears in  $F$  as either positive or negative (but not both). A formula  $F$  is “monotone” iff it is monotone in all variables. Otherwise  $F$  is non-monotone.

### 2.1 WEIGHTED MODEL COUNTING

Given a propositional formula  $F$ , a satisfying assignment or *model* of  $F$  is a truth assignment to all variables in

$F$  such that  $F$  evaluates to *true* ( $\mathbf{x} \models F$ ). The problem of determining if there exists a satisfying assignment  $\mathbf{x}$  for  $F$  is called the *Boolean satisfiability problem* or SAT. *Propositional model counting* or #SAT is the task of computing the number of models of  $F$ . This is the canonical #P-complete problem that generalizes SAT.

Weighted model counting (WMC) (Chavira and Darwiche, 2008; Sang et al., 2005) extends model counting by associating the following probability distribution (weight function)  $\phi_i$  to each propositional variable  $X_i$ :

$$\phi_i(X_i) = \begin{cases} p_i & \text{if } X_i \text{ evaluates to } 1 \\ \bar{p}_i & \text{otherwise} \end{cases},$$

where  $p_i \in [0, 1]$  and  $\bar{p}_i \triangleq 1 - p_i$ .<sup>1</sup> The functions  $\phi_i$  yield a *weighted representation*  $\mathcal{F}$  of the CNF  $F$  and is called WCNF. Formally,  $\mathcal{F}$  is a triple  $\langle \mathbf{X}, \Phi, \mathbf{C} \rangle$ , where  $\mathbf{X}$  is a set of  $n$  Boolean variables in  $F$ ,  $\Phi$  is a set of weight functions  $\phi_i$  associated with each Boolean variable  $X_i \in \mathbf{X}$  and  $\mathbf{C}$  is a set of clauses of  $F$ .  $\mathcal{F}$  represents the following probability distribution

$$P_{\mathcal{F}}(\mathbf{x}) = \begin{cases} \frac{1}{Z_{\mathcal{F}}} \prod_{i=1}^n \phi_i(X_i = x_i) & \text{if } \mathbf{x} \models F \\ 0 & \text{otherwise} \end{cases},$$

where  $Z_{\mathcal{F}}$  is the *partition function*, also referred to as the *weighted model count* (WMC) of  $\mathcal{F}$ , and is given by

$$Z_{\mathcal{F}} = \sum_{(\mathbf{x} \in \Omega \wedge \mathbf{x} \models F)} \prod_{i=1}^n \phi_i(X_i = x_i).$$

When  $p_i = 1/2$  for all variables, the product  $2^n Z_{\mathcal{F}}$  equals the special case of (unweighted) model count of  $F$ .

### 2.2 GRAPHICAL MODELS

Graphical models (GM) provide a compact representation of joint probability distributions over a set of variables  $\mathbf{X}$ . For simplicity, we will focus on pairwise binary Markov networks since every GM can be converted to this form (cf. (Koller and Friedman, 2009)). Let  $\mathcal{I} \subseteq \mathcal{A}$  where  $\mathcal{A}$  denotes the set of all pairs  $(i, j)$  such that  $i < j$  and  $1 \leq i, j \leq n$ . In a pairwise graphical model, we associate a potential function  $\psi_{i,j}$  over each pair  $(i, j) \in \mathcal{I}$ . The probability distribution is given by

$$P(\mathbf{x}) = \frac{1}{Z} \prod_{(i,j) \in \mathcal{I}} \psi_{i,j}(x_i, x_j),$$

where  $Z$  is the normalization constant (partition function) and  $(x_i, x_j)$  is the projection of  $\mathbf{x}$  on  $\{X_i, X_j\}$ .

<sup>1</sup>WMC is typically defined by attaching weights to literals, and the corresponding potential function over each variable is constructed by exponentiating the weights. We consider an equivalent representation in which the potential function is normalized to yield a probability distribution.

Table 1: Clauses for w2CNF Encoding of a GM

$(\bar{X}_i \vee Y_{i,j,1})$	$(\bar{X}_j \vee Y_{i,j,1})$
$(\bar{X}_i \vee Y_{i,j,2})$	$(X_j \vee Y_{i,j,2})$
$(X_i \vee Y_{i,j,3})$	$(\bar{X}_j \vee Y_{i,j,3})$
$(X_i \vee Y_{i,j,4})$	$(X_j \vee Y_{i,j,4})$

### 2.3 WCNF ENCODING OF A GM

We describe here a possible translation from GM to WCNF. For more details see (Chavira and Darwiche, 2008; Gogate and Domingos, 2010, 2016). Since we focus on pairwise binary GMs, we can convert them to WCNFs in which each clause has at most two literals. We will refer to such WCNFs as w2CNF.

Given a GM, we can construct an equivalent w2CNF as follows. We start with a w2CNF  $\mathcal{F}$  defined over the variables  $\mathbf{X}$  of the GM such that the set of clauses  $\mathbf{C}$  of  $\mathcal{F}$  is empty and  $p_{X_i} = 0.5$  for each variable  $X_i \in \mathbf{X}$ . Then, for each pairwise binary potential  $\psi_{i,j}$  in the GM such that  $\psi_{ij} : X_i = 0, X_j = 0 \rightarrow w_{i,j,1}$ ,  $\psi_{ij} : X_i = 0, X_j = 1 \rightarrow w_{i,j,2}$ ,  $\psi_{ij} : X_i = 1, X_j = 0 \rightarrow w_{i,j,3}$ ,  $\psi_{ij} : X_i = 1, X_j = 1 \rightarrow w_{i,j,4}$ , we add a variable for each weight to  $\mathcal{F}$ . We will denote the variables associated with  $w_{i,j,1}$ ,  $w_{i,j,2}$ ,  $w_{i,j,3}$  and  $w_{i,j,4}$  by  $Y_{i,j,1}$ ,  $Y_{i,j,2}$ ,  $Y_{i,j,3}$  and  $Y_{i,j,4}$  respectively. Utilizing these weight variables, we add the the clauses given in Table 1 to  $\mathbf{C}$  for  $k = 1, \dots, 4$ . We also add the following probability distribution for each variable  $Y_{i,j,k}$

$$\phi(y_{i,j,k}) = \begin{cases} \frac{w_{i,j,k} - 1}{w_{i,j,k}} & \text{if } y_{i,j,k} \text{ is false or } 0 \\ \frac{1}{w_{i,j,k}} & \text{otherwise} \end{cases}$$

Note that when  $w_{i,j,k} < 1$ ,  $\phi(y_{i,j,k})$  will be negative. To avoid this condition, we can easily rescale the potentials of the GM by multiplying them with an appropriate constant. Also, zero weights can be handled by adding the corresponding negated assignment as a clause to  $\mathbf{C}$ . For example, if  $w_{i,j,1} = 0$ , we add the clause  $X_i \vee X_j$  to  $\mathbf{C}$ . Using previous work (Chavira and Darwiche, 2008; Gogate and Domingos, 2010), it is straight-forward to show that:

**Proposition 2.** w2CNF output by Encoding 1 represents the same probability distribution over  $\mathbf{X}$  as the input GM.

### 2.4 MINI-BUCKET ELIMINATION

We can utilize inference algorithms such as bucket or variable elimination (Dechter, 1996; Zhang and Poole, 1994) to compute the weighted model count of a w2CNF. However, since the complexity of using such algorithms is in general exponential in the treewidth, a more practical approach is to approximate the task by introducing

relaxations techniques that control model complexity (i.e., the induced width given a fixed elimination order). Mini-bucket (MB) (Dechter and Rish, 2003) is one such approximate scheme that builds on bucket elimination (BE) for generating upper (or lower) bounds on the partition function or weighted model count. We will use the following running example to illustrate BE and MB for WMC.

**Example 3.** Consider the w2CNF  $\mathcal{F}$  such that  $\mathbf{X} = \{X_1, Y_2, Y_3\}$ ,  $\mathbf{C} = \{(X_1 \vee Y_2), (X_1 \vee Y_3)\}$  and  $\Phi = \{\phi_1, \phi_2, \phi_3\}$ . For simplicity we denote  $\phi_1$  for  $\phi_1(X_1)$ , etc. We can convert the clauses and potentials of  $\mathcal{F}$  to the following two potentials yielding a more convenient form for BE.

$X_1$	$Y_2$	$\psi_{12}(X_1, Y_2)$	$X_1$	$Y_3$	$\psi_{13}(X_1, Y_3)$
0	0	0	0	0	0
0	1	$\bar{p}_1 p_2$	0	1	$p_3$
1	0	$p_1 \bar{p}_2$	1	0	$\bar{p}_3$
1	1	$p_1 p_2$	1	1	$p_3$

Without loss of generality, we assume the elimination ordering as  $[X_1, Y_2, Y_3]$  (although it is clearly not optimal, it will help us illustrate the main ideas). BE begins by creating  $|\mathbf{X}|$  number of buckets and groups the functions by placing each function involving some variable  $X_i$  (or  $Y_i$  in our example) in a bucket  $\mathbf{B}_{X_i}$  according to the position of  $X_i$  in the ordering. The resulting computation is  $Z_{\mathcal{F}}^{\text{BE}} = \sum_{Y_3} \sum_{Y_2} \sum_{X_1} \psi_{12} \psi_{13}$  where  $\mathbf{B}_{X_1} = \{\psi_{12}, \psi_{13}\}$  is first processed by taking the product of the two potentials and summing out variable  $X_1$ . The resulting new potential  $\psi'_{23}$  is placed in bucket  $\mathbf{B}_{Y_2}$  in which variable  $Y_2$  is summed out. Summing out  $Y_3$  from the subsequent function  $\psi'_3$  yields  $Z_{\mathcal{F}}^{\text{BE}}$ .

MB follows similarly. However, MB partitions each bucket into two or more so called mini-buckets according to an input parameter called the  $i$ -bound, which defines the maximum number ( $i$ -bound + 1) of variables in each mini-bucket. The mini-buckets are then processed independently. To obtain an upper bound, the sum-product operation is performed on one of the mini-buckets and the max-product for the remaining (min-product for lower bound). Using  $i$ -bound = 1,  $\mathbf{B}_{X_1}$  is split into two mini-buckets  $\mathbf{B}'_{X_1} = \{\psi_{12}\}$  and  $\mathbf{B}''_{X_1} = \{\psi_{13}\}$ . One possible resulting computation is

$$\underbrace{\sum_{Y_3 Y_2} \left( \sum_{X_1} \psi_{12} \right) \left( \min_{X_1} \psi_{13} \right)}_{Z_{\mathcal{F}}^{\text{MB}(L)}} \leq \sum_{Y_3} \sum_{Y_2} \sum_{X_1} \psi_{12} \psi_{13} \leq \underbrace{\sum_{Y_3 Y_2} \left( \sum_{X_1} \psi_{12} \right) \left( \max_{X_1} \psi_{13} \right)}_{Z_{\mathcal{F}}^{\text{MB}(U)}}$$

where the MB upper bound on the partition function,



$Z_{\mathcal{F}}^{\text{MB}(U)}$ , is computed by maxing out  $X_1$  from  $\psi_{13}$  independently from summing out  $X_1$  from  $\psi_{12}$ . Summing out  $Y_2$  and  $Y_3$  from the resulting two new potentials,  $\psi'_2, \psi'_3$ , and taking their product gives the upper bound. The lower bound,  $Z_{\mathcal{F}}^{\text{MB}(L)}$ , is computed similarly using min instead of max.

MB is a fast and simple algorithm for computing upper (or lower) bounds. The resulting complexity of inference is exponential in the  $i$ -bound. Lower  $i$ -bound values translates to simpler models and provides the trade-off between complexity and accuracy.

Next, we present the idea of dissociation based oblivious bounds for the case of monotone W2CNF and extend it to the non-monotone case by exploiting logical structure in Section 4. As mentioned earlier, in this paper, we focus on the case where variables are dissociated until the resulting formula is a tree. In other words, our scheme is comparable to the case when the  $i$ -bound in MB equals 1.

### 3 DISSOCIATION

Our task is to compute the WMC  $Z_{\mathcal{F}}$  of a given WCNF  $\mathcal{F}$ . Since the problem is computationally intractable in general (e.g., high treewidth), approximate methods are required. In this paper we use a *bounded inference* approach, where we approximate the original  $\mathcal{F}$  with  $\mathcal{F}'$  from which the upper (or lower) bounds on  $Z_{\mathcal{F}}$  can be computed efficiently. We build upon (Gatterbauer and Suciu, 2014, 2017) which presents a bounding scheme called *dissociation* that can be applied to WMC. The derived bounds are *oblivious* to the set of weight functions  $\phi_i$ , i.e. they can be calculated by only observing a limited subset of clauses. However, these bounds only apply to monotone formulas, whereas we are interested in extending the underlying ideas to more general non-monotone formulas (Section 4). Here, we first give a general intuition of prior results followed by the formal definition and then present optimal oblivious bounds for monotone formulas.

At a high level, dissociation is the process of replacing an existing variable  $X_i$  in  $\mathcal{F}$  with new variables  $X_{i,1}, \dots, X_{i,d}$  and assigning them new probability distributions. The technique is closely related to *variable or node splitting* (Choi et al., 2007) in which the new variables are referred to as clones. The partitioning of mini-buckets can also be classified under the general notion of variable splitting.

By creating new variables, we are implicitly ignoring (or relaxing) a set of equality constraints (Choi and Darwiche, 2009). However, we can recover the set by defining and incorporating the function  $\varphi(X_{i,1} = x_{i,1}, \dots, X_{i,n} = x_{i,d})$  which evaluates to 1 iff  $x_{i,1} = \dots = x_{i,d}$ , and 0 other-

wise, for the  $d$  copies  $X_{i,j}, j \in [d]$  of variable  $X_i$ , and  $x_{i,j}$  being the corresponding truth assignment. We can also incorporate equivalence clauses for each new pair of variables into a formula with the new clauses. For example, consider the formula  $F = (X_1 \vee Y_1)(X_1 \vee Y_2)$ . We can create the equivalent formula  $F' = (X_{1,1} \vee Y_1)(X_{1,2} \vee Y_2)(X_{1,1} \Leftrightarrow X_{1,2})$  using copies of  $X_1$  for the unweighted model counting case. We see two issues arising. First, for general 2-CNF formulas, we will require  $d - 1$  equality constraints (equivalence ( $\Leftrightarrow$ ) formulas). Second, it is not immediately clear on how to integrate the weight functions so that weighted model counts can be computed using this scheme.

Dissociation expands on the notion of variable duplication and provides an *algebraic framework* to analyze and approximate the aforementioned set of equality constraints. The result is a novel class of inequalities to construct upper (or lower) bounds on the WMC. We first give the formal definition of dissociation for W2CNF.

**Definition 4.** (Dissociation). *Let  $\mathcal{F} = \langle \mathbf{X}, \Phi, \mathbf{C} \rangle$ . Select a variable  $X_i \in \mathbf{X}$  and let  $C(X_i) \subseteq \mathbf{C}$  be the subset of all clauses that involve variable  $X_i$ . We say  $\mathcal{F}' = \langle \mathbf{X}', \Phi', \mathbf{C}' \rangle$  is a dissociation of  $\mathcal{F}$  on  $X_i$  iff*

- $\mathbf{X}' = \mathbf{X} \setminus X_i \cup X_{i,1} \cup \dots \cup X_{i,d}$  with  $d \leq |C(X_i)|$ ,
- $\Phi' = \Phi \setminus \phi_i \cup \phi_{i,1} \cup \dots \cup \phi_{i,d}$ , and
- $\mathbf{C}'[\theta_{X_i}(\mathbf{X}')] = \mathbf{C}[\mathbf{X}]$  with  $\theta_{X_i}$  being the substitution  $\theta_{X_i}[\{(X_{i,j}/X_i), j \in [d]\}]$ .

We say a dissociation is full if  $d = |C(X_i)|$ .

**Example 5.** (Dissociation). *Consider  $\mathcal{F}$  from example 3. Dissociating  $X_1$  results in adding two new variables,  $\mathbf{X}' = \mathbf{X} \setminus X_1 \cup X_{1,1} \cup X_{1,2}$ , and two new associated weight functions  $\Phi' = \Phi \setminus \phi_1 \cup \phi_{1,1} \cup \phi_{1,2}$ . Applying the substitution  $\theta_{X_1}[(X_{1,1}/X_1), (X_{1,2}/X_1)]$  on  $C(X_i)$  results in  $\mathbf{C}' = \mathbf{C} \setminus C(X_i) \cup (X_{1,1} \vee Y_2) \cup (X_{1,2} \vee Y_3)$ .*

Once we have defined the new weight functions (for dissociated variables), the question we are interested in is how to parameterize the new functions in order to obtain guaranteed upper (or lower) bounds. In particular, we are interested in *oblivious bounds*, i.e. when the *new probabilities are chosen independently of the probabilities of all other variables*. We achieve that by considering all possible valuations (or truth assignments) of the non-dissociated variables, The assignments give rise to a set of inequalities which are then evaluated to develop necessary and sufficient conditions for upper (or lower) bounds. We next illustrate with an example.

**Example 6.** (Oblivious bounds). *Consider the two sets of clauses,  $\{(X_1 \vee Y_2), (X_1 \vee Y_3)\}$  and  $\{(X_{1,1} \vee Y_2), (X_{1,2} \vee Y_3)\}$  from examples 3 and 5. We analyze the  $2^2 = 4$  possible truth assignments to the non-dissociated variables*

Table 2: Dissociation valuation analysis (example 6).

$Y_2$	$Y_3$	$X_1$	$X_{1;1}$	$X_{1;2}$	$\phi_1$	$\phi_{1;1}, \phi_{1;2}$
0	0	1	1	1	$p_1$	$p_{1;1}p_{1;2}$
0	1	1	1	*	$p_1$	$p_{1;1}$
1	0	1	*	1	$p_1$	$p_{1;2}$
1	1	*	*	*	1	1

$Y_2$  and  $Y_3$ . Table 2 shows each possible valuation of  $Y_2$  and  $Y_3$  and the corresponding assignments to  $X_1, X_{1;1}$  and  $X_{1;2}$  required to satisfy the clauses. We also show the weights (probabilities) of the original (column  $\phi_1$ ) and dissociated formulas (column  $\phi_{1;1}\phi_{1;2}$ ).

As example, consider the assignment,  $Y_2 = 0 \wedge Y_3 = 1$ : The assignment  $X_1 = 1$  is required to satisfy  $\mathcal{F}$ , resulting in the term  $p_1\bar{p}_2p_3$ . The assignments ( $X_{1;1} = 1 \wedge X_{1;2} = 0$ ) or ( $X_{1;1} = 1 \wedge X_{1;2} = 1$ ) are required to satisfy  $\mathcal{F}'$ , i.e.  $X_{1;2}$  can take any assignment (\*), resulting in the term  $p_{1;1}\bar{p}_2p_3$ . Utilizing the two terms, simplifying by removing the common terms ( $\bar{p}_2p_3$ ) and assuming that we are interested in computing lower bounds, we create the inequality  $p_1 \geq p_{1;1}$ . Repeating the same analysis for the three remaining cases results in the inequalities  $p_1 \geq p_{1;1}p_{1;2}$  and  $p_1 \geq p_{1;2}$ . The last case  $1 \geq 1$  is trivially satisfied. Combining the resulting inequalities, and doing a similar analysis for computing upper bounds (where we replace  $\geq$  by  $\leq$ ) gives rise to the following conditions for oblivious (U)pper and (L)ower bounds:

- $U: (p_1 \leq p_{1;1}p_{1;2}) \wedge (p_1 \leq p_{1;1}) \wedge (p_1 \leq p_{1;2})$ .
- $L: (p_1 \geq p_{1;1}p_{1;2}) \wedge (p_1 \geq p_{1;1}) \wedge (p_1 \geq p_{1;2})$ .

Notice the valuation process creates  $2^{|C(X_i)|}$  inequalities, one for each truth assignment. However, we can simplify the conditions by removing subsumed inequalities.

**Definition 7.** (Subsumed inequality). We say an inequality  $I_i$  subsumes inequality  $I_j$  ( $i \neq j$ ) iff  $I_i \Rightarrow I_j$ , i.e. satisfying  $I_i$  also satisfies  $I_j$ .

**Example 8.** Consider the upper and lower bound conditions in example 6. For the upper bound, clearly  $p_1 \leq p_{1;1}p_{1;2}$  subsumes the remaining inequalities since  $\forall p_1, p_{1;1}, p_{1;2} \in [0, 1] : (p_1 \leq p_{1;1}p_{1;2}) \Rightarrow (p_1 \leq p_{1;1}) \wedge (p_1 \leq p_{1;2})$ . For the lower bound, clearly  $(p_1 \geq p_{1;1}) \wedge (p_1 \geq p_{1;2})$  subsumes the remaining inequality since  $\forall p_1, p_{1;1}, p_{1;2} \in [0, 1] : ((p_1 \geq p_{1;1}) \wedge (p_1 \geq p_{1;2})) \Rightarrow (p_1 \geq p_{1;1}p_{1;2})$ . Therefore, we can reduce the required conditions for the oblivious bounds to:

- $U: p_1 \leq p_{1;1}p_{1;2}$ .
- $L: (p_1 \geq p_{1;1}) \wedge (p_1 \geq p_{1;2})$ .

Following the preceding analysis, we can now state the conditions for oblivious bounds for monotone w2CNF.

**Theorem 9.** (Gatterbauer and Suciu, 2014) (Oblivious bounds for monotone w2CNF). Let  $\mathcal{F}$  be a monotone w2CNF. Let  $\mathcal{F}'$  be the result of applying a series of dissociation steps on  $\mathcal{F}$ . For every set of weight functions defined for a dissociate variable, namely  $X_{i;1}, \dots, X_{i;d}$  and  $\{\phi_{i;1}, \dots, \phi_{i;d}\}$  with  $d > 1$ , we have the following oblivious bounds:

- $U: \prod_{j=1}^d p_{i;j} \geq p_i$ .
- $L: \forall j : p_{i;j} \leq p_i$ .

Optimal oblivious bounds are defined as those that are not dominated, i.e. they cannot be improved without knowledge of the probabilities of all other variables. They are obtained by replacing inequality with equality. Notice that optimal oblivious lower bounds are uniquely defined,  $\forall j : p_{i;j} = p_i$ , whereas there are infinitely many optimal oblivious upper bounds, e.g. symmetric ones:  $\forall j : p_{i;j} = \sqrt[d]{p_i}$ , and finding the best one requires access to all other probabilities (den Heuvel et al., 2018).

Note that optimal oblivious bounds are different from augmented mini-buckets (AMB) (Liu, 2014). For example, in AMB for computing upper bounds, the potential over each dissociated variable is initialized to  $\phi_{i;j}(X_{i;j} = 1) = \phi_i(X_i = 1)^{1/d}$  and  $\phi_{i;j}(X_{i;j} = 0) = \phi_i(X_i = 0)^{1/d}$  where we have  $d$  dissociations. A better initialization would be  $\phi_{i;j}(X_{i;j} = 1) = \phi_i(X_i = 1)^{1/d}$ , and  $\phi_{i;j}(X_{i;j} = 0) = 1 - \phi_i(X_i = 1)^{1/d}$ .

### 3.1 COMPARISON WITH MINI-BUCKET

We use example 3 to analyze the base case bounds for monotone dissociation ( $X_1$  to  $X_{1;1}$  and  $X_{1;2}$ ) and compare it with MB ( $i$ -bound = 1).

**Lower bound.** Dissociation results in the partition function  $Z_{\mathcal{F}'}^{\text{DIS}(U)} = p_2p_3 + p_{1;1}\bar{p}_2p_3 + p_{1;2}p_2\bar{p}_3 + p_{1;1}p_{1;2}\bar{p}_2\bar{p}_3$ . The two possible partition functions according to MB are (1)  $\sum_{X_1} \psi_1 \min_{X_1} \psi_2 \Rightarrow Z_{\mathcal{F}}^{\text{MB}(L1)} = p_2p_3 + p_{1;1}\bar{p}_2p_3$ ; (2)  $\min_{X_1} \psi_1 \sum_{X_1} \psi_2 \Rightarrow Z_{\mathcal{F}}^{\text{MB}(L2)} = p_2 \min(\bar{p}_1, p_1)(1 + p_3)$ . Clearly,  $Z_{\mathcal{F}'}^{\text{DIS}(L)} \geq Z_{\mathcal{F}}^{\text{MB}(L)} \forall p_1, p_{1;1}, p_{1;2}, p_2, p_3 \in [0, 1]$ .

**Upper bound.** Notice there exist an infinite number of settings to  $p_{1;1}$  and  $p_{1;2}$  that satisfy  $p_{1;1}p_{1;2} = p_1$  under dissociation. We analyze two possible cases. (1)  $(p_{1;1} = p_1) \wedge (p_{1;2} = 1) \Rightarrow Z_{\mathcal{F}'}^{\text{DIS}(U1)} = p_{1;1} + \bar{p}_{1;1}p_2$ ; (2)  $(p_{1;2} = p_1) \wedge (p_{1;1} = 1) \Rightarrow Z_{\mathcal{F}'}^{\text{DIS}(U2)} = p_{1;2} + \bar{p}_{1;2}p_3$ . The two possible partition functions according to MB are (1)  $\sum_{X_1} \psi_1 \max_{X_1} \psi_2 \Rightarrow Z_{\mathcal{F}}^{\text{MB}(U1)} = p_1 + \bar{p}_1p_1$ ; (2)  $\max_{X_1} \psi_1 \sum_{X_1} \psi_2 \Rightarrow Z_{\mathcal{F}}^{\text{MB}(U2)} = (1 + p_3)(p_2 \max(\bar{p}_1, p_1) + p_1\bar{p}_1)$ . We first observe the bounds are equivalent between dissociation and MB in setting

(1) and also for (2) if the functions are unweighted (e.g.,  $\forall i p_i = 1/2$ ). However, note there exist more degrees of freedom (solutions) for dissociation, and this example simply demonstrates one such setting for which we observe equivalency under certain conditions.

## 4 DISSOCIATION FOR NON-MONOTONE FORMULAS

In this section, we extend dissociation bounds from the monotone case to arbitrary non-monotone w2CNFs. Unlike monotone w2CNFs, we can apply logical inference techniques such as resolution and unit propagation to reduce non-monotone w2CNFs which in turn may improve our dissociation-based bounds. Moreover, logical propagation can be applied as a pre-processing step before dissociating a variable  $X_i$ .

### 4.1 PREPROCESSING

We say that a w2CNF  $\mathcal{F}$  is *minimal* if the following steps are applied to its set of clauses  $\mathbf{C}$  until convergence.

1. **(Binary) Resolution:** If  $\mathbf{C}$  contains two clauses of the form  $L_i \vee L_j$  and  $\bar{L}_i \vee L_k$ , where  $L_i, L_j$  and  $L_k$  are literals of variables  $X_i, X_j$  and  $X_k$  respectively, we add the clause  $L_j \vee L_k$  to  $\mathbf{C}$ .
2. **Unit Resolution:** If  $\mathbf{C}$  contains two clauses of the form  $L_i \vee L_j$  and  $\bar{L}_i \vee L_j$ , where  $L_i$  and  $L_j$  are literals of variables  $X_i$  and  $X_j$  respectively, we add the unit clause  $L_j$  to  $\mathbf{C}$ .
3. **Clause Deletion and Reduction:** If  $\mathbf{C}$  contains a unit clause  $L_i$  where  $L_i$  is a literal of  $X_i$  then we delete all clauses of the form  $L_i \vee L_j$  and remove  $\bar{L}_i$  from all clauses that mention  $\bar{L}_i$ . If  $\mathbf{C}$  contains both unit clauses  $L_i$  and  $\bar{L}_i$ ,  $\mathbf{C}$  is inconsistent and we return a lower/upper bound of 0.<sup>2</sup>

**Example 10 (Minimal formula).** Consider  $\mathbf{C} = \{(X_1 \vee \bar{X}_2), (X_1 \vee X_2), (\bar{X}_2 \vee Y_4), (X_1 \vee X_3), (X_3 \vee X_5)\}$ .  $\mathbf{C}$  is not minimal and we can make it minimal using the aforementioned steps. After applying Unit Resolution on the first two clauses, we get  $\mathbf{C} = \{(X_1), (X_1 \vee \bar{X}_2), (X_1 \vee X_2), (\bar{X}_2 \vee Y_4), (X_1 \vee X_3), (X_3 \vee X_5)\}$ . After applying Clause deletion and Reduction, we get  $\mathbf{C} = \{(X_1), (\bar{X}_2 \vee Y_4), (X_3 \vee X_5)\}$ , which is minimal.

### 4.2 TYPES OF NON-MONOTONE FORMULAS

In the sequel, we assume that the input w2CNF  $\mathcal{F}$  to our algorithm is minimal. To formulate oblivious bounds for non-monotone w2CNF, we first establish a *canonical*

<sup>2</sup>Note that our scheme will return an upper bound of 0 only when  $\mathbf{C}$  is inconsistent.

*representation* that helps us take advantage of symmetry and reduces the number of cases (inequalities) we need to consider for our proposed oblivious bounds. Specifically, given a candidate *dissociation variable*  $X_i$ , we convert the set of clauses  $\mathbf{C}$  into a canonical representation:

**Definition 11 (Canonical representation).** We say that  $\mathcal{F}$  is *canonical w.r.t. a variable  $X_i$*  if  $\mathcal{F}$  is minimal and all clauses in  $C(X_i)$  satisfy the following two properties:

1. If a variable  $Y_j$  appears only once in  $C(X_i)$  then it only appears positively, i.e. it appears in clauses of the form  $X_i \vee Y_j$  or  $\bar{X}_i \vee Y_j$  (but not of the form  $X_i \vee \bar{Y}_j$  or  $\bar{X}_i \vee \bar{Y}_j$ ).
2. If a variable  $Y_j$  appears twice in  $C(X_i)$ , then it appears in the following two clauses  $X_i \vee Y_j$  and  $\bar{X}_i \vee \bar{Y}_j$  (but not in the clauses  $\bar{X}_i \vee Y_j$  and  $X_i \vee \bar{Y}_j$ ).

Note that since  $\mathcal{F}$  is minimal,  $Y_j$  cannot appear more than twice in  $C(X_i)$ , nor twice with the same sign. If  $C(X_i)$  is not in canonical form, we can easily make it canonical by using the following procedure:

- If  $Y_j$  violates either condition (1) or (2) in definition 11, then replace  $Y_j$  by a new variable  $Y_k$  in all clauses of  $\mathcal{F}$  (where  $Y_j$  appears) such that  $Y_k = \bar{Y}_j$ , and set  $\phi(Y_k) = \phi(\bar{Y}_j)$  and  $\phi(\bar{Y}_k) = \phi(Y_j)$ .

**Example 12 (Canonical representation).** Consider  $\mathbf{C} = \{(X_1 \vee \bar{Y}_2), (\bar{X}_1 \vee Y_2), (X_1 \vee \bar{Y}_3)\}$ .  $\mathbf{C}$  is not in canonical form w.r.t.  $X_1$  because  $Y_2$  and  $Y_3$  violate the second and first property respectively in definition 11. To convert it to canonical form, set  $Y_4 = \bar{Y}_2, Y_5 = \bar{Y}_3, \phi(Y_4) = \phi(\bar{Y}_2), \phi(\bar{Y}_4) = \phi(Y_2), \phi(Y_5) = \phi(\bar{Y}_3)$  and  $\phi(\bar{Y}_5) = \phi(Y_3)$ . Thus, the canonical representation of  $\mathbf{C}$  is the set  $\{(X_1 \vee Y_4), (\bar{X}_1 \vee \bar{Y}_4), (X_1 \vee Y_5)\}$ .

We call variables  $Y_j$  which appear only once in  $C(X_i)$  *single-occurrence neighbors* of  $X_i$  and those which appear twice *two-occurrence neighbors*.

### 4.3 CHARACTERIZING OBLIVIOUS BOUNDS

We now derive oblivious bounds based on whether  $C(X_i)$  has two-occurrence neighbors or not. In the following, let  $\mathcal{F}$  denote a w2CNF that is canonical w.r.t.  $X_i$  and let  $\mathcal{F}'$  be the result of applying a series of dissociation steps on  $\mathcal{F}$ . Let  $Y_j$  be a single-occurrence neighbor of  $X_i$ . Let  $S^+$  and  $S^-$  denote the set of indices of the dissociated variables that appear in clauses  $(X_i \vee Y_j)$  and  $(\bar{X}_i \vee Y_j)$  respectively in  $C(X_i)$ . Let  $Y_k$  be a two-occurrence neighbor of  $X_i$ . Let  $T^+$  and  $T^-$  denote the set of indices of the dissociated variables in clauses  $X_i \vee Y_k$  and  $\bar{X}_i \vee \bar{Y}_k$  respectively in  $C(X_i)$ . (We use  $S$  and  $T$  to refer to “single-occurrence” and “two-occurrence” variables, respectively.)

**Example 13 (Indices).** Consider  $\mathbf{C} = \{(X_1 \vee Y_5), (X_1 \vee Y_8), (\bar{X}_1 \vee Y_6), (X_1 \vee Y_7), (\bar{X}_1 \vee \bar{Y}_7), (X_1 \vee$

$Y_9), (\overline{X_1 \vee Y_9})\}$ . After applying dissociation on  $X_1$ , we get  $C(X'_1) = \{(X_{1;1} \vee Y_5), (X_{1;2} \vee Y_8), (\overline{X_{1;3}} \vee Y_6), (X_{1;4} \vee Y_7), (\overline{X_{1;5}} \vee Y_7), (X_{1;6} \vee Y_9), (\overline{X_{1;7}} \vee Y_9)\}$ . Then  $S^+ = \{1, 2\}$ ,  $S^- = \{3\}$ ,  $T^+ = \{4, 6\}$ , and  $T^- = \{5, 7\}$ .

We next analyze the two possible non-monotone cases in Theorems 14 and 16. The proofs are presented in an extended version of the paper.

The first case is when  $C(X_i)$  has only single-occurrence neighbors (but no two-occurrence neighbors). This generalizes the monotone case, in which only one type of single-occurrence variables are present. In particular, in the monotone case either clauses of the form  $(X_i \vee Y_j)$  or  $(\overline{X_i} \vee Y_j)$  are present but not both while in the non-monotone case both clauses can be present in  $C(X_i)$ . Note that bounds given in Theorem 9 are a special case of the bounds in Theorem 14 presented next.

**Theorem 14.** (Oblivious bounds for w2CNFs having only single-occurrence neighbors w.r.t.  $X_i$ ). For a given variable  $X_i$ , if  $\mathcal{F}$  contains only single-occurrence neighbors but no two-occurrence neighbors then we have the following oblivious bounds for  $X_i$ :

- $U$ :  $\left( \prod_{j \in S^+} p_{i;j} \geq p_i \right) \wedge \left( \prod_{j \in S^-} \overline{p_{i;j}} \geq \overline{p_i} \right)$
- $L$ : Either of following two conditions hold:
  1.  $(\forall j \in S^+ : p_{i;j} \leq p_i) \wedge (\forall j \in S^- : \overline{p_{i;j}} = 0)$
  2.  $(\forall j \in S^- : \overline{p_{i;j}} \leq \overline{p_i}) \wedge (\forall j \in S^+ : p_{i;j} = 0)$

Optimal oblivious bounds are obtained by replacing inequality with equality in the bound conditions.

**Example 15.** Consider  $C(X'_1) = \{(X_{1;1} \vee Y_2), (\overline{X_{1;2}} \vee Y_3), (X_{1;3} \vee Y_4), (\overline{X_{1;4}} \vee Y_5)\}$ . Theorem 14 gives the conditions for upper and lower oblivious bounds as:

- $U$ :  $(p_{1;1} p_{1;3} \geq p_1) \wedge (\overline{p_{1;2}} \overline{p_{1;4}} \geq \overline{p_1})$ .
- $L$ : Either of following two conditions hold:
  1.  $(p_{1;1} \leq p_1) \wedge (p_{1;3} \leq p_1) \wedge (\overline{p_{1;2}} = \overline{p_{1;4}} = 0)$
  2.  $(\overline{p_{1;2}} \leq \overline{p_1}) \wedge (\overline{p_{1;4}} \leq \overline{p_1}) \wedge (p_{1;1} = p_{1;3} = 0)$

Our second non-monotone case is when  $\mathcal{F}$  has at least one two-occurrence neighbor. Intuitively, dissociated variables which form clauses with two-occurrence neighbors are more constrained than those that appear with single-occurrence neighbors. Thus, there are more constraints on probabilities associated with two-occurrence neighbors (indexed by  $T^+$  and  $T^-$ ) than those associated with single-occurrence neighbors (indexed by  $S^+$  and  $S^-$ ); see conditions 1. and 2. in Theorem 16.

**Theorem 16.** (Oblivious bounds for w2CNFs having two-occurrence neighbors w.r.t.  $X_i$ ). For a given variable  $X_i$ , if  $\mathcal{F}$  contains at least one two-occurrence neighbor then we have the following oblivious bounds for  $X_i$ :

---

**Algorithm 1:** (DIS) Dissociation Bounds for WMC

---

**Input:** w2CNF  $\mathcal{F} = \langle \mathbf{X}, \Phi, \mathbf{C} \rangle$ ,

Variable ordering  $o = [X_1, X_2, \dots, X_{|\mathbf{X}|}]$

**Output:** Lower (or upper) bound on the WMC

1. Initialize:  $Z_B = 1$  (Bound on the partition function)

2. for  $i = 1$  to  $|\mathbf{X}|$  do

    2a. Convert  $\mathcal{F}$  to a minimal  $\mathcal{F}$

    2b. Convert  $C(X_i)$  to canonical form

    2c. if  $\mathbf{C}$  is inconsistent then

        ⊥ return 0

    else if  $C(X_i) = \{X_i\}$  then

        ⊥  $Z_B = Z_B \times p_i$

    else if  $C(X_i) = \{\overline{X_i}\}$  then

        ⊥  $Z_B = Z_B \times \overline{p_i}$

    else if  $C(X_i)$  has two-occurrence neighbors then

        ⊥ Update  $Z_B$  using Theorem 16

    else if  $C(X_i)$  has single-occurrence neighbors then

        ⊥ Update  $Z_B$  using Theorem 14

return  $Z_B$

---

- $U$ :  $\left( \prod_{j \in (S^+ \cup T^+)} p_{i;j} \geq p_i \right) \wedge \left( \prod_{j \in (S^- \cup T^-)} \overline{p_{i;j}} \geq \overline{p_i} \right)$

•  $L$ : Either of following three conditions hold:

1.  $\left( \prod_{j \in T^+} p_{i;j} \leq p_i \right) \wedge (\forall j \in (S^- \cup T^-) : \overline{p_{i;j}} = 0)$
2.  $\left( \prod_{j \in T^-} \overline{p_{i;j}} \leq \overline{p_i} \right) \wedge (\forall j \in (S^+ \cup T^+) : p_{i;j} = 0)$
3. If  $|T^+| = |T^-| = 1$  and  $T^+ = \{a\} \wedge T^- = \{b\}$ :
  - $(p_{i;a} \leq p_i) \wedge (\forall j \in S^- : \overline{p_{i;j}} = 0) \wedge$
  - $(\overline{p_{i;b}} \leq \overline{p_i}) \wedge (\forall j \in S^+ : p_{i;j} = 0)$

Optimal oblivious bounds are obtained by replacing inequality with equality in the bound conditions.

**Example 17.** Consider  $C(X'_1) = \{(X_{1;1} \vee Y_3), (X_{1;2} \vee Y_4), (\overline{X_{1;3}} \vee Y_4), (X_{1;4} \vee Y_5), (\overline{X_{1;5}} \vee Y_6)\}$ . Theorem 16 gives the following conditions for upper and lower oblivious bounds:

- $U$ :  $(p_{1;1} p_{1;2} p_{1;4} \geq p_1) \wedge (\overline{p_{1;3}} \overline{p_{1;5}} \geq \overline{p_1})$

•  $L$ : Either of the following three conditions hold:

1.  $(p_{1;2} \leq p_1) \wedge (\overline{p_{1;3}} = \overline{p_{1;5}} = 0)$
2.  $(\overline{p_{1;3}} \leq \overline{p_1}) \wedge (p_{1;1} = p_{1;2} = p_{1;4} = 0)$
3.  $(p_{1;2} \leq p_1) \wedge (\overline{p_{1;3}} \leq \overline{p_1}) \wedge (p_{1;1} = p_{1;4} = \overline{p_{1;5}} = 0)$

Table 3 summarizes the oblivious bound conditions. Theorems 14 and 16 yield the algorithm given in Algorithm 1 for bounding the partition function of a given w2CNF.

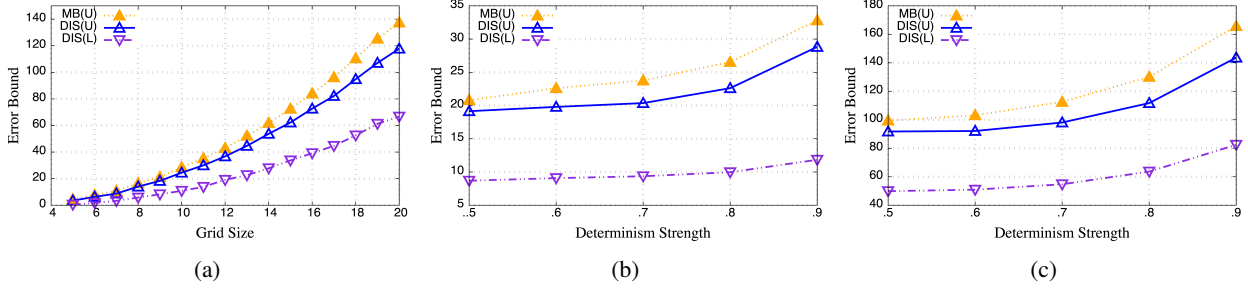


Figure 1: Upper bound estimates for dissociation DIS(U) and mini-bucket MB(U), and lower bound estimates for dissociation DIS(L). Error bound by varying (a) grid size (b) level of determinism for  $10 \times 10$  grid (c)  $20 \times 20$  grid. Lower value is better.

Table 3: Summary of oblivious bound conditions.  $T$ : whether  $C(X_i)$  has two-occurrence neighbors,  $S^+$  and  $S^-$ : whether  $C(X_i)$  has single-occurrence neighbors which appear in clauses  $(X_i \vee Y_j)$  and  $(\overline{X_i} \vee Y_j)$  respectively. An entry in a cell means that neighbors of the respective types are either present ( $\checkmark$ ), absent ( $\times$ ), or either present or absent ( $*$ ). Bold text in Case and Solution columns denote novel contributions of this paper while normal font text indicates previous work.

$S^+$	$S^-$	$T$	Case	Solution
$\checkmark$	$\times$	$\times$	Monotone	Theorems 9 & 14
$\times$	$\checkmark$	$\times$		
$\checkmark$	$\checkmark$	$\times$	<b>Single-occurrence</b>	<b>Theorem 14</b>
$*$	$*$	$\checkmark$	<b>Two-occurrence</b>	<b>Theorem 16</b>

## 5 EXPERIMENTS

We evaluated the performance of DIS (see Algorithm 1) and compared it with MB on generated synthetic datasets and benchmark datasets from the UAI 2008 probabilistic inference competition repository (<http://graphmod.ics.uci.edu/uai08>) for the task of computing upper and lower bounds on the weighted model count (or partition function). All experiment were conducted on quad-core Intel i7 based machines with 24GB RAM running Ubuntu.

### 5.1 SYNTHETIC DATASETS

We generated non-monotone w2CNF formulas encoded as  $m \times m$  grid structure graphical models parameterized by univariate and pairwise binary potentials. We then compared error bound performance of DIS and MB ( $i$ -bound = 1) from the aspects of (1) varying grid sizes under random weight function settings; and (2) varying weight function settings according to determinism strength under fixed grid sizes. For each model, we computed the true weighted model count  $Z^*$ . We then compared each algorithm’s approximated bound  $Z^{\text{algo}}$  and calculated the error bound as  $\log(Z^*/Z^{\text{algo}})$  for the lower bound and the

same negated for the upper bound. A lower error bound value is better. For each setting, we generated 50 random problem instances and ran DIS and MB 100 times for each instance. From the 100 solutions, we selected the best, namely either the lowest upper bound or the highest lower bound. We then computed the average error bound across the 50 problem instances.

**Grid size.** We generated  $m \times m$  grids using values of  $m = \{5, 6, 7, \dots, 20\}$ . For the weight function values, we sampled from an uniform  $U(0, 1)$  distribution. We also uniformly generated the clauses. The results are shown in Figure 1a. For the upper bound, DIS noticeably begins to outperform MB starting at around grid size  $10 \times 10$  and the performance gap widens as the grid size increases. Since MB utilizes the max function, it has a higher tendency to overestimate the upper bound. This was accomplished only by setting the weight function values to the  $k$ -th root (e.g.,  $p_{X_{1,1}} = p_{X_{1,2}} = \sqrt[p_{X_1}]{|C(X_1)|} = 2$ ). We would expect the performance gap to be wider, favoring DIS, by optimizing the inequalities. For the lower bound, MB produced 0 for all problems and thus was not plotted. MB has a high tendency to converge to the so called degenerate solution (i.e., 0) due to the min function. The lower bound for DIS is tighter, as compared to the upper bounds, since the settings to the lower bound inequalities do not need to be optimized.

**Determinism strength.** We analyzed the performance of DIS and MB according to various levels of determinism, namely the distance from uniform .5 (unweighted) towards 0 and 1. To accomplish this, we set all weight functions to the same value  $p_X \in \{.5, .6, .7, .8, .9\}$ . The results are shown in Figures 1b and 1c. For the lower bound, MB produced 0 for all problems and thus was not plotted. The overall relative performance comparison is similar to that of varying grid size. Again, the lower bound performance for DIS is tighter and all bounds had higher bound error as the determinism strength increased. Intuitively, as the gap between  $p_{X_i}$  and  $\overline{p_{X_i}}$  widens, the tendency to overestimate (underestimate) the upper (lower) bound increases.

Table 4: The log relative upper bound between dissociation DIS(U) and mini-bucket MB(U) on UAI 2008 repository problem instances. Lower value is better for DIS.

Instance	$\log \frac{Z^{\text{DIS}(U)}}{Z^{\text{MB}(U)}}$	Instance	$\log \frac{Z^{\text{DIS}(U)}}{Z^{\text{MB}(U)}}$
sg2-17	-277.8	orc111	-87.6
sg7-11	-293.4	orc175	-96.3
sg8-18	-281.9	orc180	-124.4
sg9-24	-292.8	orc203	-111.0
sg17-4	-303.3	orc218	-4.4
smk10	-50.9	orc62	-393.4
smk20	-165.9	orc154	-97.0
orc42	-119.6	orc225	-137.3
orc45	-261.1	orc139	-155.0

Table 5: The log relative lower bound between ground truth and Dissociation DIS(L) on UAI 2008 repository problem instances. Lower value is better for DIS.

Instance	$\log \frac{Z^*}{Z^{\text{DIS}(L)}}$	Instance	$\log \frac{Z^*}{Z^{\text{DIS}(L)}}$
sg2-17	732.4	orc111	209.8
sg7-11	759.4	orc175	342.6
sg8-18	727.3	orc180	375.0
sg9-24	774.5	orc203	346.8
sg17-4	752.1	orc218	18.2
smk10	191.3	orc62	—
smk20	799.8	orc154	354.7
orc42	407.9	orc225	499.7
orc45	747.8	orc139	576.6

## 5.2 UAI INFERENCE DATASETS

We also compared DIS to MB on the segmentation (sg), promedas (orc) and smokers (smk) dataset from the UAI 2008 repository. The variables in the models are binary and the number of variables range from  $\sim 100$  to 1000. We converted the non-pairwise models to pairwise models and then encoded them as W2CNF. We used  $i$ -bound = 1 for MB. We ran DIS and MB 100 times and similarly, we selected the best. For the upper bound, we evaluated using the log relative upper bound, namely  $\log(Z^{\text{DIS}(U)}/Z^{\text{MB}(U)})$ . Lower value is better for DIS. The results are shown in Table 4. DIS outperforms MB by a wide margin on the majority of the datasets. The solution quality of DIS for sg was quite consistent while for orc it had higher variance. For the lower bound, we evaluated dissociation’s lower bound against the ground truth, namely  $\log(Z^*/Z^{\text{DIS}(L)})$ . MB produced 0 for all problems and thus was not shown. The results are shown in Table 5 (orc62 was not tractable).

In summary, DIS performs consistently better than MB on harder WMC problems. In particular, the lower bounds

output by DIS are always better than MB.

## 6 CONCLUSION AND FUTURE WORK

We proposed an approximate, oblivious bounding scheme for WMC, extending the idea of dissociation to non-monotone formulas and exploiting logical structure. Dissociation yields a novel set of inequalities for which upper and lower bounds can be derived efficiently. Empirically, we showed that our method outperforms mini-buckets—a popular oblivious bounding scheme—on various datasets. The lower bounds are robust since they do not require optimization (in the monotone case). For upper bounds, we utilized naïve settings, namely the  $k$ -th root applied to the parameter of a dissociated variable.

For future work, we are interested in obtaining better (tighter) upper and lower bounds. To do so, we can leverage four powerful complementary techniques described in literature (cf. (Gogate and Domingos, 2011, 2013; Ihler et al., 2012; Lam et al., 2014; Liu and Ihler, 2011; Ping et al., 2015)): cost-shifting (or re-parameterization), higher  $i$ -bound, quantization and Hölder’s inequality. For instance, applying Hölder’s inequality to our running example (see Example 3) gives the optimization problem  $\min_{\omega} (p_{X_1}^{\omega} + (\overline{p_{X_1} p_{Y_2}})^{\omega})^{1/\omega} (1 + p_{Y_3}^{(1-\omega)})^{(1/1-\omega)}$  such that  $0 \leq \omega \leq 1$ . We can also apply Hölder’s inequality to dissociation which alternatively gives us the optimization problem  $\min_{p_{X_{1,1}}, p_{X_{1,2}}, \omega} (p_{X_{1,1}}^{\omega} + (\overline{p_{X_{1,1}} p_{Y_2}})^{\omega})^{1/\omega} (p_{X_{1,2}}^{(1-\omega)} + (\overline{p_{X_{1,2}} p_{Y_3}})^{(1-\omega)})^{(1/1-\omega)}$  such that  $p_{X_{1,1}} p_{X_{1,2}} = p_{X_1}$  and  $0 \leq \omega \leq 1$ . We are particularly interested in developing algorithms to optimize the latter problem and to determine which formulation will consistently yield tighter upper and lower bounds. Another line of future work is investigating the utility of our approach when applied to other inference tasks such as maximum a posteriori (MAP) estimation and marginal maximum a posteriori (MMAP) estimation.

### Acknowledgments

This work was supported by the DARPA Explainable Artificial Intelligence (XAI) Program under contract number N66001-17-2-4032, and by the National Science Foundation grants IIS-1652835, IIS-1528037, and IIS-1762268.

### References

- Chavira, M. and Darwiche, A. (2008). On probabilistic inference by weighted model counting. *Artificial Intelligence*, 172(6-7):772–799.
- Choi, A., Chavira, M., and Darwiche, A. (2007). Node splitting: A scheme for generating upper bounds in

- Bayesian networks. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence*, pages 57–66.
- Choi, A. and Darwiche, A. (2009). Relax then compensate: On max-product belief propagation and more. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*, pages 351–359.
- Dechter, R. (1996). Bucket elimination, a unifying framework for probabilistic inference. In *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence*, pages 211–219.
- Dechter, R. and Rish, I. (2003). Mini-buckets: A general scheme for bounded inference. *Journal of the ACM*, 50(2):107–153.
- den Heuvel, M. V., Gatterbauer, W., Theobald, M., and Geerts, F. (2018). A general framework for anytime approximation in probabilistic databases. In *8th International Workshop on Statistical Relational AI*. (to appear).
- Ermon, S., Gomes, C. P., Sabharwal, A., and Selman, B. (2013). Embed and project: Discrete sampling with universal hashing. In *Advances in Neural Information Processing Systems 26*.
- Gatterbauer, W. and Suciu, D. (2014). Oblivious bounds on the probability of boolean functions. *ACM Trans. Database Syst.*, 39(1):5:1–5:34.
- Gatterbauer, W. and Suciu, D. (2017). Dissociation and propagation for approximate lifted inference with standard relational database management systems. *VLDB J.*, 26(1):5–30.
- Gogate, V. and Dechter, R. (2007). Approximate counting by sampling the backtrack-free search space. In *Proceedings of the 22nd Conference on Artificial Intelligence*, pages 198–203.
- Gogate, V. and Dechter, R. (2011). SampleSearch: Importance sampling in presence of determinism. *Artificial Intelligence*, 175(2):694–729.
- Gogate, V. and Domingos, P. (2010). Formula-based probabilistic inference. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, pages 210–219.
- Gogate, V. and Domingos, P. (2011). Approximation by Quantization. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pages 247–255.
- Gogate, V. and Domingos, P. (2013). Structured Message Passing. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, pages 252–261.
- Gogate, V. and Domingos, P. M. (2016). Probabilistic theorem proving. *Communications of the ACM*, 59(7):107–115.
- Ihler, A., Flerova, N., Dechter, R., and Otten, L. (2012). Join-graph based cost-shifting schemes. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*, pages 397–406.
- Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.
- Lam, W., Kask, K., Dechter, R., and Ihler, A. (2014). Beyond static mini-bucket: Towards integrating with iterative cost-shifting based dynamic heuristics. In *Proceedings of the 7th Annual Symposium of Combinatorial Search*, pages 105–113.
- Liu, Q. (2014). *Reasoning and Decisions in Probabilistic Graphical Models—A Unified Framework*. University of California, Irvine.
- Liu, Q. and Ihler, A. (2011). Bounding the partition function using Hölder’s inequality. In *Proceedings of the 28th International Conference on Machine Learning*, pages 849–856.
- Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.
- Ping, W., Liu, Q., and Ihler, A. (2015). Decomposition bounds for marginal map. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, pages 3267–3275.
- Sang, T., Bearne, P., and Kautz, H. (2005). Performing Bayesian inference by weighted model counting. In *Proceedings of the 20th National Conference on Artificial Intelligence*, pages 475–481.
- Zhang, L. and Malik, S. (2002). The quest for efficient Boolean satisfiability solvers. In *Proceedings of the 14th International Conference on Computer Aided Verification*, pages 17–36.
- Zhang, N. L. and Poole, D. (1994). A simple approach to Bayesian network computations. In *Proceedings of the 10th Canadian Conference on Artificial Intelligence*, pages 171–178.

---

# Averaging Weights Leads to Wider Optima and Better Generalization

---

Pavel Izmailov\*<sup>1</sup> Dmitrii Podoprikin\*<sup>2,3</sup> Timur Garipov\*<sup>4,5</sup> Dmitry Vetrov<sup>2,3</sup> Andrew Gordon Wilson<sup>1</sup>

<sup>1</sup>Cornell University, <sup>2</sup>Higher School of Economics, <sup>3</sup>Samsung-HSE Laboratory,

<sup>4</sup>Samsung AI Center in Moscow, <sup>5</sup>Lomonosov Moscow State University

## Abstract

Deep neural networks are typically trained by optimizing a loss function with an SGD variant, in conjunction with a decaying learning rate, until convergence. We show that simple averaging of multiple points along the trajectory of SGD, with a cyclical or constant learning rate, leads to better generalization than conventional training. We also show that this *Stochastic Weight Averaging* (SWA) procedure finds much broader optima than SGD, and approximates the recent *Fast Geometric Ensembling* (FGE) approach with a single model. Using SWA we achieve notable improvement in test accuracy over conventional SGD training on a range of state-of-the-art residual networks, PyramidNets, DenseNets, and Shake-Shake networks on CIFAR-10, CIFAR-100, and ImageNet. In short, SWA is extremely easy to implement, improves generalization, and has almost no computational overhead.

## 1 INTRODUCTION

With a better understanding of the loss surfaces for multilayer networks, we can accelerate the convergence, stability, and accuracy of training procedures in deep learning. Recent work [Garipov et al., 2018, Draxler et al., 2018] shows that local optima found by SGD can be connected by simple curves of near constant loss. Building upon this insight, Garipov et al. [2018] also developed *Fast Geometric Ensembling* (FGE) to sample multiple nearby points in weight space to create high performing ensembles in the time required to train a single DNN.

FGE uses a high frequency cyclical learning rate with SGD to select networks to ensemble. In Figure 1 (left)

we see that the weights of the networks ensembled by FGE are on the periphery of the most desirable solutions. This observation suggests it is promising to average these points in *weight space*, and use a network with these averaged weights, instead of forming an ensemble by averaging the outputs of networks in *model space*. Although the general idea of maintaining a running average of weights traversed by SGD dates back to Ruppert [1988], this procedure is not typically used to train neural networks. It is sometimes applied as an exponentially decaying running average in combination with a decaying learning rate (where it is called an exponential moving average), which smooths the trajectory of conventional SGD but does not perform very differently. However, we show that an equally weighted average of the points traversed by SGD with a cyclical or constant learning rate, which we refer to as *Stochastic Weight Averaging* (SWA), has many surprising and promising features for training deep neural networks, leading to a better understanding of the geometry of their loss surfaces. Indeed, SWA with cyclical or constant learning rates can be used as a drop-in replacement for standard SGD training of multilayer networks — but with improved generalization and essentially no overhead. In particular:

- We show that SGD with cyclical [e.g., Loshchilov and Hutter, 2017] and constant learning rates traverses regions of weight space corresponding to high-performing networks. We find that while these models are moving around this optimal set they never reach its central points. We show that we can move into this more desirable space of points by averaging the weights proposed over SGD iterations.
- While FGE ensembles [Garipov et al., 2018] can be trained in the same time as a single model, test predictions for an ensemble of  $k$  models requires  $k$  times more computation. We show that SWA can be interpreted as an approximation to FGE ensembles but with the test-time, convenience, and interpretability of a single model.

---

\*Equal contribution.



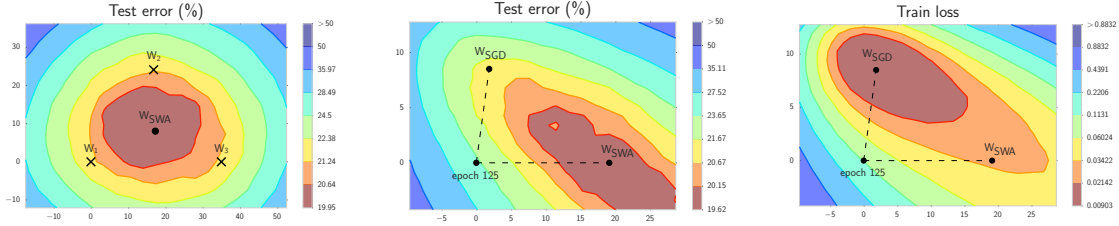


Figure 1: Illustrations of SWA and SGD with a Preactivation ResNet-164 on CIFAR-100<sup>1</sup>. **Left:** test error surface for three FGE samples and the corresponding SWA solution (averaging in weight space). **Middle and Right:** test error and train loss surfaces showing the weights proposed by SGD (at convergence) and SWA, starting from the same initialization of SGD after 125 training epochs.

- We demonstrate that SWA leads to solutions corresponding to wider optima than SGD. Keskar et al. [2017] and Hochreiter and Schmidhuber [1997] conjecture that the width of the optima is critically related to generalization. We illustrate that the loss on the train is shifted with respect to the test error (Figure 1, middle and right panels, and sections 3, 4). We show that SGD generally converges to a point near the boundary of the wide flat region of optimal points. SWA on the other hand is able to find a point centered in this region, often with slightly worse train loss but with substantially better test error.
- SWA achieves notable improvement for training a broad range of architectures over several consequential benchmarks. In particular, running SWA for just 10 epochs on ImageNet we are able to achieve 0.8% improvement for ResNet-50 and DenseNet-161, and 0.6% improvement for ResNet-150. We achieve improvement of over 1.3% on CIFAR-100 and of over 0.4% on CIFAR-10 with Preactivation ResNet-164, VGG-16 and Wide ResNet-28-10. We also achieve substantial improvement for the recent Shake-Shake Networks and PyramidNets.
- SWA is extremely easy to implement and has virtually no computational overhead compared to the conventional training schemes.
- We provide our implementation of SWA at <https://github.com/timgaripov/swa>.

## 2 RELATED WORK

This paper is fundamentally about better understanding the geometry of loss surfaces and generalization in deep learning. We follow the trajectory of weights traversed by SGD, leading to new geometric insights and the intuition that SWA will lead to better results than standard training. Empirically, we make the discovery that SWA

notably improves training of many state-of-the-art deep neural networks over a range of consequential benchmarks, with essentially no overhead.

The procedures for training neural networks are constantly being improved. New methods are being proposed for architecture design, regularization and optimization. The SWA approach is related to work in both optimization and regularization.

In optimization, there is great interest in how different types of local optima affect generalization in deep learning. Keskar et al. [2017] claim that SGD is more likely to converge to broad local optima than batch gradient methods, which tend to converge to sharp optima. Moreover, they argue that the broad optima found by SGD are more likely to have good test performance, even if the training loss is worse than for the sharp optima. On the other hand Dinh et al. [2017] argue that all the known definitions of sharpness are unsatisfactory and cannot on their own explain generalization. Chaudhari et al. [2017] propose the Entropy-SGD method that explicitly forces optimization towards wide valleys. They report that although the optima found by Entropy-SGD are wider than those found by conventional SGD, the generalization performance is still comparable.

The SWA method is based on averaging multiple points along the trajectory of SGD with cyclical or constant learning rates. The general idea of maintaining a running average of weights proposed by SGD was first considered in convex optimization by Ruppert [1988] and later by Polyak and Juditsky [1992]. However, this procedure is not typically used to train neural networks. Practi-

<sup>1</sup> Suppose we have three weight vectors  $w_1, w_2, w_3$ . We set  $u = (w_2 - w_1)$ ,  $v = (w_3 - w_1) - \langle w_3 - w_1, w_2 - w_1 \rangle / \|w_2 - w_1\|^2 \cdot (w_2 - w_1)$ . Then the normalized vectors  $\hat{u} = u / \|u\|$ ,  $\hat{v} = v / \|v\|$  form an orthonormal basis in the plane containing  $w_1, w_2, w_3$ . To visualize the loss in this plane, we define a Cartesian grid in the basis  $\hat{u}, \hat{v}$  and evaluate the networks corresponding to each of the points in the grid. A point  $P$  with coordinates  $(x, y)$  in the plane would then be given by  $P = w_1 + x \cdot \hat{u} + y \cdot \hat{v}$ .

tioners instead sometimes use an exponentially decaying running average of the weights found by SGD with a decaying learning rate, which smooths the trajectory of SGD but performs comparably.

SWA is making use of multiple samples gathered through exploration of the set of points corresponding to high performing networks. To enforce exploration we run SGD with constant or cyclical learning rates. Mandt et al. [2017] show that under several simplifying assumptions running SGD with a constant learning rate is equivalent to sampling from a Gaussian distribution centered at the minimum of the loss, and the covariance of this Gaussian is controlled by the learning rate. Following this explanation from [Mandt et al., 2017], we can interpret points proposed by SGD as being constrained to the surface of a sphere, since they come from a high dimensional Gaussian distribution. SWA effectively allows us to go inside the sphere to find higher density solutions.

In a procedure called Fast Geometric Ensembling (FGE), Garipov et al. [2018] showed that using a cyclical learning rate it is possible to gather models that are spatially close to each other but produce diverse predictions. They used the gathered models to train ensembles with no computational overhead compared to training a single DNN model. In recent work Neklyudov et al. [2018] also discuss an efficient approach for model averaging of Bayesian neural networks. SWA was inspired by following the trajectories of FGE proposals, in order to find a single model that would approximate an FGE ensemble, but provide greater interpretability, convenience, and test-time scalability.

Dropout [Srivastava et al., 2014] is an extremely popular approach to regularizing DNNs. Across each mini-batch used for SGD, a different architecture is created by randomly dropping out neurons. The authors make analogies between dropout, ensembling, and Bayesian model averaging. At test time, an ensemble approach is proposed, but then approximated with similar results by multiplying each connection by the dropout rate. At a high level, SWA and Dropout are both at once regularizers and training procedures, motivated to approximate an ensemble. Each approach implements these high level ideas quite differently, and as we show in our experiments, can be combined for improved performance.

### 3 STOCHASTIC WEIGHT AVERAGING

We present Stochastic Weight Averaging (SWA) and analyze its properties. In section 3.1, we consider trajectories of SGD with a constant and cyclical learning rate, which helps understand the geometry of SGD training for neural networks, and motivates the SWA procedure. Then in section 3.2 we present the SWA algorithm in

detail, in section 3.3 we derive its complexity, and in section 3.4 we analyze the width of optima found by SWA versus conventional SGD training. In section 3.5 we then examine the relationship between SWA and the recently proposed Fast Geometric Ensembling [Garipov et al., 2018]. Finally, in section 3.6 we consider SWA from the perspective of stochastic convex optimization.

We note the name SWA has two meanings: on the one hand, it is an average of SGD weights. On the other, with a cyclical or constant learning rate, SGD proposals are approximately sampling from the loss surface of the DNN, leading to stochastic weights.

#### 3.1 ANALYSIS OF SGD TRAJECTORIES

SWA is based on averaging the samples proposed by SGD using a learning rate schedule that allows exploration of the region of weight space corresponding to high-performing networks. In particular we consider cyclical and constant learning rate schedules.

The cyclical learning rate schedule that we adopt is inspired by Garipov et al. [2018] and Smith and Topin [2017]. In each cycle we linearly decrease the learning rate from  $\alpha_1$  to  $\alpha_2$ . The formula for the learning rate at iteration  $i$  is given by

$$\alpha(i) = (1 - t(i))\alpha_1 + t(i)\alpha_2,$$

$$t(i) = \frac{1}{c} (\text{mod}(i - 1, c) + 1).$$

The base learning rates  $\alpha_1 \geq \alpha_2$  and the cycle length  $c$  are the hyper-parameters of the method. Here by iteration we assume the processing of one batch of data. Figure 2 illustrates the cyclical learning rate schedule and the test error of the corresponding points. Note that unlike the cyclical learning rate schedule of Garipov et al. [2018] and Smith and Topin [2017], here we propose to use a discontinuous schedule that jumps directly from the minimum to maximum learning rates, and does *not* steadily increase the learning rate as part of the cycle. We use this more abrupt cycle because for our purposes exploration is more important than the accuracy of individual proposals. For even greater exploration, we also consider constant learning rates  $\alpha(i) = \alpha_1$ .

We run SGD with cyclical and constant learning rate schedules starting from a pretrained point for a Preactivation ResNet-164 on CIFAR-100. We then use the first, middle and last point of each of the trajectories to define a 2-dimensional plane in the weight space containing all affine combinations of these points. In Figure 3 we plot the loss on train and error on test for points in these planes. We then project the other points of the trajectory to the plane of the plot. Note that the trajectories do not generally lie in the plane of the plot, except for the

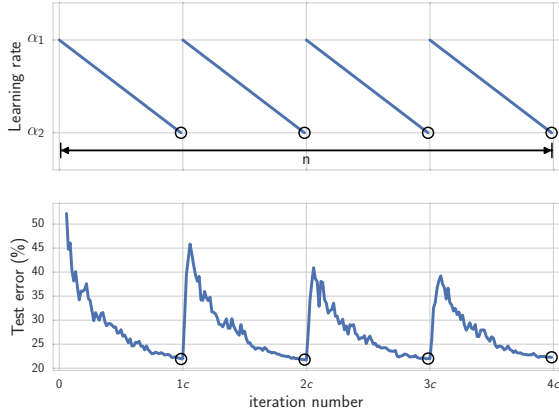


Figure 2: **Top:** cyclical learning rate as a function of iteration. **Bottom:** test error as a function of iteration for cyclical learning rate schedule with Preactivation-ResNet-164 on CIFAR-100. Circles indicate iterations corresponding to the minimum learning rates.

first, last and middle points, showed by black crosses in the figure. Therefore for other points of the trajectories it is not possible to tell the value of train loss and test error from the plots.

The key insight from Figure 3 is that both methods explore points close to the periphery of the set of high-performing networks. The visualizations suggest that both methods are doing exploration in the region of space corresponding to DNNs with high accuracy. The main difference between the two approaches is that the individual proposals of SGD with a cyclical learning rate schedule are in general much more accurate than the proposals of a fixed-learning rate SGD. After making a large step, SGD with a cyclical learning rate spends several epochs fine-tuning the resulting point with a decreasing learning rate. SGD with a fixed learning rate on the other hand is always making steps of relatively large sizes, exploring more efficiently than with a cyclical learning rate, but the individual proposals are worse.

Another important insight we can get from Figure 3 is that while the train loss and test error surfaces are qualitatively similar, they are not perfectly aligned. The shift between train and test suggests that more robust central points in the set of high-performing networks can lead to better generalization. Indeed, if we average several proposals from the optimization trajectories, we get a more robust point that has a substantially higher test performance than the individual proposals of SGD, and is essentially centered on the shifted mode for test error. We further discuss the reasons for this behaviour in sections 3.4, 3.5, 3.6.

### 3.2 SWA ALGORITHM

We now present the details of the Stochastic Weight Averaging algorithm, a simple but effective modification for training neural networks, motivated by our observations in section 3.1.

Following Garipov et al. [2018], we start with a pre-trained model  $\hat{w}$ . We will refer to the number of epochs required to train a given DNN with the conventional training procedure as its training budget and will denote it by  $B$ . The pretrained model  $\hat{w}$  can be trained with the conventional training procedure for full training budget or reduced number of epochs (e.g.  $0.75B$ ). In the latter case we just stop the training early without modifying the learning rate schedule. Starting from  $\hat{w}$  we continue training, using a cyclical or constant learning rate schedule. When using a cyclical learning rate we capture the models  $w_i$  that correspond to the minimum values of the learning rate (see Figure 2), following Garipov et al. [2018]. For constant learning rates we capture models at each epoch. Next, we average the weights of all the captured networks  $w_i$  to get our final model  $w_{\text{SWA}}$ .

Note that for cyclical learning rate schedule, the SWA algorithm is related to FGE [Garipov et al., 2018], except that instead of averaging the predictions of the models, we average their weights, and we use a different type of learning rate cycle. In section 3.5 we show how SWA can approximate FGE, but with a single model.

**Batch normalization.** If the DNN uses batch normalization [Ioffe and Szegedy, 2015], we run one additional pass over the data, as in Garipov et al. [2018], to compute the running mean and standard deviation of the activations for each layer of the network with  $w_{\text{SWA}}$  weights after the training is finished, since these statistics are not collected during training. For most deep learning libraries, such as PyTorch or Tensorflow, one can typically collect these statistics by making a forward pass over the data in training mode.

The SWA procedure is summarized in Algorithm 1.

### 3.3 COMPUTATIONAL COMPLEXITY

The time and memory overhead of SWA compared to conventional training is negligible. During training, we need to maintain a copy of the running average of DNN weights. Note however that the memory consumption in storing a DNN is dominated by its activations rather than its weights, and thus is only slightly increased by the SWA procedure, even for large DNNs (e.g., on the order of 10%). After the training is complete we only need to store the model that aggregates the average, leading to the same memory requirements as standard training.

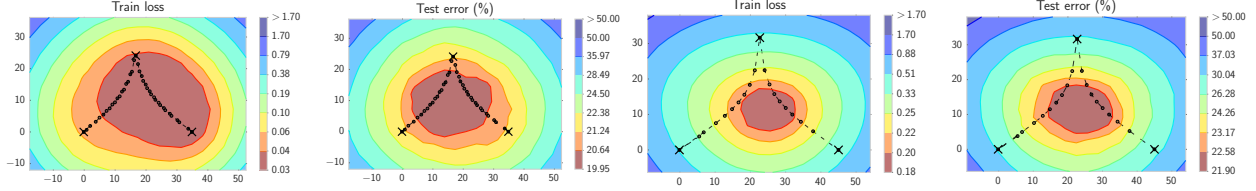


Figure 3: The  $L_2$ -regularized cross-entropy train loss and test error surfaces of a Preactivation ResNet-164 on CIFAR-100 in the plane containing the first, middle and last points (indicated by black crosses) in the trajectories with **(left two)** cyclical and **(right two)** constant learning rate schedules.

---

### Algorithm 1 Stochastic Weight Averaging

---

**Require:**

weights  $\hat{w}$ , LR bounds  $\alpha_1, \alpha_2$ ,  
 cycle length  $c$  (for constant learning rate  $c = 1$ ), number of iterations  $n$

**Ensure:**  $w_{\text{SWA}}$

```

 $w \leftarrow \hat{w}$  {Initialize weights with  $\hat{w}$ }
 $w_{\text{SWA}} \leftarrow w$ 
for  $i \leftarrow 1, 2, \dots, n$  do
   $\alpha \leftarrow \alpha(i)$  {Calculate LR for the iteration}
   $w \leftarrow w - \alpha \nabla \mathcal{L}_i(w)$  {Stochastic gradient update}
  if  $\text{mod}(i, c) = 0$  then
     $n_{\text{models}} \leftarrow i/c$  {Number of models}
     $w_{\text{SWA}} \leftarrow \frac{w_{\text{SWA}} \cdot n_{\text{models}} + w}{n_{\text{models}} + 1}$  {Update average}
  end if
end for
{Compute BatchNorm statistics for  $w_{\text{SWA}}$  weights}

```

---

During training extra time is only spent to update the aggregated weight average. This operation is of the form

$$w_{\text{SWA}} \leftarrow \frac{w_{\text{SWA}} \cdot n_{\text{models}} + w}{n_{\text{models}} + 1},$$

and it only requires computing a weighted sum of the weights of two DNNs. As we apply this operation at most once per epoch, SWA and SGD require practically the same amount of computation. Indeed, a similar operation is performed as a part of each gradient step, and each epoch consists of hundreds of gradient steps.

### 3.4 OPTIMA WIDTH

Keskar et al. [2017] and Chaudhari et al. [2017] conjecture that the width of a local optimum is related to generalization. The general explanation for the importance of width is that the surfaces of train loss and test error are shifted with respect to each other and it is thus desirable to converge to the modes of broad optima, which stay approximately optimal under small perturbations. In this section we compare the solutions found by SWA and SGD and show that SWA generally leads to much wider optima.

Let  $w_{\text{SWA}}$  and  $w_{\text{SGD}}$  denote the weights of DNNs trained using SWA and conventional SGD, respectively. Consider the rays

$$w_{\text{SWA}}(t, d) = w_{\text{SWA}} + t \cdot d,$$

$$w_{\text{SGD}}(t, d) = w_{\text{SGD}} + t \cdot d,$$

which follow a direction vector  $d$  on the unit sphere, starting at  $w_{\text{SWA}}$  and  $w_{\text{SGD}}$ , respectively. In Figure 4 we plot train loss and test error of  $w_{\text{SWA}}(t, d_i)$  and  $w_{\text{SGD}}(t, d_i)$  as a function of  $t$  for 10 random directions  $d_i, i = 1, 2, \dots, 10$  drawn from a uniform distribution on the unit sphere. For this visualization we use a Preactivation ResNet-164 on CIFAR-100.

First, while the loss values on train for  $w_{\text{SGD}}$  and  $w_{\text{SWA}}$  are quite similar (and in fact  $w_{\text{SGD}}$  has a slightly lower train loss), the test error for  $w_{\text{SGD}}$  is lower by 1.5% (at the converged value corresponding to  $t = 0$ ). Further, the shapes of both train loss and test error curves are considerably wider for  $w_{\text{SWA}}$  than for  $w_{\text{SGD}}$ , suggesting that SWA indeed converges to a wider optimum: we have to step much further away from the solution found by  $w_{\text{SWA}}$  to increase error by a given amount. We even see the error curve for SGD has an inflection point that is not present for these distances with SWA.

Notice that in Figure 4 any of the random directions from  $w_{\text{SGD}}$  increase test error. However, we know that the direction from  $w_{\text{SGD}}$  to  $w_{\text{SWA}}$  would decrease test error, since  $w_{\text{SWA}}$  has considerably lower test error than  $w_{\text{SGD}}$ . In other words, the path from  $w_{\text{SGD}}$  to  $w_{\text{SWA}}$  is qualitatively different from all directions shown in Figure 4, because along this direction  $w_{\text{SGD}}$  is far from optimal. We therefore consider the line segment connecting  $w_{\text{SGD}}$  and  $w_{\text{SWA}}$ :

$$w(t) = t \cdot w_{\text{SGD}} + (1 - t) \cdot w_{\text{SWA}}.$$

In Figure 5 we plot the train loss and test error of  $w(t)$  as a function of signed distance from  $w_{\text{SWA}}$  for Preactivation ResNet-164 and VGG-16 on CIFAR-100.

We can extract several key insights about  $w_{\text{SWA}}$  and  $w_{\text{SGD}}$  from Figure 5. First, the train loss and test error plots are indeed substantially shifted, and the point obtained

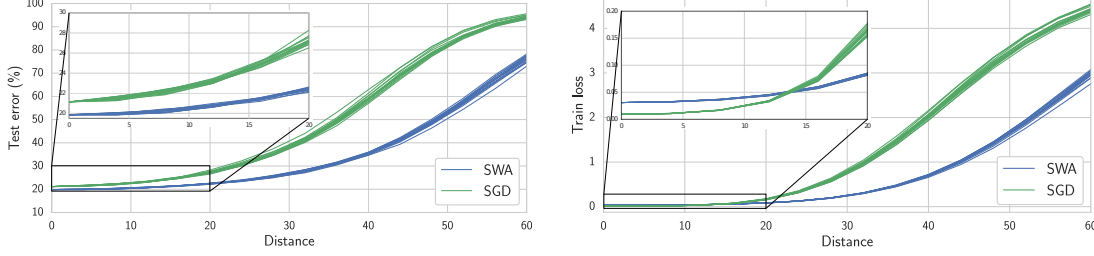


Figure 4: **(Left)** Test error and **(Right)**  $L_2$ -regularized cross-entropy train loss as a function of a point on a random ray starting at SWA (blue) and SGD (green) solutions for Preactivation ResNet-164 on CIFAR-100. Each line corresponds to a different random ray.

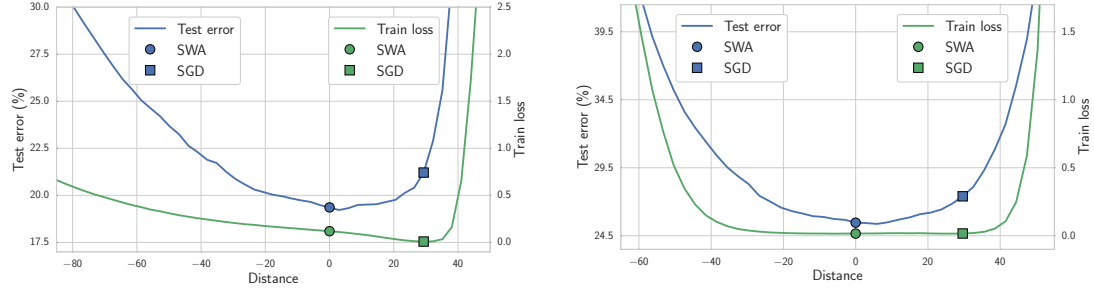


Figure 5:  $L_2$ -regularized cross-entropy train loss and test error as a function of a point on the line connecting SWA and SGD solutions on CIFAR-100. **Left:** Preactivation ResNet-164. **Right:** VGG-16.

by minimizing the train loss is far from optimal on test. Second,  $w_{\text{SGD}}$  lies near the boundary of a wide flat region of the train loss. Further, the loss is very steep near  $w_{\text{SGD}}$ .

Keskar et al. [2017] argue that the loss near sharp optima found by SGD with very large batches are actually flat in most directions, but there exist directions in which the optima are extremely steep. They conjecture that because of this sharpness the generalization performance of large batch optimization is substantially worse than that of solutions found by small batch SGD. Remarkably, in our experiments in this section we observe that there exist directions of steep ascent even for small batch optima, and that SWA provides even wider solutions (at least along random directions) with better generalization.

### 3.5 CONNECTION TO ENSEMBLING

Garipov et al. [2018] proposed the Fast Geometric Ensembling (FGE) procedure for training ensembles in the time required to train a single model. Using a cyclical learning rate, FGE generates a sequence of points that are close to each other in the weight space, but produce diverse predictions. In SWA instead of averaging the predictions of the models we average their weights. However, the predictions proposed by FGE ensembles and SWA models have similar properties.

Let  $f(\cdot)$  denote the predictions of a neural network

parametrized by weights  $w$ . We will assume that  $f$  is a scalar (e.g. the probability for a particular class) twice continuously differentiable function with respect to  $w$ .

Consider points  $w_i$  proposed by FGE. These points are close in the weight space by design, and concentrated around their average  $w_{\text{SWA}} = \frac{1}{n} \sum_{i=1}^n w_i$ . We denote  $\Delta_i = w_i - w_{\text{SWA}}$ . Note  $\sum_{i=1}^n \Delta_i = 0$ . Ensembling the networks corresponds to averaging the function values

$$\bar{f} = \frac{1}{n} \sum_{i=1}^n f(w_i).$$

Consider the linearization of  $f$  at  $w_{\text{SWA}}$ .

$$f(w_j) = f(w_{\text{SWA}}) + \langle \nabla f(w_{\text{SWA}}), \Delta_j \rangle + O(\|\Delta_j\|^2),$$

where  $\langle \cdot, \cdot \rangle$  denotes the dot product. Thus, the difference between averaging the weights and averaging the predictions

$$\begin{aligned} \bar{f} - f(w_{\text{SWA}}) &= \frac{1}{n} \sum_{i=1}^n (\langle \nabla f(w_{\text{SWA}}), \Delta_i \rangle + O(\|\Delta_i\|^2)) \\ &= \left\langle \nabla f(w_{\text{SWA}}), \frac{1}{n} \sum_{i=1}^n \Delta_i \right\rangle + O(\Delta^2) = O(\Delta^2), \end{aligned}$$

where  $\Delta = \max_{i=1}^n \|\Delta_i\|$ . Note that the difference between the predictions of different perturbed networks is

$$f(w_i) - f(w_j) = \langle \nabla f(w_{\text{SWA}}), \Delta_i - \Delta_j \rangle + O(\Delta^2),$$

and is thus of the first order of smallness, while the difference between averaging predictions and averaging weights is of the second order of smallness. Note that for the points proposed by FGE the distances between proposals are relatively small by design, which justifies the local analysis.

To analyze the difference between ensembling and averaging the weights of FGE proposals in practice, we run FGE for 20 epochs and compare the predictions of different models on the test dataset with a Preactivation ResNet-164 [He et al., 2016] on CIFAR-100. The norm of the difference between the class probabilities of consecutive FGE proposals averaged over the test dataset is 0.126. We then average the weights of the proposals and compute the class probabilities on the test dataset. The norm of difference of the probabilities for the SWA model and the FGE ensemble is 0.079, which is substantially smaller than the difference between the probabilities of consecutive FGE proposals. Further, the fraction of objects for which consecutive FGE proposals output the same labels is not greater than 87.33%. For FGE and SWA the fraction of identically labeled objects is 95.26%.

The theoretical considerations and empirical results presented in this section suggest that SWA can approximate the FGE ensemble with a single model.

### 3.6 CONNECTION TO CONVEX MINIMIZATION

Mandt et al. [2017] showed that under strong simplifying assumptions SGD with a fixed learning rate approximately samples from a Gaussian distribution centered at the minimum of the loss. Suppose this is the case when we run SGD with a fixed learning rate for training a DNN.

Let us denote the dimensionality of the weight space of the neural network by  $d$ . Denote the samples produced by SGD by  $w_i$ ,  $i = 1, 2, \dots, k$ . Assume the points  $w_i$  are concentrated around the local optimum  $\hat{w}$ . The SWA solution is given by  $w_{\text{SWA}} = \frac{1}{n} \sum_{i=1}^k w_i$ . The points  $w_i$  are samples from a multidimensional Gaussian  $\mathcal{N}(\hat{w}, \Sigma)$  for some covariance matrix  $\Sigma$  defined by the curvature of the loss, batch size and the learning rate. Note that the samples from a multidimensional Gaussian are concentrated on the ellipsoid

$$\left\{ z \in \mathbb{R}^d \mid \|\Sigma^{-\frac{1}{2}}(z - \hat{w})\| = \sqrt{d} \right\},$$

and the probability mass for a sample to end up inside the ellipsoid near  $\hat{w}$  is negligible. On the other hand,  $w_{\text{SWA}}$  is guaranteed to converge to  $\hat{w}$  as  $k \rightarrow \infty$ .

Moreover, Polyak and Juditsky [1992] showed that averaging SGD proposals achieves the best possible conver-

gence rate among all stochastic gradient algorithms. The proof relies on the convexity of the underlying problem and in general there are no convergence guarantees if the loss function is non-convex [see e.g. Ghadimi and Lan, 2013]. While DNN loss functions are known to be non-convex [e.g. Choromanska et al., 2015], over the trajectory of SGD these loss surfaces are approximately convex [e.g. Goodfellow et al., 2015]. However, even when the loss is locally non-convex, SWA can improve *generalization*. For example, in Figure 5 we see that SWA converges to a central point of the training loss.

## 4 EXPERIMENTS

We compare SWA against conventional SGD training on CIFAR-10, CIFAR-100 and ImageNet ILSVRC-2012 [Russakovsky et al., 2012]. We also compare to Fast Geometric Ensembling (FGE) [Garipov et al., 2018], but we note that FGE is an ensemble whereas SWA corresponds to a single model. Conventional SGD training uses a standard decaying learning rate schedule (details in the Appendix) until convergence. We found an exponentially decaying average of SGD to perform comparably to conventional SGD at convergence. We release the code for reproducing the results in this paper at <https://github.com/timgaripov/swa>.

### 4.1 CIFAR DATASETS

For the experiments on CIFAR datasets we use VGG-16 [Simonyan and Zisserman, 2014], a 164-layer Preactivation-ResNet [He et al., 2016] and Wide ResNet-28-10 [Zagoruyko and Komodakis, 2016] models. Additionally, we experiment with the recent Shake-Shake-2x64d [Gastaldi, 2017] on CIFAR-10 and PyramidNet-272 (bottleneck,  $\alpha = 200$ ) [Han et al., 2016] on CIFAR-100. All models are trained using  $L_2$ -regularization, and VGG-16 also uses dropout.

For each model we define *budget* as the number of epochs required to train the model until convergence with conventional SGD training, such that we do not see improvement with SGD beyond this budget. We use the same budgets for VGG, Preactivation ResNet and Wide ResNet models as Garipov et al. [2018]. For Shake-Shake and PyramidNets we use the budgets indicated by the papers that proposed these models [Gastaldi, 2017, Han et al., 2016]. We report the results of SWA training within 1, 1.25 and 1.5 budgets of epochs.

For VGG, Wide ResNet and Preactivation-ResNet models we first run standard SGD training for  $\approx 75\%$  of the training budget, and then use the weights at the last epoch as an initialization for SWA with a fixed learning rate schedule. We ran SWA for 0.25, 0.5 and 0.75 budget to complete the training within 1, 1.25 and 1.5 budgets

Table 1: Accuracies (%) of SWA, SGD and FGE methods on CIFAR-100 and CIFAR-10 datasets for different training budgets. Accuracies for the FGE ensemble are from Garipov et al. [2018].

DNN (Budget)	SGD	FGE (1 Budget)	SWA		
			1 Budget	1.25 Budgets	1.5 Budgets
CIFAR-100					
VGG-16 (200)	72.55 ± 0.10	74.26	73.91 ± 0.12	74.17 ± 0.15	74.27 ± 0.25
ResNet-164 (150)	78.49 ± 0.36	79.84	79.77 ± 0.17	80.18 ± 0.23	80.35 ± 0.16
WRN-28-10 (200)	80.82 ± 0.23	82.27	81.46 ± 0.23	81.91 ± 0.27	82.15 ± 0.27
PyramidNet-272 (300)	83.41 ± 0.21	–	–	83.93 ± 0.18	84.16 ± 0.15
CIFAR-10					
VGG-16 (200)	93.25 ± 0.16	93.52	93.59 ± 0.16	93.70 ± 0.22	93.64 ± 0.18
ResNet-164 (150)	95.28 ± 0.10	95.45	95.56 ± 0.11	95.77 ± 0.04	95.83 ± 0.03
WRN-28-10 (200)	96.18 ± 0.11	96.36	96.45 ± 0.11	96.64 ± 0.08	96.79 ± 0.05
ShakeShake-2x64d (1800)	96.93 ± 0.10	–	–	97.16 ± 0.10	97.12 ± 0.06

respectively.

For Shake-Shake and PyramidNet architectures we do not report the results in one budget. For these models we use a full budget to get an initialization for the procedure, and then train with a cyclical learning rate schedule for 0.25 and 0.5 budgets. We used long cycles of small learning rates for Shake-Shake, because this architecture already involves many stochastic components.

We present the details of the learning rate schedules for each of these models in the Appendix.

For each model we also report the results of conventional SGD training, which we denote by **SGD**. For VGG, Preactivation ResNet and Wide ResNet we also provide the results of the **FGE** method with one budget reported in Garipov et al. [2018]. Note that for FGE we report the accuracy of an ensemble of 6 to 12 networks, while for SWA we report the accuracy of a single model.

We summarize the experimental results in Table 1. For all models we report the mean and standard deviation of test accuracy over 3 runs. In all conducted experiments SWA substantially outperforms SGD in one budget, and improves further, as we allow more training epochs. Across different architectures we see consistent improvement by  $\approx 0.5\%$  on CIFAR-10 (excluding Shake-Shake, for which SGD performance is already extremely high) and by 0.75-1.5% on CIFAR-100. Amazingly, SWA is able to achieve comparable or better performance than FGE ensembles with just one model. On CIFAR-100 SWA usually needs more than one budget to get results comparable with FGE ensembles, but on CIFAR-10 even with 1 budget SWA outperforms FGE.

## 4.2 IMAGENET

On ImageNet we experimented with ResNet-50, ResNet-152 [He et al., 2016] and DenseNet-161 [Huang et al.,

2017]. For these architectures we used pretrained models from `PyTorch.torchvision`. For each of the models we ran SWA for 10 epochs with a cyclical learning rate schedule with the same parameters for all models (the details can be found in the Appendix), and report the mean and standard deviation of test error averaged over 3 runs. The results are shown in Table 2.

Table 2: Accuracies (%) on ImageNet dataset for SWA and SGD with different architectures.

DNN	SGD	SWA	
		5 epochs	10 epochs
ResNet-50	76.15	76.83 ± 0.01	76.97 ± 0.05
ResNet-152	78.31	78.82 ± 0.01	78.94 ± 0.07
DenseNet-161	77.65	78.26 ± 0.09	78.44 ± 0.06

For all 3 architectures SWA provides consistent improvement by 0.6-0.9% over the pretrained models.

## 4.3 EFFECT OF THE LEARNING RATE SCHEDULE

In this section we explore how the learning rate schedule affects the performance of SWA. We run experiments on Preactivation ResNet-164 on CIFAR-100. For all schedules we use the same initialization from a model trained for 125 epochs using the conventional SGD training. As a baseline we use a fully-trained model trained with conventional SGD for 150 epochs.

We consider a range of constant and cyclical learning rate schedules. For cyclical learning rates we fix the cycle length to 5, and consider the pairs of base learning rate parameters  $(\alpha_1, \alpha_2) \in \{(10^{-1}, 10^{-3}), (5 \cdot 10^{-2}, 5 \cdot 10^{-4}), (10^{-2}, 10^{-4}), (5 \cdot 10^{-3}, 5 \cdot 10^{-5})\}$ . Among the constant learning rates we consider  $\alpha_1 \in \{10^{-1}, 5 \cdot 10^{-2}, 10^{-2}, 10^{-3}\}$ .

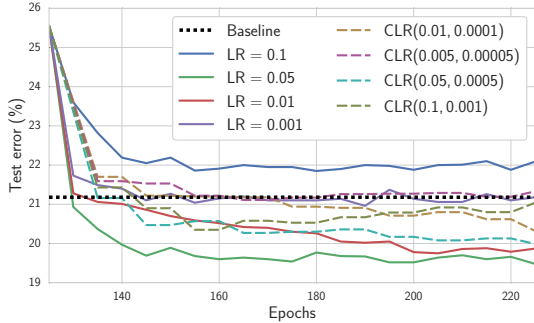


Figure 6: Test error as a function of training epoch for SWA with different learning rate schedules with a Pre-activation ResNet-164 on CIFAR-100.

We plot the test error of the SWA procedure for different learning rate schedules as a function of the number of training epochs in Figure 6.

We find that in general the more aggressive constant learning rate schedule leads to faster convergence of SWA. In our experiments we found that setting the learning rate to some intermediate value between the largest and the smallest learning rate used in the annealing scheme in conventional training usually gave us the best results. The approach is however universal and can work well with different learning rate schedules tailored for particular tasks.

#### 4.4 DNN TRAINING WITH A FIXED LEARNING RATE

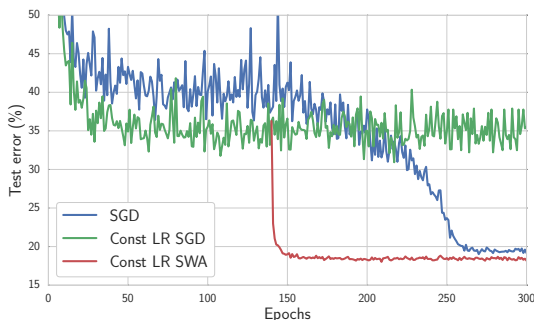


Figure 7: Test error as a function of training epoch for constant (green) and decaying (blue) learning rate schedules for a Wide ResNet-28-10 on CIFAR-100. In red we average the points along the trajectory of SGD with constant learning rate starting at epoch 140.

In this section we show that it is possible to train DNNs from scratch with a fixed learning rate using SWA. We run SGD with a fixed learning rate of 0.05 on a Wide ResNet-28-10 [Zagoruyko and Komodakis, 2016] for 300 epochs from a random initialization on CIFAR-100.

We then averaged the weights at the end of each epoch from epoch 140 and until the end of training. The final test accuracy of this SWA model was 81.7.

Figure 7 illustrates the test error as a function of the number of training epochs for SWA and conventional training. The accuracy of the individual models with weights averaged by SWA stays at the level of  $\approx 65\%$  which is 16% less than the accuracy of the SWA model. These results correspond to our intuition presented in section 3.6 that SGD with a constant learning rate oscillates around the optimum, but SWA converges.

While being able to train a DNN with a fixed learning rate is a surprising property of SWA, for practical purposes we recommend initializing SWA from a model pre-trained with conventional training (possibly for a reduced number of epochs), as it leads to faster and more stable convergence than running SWA from scratch.

## 5 DISCUSSION

We have presented Stochastic Weight Averaging (SWA) for training neural networks. SWA is extremely easy to implement, architecture-agnostic, and improves generalization performance at virtually no additional cost over conventional training.

There are so many exciting directions for future research. SWA does not require each weight in its average to correspond to a good solution, due to the geometry of weights traversed by the algorithm. It therefore may be possible to develop SWA for much faster convergence than standard SGD. One may also be able to combine SWA with large batch sizes while preserving generalization performance, since SWA discovers much broader optima than conventional SGD training. Furthermore, a cyclic learning rate enables SWA to explore regions of high posterior density over neural network weights. Such learning rate schedules could be developed in conjunction with stochastic MCMC approaches, to encourage exploration while still providing high quality samples. One could also develop SWA to average whole regions of good solutions, using the high-accuracy curves discovered in Garipov et al. [2018].

A better understanding of the loss surfaces for multilayer networks will help continue to unlock the potential of these rich models. We hope that SWA will inspire further progress in this area.

**Acknowledgements.** This work was supported by NSF IIS-1563887, Samsung Research, Samsung Electronics and Russian Science Foundation grant 17-11-01027. We also thank Vadim Bereznyuk for helpful comments.



## References

- P. Chaudhari, Anna Choromanska, S. Soatto, Yann LeCun, C. Baldassi, C. Borgs, J. Chayes, Levent Sagun, and R. Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. In *International Conference on Learning Representations (ICLR)*, 2017.
- Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics*, pages 192–204, 2015.
- Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *International Conference on Machine Learning*, pages 1019–1028, 2017.
- Felix Draxler, Kambis Veschgini, Manfred Salmhofer, and Fred Hamprecht. Essentially no barriers in neural network energy landscape. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1308–1317, 2018.
- Timur Garipov, Pavel Izmailov, Dmitrii Podoprikin, Dmitry P Vetrov, and Andrew Gordon Wilson. Loss surfaces, mode connectivity, and fast ensembling of dnns. *arXiv preprint arXiv:1802.10026*, 2018.
- Xavier Gastaldi. Shake-shake regularization. *arXiv preprint arXiv:1705.07485*, 2017.
- Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.
- Ian J Goodfellow, Oriol Vinyals, and Andrew M Saxe. Qualitatively characterizing neural network optimization problems. *International Conference on Learning Representations*, 2015.
- Dongyoon Han, Jiwhan Kim, and Junmo Kim. Deep pyramidal residual networks. *arXiv preprint arXiv:1610.02915*, 2016.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997.
- Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, volume 1, page 3, 2017.
- Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.
- Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *International Conference on Learning Representations*, 2017.
- Ilya Loshchilov and Frank Hutter. Sgdr: stochastic gradient descent with restarts. *International Conference on Learning Representations*, 2017.
- Stephan Mandt, Matthew D Hoffman, and David M Blei. Stochastic gradient descent as approximate bayesian inference. *The Journal of Machine Learning Research*, 18(1):4873–4907, 2017.
- Kirill Neklyudov, Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. Variance networks: When expectation does not meet your expectations. *arXiv preprint arXiv:1803.03764*, 2018.
- Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.
- David Ruppert. Efficient estimations from a slowly convergent robbins-monro process. Technical report, Cornell University Operations Research and Industrial Engineering, 1988.
- Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2012.
- Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- Leslie N Smith and Nicholay Topin. Exploring loss function topology with cyclical learning rates. *arXiv preprint arXiv:1702.04283*, 2017.
- Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.
- Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

---

# Block-Value Symmetries in Probabilistic Graphical Models

---

Gagan Madan, Ankit Anand, Mausam and Parag Singla

Indian Institute of Technology Delhi

gagan.madan1@gmail.com, {ankit.anand, mausam, parags}@cse.iitd.ac.in

## Abstract

One popular way for lifted inference in probabilistic graphical models is to first merge symmetric states into a single cluster (orbit) and then use these for downstream inference, via variations of orbital MCMC [Niepert, 2012]. These orbits are represented compactly using permutations over variables, and variable-value (VV) pairs, but they can miss several state symmetries in a domain.

We define the notion of permutations over block-value (BV) pairs, where a block is a set of variables. BV strictly generalizes VV symmetries, and can compute many more symmetries for increasing block sizes. To operationalize use of BV permutations in lifted inference, we describe 1) an algorithm to compute BV permutations given a block partition of the variables, 2) BV-MCMC, an extension of orbital MCMC that can sample from BV orbits, and 3) a heuristic to suggest good block partitions. Our experiments show that BV-MCMC can mix much faster compared to vanilla MCMC and orbital MCMC.

## 1 INTRODUCTION

A *lifted inference* algorithm for probabilistic graphical models (PGMs) performs inference on a smaller model, which is constructed by merging together states (or variables) of the original model [Poole, 2003; de Salvo Braz *et al.*, 2005; Kimmig *et al.*, 2015]. Two main kinds of lifted inference algorithms exist: those where lifting is tied to an existing inference procedure such as belief propagation [Singla and Domingos, 2008; Kersting *et al.*, 2009], Gibbs sampling [Venugopal and Gogate, 2012], weighted model counting [Gogate and Domingos, 2011],

variational inference [Bui *et al.*, 2013] and linear programming [Mladenov *et al.*, 2012]; and those that merge symmetric states/variables independent of the procedure [Niepert, 2012; Van den Broeck and Niepert, 2015; Anand *et al.*, 2016].

One approach for generating symmetries is by computing isomorphism over a graphical representation of the PGM. This merges symmetric states into a single cluster (orbit), which is compactly represented as permutations over a polynomial representation. Permutations over variables [Niepert, 2012] and over variable-value (VV) pairs [Anand *et al.*, 2017] have been studied, with latter being a generalization of the former, capturing many more state symmetries. While more general, VV permutations clearly do not capture all possible state symmetries in a domain. For example, state  $s_1 = (0, 0, 0, 0)$  is symmetric to  $s_2 = (0, 1, 1, 1)$  in Figure 1(b), but VV permutations cannot represent it.

A natural question arises: are there more general representations which can capture (a subset of) these larger set of symmetries? We note that the problem of computing all possible symmetries is intractable since there is an exponential number of permutations over an exponentially large state space, each of which could be a symmetry (or not). Nevertheless, we hope there are representations which can capture additional symmetries compared to current approaches in bounded polynomial time. More so, it would be interesting to come up with a representation that enables computation of larger and larger sets of symmetries, while paying additional costs, which could be controlled as a function of a parameter of the representation.

As a significant step toward this research question, we develop the novel notion of symmetries defined over *block-value (BV) pairs*. Here, a block is a set of variables, and its value is an assignment to these variables. Intuitively, BV pairs can capture all such VV pairs that are not permuted independently, instead, are

$X_1=0$	$X_2=1$	$\Phi$
0	0	a
0	1	b
1	0	c
1	1	c

(a)

$X_1=0$	$X_2=1$	$\Phi$	$X_3=0$	$X_4=1$	$\Phi$
0	0	a	0	0	b
0	1	b	0	1	d
1	0	c	1	0	c
1	1	d	1	1	a

(b)

Figure 1: Block-Value Symmetries (a) BV Symmetries within a block (b) BV Symmetries across blocks

permuted in subsets together. For example, it can capture symmetry of states  $s_1$  and  $s_2$  via a BV permutation which maps  $\{(X_1, 0), (X_2, 0)\} \leftrightarrow \{(X_3, 1), (X_4, 1)\}$  and  $\{(X_1, 0), (X_2, 1)\} \leftrightarrow \{(X_3, 0), (X_4, 0)\}$ .

Clearly, symmetries defined over BV pairs are a strict generalization of those over VV pairs, since each VV pair is a BV pair with a block of size 1. Our blocks can be of varying sizes and the size of each block essentially controls the set of symmetries that can be captured; larger the blocks, more the symmetries, coming at an additional cost (exponential in the max size of a block).

In this paper, we formally develop the notion of symmetries as permutations defined over a subset of BV pairs. Some of these permutations will be invalid (when blocks overlap with each other) and their application may lead to inconsistent state. In order to ensure valid permutations, we require that the blocks come from a disjoint set of blocks, referred to as a *block partition*. Given a block partition, we show how to compute the corresponding set of symmetries by reducing the problem to one of graph isomorphism. We also show that our BV symmetries can be thought of as VV symmetries, albeit over a transformed graphical model, where the new variables represent the blocks in the original graph.

Next, we show that jointly considering symmetries obtained from different block partitions can result in capturing symmetries not obtainable from any single one. Since, there is an exponential number of such block partitions, we provide an efficient heuristic for obtaining a promising partition of blocks, referred to as a *candidate set*.

Use of BV symmetries in an MCMC framework requires uniform sampling of a state from each orbit, i.e., a set of symmetric states. This turns out to be a non-trivial task when the orbits are defined over symmetries corresponding to different block partitions. In response, we design an aggregate Markov chain which samples from orbits corresponding to each (individual) candidate set in turn.

We prove that our aggregate Markov chain converges to the desired distribution. As a proof of the utility of our BV symmetries, we show that their usage results in significantly faster mixing times on two different domains.

The outline of this paper is as follows. We start with some background on variable and VV symmetries in Section 2. This is followed by the exposition of our symmetries defined over BV pairs (Section 3). Section 4 describes our algorithm for using BV symmetries in MCMC. This is followed by our heuristic to compute promising candidate sets in Section 5. We present our experimental evaluation (Section 6) and conclude the paper with directions for future work.

## 2 BACKGROUND

Let  $\mathcal{X} = \{X_1, X_2, \dots, X_n\}$  denote a set of discrete valued random variables. We will use the symbol  $x_i$  to denote the value taken by the variable  $X_i$ . We will assume that each of the variables comes from the same domain  $\mathcal{D}$ . A state  $s \in \mathcal{D}^n$  is an assignment to all the variables in the set  $\mathcal{X}$ . Further,  $s(X_i) = x_i$  gives the value of variable  $X_i$  in state  $s$ . We will use  $\mathcal{S}$  to denote the set of all possible states.

A Graphical Model [Koller and Friedman, 2009] is a set of pairs  $\{(f_j, w_j)\}_{j=1}^m$  where  $f_j$  is a feature function defined over the variables in the set  $\mathcal{X}$  and  $w_j$  is its associated weight.

**Definition 1.** Action of  $\theta$  on  $\mathcal{G}$  results in a new graphical model where the occurrence of  $X_i$  in each feature  $f_j$  in  $\mathcal{G}$  is replaced by  $\theta(X_i)$ . Given a graphical model  $\mathcal{G}$ , a permutation  $\theta$  of the variables in  $\mathcal{X}$  is said to be a **variable symmetry** of  $\mathcal{G}$  if the action of  $\theta$  on  $\mathcal{G}$  results back in  $\mathcal{G}$ .

Given a state  $s \in \mathcal{S}$ , the action of  $\theta$  on  $s$ , denoted by  $\theta(s)$ , results in a new state  $s'$  such that  $\forall X_i, X_j \in \mathcal{X}$  if  $\theta(X_i) = X_j$  and  $s(X_j) = x_j$  then  $s'(X_i) = x_j$ .

The set of all variable symmetries forms a group called *the variable automorphic group* of  $\mathcal{G}$  and is denoted by  $\Theta$ .  $\Theta$  partitions the states into equivalence classes or orbits which are as defined below.

**Definition 2.** Given a variable automorphic group  $\Theta$ , the **orbit** of a state  $s$  under the effect of  $\Theta$  is defined as  $\Gamma_{\Theta}(s) = \{\theta(s) | \theta \in \Theta\}$ .

Intuitively, the orbit of a state  $s$  is set of all states reachable from  $s$  under the action of any permutation in the automorphic group.

We note that variable symmetries are probability preserving transformations [Niepert, 2012]. Let  $\mathcal{P}$  denote the distribution defined by a graphical model  $\mathcal{G}$  where  $\mathcal{P}(s)$  is the probability of a state  $s$ .

**Theorem 1.** If  $\Theta$  is a variable automorphic group of  $\mathcal{G}$ , then  $\forall s \in \mathcal{S}, \forall \theta \in \Theta, \mathcal{P}(s) = \mathcal{P}(\theta(s))$ .

Anand et al. [2017] extend the notion of variable symmetries to those defined over variable value (VV) pairs. Let  $(X_i, x_i)$  denote a VV pair and let  $\mathcal{X}_V$  denote the set of all possible such pairs. Let  $\phi$  denote a permutation over the set  $\mathcal{X}_V$ . Action of  $\phi$  on state  $s$ , denoted by  $\phi(s)$ , results in a state  $s'$ , such that  $\forall X_i, X_j \in \mathcal{X}$ , if  $\phi(X_i, s(X_i)) = (X_j, x_j)$ , then  $s'(X_j) = x_j$ .

There are some VV permutations which when applied to a state  $s$  may result in an inconsistent state. For instance, let  $\phi(X_0, 0) = (X_0, 0)$ ,  $\phi(X_1, 1) = (X_0, 1)$  and  $s = (0, 1)$ , then  $\phi(s)$  results in an inconsistent state with multiple values being assigned to  $X_0$ . Therefore, the notion of valid VV permutation needs to be defined which when applied to any state  $s \in \mathcal{S}$  always results in a consistent state  $s'$  [Anand et al., 2017].

**Definition 3.** A VV permutation  $\phi$  over  $\mathcal{X}_V$  is said to be a **valid VV permutation** if whenever there exists a VV pair  $(X_i, x_i)$  such that  $\phi(X_i, x_i) = (X_j, x_j)$ , then for all the VV pairs of the form  $(X_i, x'_i)$  where  $x'_i \in \mathcal{D}_i$ ,  $\phi(X_i, x_i) = (X_j, x'_j)$  where  $x'_j \in \mathcal{D}_j$ .

**Definition 4.** Action of  $\phi$  on  $\mathcal{G}$  results in a new graphical model where the occurrence of  $(X_i, x_i)$  in each feature  $f_j$  in  $\mathcal{G}$  is replaced by  $\phi(X_i, x_i)$ . We say that  $\phi$  is a **VV symmetry** of  $\mathcal{G}$ , if action of  $\phi$  on  $\mathcal{G}$  results back in  $\mathcal{G}$ .

Similar to variable symmetries, the set of all VV symmetries form a group called the VV automorphic group of  $\mathcal{G}$  and is denoted by  $\Phi$ . Analogously,  $\Phi$  partitions the states into orbits defined as  $\Gamma_{\Phi}(s) = \{\phi(s) | \forall \phi \in \Phi\}$ .

In the following, we will often refer to the automorphic groups  $\Theta$  and  $\Phi$  as symmetry groups of  $\mathcal{G}$ . It can be easily seen that VV symmetries subsume variable symmetries and like variable symmetries, they are also probability preserving transformations.

**Theorem 2.** If  $\Phi$  is a VV automorphic group of  $\mathcal{G}$ , then  $\forall s \in \mathcal{S}, \forall \phi \in \Phi, \mathcal{P}(s) = \mathcal{P}(\phi(s))$

The orbits so obtained through variable (VV) symmetries can then be exploited for faster mixing by Markov Chain Monte Carlo (MCMC) based methods as described below.

## 2.1 Orbital-MCMC

Markov Chain Monte Carlo (MCMC) methods [Koller and Friedman, 2009] are one of the popular algorithms for approximate inference in Probabilistic Graphical Models. Starting with a random state, these methods set up a Markov chain over the state space whose stationary distribution is same as the desired distribution. Convergence is guaranteed in the limit of a large number of samples coming from the Markov chain.

Orbital MCMC and VV-MCMC improve MCMC methods by exploiting Variable and VV symmetries, respectively. Given a Markov chain  $\mathcal{M}$  and a symmetry group  $\Phi$ , starting from a sample  $s_t$ , any subsequent sample is obtained in 2 steps: a) An intermediate state  $s'$  is obtained according to  $\mathcal{M}$  b) The next sample  $s_{t+1}$  is obtained by sampling a state uniformly from the orbit (Variable or VV) of the intermediate state  $s'$ . Sampling a state from the orbit of the intermediate state is done using the Product Replacement Algorithm [Celler et al., 1995; Pak, 2000]. This two step chain so obtained converges to the true stationary distribution and has been shown to have better mixing both theoretically [Niepert, 2012] and empirically [Niepert, 2012; Anand et al., 2017]. The key insight exploited by these algorithms is the fact that all the states in any given orbit have the same probability.

## 3 BLOCK-VALUE SYMMETRIES

In this section, we will present symmetries defined over blocks of variables, referred to as *BV Symmetries* which strictly generalize the earlier notions of symmetries defined over VV pairs. As a motivating example, Figure 1 shows two Graphical Models  $\mathcal{G}_1$  and  $\mathcal{G}_2$ . For ease of explanation these have been represented in terms of potential tables. These can easily be converted to the weighted feature representation, as defined previously. In  $\mathcal{G}_1$ , state  $(1, 0)$  has the same joint probability as  $(1, 1)$  and in  $\mathcal{G}_2$ , state  $(0, 0, 0, 0)$  has the same joint probability as  $(0, 1, 1, 1)$ . However, none of these can be captured by Variable or VV symmetries. We start with some definitions.

**Definition 5.** Let  $B = \{X_1, X_2, \dots, X_r\}$  denote a set of variables ( $X_i \in \mathcal{X}$ ) which we will refer to as a **block**. Similarly, let  $b = \{x_1, x_2, \dots, x_r\}$  denote a set of (cor-

responding) assignments to the variables in the block  $B$ . Then, we refer to the pair  $(B, b)$  as a **Block-Value (BV) pair**.

**Definition 6.** A BV pair  $(B, b)$  is said to be **consistent** with a state  $s$  if  $\forall X_i \in B, s(X_i) = x_i$  where  $x_i$  is the value for variable  $X_i$  in block  $B$ .

Let  $\Delta_V^r$  denote some subset of all possible BV pairs defined over blocks of size less than equal to  $r$ . For ease of notation, we will drop superscript  $r$  and denote  $\Delta_V^r$  as  $\Delta_V$  where  $r$  is a pre-specified constant for maximum block size. Then, we are interested in defining permutations over the elements of the set  $\Delta_V$ . Considering any set of block-value pairs in  $\Delta_V$  and allowing permutation among them may lead to inconsistent states. Consider a graphical model defined over four variables:  $\{X_1, X_2, X_3, X_4\}$ . Let us consider all possible blocks of size  $\leq 2$ . Then, a BV permutation permuting the singleton block  $\{X_1\}$  to itself (with identity mapping on values) while at the same time, permuting the block  $\{X_1, X_3\}$  to the block  $\{X_2, X_4\}$  is clearly inconsistent since  $X_1$ 's value can not be determined uniquely. A natural way to avoid this inconsistency is to restrict each variable to be a part of single block while applying permutations. Therefore, we restrict our attention to sets of blocks which are non overlapping.

**Definition 7.** Let  $\Delta = \{B_1, B_2, \dots, B_L\}$  denote a set of blocks. We define  $\Delta$  to be a **partition** if each variable  $X_i \in \mathcal{X}$  appears in exactly one block in  $\Delta$ . For a partition  $\Delta$ , we define the **block value set**  $\Delta_V$  as a set of BV pairs where each block  $B_l \in \Delta$  is present with all of its possible assignments.

We would now like to define permutations over the block value set  $\Delta_V$ , which we refer to as *BV-permutations*. To begin, we define the action of a BV-permutation  $\psi : \Delta_V \rightarrow \Delta_V$  on a state  $s$ . The action of a BV-permutation  $\psi : \Delta_V \rightarrow \Delta_V$  on a state  $s$  results in a state  $s' = \psi(s)$  such that  $\forall (B, b) \in \Delta_V, (B, b)$  is consistent with  $s$  if and only if  $\psi(B, b)$  is consistent with  $s'$

However, similar to the case of VV symmetries, any bijection from  $\Delta_V \rightarrow \Delta_V$  may not always result in a consistent state. For instance, consider a graphical model with 4 variables. Let the partition  $\Delta = \{(X_1, X_2), (X_3, X_4)\}$ . Consider the state  $s = (0, 1, 1, 0)$ . In case  $\psi$  is defined as  $\psi(\{X_1, X_2\}, \{0, 1\}) = (\{X_1, X_2\}, \{1, 0\})$  and  $\psi(\{X_3, X_4\}, \{1, 0\}) = (\{X_1, X_2\}, \{1, 1\})$ , the action of  $\psi$  results in an inconsistent state, since the action of  $\psi$  would result in a state with  $X_2$  equal to both 0 and 1 simultaneously. To address this issue, we define a BV-permutation to be valid only under certain conditions.

**Definition 8.** A BV-permutation  $\psi : \Delta_V \rightarrow \Delta_V$  is said

to be **valid** if  $\forall (B_i, b_i) \in \Delta_V, \psi(B_i, b_i) = (B_j, b_j) \Rightarrow \forall b'_i, \exists b'_j$  such that  $\psi(B_i, b'_i) = (B_j, b'_j)$

Intuitively a BV-permutation  $\psi$  is valid if it maps all assignments of a block  $B$  to assignments of a fixed block  $B'$ .

Presently, it is tempting to define a new graphical model where each block is a multi valued variable, with domain of this variable describing all of the possible assignments. This would be useful in a lucid exposition of symmetries. To do this we must suitably transform the set of features as well to this new set of variables. Given a block partition  $\Delta$ , we transform the set of features  $f_j$  such that for each block either all the variables in this block appear in the feature or none of them appear in the feature, while keeping all features logically invariant. We denote the set of all variables over which feature  $f_j$  is defined as  $\mathcal{V}(f_j)$ . Further, for a block  $B_l$  and a feature  $f_j$ , let  $\bar{B}_l = B_l - \mathcal{V}(f_j)$  i.e  $\bar{B}_l$  contains the additional variables in the block which are not part of feature  $f_j$ .

**Definition 9.** Given a variable  $X_i$ , which appears in a block  $B_l \in \Delta$  and a feature  $f_j$ , a **block consistent representation** of the feature, denoted by  $f'_j$ , is defined over the variables  $\mathcal{V}(f_j) \cup \bar{B}_l$ , such that,  $f'_j(\mathbf{x}_j, \bar{b}_l) = f_j(\mathbf{x}_j)$  where  $\mathbf{x}_j, \bar{b}_l$  denote an assignment to all the variables in  $\mathcal{V}(f_j)$  and  $\bar{B}_l$ , respectively.

For instance consider the feature  $f = (X_2)$ . Let the block  $B_l$  be  $\{(X_1, X_2)\}$ . Then the block consistent feature  $f'$  is given by  $f' = (X_1 \wedge X_2) \vee (\neg X_1 \wedge X_2)$ .

We extend the idea of block consistent representation to get a partition consistent representation  $\hat{f}_j$ .

**Definition 10.** A **partition consistent representation** of a feature  $f_j$ ,  $\hat{f}_j$  is defined by iteratively converting the feature  $f_j$  to its block consistent representation for each  $X_i \in \mathcal{V}(f_j)$ .

The set of partition consistent features  $\{(\hat{f}_j, w_j)\}_{j=1}^m$  has the property that for all  $B_l \in \Delta, B_l \subseteq \text{Var}(\hat{f}_j)$  or  $B_l \cap \text{Var}(\hat{f}_j) = \phi$ , i.e. all variables in each block either appear completely, or do not appear at all in any given feature. This property allows us to define a transformed graphical model  $\hat{\mathcal{G}}$  over a set of multi valued variables  $\mathcal{Y}$ , where each variable  $Y_l \in \mathcal{Y}$  represents a block  $B_l \in \Delta$ . The domain size of  $Y_l$  is the number of possible assignments of the variables in the block  $B_l$ . The set of features in this new model is simply the set of transformed features  $\{(\hat{f}_j, w_j)\}_{j=1}^m$ . As the blocks are non overlapping, such a transformation can always be carried out.

Since the transformation of features to partition consistent features always preserves logical equivalence, it seems natural to wonder about the relationship between

the graphical models  $\mathcal{G}$  and  $\hat{\mathcal{G}}$ . We first note that each state  $s$  in  $\mathcal{G}$  can be mapped to a unique state  $\hat{s}$  in  $\hat{\mathcal{G}}$  by simply iterating over all the blocks  $B_l \in \Delta$ , checking which BV pair  $(B_l, b_l)$  is consistent with the state  $s$  and assigning the appropriate value  $y_l$  to the corresponding variable  $Y_l$ . In a similar manner, each state  $\hat{s} \in \hat{\mathcal{G}}$  can be mapped to a unique state in  $s \in \mathcal{G}$ .

**Theorem 3.** *Let  $s$  denote a state in  $\mathcal{G}$  and let  $\hat{s}$  be the corresponding state in  $\hat{\mathcal{G}}$ . Then, this correspondence is probability preserving i.e.,  $\mathcal{P}(s) = \hat{\mathcal{P}}(\hat{s})$  where  $\mathcal{P}$  and  $\hat{\mathcal{P}}$  are the distributions defined by  $\mathcal{G}$  and  $\hat{\mathcal{G}}$ , respectively.*

Similar to the mapping between states, every BV-permutation  $\psi$  of  $\mathcal{G}$  corresponds to an equivalent VV-permutation  $\hat{\phi}$  of  $\hat{\mathcal{G}}$  obtained by replacing each BV pair in  $\mathcal{G}$  by the corresponding VV pair in  $\hat{\mathcal{G}}$  (and vice-versa). Since the distributions defined by the two graphical models are equivalent, we can define BV symmetries in  $\mathcal{G}$  as follows:

**Definition 11.** *Under a given partition  $\Delta$ , a BV-permutation  $\psi$  of a graphical model  $\mathcal{G}$  is a **BV-symmetry** of  $\mathcal{G}$  if the corresponding permutation  $\hat{\phi}$  under  $\hat{\mathcal{G}}$  is a VV-symmetry of  $\hat{\mathcal{G}}$ .*

We can now state the following results for BV-symmetries.

**Theorem 4.** *BV-symmetries are probability preserving transformations, i.e., for a BV-symmetry  $\psi$ ,  $\mathcal{P}(s) = \mathcal{P}(\psi(s))$  for all states  $s \in \mathcal{S}$ .*

It is easy to see that the set of all BV symmetries under a given partition  $\Delta$  form a group  $\Psi$ . Similar to the VV orbits, we define the BV orbit of a state  $s$  as  $\Gamma_\Psi(s) = \{\psi(s) | \psi \in \Psi\}$ .

When the partition  $\Delta$  is such that each variable appears in a block by itself, all the BV-symmetries are nothing but VV-symmetries.

**Theorem 5.** *Any VV-symmetry can be represented as a BV-symmetry for an appropriate choice of  $\Delta$ .*

### Computing BV Symmetries

Since BV symmetry on a graphical model  $\mathcal{G}$  is defined in terms of VV symmetry of a transformed graphical model  $\hat{\mathcal{G}}$ , BV symmetry can be trivially computed by constructing the transformed graphical model and then computing VV symmetry on  $\hat{\mathcal{G}}$  as described by Anand et al. [2017].

## 4 AGGREGATE ORBITAL MARKOV CHAINS

Given a block partition  $\Delta$ , BV symmetry group  $\Psi$  of  $\mathcal{G}$  can be found by computing VV symmetry group  $\Phi$  in the

auxiliary graphical model  $\hat{\mathcal{G}}$ . We further setup a Markov chain *BV-MCMC*( $\alpha$ ) over  $\Psi$  to exploit BV symmetries where  $\alpha \in [0, 1]$  is a parameter.

**Definition 12.** *Given a graphical model  $\mathcal{G}$ , a Markov chain  $\mathcal{M}$  and a BV symmetry group  $\Psi$ , one can define a **BV-MCMC**( $\alpha$ ) **Markov chain**  $\mathcal{M}'$  as follows: From the current sample  $s_t$*

- a) *Sample a state  $s'$  from original Markov chain  $\mathcal{M}$*
- b) i) *With probability  $\alpha$ , sample a state  $s_{t+1} = \Gamma_\Psi(s')$  uniformly from BV orbit of  $s'$  and return  $s_{t+1}$  as next sample.*
- ii) *With probability  $1 - \alpha$ , set state  $s_{t+1} = s'$  and return it as the next sample*

*BV-MCMC*( $\alpha$ ) Markov chain is defined similar to VV-MCMC except that it takes an orbital move only with probability  $\alpha$  instead of taking it always. For  $\alpha = 1$ , it is similar to VV-MCMC, and reduces to the original Markov chain  $\mathcal{M}$  for  $\alpha = 0$ . When  $\alpha = 1$ , sometimes, it is observed that the gain due to symmetries is overshadowed by the computational overhead of the orbital step. The parameter  $\alpha$  captures a compromise between these two contradictory effects.

**Theorem 6.** *Given a Graphical Model  $\mathcal{G}$ , if the original Markov chain  $\mathcal{M}$  is regular, then, *BV-MCMC*( $\alpha$ ) Markov chain  $\mathcal{M}'$ , constructed as above, is regular and converges to the unique stationary distribution of the original Markov chain  $\mathcal{M}$ .*

It should be noted that two different block partitions may capture different BV symmetries and hence may have different BV symmetry groups. In order to fully utilize all symmetries which may be present in multiple block partitions, we propose the idea of **Aggregate Orbital Markov Chain**.

Consider  $K$  different block partitions  $\Delta_1, \Delta_2, \dots, \Delta_K$ . We set up  $K$  independent *BV-MCMC*( $\alpha$ ) Markov chains, where each chain generates samples as per *BV-MCMC*( $\alpha$ ) corresponding to partition  $\Delta_k$ . Let these chains be  $\mathcal{M}'_1, \mathcal{M}'_2, \dots, \mathcal{M}'_K$ , and let the corresponding automorphism groups be  $\Psi_1, \Psi_2, \dots, \Psi_K$ . Given an intermediate state  $s'$ , we would like to sample uniformly from the union of orbits  $\bigcup_k \Psi_k(s')$ . Since these orbits may overlap with each other, sampling a state uniformly from the union of orbits is unclear. We circumvent this problem by setting up a new Markov chain, **Aggregate Orbital Markov Chain**. This Aggregate Orbital Markov Chain utilizes all available symmetries and converges to the true stationary distribution.

**Definition 13.** *Given  $K$  different *BV-MCMC*( $\alpha$ ) Markov chains,  $\mathcal{M}'_1, \mathcal{M}'_2, \dots, \mathcal{M}'_K$ , an **Aggregate Orbital Markov Chain**  $\mathcal{M}^*$  can be constructed in the following way: Starting from state  $s_t$  a) Sample a*

BV-MCMC( $\alpha$ ) Markov chain  $\mathcal{M}'_k$  uniformly from  $\mathcal{M}'_1, \mathcal{M}'_2, \dots, \mathcal{M}'_K$  b) Sample a state  $s_{t+1}$  according to  $\mathcal{M}'_k$ .

**Theorem 7.** *The aggregate orbital Markov chain  $\mathcal{M}^*$  constructed from  $K$  BV-MCMC( $\alpha$ ) Markov chains,  $\mathcal{M}'_1, \mathcal{M}'_2, \dots, \mathcal{M}'_K$ , all of which have stationary distribution  $\pi$ , is regular and converges to the same stationary distribution  $\pi$ .*

*Proof.* Given each of BV-MCMC( $\alpha$ ) Markov chains  $\mathcal{M}'_k$  are regular, firstly, we prove that the aggregate Markov chain is regular. In each step of aggregate chain, one of the BV-MCMC( $\alpha$ ) is applied and since, there is non-zero probability of returning to the same state in BV-MCMC( $\alpha$ ) chain, there is non-zero probability of returning to the same state in  $\mathcal{M}^*$ . Hence, aggregate chain so defined is regular and therefore, it converges to a unique stationary distribution. [Koller and Friedman, 2009].

The only fact that remains to be shown is that the stationary distribution of  $\mathcal{M}^*$  is  $\pi$ . Let  $T^*(s \rightarrow s')$  represent the transition probability of going from state  $s$  to  $s'$  in aggregate chain  $\mathcal{M}^*$ . We need to show that

$$\pi(s') = \sum_{s \in \mathcal{S}} \pi(s) * T^*(s \rightarrow s') \quad (1)$$

Let  $T_k(s \rightarrow s')$  represent the transition probability of going from state  $s$  to  $s'$  in  $\mathcal{M}'_k$

$$\sum_{s \in \mathcal{S}} \pi(s) * T^*(s \rightarrow s') = \sum_{s \in \mathcal{S}} \pi(s) * \frac{1}{K} * \sum_{k=1}^K T_k(s \rightarrow s') \quad (2)$$

$$= \frac{1}{K} \sum_{k=1}^K \sum_{s \in \mathcal{S}} \pi(s) * T_k(s \rightarrow s') = \frac{1}{K} \sum_{k=1}^K \pi(s') = \pi(s') \quad (3)$$

Equation 2 follows from the definition of aggregate chain while equation 3 holds since  $\mathcal{M}'_k$  converges to stationary distribution  $\pi$ .  $\square$

Aggregate Markov chain  $\mathcal{M}^*$  so obtained not only converges to the correct stationary distribution but also results in faster mixing since it can exploit the symmetries associated with each of the individual orbital Markov chains.

## 5 HEURISTICS FOR BLOCK PARTITIONS

We have so far computed BV symmetries given a specific block partition. We now discuss our heuristic that suggests candidate block partitions for downstream symme-

try computation (see supplementary material for pseudo-code). At a high level, our heuristic has the following two desiderata. Firstly, it ensures that there are no overlapping blocks, i.e., one variable is always in one block. Secondly, it guesses which blocks might exhibit BV-symmetries, and encourages such blocks in a partition.

The heuristic takes the hyperparameter  $r$ , the maximum size of a block, as an input. It considers only those blocks (upto size  $r$ ) in which for each variable in the block, there exists at least one other variable from the same block, such that some clause in  $\mathcal{G}$  contains both of them. This prunes away blocks in which variables do not directly interact with each other, and thus are unlikely to produce symmetries. Note that these candidate blocks can have overlapping variables and hence not all can be included in a block partition.

For these candidate blocks, for each block-value pair, the heuristic computes a weight signature. The weight signature is computed by multiplying weights of all the clauses that are made true by the specific block-value assignment. The heuristic then buckets all BV pairs of the same size based on their weight signatures. The cardinality of each bucket (i.e., the number of BV pairs of the same size that have the same weight signature) is calculated and stored.

The heuristic samples a block partition as follows. At each step it samples a bucket with probability proportional to its cardinality and once a bucket is selected, then it samples a block from that bucket uniformly at random, as long as the sampled block doesn't conflict with existing blocks in the current partition i.e., it has no variables in common with them. This process is repeated until all variables are included in the partition. In the degenerate case, if a variable can't be sampled from any block of size 2 or higher, then it gets sampled as an independent block of size 1. Once a partition is fully sampled, it is stored and the process is reset to generate another random block partition.

This heuristic encourages sampling of blocks that are part of a larger bucket in the hope that multiple blocks from the same bucket will likely yield BV symmetries in the downstream computation. At the same time, the non-conflicting condition and existence of single variable blocks jointly ensure that each sample is indeed a bona fide block partition.

## 6 EXPERIMENTS

Our experiments attempt to answer two key research questions. (1) Are there realistic domains where BV symmetries exist but VV symmetries do not? (2) For

Domain	Rules	Weights	Variables
Job Search	$\forall x \text{ TakesML}(x) \wedge \text{GetsJob}(x)$	$+w_1$	$\text{TakesML}(x),$ $\text{GetsJob}(x),$ $\text{Connected}(x,y)$
	$\forall x \neg \text{TakesML}(x) \wedge \text{GetsJob}(x)$	$+w_2$	
	$\forall (x,y) \text{ Connected}(x,y) \wedge \text{TakesML}(x) \Rightarrow \text{TakesML}(y)$	$w_3$	
Student Curriculum	$\forall x \text{ Maths}(x) \wedge \text{CS}(x)$	$+w_1$	$\text{Maths}(x)$ $\text{CS}(x)$
	$\forall x \text{ Maths}(x) \wedge \neg \text{CS}(x)$	$+w_2$	
	$\forall x \neg \text{Maths}(x) \wedge \text{CS}(x)$	$+w_3$	
	$\forall x \neg \text{Maths}(x) \wedge \neg \text{CS}(x)$	$+w_4$	
	$\forall (x,y) \in \text{Friends}, \text{Maths}(x) \Rightarrow \text{Maths}(y)$	$w$	
	$\forall (x,y) \in \text{Friends}, \text{CS}(x) \Rightarrow \text{CS}(y)$	$w$	

Table 1: Description of the two domains used in experiments. A weight of the form  $+w_1$  indicates that the weight is randomly sampled for each object.

such domains, how much faster can an MCMC chain mix when using BV symmetries compared to when using VV symmetries or not using any symmetries?

## 6.1 Domains

To answer the first question, we construct two domains. The first domain models the effect of an academic course on an individual’s employability, whereas the second domain models the choices a student makes in completing their course credits. Both domains additionally model the effect of one’s social network in these settings. Table 1 specifies the weighted first order formulas for both the domains.

**Job Search:** In this domain, there are  $N$  people on a social network, looking for a job. Given the AI hype these days, their employability is directly linked with whether they have learned machine learning (ML) or not. Each person  $x$  has an option of taking the ML course, which is denoted by  $\text{TakesML}(x)$ . Furthermore, the variable  $\text{Connected}(x,y)$  denotes whether two people  $x$  and  $y$  are connected in the social network or not. Finally, the variable  $\text{GetsJob}(x)$  denotes whether  $x$  gets employment or not.

In this Markov Logic Network (MLN) [Domingos and Lowd, 2009], each person  $x$  participates in three kinds of formulas. The first one with weight  $w_1$  indicates the (un-normalized) probability of the person getting a job and taking the ML course ( $\text{TakesML}(x) \wedge \text{GetsJob}(x)$ ). The second formula with weight  $w_2$  indicates the chance of the person getting a job while not taking the course ( $\neg \text{TakesML}(x) \wedge \text{GetsJob}(x)$ ). Our domain assigns different weights  $w_1$  and  $w_2$  for each person, modeling the fact that each person may have a different capacity to learn ML, and that other factors may also determine whether they get a job or not. Finally,  $x$  is more likely to take the course if their friends take the course. This is modeled by an additional formula for each pair  $(x,y)$ , with a fixed weight  $w_3$ .

In this domain, there are hardly any VV symmetries, since every  $x$  will likely have different weights. However there are *intra-block* BV symmetries for the block  $(\text{TakesML}(x), \text{GetsJob}(x))$  for every  $x$ . This is because within the potential table of this block the block values  $(0, 0)$  and  $(1, 0)$  are symmetric and can be permuted.

**Student Curriculum:** In this domain, there are  $N$  students who need to register for two courses, one from Mathematics and one from Computer Science to complete their course credits. There are two courses (basic or advanced) on offer in both disciplines. Variables  $\text{Math}(x)$  and  $\text{CS}(x)$  denote whether the student  $x$  would take the advanced course in each discipline. Since courses for Mathematics and CS could be related, each student needs to give a joint preference amongst the 4 available options. This is modeled as a potential table over  $(\text{Math}(x), \text{CS}(x))$  with weights chosen randomly from a fixed set of parameters. Further, some students may also be friends. Since students are more likely to register in courses with their friends, we model this as an additional formula, which increases the probability of registering for a course in case a friend registers for the same.

In this domain, VV pairs can only capture symmetries when the potential tables (over  $\text{Math}$  and  $\text{CS}$ ) for two students are exactly the same. However, there are a lot more *inter-block* BV symmetries since it is more likely to find pairs of students, whose potential tables use the same set of weights, but in a different order.

## 6.2 Comparison of MCMC Convergence

We now answer our second research question by comparing the convergence of three Markov chains – Vanilla-MCMC, VV-MCMC, and BV-MCMC( $\alpha$ ). All three use Gibbs sampling as the base MCMC chain. All experiments are done on Intel Core i7 machines. Following previous work, and for fair comparison, we implement



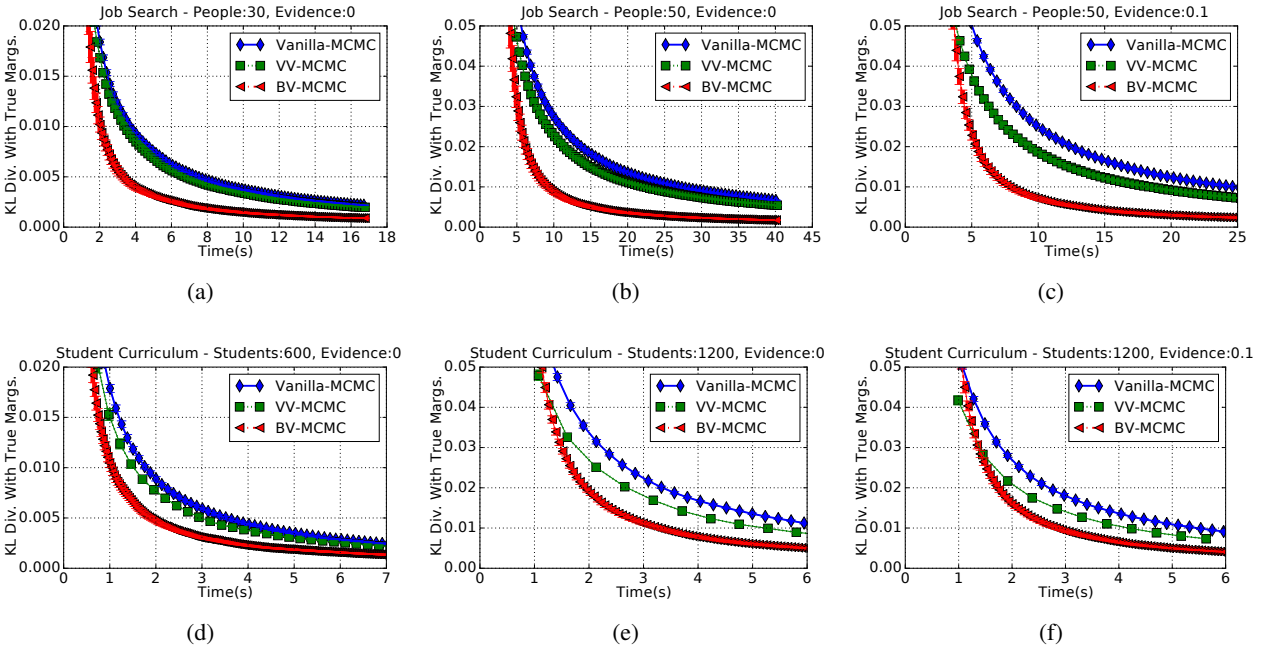


Figure 2: BV-MCMC( $\alpha = 1$ ) and BV-MCMC( $\alpha = 0.02$ ) outperforms VV-MCMC and Vanilla MCMC on Job Search and Student Curriculum domains respectively with different size and evidence variations

all the three Markov chains in group theoretic package - GAP [GAP, 2015]. This allows the use of off-the-shelf group theoretic operations. The code for generating candidate lists is written in C++. We solve graph isomorphism problems using the Saucy software [Darga *et al.*, 2008]. We release our implementation for future use by the community <sup>1</sup>.

In all experiments, we keep the maximum block size in a block partition to be two. For each chain we plot the KL divergence of true marginals and computed marginals for different runtimes. We estimate true marginals by running the Gibbs sampling algorithm for a sufficiently long period of time. Each algorithm is run 20 times to compute error bars indicating 95% confidence interval.

For VV-MCMC and BV-MCMC, the run time on x-axis includes the pre-processing time of computing symmetries as well. For BV-MCMC, this includes the time for generating candidate lists, running Saucy for each candidate list, and initializing the Product Replacement algorithm for each candidate lists. The total preprocessing time for Job Search domain is around 1.6 sec and for Student Curriculum domain is around 0.6 sec.

Figure 2 shows that BV-MCMC substantially outperforms VV-MCMC and Vanilla-MCMC in both the domains. The parameter  $\alpha$  is set to 1.0 for Job Search Do-

main and 0.02 for Student Curriculum Domain. Since these domains do not have many VV-Symmetries, VV-MCMC only marginally outperforms Vanilla MCMC. On the other hand BV-MCMC is able to exploit a considerably larger number of symmetries and leads to faster mixing. BV-MCMC scales well with domain size, significantly outperforming other algorithms as domain size is changed from 30 to 50 people in Job Search and 600 to 1200 in Student Curriculum domain. This is particularly due to more symmetries being captured by BV-MCMC for larger domain sizes. <sup>2</sup>

Figure 2(c) and 2(f) plot the variation with introduction of 10% evidence in each domain. BV MCMC still outperforms VV-MCMC and Vanilla-MCMC and is robust to presence of evidence.

Finally, we also test the sensitivity of BV-MCMC with the  $\alpha$  parameter. Figure 3 plots this variation on both these domains. We find that for Job Search, a high value  $\alpha = 1$  performs the best, whereas a lower value is better in Student Curriculum. This is because Job Search mostly has intra-block BV symmetries, which can be computed and applied efficiently. This makes sampling an orbital step rather efficient. On the other hand, for Student Curriculum, the inter-block symmetry between different pairs of people makes the orbital step costlier, and reducing the fraction of times an orbital move is taken

<sup>1</sup><https://github.com/dair-iitd/bv-mcmc>

<sup>2</sup>Most of the error-bars are negligible in size.

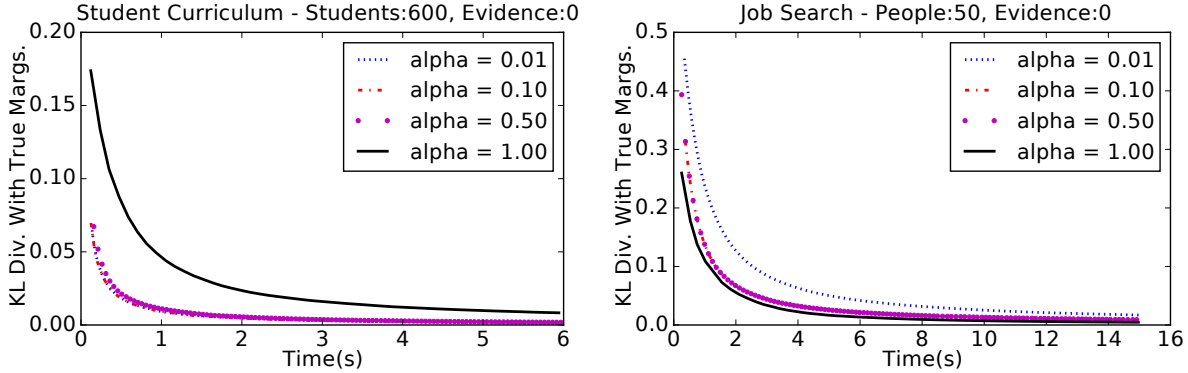


Figure 3: Variation on  $\alpha$   $\alpha < 1$  is significantly better than  $\alpha = 1$  in Student-Curriculum domain while  $\alpha = 1$  is best in Job-Search domains

improves the overall performance.

## 7 CONCLUSIONS

Permutations defined over variables or variable-value (VV) pairs miss a significant fraction of state symmetries. We define permutations over block-value (BV) pairs, which enable a subset of variables (block) and their assignment to jointly permute to another subset. This representation is exponential in the size of the maximum block  $r$ , but captures more and more state symmetries with increasing  $r$ .

Novel challenges arise when building the framework and algorithms for BV permutations. First, we recognize that all BV permutations do not lead to valid state symmetries. For soundness, we impose a sufficient condition that each BV permutation must be defined on blocks with non-overlapping variables. Second, to compute BV symmetries, we describe a graph-isomorphism based solution. But, this solution expects a block partition as an input, and we cannot run it over all possible block partitions as they are exponential in number. In response, we provide a heuristic that outputs candidate block partitions, which will likely lead to BV symmetries. Finally, since the orbits from different block partitions may have overlapping variables, they cannot be explicitly composed in compact form. This makes it difficult to uniformly sample from the aggregate orbit (aggregated over all block partitions). To solve this challenge, we modify the Orbital MCMC algorithm so that in the orbital step, it uniformly samples from the orbit from any one of the block partitions (BV-MCMC). We prove that this aggregate Markov chain also converges to the true posterior.

Our experiments show that there exist domains in which BV symmetries exist but VV symmetries may not. We find that BV-MCMC mixes much more rapidly than base

MCMC or VV-MCMC, due to the additional mixing from orbital BV moves. Overall, our work provides a unified representation for existing research on permutation groups for state symmetries. In the future, we wish to extend this notion to approximate symmetries, so that they can be helpful in many more realistic domains as done in earlier works [Habeeb *et al.*, 2017].

## ACKNOWLEDGEMENTS

We thank anonymous reviewers for their comments and suggestions and Happy Mittal for useful discussions. Ankit Anand is supported by the TCS Fellowship. Mausam is supported by grants from Google and Bloomberg. Parag Singla is supported by the DARPA Explainable Artificial Intelligence (XAI) Program with number N66001-17-2-4032. Both Mausam and Parag Singla are supported by the Visvesvaraya Young Faculty Fellowships by Govt. of India and IBM SUR awards. Any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views or official policies, either expressed or implied, of the funding agencies.

## References

- [Anand *et al.*, 2016] A. Anand, A. Grover, Mausam, and P. Singla. Contextual Symmetries in Probabilistic Graphical Models. In *IJCAI*, 2016.
- [Anand *et al.*, 2017] A. Anand, R. Noothigattu, P. Singla, and Mausam. Non-Count Symmetries in Boolean & Multi-Valued Prob. Graphical Models. In *AISTATS*, 2017.
- [Bui *et al.*, 2013] H. Bui, T. Huynh, and S. Riedel. Automorphism groups of graphical models and lifted variational inference. In *UAI*, 2013.
- [Celler *et al.*, 1995] F. Celler, C. R. Leedham-Green, S. H. Murray, A. C Niemeyer, and E. A O’Brien. Generating random elements of a finite group. *Communications in algebra*, 23(13):4931–4948, 1995.
- [Darga *et al.*, 2008] P. T. Darga, K. A. Sakallah, and I. L. Markov. Faster Symmetry Discovery using Sparsity of Symmetries. In *Design Automation Conference*, 2008.
- [de Salvo Braz *et al.*, 2005] R. de Salvo Braz, E. Amir, and D. Roth. Lifted First-Order Probabilistic Inference. In *IJCAI*, 2005.
- [Domingos and Lowd, 2009] P. Domingos and D. Lowd. *Markov Logic: An Interface Layer for Artificial Intelligence*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2009.
- [GAP, 2015] The GAP Group. *GAP – Groups, Algorithms, and Programming, Version 4.7.9*, 2015.
- [Gogate and Domingos, 2011] V. Gogate and P. Domingos. Probabilistic Theorem Proving. In *UAI*, 2011.
- [Habeeb *et al.*, 2017] Haroun Habeeb, Ankit Anand, Mausam Mausam, and Parag Singla. Coarse-to-fine Lifted MAP Inference in Computer Vision. In *IJCAI*, 2017.
- [Kersting *et al.*, 2009] K. Kersting, B. Ahmadi, and S. Natarajan. Counting Belief Propagation. In *UAI*, 2009.
- [Kimmig *et al.*, 2015] A. Kimmig, L. Mihalkova, and L. Getoor. Lifted Graphical Models: A Survey. *Machine Learning*, 99(1):1–45, 2015.
- [Koller and Friedman, 2009] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.
- [Mladenov *et al.*, 2012] M. Mladenov, B. Ahmadi, and K. Kersting. Lifted Linear Programming. In *AISTATS*, 2012.
- [Niepert and den Broeck, 2014] Mathias Niepert and Guy Van den Broeck. Tractability through Exchangeability: A New Perspective on Efficient Probabilistic Inference. In *AAAI*, 2014.
- [Niepert, 2012] M. Niepert. Markov Chains on Orbits of Permutation Groups. In *UAI*, 2012.
- [Pak, 2000] I. Pak. The Product Replacement Algorithm is Polynomial. In *Foundations of Computer Science*, 2000.
- [Poole, 2003] D. Poole. First-Order Probabilistic Inference. In *IJCAI*, 2003.
- [Singla and Domingos, 2008] P. Singla and P. Domingos. Lifted First-Order Belief Propagation. In *AAAI*, 2008.
- [Van den Broeck and Niepert, 2015] G. Van den Broeck and M. Niepert. Lifted Probabilistic Inference for Asymmetric Graphical Models. In *AAAI*, 2015.
- [Venugopal and Gogate, 2012] D. Venugopal and V. Gogate. On Lifting the Gibbs Sampling Algorithm. In *NIPS*, 2012.

---

# Max-margin learning with the Bayes factor

---

Rahul G. Krishnan  
MIT

Arjun Khandelwal  
MIT

Rajesh Ranganath  
NYU

David Sontag  
MIT

## Abstract

We propose a new way to answer probabilistic queries that span multiple datapoints. We formalize reasoning about the similarity of different datapoints as the evaluation of the Bayes Factor within a hierarchical deep generative model that enforces a separation between the latent variables used for representation learning and those used for reasoning. Under this model, we derive an intuitive estimator for the Bayes Factor that represents similarity as the amount of overlap in representation space shared by different points. The estimator we derive relies on a query-conditional latent reasoning network, that parameterizes a distribution over the latent space of the deep generative model. The latent reasoning network is trained to amortize the posterior-predictive distribution under a hierarchical model using supervised data and a max-margin learning algorithm. We explore how the model may be used to focus the data variations captured in the latent space of the deep generative model and how this may be used to build new algorithms for few-shot learning.

## 1 INTRODUCTION

How do we frame the problem of selecting, from a target set, an object most similar to a given query set? For example—given a red chair, a blue chair and a black chair, we would rank chairs in the target set highly. At the same time, given a red chair, a red car and a red shirt, we would rank red objects highly. Between the two tasks, our understanding of the *data* has not changed; what has changed is our understanding of the *task based on the context* given by the query. The query highlights the relevant property of the data that is needed for solving a specific task. Such

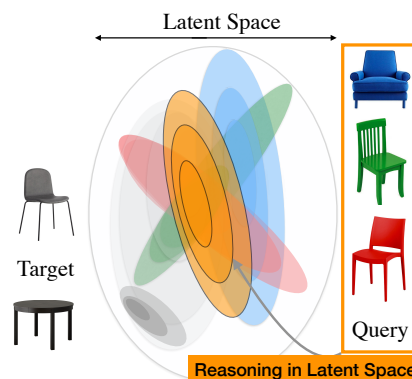


Figure 1: **Comparing objects in representational space:** On the left is a target set that will be ranked based on similarity to the query  $Q$  (right). The colour of each object is matched to a distribution in representation space. In orange is the output of the *latent reasoning network* – it represents the common factor of variation shared by  $Q$ . The black chair should rank higher than the black table; here its distribution (in representation space) overlaps more with the output of the latent reasoning network.

tasks appear in few-shot learning, where the goal is ranking objects according to their similarity to a given query set and in healthcare where a task may be finding similar patients to a given cohort.

To answer such queries, we could train discriminative models attuned to answering set-conditional queries at test time (e.g. Vinyals *et al.* (2016)). Or we could encode class separability in the structure of a generative model (Edwards & Storkey, 2017) and use inference for prediction. We take a different approach to the problem.

We learn a generic representation space (using unsupervised data) that is warped (using supervised data) for potentially different test-time problems. The task of scoring objects given a query is decomposed into two subtasks. The first determines the common property shared by items in the query set and represents the property as a region in representation space. In Figure 1, we visualize such

a hypothetical space. On the right is a query comprising chairs of different colors and (in orange) a region of space that characterizes the property (in this case, a likeness to a chair) common to items in the query. The second task is to score a target item based on how much it expresses the region of representation space shared by items in the query. For the two candidate target points in Figure 1 (left), the black chair would rank highly since its representation has more in common with the property encapsulated by the query.

Here, we will use the latent space of deep generative models (Rezende *et al.*, 2014; Kingma & Welling, 2014) as our representation space. In such models, one can do posterior inference to map from raw data to a distribution in latent space. Then, to find commonalities among query items, we introduce a *latent reasoning network* (LRN). The LRN takes a query as input and constructs a probability distribution over the latent space that *summarizes* the representations of the query points into a single distribution. Figure 1 (orange) depicts what the output of the LRN might look like. We design a neural architecture for the LRN based on Zaheer *et al.* (2017) so that it does not depend on the size of the query set. To score the latent space of a target item, we propose using the logarithm of the Bayes Factor (Jeffreys, 1998). The Bayes Factor measures how conditioning on the query alters the likelihood of a target point. Our approach is inspired by Bayesian Sets Ghahramani & Heller (2005) where data was assumed to be modeled by a hierarchical exponential family distribution and the likelihood ratio of the joint distribution and product of marginals was shown to be a useful measure of similarity.

The latent (representation) space of a deep generative models learned with unsupervised data is typically non-identifiable. i.e. there will exist multiple good (from the perspective of log-likelihood) representation spaces. Each corresponds to a different notion of similarity and a different way of grouping points. However, queries provide extra information: they reveal which points should be close together in latent space. We take advantage of this and propose a supervised max-margin learning algorithm for the LRN such that scores given to items in the query are larger than scores unrelated to the query.

We obtain a coupled set of models: in which one model is a deep generative model of the data whilst the other reshapes the latent space of the first and serves to answer queries about similarity judgements between datapoints. We study how the proposed approach can tune the latent space of deep generative models and be used to build new types of models for few-shot learning. We begin in Section 2 by motivating the Bayes Factor as a viable tool for computing similarity.

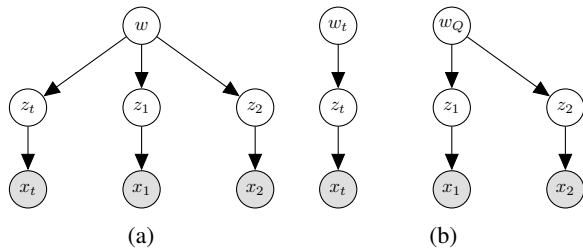


Figure 2: **Hypothesis testing with Deep Generative Models:** (a) The **Reasoning Model**, here, depicting the hypothesis that the set  $\{x_t, Q = \{x_1, x_2\}\}$  was generated jointly; (b) the two figures represent the hypothesis that  $x_t$  and  $Q$  were generated independently under different realizations of  $w$  (the random variable that captures the property shared across datapoints).

## 2 FROM REPRESENTATION LEARNING TO REASONING

Here, we consider the problem of scoring elements in a set based on how similar they are to a given query. Suppose we are given a dataset  $\mathcal{D} = \{x_1, \dots, x_N\}$ ,  $x_i \in \mathbb{R}^n$ ,  $x_i \in \mathcal{D}$ . Then for a query  $Q = \{x_1, \dots, x_Q\}$ ;  $|Q| = Q$ , we wish to assign to each  $x_t \in \mathcal{D}$  a score  $\text{score}(x_t, Q)$  that denotes how *similar*  $x_t$  is to elements of the query  $Q$ .

### 2.1 The Data Model

A simple way to quantify how similar objects are (here, between  $Q$  and  $x_t$ ) might be to take the pairwise Euclidian distance between them. For complex, high dimensional data that do not lie on a Euclidian manifold, such a metric may fail to capture interesting regularity between data.

Alternatively, we can use a latent variable model to construct a representation of data. The latent variable then becomes a low-dimensional sufficient statistic the raw data when quantifying similarity. The simplest latent variable model we will consider has the following generative process:  $z \sim p_{\text{dm}}(z)$ ;  $x \sim p_{\text{dm}}(x; f(z; \theta))$  where  $p_{\text{dm}}(z)$  is a simple distribution such as  $\mathcal{N}(0, I)$ . The use of MLPs in the conditional distributions allow the model to fit highly complex data despite the use of a simple prior. When  $f$  is parameterized by a Multi-Layer Perceptron (MLP), the resulting model is a deep generative model. We will refer to this model (Kingma & Welling, 2014; Rezende *et al.*, 2014) as the **Data Model** (with probabilities denoted with subscript  $\text{dm}$ ).

The generative process assumes datapoints are drawn independently. Using variational inference with an inference network (Hinton *et al.*, 1995) to approximate the posterior distribution,  $p_{\text{rm}}(z|x)$ , the model can be learned by maximizing a lower bound on the log-likelihood of the

data obtained using Jensen’s inequality:

$$\log p_{\text{dm}}(x; \theta) \geq \mathbb{E}_{q_{\text{dm}}(z|x; \phi)} [\log p_{\text{dm}}(x|z; \theta)] - \text{KL}(q_{\text{dm}}(z|x; \phi) || p_{\text{dm}}(z)) = \mathcal{L}(x; \theta, \phi), \quad (1)$$

With a Gaussian distribution as the variational approximation:  $q_{\text{dm}}(z|x; \phi) \sim \mathcal{N}(\mu_{\phi}(x), \Sigma_{\phi}(x))$  where  $\mu_{\phi}(x), \Sigma_{\phi}(x)$  are (differentiable, parametric, with parameters  $\phi$ ) functions of the observation  $x$ . Eq. 1 is differentiable in  $\theta, \phi$  (Kingma & Welling, 2014; Rezende *et al.*, 2014) and the model parameters  $(\theta, \phi)$  can be learned via gradient ascent on  $\mathcal{L}(x; \theta, \phi)$ .

With the variational approximation,  $q_{\text{dm}}(z|x; \phi)$ , to map from data to latent space, would computing overlap in the posterior distributions of points in  $\mathcal{Q}$  and  $x_t$  suffice to identify similar points? The answer is *sometimes*. While unsupervised learning will tend to put *similar* points together, the notion of similarity encoded in the latent space need not correspond to the notion of similarity required for a task at test time. We require a way to guide the structure of the latent space to be better suited for a task.

## 2.2 The Reasoning Model

Introducing hierarchy into the generative process is one way to guide the structure of latent variables. In Figure 2 (b) is a simple hierarchical model that makes explicit the insight that similar datapoints should have similar latent spaces. It defines the following generative process for a set of similar objects  $\mathcal{Q}$ :  $p_{\text{rm}}(\mathcal{Q}) = \int_w \int_z p_{\text{rm}}(w) \prod_{q=1}^Q p_{\text{rm}}(z_q|w) p_{\text{rm}}(x_q|z_q)$ . The random variable  $w$  defines the context of  $\mathcal{Q}$ . It may denote the label or class identity of points in  $\mathcal{Q}$  but more broadly is a representation of the properties that points in  $\mathcal{Q}$  satisfy. For notational convenience and because we can express reasoning about similarity as a probabilistic query in this model, we refer to it as the **Reasoning Model**.

The Neural Statistician (Edwards & Storkey, 2017) uses  $\text{KL}(p(w|x_t) || p(w|\mathcal{Q}))$  to quantify the similarity between  $x_t$  and  $\mathcal{Q}$  in a model similar to the one in Figure 2 (b). In this work, we pose the estimation of similarity between objects as hypothesis testing in a hierarchical deep generative model. The conditional independences in Figure 2 (b) enforce that  $x_t$  is independent of  $w$  given  $z_t$ , i.e. the per-data-point latent variables serve as a sufficient statistic to quantify comparisons between multiple datapoints. The conditional density  $p(x_t|z_t)$  is a map from the representation space to the data while  $p(z_t|w)$  dictates how the latent space of a datapoint behaves as a function of property encoded in  $w$ .

## 2.3 Bayes Factor

To score the similarity between two objects (in this case  $x_t$  and set  $\mathcal{Q}$ ) under the Reasoning Model, we turn to the likelihood ratio between the joint distribution of  $x_t$  and  $\mathcal{Q}$  and the product of their marginals. If  $x_t$  and  $\mathcal{Q}$  are drawn from the same joint distribution, then there exists a random variable  $w$  that governs the distribution of the *latent* spaces  $z_t, z_1, \dots, z_Q$ . With slight abuse of notation <sup>1</sup>, Figure 2 (a) depicts this scenario when  $\mathcal{Q} = \{x_1, x_2\}$ . If  $x_t$  and  $\mathcal{Q}$  are not similar, then their latent spaces will have different distributions, and they are children of different realizations of  $w$  (see Figure 2 (b)). With that in mind, the score function we use to measure similarity is given by (**Bayes Factor**):

$$\frac{p(x_t, \mathcal{Q})}{p(x_t)p(\mathcal{Q})} = \frac{p(x_t|\mathcal{Q})}{p(x_t)} = \text{score}(x_t, \mathcal{Q}) \quad (2)$$

The log-score is the pointwise mutual information (Fano, 1949), a measure of association that is frequently used in applications such as natural language processing (Church & Hanks, 1990). The Bayes Factor normalizes the posterior predictive density of the target point conditioned on the query by the target’s marginal likelihood under the model. It also has an information theoretic interpretation. Letting  $h(x) = -\log p(x)$  denote the self-information (or surprisal), then  $\log \text{score}(x_t, \mathcal{Q}) = h(x_t) - h(x_t|\mathcal{Q})$  intuitively denotes the surprise (quantified in nats or bits) from observing  $x_t$  when having already observed  $\mathcal{Q}$ .

**Similarity in Latent Space:** Equation 2 captures an intuitive notion of similarity but evaluating  $p(x_t)$ , the marginal density of the target, is typically intractable (except in hierarchical models that lie in the exponential family (Ghahramani & Heller, 2005)). Furthermore, an importance sampling based Monte-Carlo estimator for  $p(x_t)$  will involve a high-dimensional integral in the data  $x_t$ . We therefore propose the following decomposition of the score function that evaluates the Bayes Factor in the target datapoint’s (lower dimensional) latent space:

$$\begin{aligned} \frac{p_{\text{rm}}(x_t|\mathcal{Q})}{p_{\text{rm}}(x_t)} &= \frac{1}{p_{\text{rm}}(x_t)} \int_{z_t} p_{\text{rm}}(x_t, z_t|\mathcal{Q}) \\ &= \frac{1}{p_{\text{rm}}(x_t)} \int_{z_t} p_{\text{rm}}(x_t|z_t) p_{\text{rm}}(z_t|\mathcal{Q}) \\ &= \frac{1}{p_{\text{rm}}(x_t)} \int_{z_t} \frac{p_{\text{rm}}(z_t|x_t) p_{\text{rm}}(x_t)}{p_{\text{rm}}(z_t)} p_{\text{rm}}(z_t|\mathcal{Q}) \\ &= \int_{z_t} \underbrace{\frac{p_{\text{rm}}(z_t|x_t)}{p_{\text{rm}}(z_t)}}_{\text{Relative Posterior Likelihood}} \underbrace{p_{\text{rm}}(z_t|\mathcal{Q})}_{\text{Latent Reasoning Network}} \end{aligned} \quad (3)$$

<sup>1</sup>We re-use Figure 2 to denote both the instantiation of a hypothesis and the generative process

The estimator above formalizes the intuition for comparing points laid out in Section 1. The query-conditional posterior-predictive density over the latent space of the target datapoint,  $p_{\text{rm}}(z_t|\mathcal{Q})$ , reasons about points in the query and represents them as a density in latent space, The **Relative Posterior Likelihood**,  $\frac{p_{\text{rm}}(z_t|x_t)}{p_{\text{rm}}(z_t)}$  scores how likely the target point is to have come from the relevant part of latent space.

### 3 HIERARCHICAL MODELS WITH COMPOUND PRIORS

To compute the ratio  $\frac{p_{\text{rm}}(z_t|x_t)}{p_{\text{rm}}(z_t)}$ , we need to marginalize  $w_t$ . However, under certain assumptions about the conditional distributions in the **Reasoning Model**, we will see that approximating this ratio becomes simpler.

*Assumption 1.* Priors with Compound Distributions

$$\int_w p_{\text{rm}}(w)p_{\text{rm}}(z|w)dw = p_{\text{dm}}(z)$$

*Assumption 2.* Matching conditional likelihoods

$$p_{\text{rm}}(x|z) = p_{\text{dm}}(x|z)$$

**Lemma 1.** *Matching posterior marginals*

$$p_{\text{dm}}(z|x) = p_{\text{rm}}(z|x)$$

*Proof.* Follows from Bayes rule and Assumption 1, 2.  $\square$

**Lemma 2.** *Matching marginal likelihoods*

*Under Assumption 1 and 2:*

$$p_{\text{dm}}(x) = p_{\text{rm}}(x)$$

*Proof.*

$$\begin{aligned} p_{\text{rm}}(x) &= \int_w \int_z p_{\text{rm}}(w)p_{\text{rm}}(z|w)p_{\text{rm}}(x|z)]dzdw \\ &= \int_z p_{\text{dm}}(z)p_{\text{dm}}(x|z)dz = p_{\text{dm}}(x) \end{aligned}$$

$\square$

The conditions above state when we can take an instance of the **Data Model** discussed in Section 2.1 and transform it into an instance of the **Reasoning Model** in Section 2.2 while preserving the marginal likelihood of the data.

This transformation has a few implications. The first is when evaluating the Bayes Factor; if we work in a class of **Reasoning Models** that satisfy Assumption 1, then we can evaluate the **Relative Posterior Likelihood** using

the prior and posterior distribution of the associated **Data Model**. With Lemma 1 and Assumption 1:

$$\frac{p_{\text{rm}}(x_t|\mathcal{Q})}{p_{\text{rm}}(x_t)} = \int_{z_t} \underbrace{\frac{p_{\text{dm}}(z_t|x_t)}{p_{\text{dm}}(z_t)}}_{\text{Relative Posterior Likelihood}} \underbrace{p_{\text{rm}}(z_t|\mathcal{Q})}_{\text{Latent Reasoning Network}}$$

where  $p_{\text{dm}}(z_t)$  is typically fixed ahead of time (e.g.  $\mathcal{N}(0; \mathbb{I})$ ) and we can do inference for  $p_{\text{dm}}(z_t|x_t)$  (or approximate it using the inference network  $q_{\text{dm}}(z|x; \phi)$ ).

The second implication is that part of the **Reasoning Model**,  $p_{\text{rm}}(x|z)$ , can be learned ahead of time. This gives us the flexibility to *warm-start* the **Reasoning Model** using a *pre-trained Data Model* whose  $p_{\text{dm}}(z)$  can be expressed according to Assumption 1. In this way, even if we do not know which property will be used to organize datapoints into sets at test time, we can still learn a generic low-dimensional representation of the dataset. We will make use of this when we discuss the learning framework in Section 5. For now, what remains is how we can specify  $p_{\text{rm}}(w)$ ,  $p_{\text{rm}}(z|w)$  in order to evaluate  $p_{\text{rm}}(z_t|\mathcal{Q})$ .

### 4 LATENT REASONING NETWORKS

Although  $p_{\text{rm}}(z_t|\mathcal{Q}) = \int_w p_{\text{rm}}(z_t|w)p_{\text{rm}}(w|\mathcal{Q})dw$ , finding  $p_{\text{rm}}(w)$ ,  $p_{\text{rm}}(z|w)$  that satisfy Assumption 1 may prove challenging and so we will make use of another computational trick. To evaluate the Bayes Factor we only need a way to sample from  $p_{\text{rm}}(z_t|\mathcal{Q})$  i.e. the posterior predictive distribution given the query, of the target’s latent representation. Our strategy therefore, will instead be to parameterize and learn  $p_{\text{rm}}(z_t|\mathcal{Q})$  directly from data.

Without  $p_{\text{rm}}(w)$ ,  $p_{\text{rm}}(z|w)$ , we lose the ability to sample from the Reasoning Model but by amortizing  $p_{\text{rm}}(z_t|\mathcal{Q})$  we obtain a fast way to evaluate the Bayes Factor at test time.  $p_{\text{rm}}(z_t|\mathcal{Q})$  must *reason* about how the latent spaces of points in  $\mathcal{Q}$  are related and parameterize a distribution over the latent space of the target datapoint  $x_t$ ; this distribution must characterize the property represented by points in  $\mathcal{Q}$ . Therefore, we refer to this amortized, parameteric posterior-predictive distribution as a *Latent Reasoning Network*. Since we do not know the functional form of this distribution we will parameterize it as a non-linear function of the query  $\mathcal{Q}$ .

To construct the LRN, we require neural architectures capable of operating over sets. We make use of two primitives for such neural architectures proposed by Zaheer *et al.* (2017). These functions operate over sets of vectors  $\mathcal{Q} = \{x_1, \dots, x_Q\}$ ,  $x_q \in \mathbb{R}^n$ . We will use the notation  $\mathbb{R}^{n \times |\mathcal{Q}|}$  to denote a set of size  $|\mathcal{Q}|$  where each element is an  $n$ -dimensional vector. We design the LRN, with the following three properties:

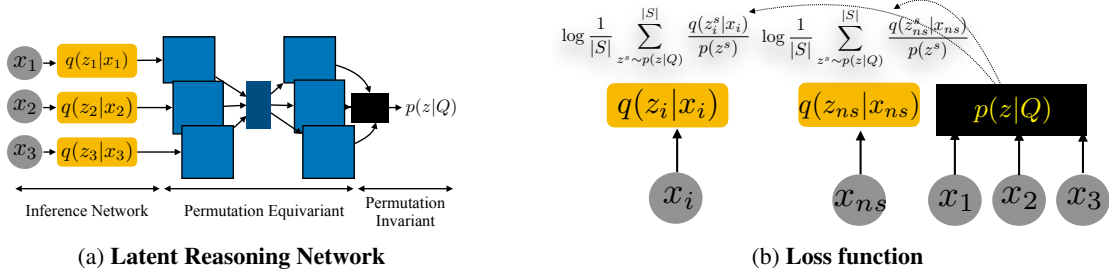


Figure 3: **Latent Reasoning Networks (LRN) and Loss function:** On the left is a diagrammatic representation of  $p_{\text{rm}}(z_t|\mathcal{Q})$ . On the right is a depiction of Monte-Carlo sampling (with samples from the LRN) to evaluate Bayes factor.  $x_i$  is a point similar to those in the query  $\mathcal{Q} = \{x_1, x_2, x_3\}$ , while  $x_{ns}$  is not. We suppress subscripts in the figure.

**A) Parameter Sharing:** We share parameters between the inference network of the Data Model and the LRN. A direct consequence of this choice is that the LRN now has the ability to change the way inference is done in the Data model. The first stage of the LRN uses the inference network of the Data Model to map from the set  $\mathcal{Q}$  to a set of each point’s variational parameters

**B) Exchangeability:** The output of the LRN must not depend on the order of elements in  $\mathcal{Q}$ . We achieve this by using the functions proposed by (Zaheer *et al.*, 2017):  $g : \mathbb{R}^{n \times |\mathcal{Q}|} \rightarrow \mathbb{R}^{m \times |\mathcal{Q}|}$  is a permutation equivariant function that maps from sets of  $n$  dimensional vectors to sets of  $m$  dimensional vectors while ensuring that if the input elements were permuted, then the output elements would also be permuted identically. The form of  $g$  is given by  $g(\mathcal{Q}) = \left[ \rho \left( W_1^{\text{eq}} x_q + W_2^{\text{eq}} (\sum_{q'} x_{q'}) \right) \right]_{q=1}^{|\mathcal{Q}|}$  where  $W_1^{\text{eq}} \in \mathbb{R}^{m \times n}$ ,  $W_2^{\text{eq}} \in \mathbb{R}^{m \times n}$  and  $\rho$  is an element-wise nonlinearity. We use compositions of the function  $g$  in the second stage of the LRN to *learn* about how the variational parameters between points in  $\mathcal{Q}$  relate to one-another and map to a set of intermediate representations.

**C) Distributions in latent space:** The network must parameterize a valid density in latent space; this is satisfied by construction. To go from the set of intermediate representations to the parameters of  $p(z_t|\mathcal{Q})$ , we leverage the following permutation invariant function:  $f(\mathcal{Q}) = \rho \left( \sum_q (W^{\text{inv}} x_q + b) \right)$ ,  $f : \mathbb{R}^{n \times |\mathcal{Q}|} \rightarrow \mathbb{R}^m$  where  $W^{\text{inv}} \in \mathbb{R}^{m \times n}$ ,  $b \in \mathbb{R}^m$  are linear operators and  $\rho$  is an elementwise non-linearity.

With  $\mu(\mathcal{Q}; \gamma, \phi)$ ,  $\Sigma(\mathcal{Q}; \gamma, \phi)$  as parametric functions of set  $\mathcal{Q}$ , we can write  $p_{\text{rm}}(z_t|\mathcal{Q}; \gamma, \phi) = \mathcal{N}(\mu(\mathcal{Q}; \gamma, \phi), \Sigma(\mathcal{Q}; \gamma, \phi))$ .  $\gamma$  denotes the parameters of the permutation equivariant and invariant layers while  $\phi$  represent the parameters shared with  $q_{\text{dm}}(z|x; \phi)$ . We visualize the LRN in Figure 3a.

## 5 LEARNING

The learning procedure we use is based on a combination of doing unsupervised learning to learn a good representation alongside a supervised max-margin loss to ground the representation for a specific task. We discuss each separately and then highlight how they are combined.

**Unsupervised Learning:** Since we use **Reasoning Models** that satisfy Assumption 1, 2, we make use of the transformation between the **Data Model** and **Reasoning Model** in Section 3. We maximize the likelihood of a given dataset using the lower-bound in Equation 1. A consequence of doing variational learning of the Data Model is that we can use  $q_{\text{dm}}(z|x; \phi)$  to approximate the Bayes Factor.

**Max-Margin Learning:** We expect that the Bayes Factor in Equation 3 takes a high value when the target point  $x_t$  is *similar* to  $\mathcal{Q}$  and a low value when  $x_t$  is dissimilar to  $\mathcal{Q}$ . But how do we know what points form  $\mathcal{Q}$ ? This will depend on the test-time task. We assume we are given labels that define the property encompassed in sets of datapoints.

*Assumption 3.* For  $L$  datapoints in  $\mathcal{D}$ , we have  $\mathcal{Y} = \{y_{x_1}, \dots, y_{x_L}\}$ ,  $y_l \in \{1, \dots, K\}$  where  $y_{x_i}$  is the label for  $x_i$  that takes one of  $K$  unique labels. We define  $\mathbb{N}_{x_i}^{\mathcal{Q}} = \{x_k \text{ s.t. } y_{x_k} \in \mathcal{Y} \ \& \ y_{x_k} = y_{x_i}\}$ ,  $\mathbb{N}_{x_i}^{\bar{\mathcal{Q}}} = \{x_k \text{ s.t. } y_{x_k} \in \mathcal{Y} \ \& \ y_{x_k} \neq y_{x_i}\}$  to be sets of datapoints that have the same label as  $x_i$  and those that do not.

We will assume that a point can only have a single label. Here, the labels characterize the property we want to base our similarity judgements on. Therefore, learn the parameters of  $p(z_t|\mathcal{Q}; \gamma, \phi)$  using the following (supervised) loss function:

$$\mathcal{L}_{\text{mm}}(x; \gamma, \phi) = \mathbb{E}_{\mathcal{Q} \sim \mathbb{N}_x^{\mathcal{Q}}} \mathbb{E}_{\mathcal{Q}_{ns} \sim \mathbb{N}_x^{\bar{\mathcal{Q}}}} \frac{1}{|\mathcal{Q}_{ns}|} \sum_{x_{ns} \in \mathcal{Q}_{ns}} \max(\log \text{score}(x_{ns}, \mathcal{Q}) - \log \text{score}(x, \mathcal{Q}) + \Delta, 0). \quad (4)$$



The loss function maximizes the difference between the log-Bayes Factor for points that lie within the set  $\mathcal{Q}$  and those that do not (they lie in  $\mathcal{Q}_{ns}$ ). The log score( $x, \mathcal{Q}$ ), in Equation 3, is evaluated via Monte-Carlo sampling and the log-sum-exp trick. The expectation is differentiable with respect to  $\gamma, \phi$  via the reparameterization trick (Kingma & Welling, 2014; Rezende *et al.*, 2014). For the margin  $\Delta$  we use the mean-squared-error between the the posterior means of  $x, x_{ns}$ . We provide a visual depiction of how the loss is evaluated using the LRN in Figure 3.

**Combined Loss:** With the unsupervised learning objective for the **Data Model** and the supervised max-margin loss function (Equation 4) for the LRN, we obtain the following loss to jointly learn  $\theta, \phi, \gamma$ :

$$\min_{\theta, \phi, \gamma} \frac{1}{N} \sum_{i=1}^N \frac{1}{C+1} [-\mathcal{L}(x_i; \theta, \phi)] + \frac{C}{C+1} \mathbb{I}[x_i \in \mathcal{Y}] \mathcal{L}_{\text{mm}}(x_i; \gamma, \phi) \quad (5)$$

where  $C$  is a regularization constant that trades off between the supervised and the unsupervised loss. The unsupervised loss learns a representation space constrained to lie close to the prior while explaining the data under the generative model. The max-margin loss modifies this representation space so that dissimilar points are kept apart. Note that Equation 5 is no longer a valid bound on the marginal likelihood of the training set (for  $C > 0$ ).

## 6 EVALUATION

The goal of this section is threefold: (1) to study whether  $p_{\text{rm}}(z|\mathcal{Q})$  is learnable from data using the max-margin learning objective—we expect this to be challenging since we learn the parameters of a model that is itself used to evaluate the the score function in the loss; (2) studying the role of parameter sharing between the inference network and the LRN – i.e. whether the latter can change the former in adversarial scenarios; and (3) studying the utility of the framework for few-shot learning.

We will release code in Keras (Chollet *et al.*, 2015). The supplementary material contains detailed information on the neural architectures of the deep generative models used in the evaluation. We learn parameters with a learning rate of 0.00005 and adaptive momentum updates given by ADAM (Kingma & Ba, 2015). We set the value  $C$  separately for each experiment. When there is a task to be solved,  $C$  can be set using the validation data. When using a pre-trained Data Model, we found it useful to anneal  $C$  from a higher to a lower value so that the task-specific supervised term can overcome (potentially)

suboptimal latent spaces learned from unsupervised data. We use the following datasets for our study:

**Synthetic Pinwheel:** A synthetic dataset of two-dimensional points arranged on a pinwheel taken from the work of Johnson *et al.* (2016). We depict the raw data in Figure 4a. The dataset is created with five labels.

**MNIST digits:** 50000 black and white images of handwritten digits (LeCun *et al.*, 1998).

**MiniImagenet:** A subsampled set of images taken from the Imagenet repository setup for the task of k-shot learning by Vinyals *et al.* (2016). We use the train-validate-test split kindly provided by Ravi & Larochelle (2016).

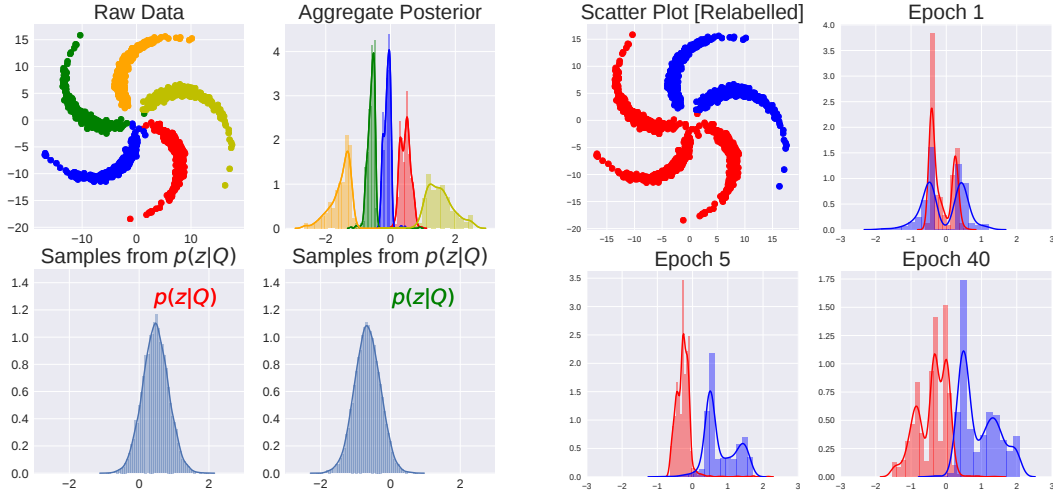
### 6.1 Learning $p(z|\mathcal{Q})$

As a sanity check, we begin by first training a deep generative model (without labels and using a one-dimensional latent space) on the Pinwheel dataset. We visualize the raw-data and learned aggregate posterior  $\sum_x q_{\text{dm}}(z|x; \phi)$  in Figure 4a (top row). We see that the unsupervised learning alone induces class separation in the aggregate posterior distribution. Using the learned model, we hold fixed parameters:  $\theta, \phi$  and learn the parameters  $\gamma$  of the LRN using the loss function in 4 with  $C = 2000$ . We form a kernel density estimate of samples from  $p_{\text{rm}}(z|\mathcal{Q})$  using randomly constructed sets of points derived from the red and green clusters. In Figure 4a (bottom row), we see that samples from the LRN correspond to regions of the latent space associated with  $\mathcal{Q}$ . On synthetic examples, the LRN finds regions of latent space corresponding to points from a query  $\mathcal{Q}$ .

### 6.2 Changing inductive biases at test-time

Previously, we worked with a model where the structure of the latent space (as seen in the aggregate posterior distribution) formed during unsupervised learning co-incident with how points were grouped into sets. Here, we study what happens where the notion of which points are similar changes at test time. We relabel the pinwheel dataset so that the yellow and orange points form one class while the green, red and blue form the other (see Figure 4b, top left). This corresponds to an *adversarial* labelling of the data since we use a deep generative model in which points in the same class are far apart in the learned latent space. If we keep  $\theta, \phi$  fixed then  $p_{\text{rm}}(z|\mathcal{Q})$  (whose output is parameterized as a unimodal Gaussian distribution) cannot capture the relevant subspace.

We have two choices here; we can either consider richer parameterizations for  $p_{\text{rm}}(z|\mathcal{Q})$  that are capable of capturing multi-modal structure in the latent space using techniques proposed by (Rezende & Mohamed, 2015), or we



(a) **Data and Aggregate Posterior:** (Top Left) Raw Pinwheel Data; (Top Right) Aggregate posterior density of a learned (unconditional) deep generative model coloured by class membership. (Bottom Row) Sampling from  $p_{\text{rm}}(z|Q)$  where the colour denotes the class membership of points in  $Q$ .

(b) **Learning dynamics:** (Top left) Visualization of *adversarially labelled data* (relative to the learned aggregate posterior in Figure 4a (top right)). The remaining plots are class coloured visualizations of the aggregate posterior (during training) while allowing the LRN to fine-tune the latent space of the DGM.

Figure 4: **Qualitative Evaluation on Pinwheel Data**

can instead allow the  $p_{\text{rm}}(z|Q)$  to *change* the underlying latent space of the generative model by back-propagating through the parameters of the inference network. Here, we opt for the latter though the former is an avenue for future work.

We minimize Equation 5 while annealing the constant  $C$  from  $1000 \rightarrow 1$  linearly through the course of training. To gain insight into the learning dynamics of the LRN during training, we visualize the aggregate posterior of the generative model (via the fine-tuned inference network) in Figure 4b through the course of training. The role of this adversarial scenario is to highlight two important points (1) unsupervised learning is typically unidentifiable and may not learn a representation appropriate to all tasks and (2) learning with the latent reasoning network can overcome a suboptimal (relative to the task at hand) representation and transform it to a more suitable one.

### 6.3 Modeling High Dimensional Data

**Inducing diversity in latent space:** Moving beyond low-dimensional data, we study learning LRNs on MNIST digits. We use a Data Model with a two-dimensional latent space for this experiment. We begin by training the model in a fully unsupervised manner and visualize the learned latent space in the form of the aggregate posterior (Figure 5a [left]). Although there is some class separability, we find that the unsupervised learning algorithm concentrates much of the probability mass together.

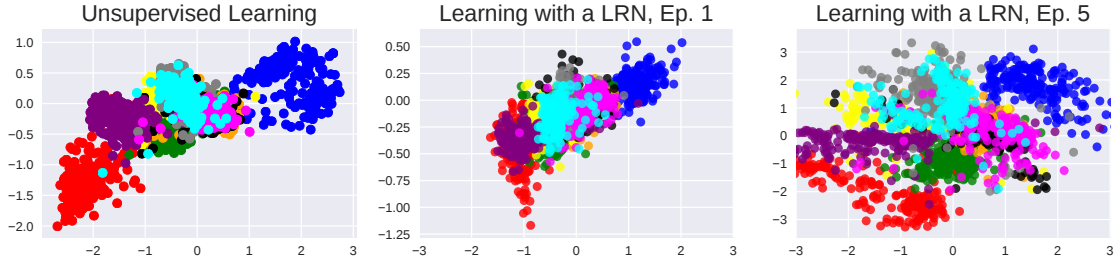
We re-learn the same model with the loss in Equation 5 where  $C$  is set to 3000 (and annealed to 1). We again visualize the new aggregate posterior distribution of the Data Model in Figure 5a (middle and right). When learning with Equation 5, the inference network uses more of the latent space in the model because the max-margin loss pushes points in different classes further apart.

**Qualitative Analysis of MNIST digits:** To validate our method, we provide visualizations on the MNIST dataset. We select a handful of labelled examples  $Q$  (Figure 5b, left) and visualize both their posterior means and samples from  $p(z|Q)$  (Figure 5b, middle). Then, for each sample from  $p_{\text{rm}}(z|Q)$ , we evaluate the fine-tuned  $p_{\text{dm}}(x|z)$  and visualize the images in Figure 5b (right). We see that the generative model fine-tuned with the learning algorithm retains its ability to generate meaningful samples.

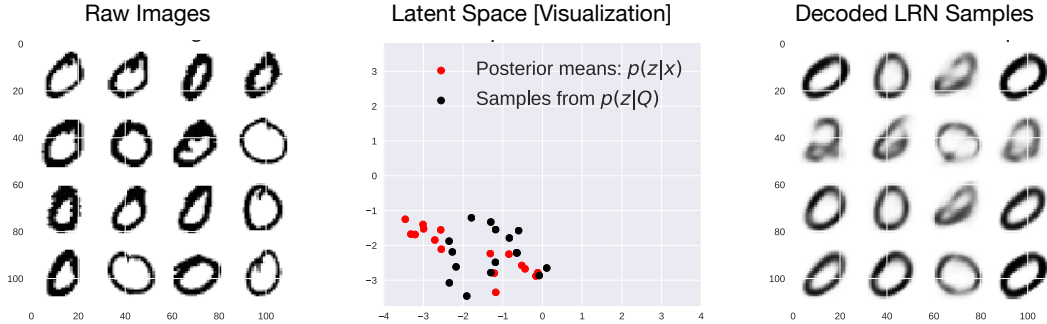
### 6.4 Few-shot learning with the Bayes Factor

The task of k-shot learning is to identify the class an object came from given a single example from 5 other classes (1-shot, 5-way). In the 5-shot, 5-way task, there are 5 examples provided from each of the 5 potential classes. We use an LRN with a deep-discriminative model to obtain near state of the art performance in few-shot learning on the MiniImagenet dataset.

Following (Bauer *et al.*, 2017), who show that discriminative models alone form powerful baselines for this task



(a) **Training dynamics for MNIST:** Aggregate (two-dimensional) posterior of deep generative model of MNIST (coloured by label). The left corresponds to a model trained with unsupervised data only; the middle & right show the aggregate posteriors for a model fine-tuned using Equation 5.



(b) **Test-time evaluation of LRN on MNIST:** On the left are a set of query points  $\mathcal{Q}$  drawn from the same class, in the middle, we visualize samples from  $q_{\text{dm}}(z|x; \phi)$  for each of the points and  $p_{\text{rm}}(z|\mathcal{Q})$ . On the right is the output of the fine-tuned conditional density  $p_{\text{dm}}(x|z)$  for samples drawn from  $p_{\text{rm}}(z|\mathcal{Q})$ .

Figure 5: **Qualitative Evaluation on MNIST**

on this dataset, we pretrain an 18 layer Resnet (He *et al.*, 2016) convolutional neural network to predict class labels at training time. We use early stopping on a validation set based on the nearest neighbor performance of the learned embeddings (obtained from the final layer of the ResNet) to identify the best model. Building a good generative model of the images in MiniImagenet is difficult and so instead, we use the *fixed* embeddings as a 256 dimensional proxy for each image. We initialize  $q_{\text{dm}}(z|x; \phi)$  with the pretrained Resnet and set up a deep generative model to maximize the likelihood of the fixed embeddings (after discriminative pre-training).

For this task, when comparing to the many different approaches proposed, it is challenging to control for both the depth of the encoder that parameterizes the representation and the various algorithmic approach used to tackle the problem using the representation. Therefore, our two take-aways from Table 1 are: (1) on the 1 shot and 5 shot task, we outperform a strong nearest neighbors baseline created using fixed (but learned) embeddings suggesting that our algorithmic approach bears promise for this task and (2) the method is competitive with other state of the art approaches.

Table 1: Accuracy on the 5-way MiniImagenet task

MODEL	1-SHOT	5-SHOT
NEAREST NEIGHBOR	$51.4 \pm 0.08$	$67.5 \pm 0.08$
OURS [RESNET18 ENCODER]	$53.5 \pm 0.08$	$68.8 \pm 0.08$
MATCHING NETWORKS (VINYALS <i>et al.</i> , 2016)	46.6	60.0
MAML (FINN <i>et al.</i> , 2017)	48.7	63.1
PROTOTYPICAL NETS (SNELL <i>et al.</i> , 2017)	49.4	68.2
METANETS (MUNKHDALAI & YU, 2017)	49.2	*
TCML (MISHRA <i>et al.</i> , 2018)	56.7	68.9

## 7 RELATED WORK

**Max Margin Learning:** Max margin parameter estimation has been widely used in machine learning (e.g. in structural SVMs (Yu & Joachims, 2009) and in discriminative Markov networks (Zhu & Xing, 2009)). (Li *et al.*, 2015) give a doubly stochastic subgradient algorithm for regularized maximum likelihood estimation when dealing with max-margin posterior constraints.

(Zaheer *et al.*, 2017) experiment with max-margin learning using a variant of the DeepSets model to predict a scalar score conditioned on a set. While (Zaheer *et al.*, 2017) cite the estimator in (Ghahramani & Heller, 2005) as motivation for their model, they do not explicitly use, parameterize, or differentiate through the Bayes Factor in a *generative* model of data.

**Inductive Transfer and Metric Learning:** Lake *et al.* (2013) use probabilistic inference in a hierarchical model to classify unseen examples by their probability of being in a new class. Instead of the Bayes Factor, they use the posterior predictive obtained via the use of a MCMC algorithm to score target points relative to a query. (Ghahramani & Heller, 2005) evaluate the Bayes factor analytically in exponential family distributions. What we gain in for sacrificing tractability is the ability to work within a richer class of models. Though not motivated within the context of a hierarchical model, (Engel *et al.*, 2018) use an adversarial loss to recognize regions of latent space that correspond to points with a specified class.

Vinyals *et al.* (2016) learn a parametric K-nearest neighbor classifiers to predict whether a target item is within the same class as  $k$ -others. (Snell *et al.*, 2017) associate a point with a prototype within a set and use it to answer whether an object is in the same class as others. The Neural Statistician (Edwards & Storkey, 2017) learns a model similar <sup>2</sup> to the **Reasoning Model** in Figure 2 (b) by maximizing the likelihood of sets  $\mathcal{Q}$ . Their method does not use the Bayes Factor to score items; it also does not permit easy initialization with pre-trained Data Models since the full model is trained with queries.

We tune the latent space of a deep generative model to enhance class separability for test time tasks. By contrast, meta learning algorithms learn to tune the parameters of an algorithm or a model. (Finn *et al.*, 2017) prime the parameters of a neural network to have high accuracy at test time using second order gradient information.

Our work has close parallels with metric-learning; here the metric learned lies in the latent space of a deep generative model. (Bar-Hillel *et al.*, 2005) proposed Relevant Component Analysis, an optimization problem that jointly performs (linear) dimensionality reduction and learns a Mahalanobis metric using queries.

## 8 DISCUSSION

We seek good, task-specific inductive biases to quantify how similar a point is to a set. We give new theoretical and practical constructs towards this goal. We break up

<sup>2</sup>Their model does not enforce the conditional independence statement  $x_t \perp\!\!\!\perp \mathcal{Q} | z_t$

the problem into two parts: learn a good representation and tune the learned representation for a specific notion of similarity. Using the latent space in a deep generative model as our representation, we use the Bayes Factor to quantify similarity.

We derive conditions under which there exists an equivalence between a generative model where data are generated independently to a hierarchical model that jointly generates sets of (similar) points. Using this insight, we derive an easy-to-evaluate estimator for the Bayes Factor; the estimator poses the comparison between a point and a set as overlap in latent space. With the Bayes Factor as a differentiable scoring mechanism, we give a max-margin learning algorithm capable of changing the inductive bias of a (potentially pre-trained) deep generative model. To evaluate the Bayes Factor, we propose a neural architecture for a *latent reasoning network*: a set conditional density that amortizes the posterior predictive distribution of a hierarchical model.

Our approach has limitations. By directly parameterizing the posterior predictive density, and not the prior  $p_{\text{rm}}(w)$  and conditional  $p_{\text{rm}}(z|w)$ , we lose the ability to sample points from the hierarchical generative model. Working with a set of models in which Assumption 1 holds may implicitly only find posterior predictive densities under relatively simple model families of  $p_{\text{rm}}(w)$  and  $p_{\text{rm}}(z|w)$ . Finally, enforcing that property identity in  $w$  is conditionally independent of the data  $x$ , given the representation  $z$ , may make for a challenging learning problem –  $z$  has to represent both the property and variability in the property conditional distribution of the data.

An avenue of future work is leveraging vast amounts of unlabeled data for representation learning informed by a small amount of supervision to guide either during learning, or after learning, the structured of the learned space. Yet another interesting direction would be to learn LRNs that parameterize distributions over multiple, per-data-point latent variables.

## ACKNOWLEDGEMENTS

The authors thank Uri Shalit, Fredrik Johansson, Adam Yala, and the anonymous reviewers for their comments that have improved the manuscript. RGK, AK, and DS are supported by a grant from SAP Corporation.

## References

- Bar-Hillel, Aharon, Hertz, Tomer, Shental, Noam, & Weinshall, Daphna. 2005. Learning a mahalanobis metric from equivalence constraints. *JMLR*.
- Bauer, Matthias, Rojas-Carulla, Mateo, Bartłomiej Świątkowski, Jakub, Schölkopf, Bernhard, & Turner, Richard E. 2017. Discriminative k-shot learning using probabilistic models. *arXiv preprint arXiv:1706.00326*.
- Chollet, François, *et al.* . 2015. *Keras*. <https://github.com/keras-team/keras>.
- Church, Kenneth Ward, & Hanks, Patrick. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*.
- Edwards, Harrison, & Storkey, Amos. 2017. Towards a neural statistician. *In: ICLR*.
- Engel, Jesse, Hoffman, Matthew, & Roberts, Adam. 2018. Latent Constraints: Learning to Generate Conditionally from Unconditional Generative Models. *In: ICLR*.
- Fano, Robert M. 1949. *The transmission of information*.
- Finn, Chelsea, Abbeel, Pieter, & Levine, Sergey. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. *ICML*.
- Ghahramani, Zoubin, & Heller, Katherine A. 2005. Bayesian sets. *In: NIPS*.
- He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, & Sun, Jian. 2016. Deep residual learning for image recognition. *In: CVPR*.
- Hinton, Geoffrey E, Dayan, Peter, Frey, Brendan J, & Neal, Radford M. 1995. The "wake-sleep" algorithm for unsupervised neural networks. *Science*.
- Jeffreys, Harold. 1998. *The Theory of Probability*. OUP Oxford.
- Johnson, Matthew, Duvenaud, David K, Wiltchko, Alex, Adams, Ryan P, & Datta, Sandeep R. 2016. Composing graphical models with neural networks for structured representations and fast inference. *In: NIPS*.
- Kingma, Diederik, & Ba, Jimmy. 2015. Adam: A method for stochastic optimization. *In: ICLR*.
- Kingma, Diederik P, & Welling, Max. 2014. Auto-encoding variational bayes. *In: ICLR*.
- Lake, Brenden M, Salakhutdinov, Ruslan R, & Tenenbaum, Josh. 2013. One-shot learning by inverting a compositional causal process. *In: NIPS*.
- LeCun, Yann, Cortes, Corinna, & Burges, Christopher JC. 1998. *The MNIST database of handwritten digits*.
- Li, Chongxuan, Zhu, Jun, Shi, Tianlin, & Zhang, Bo. 2015. Max-margin deep generative models. *Pages 1837–1845 of: Advances in neural information processing systems*.
- Mishra, Nikhil, Rohaninejad, Mostafa, Chen, Xi, & Abbeel, Pieter. 2018. Meta-learning with temporal convolutions. *ICLR*.
- Munkhdalai, Tsendsuren, & Yu, Hong. 2017. Meta networks. *ICML*.
- Ravi, Sachin, & Larochelle, Hugo. 2016. Optimization as a model for few-shot learning. *In: ICLR*.
- Rezende, Danilo Jimenez, & Mohamed, Shakir. 2015. Variational inference with normalizing flows. *In: ICML*.
- Rezende, Danilo Jimenez, Mohamed, Shakir, & Wierstra, Daan. 2014. Stochastic backpropagation and approximate inference in deep generative models. *In: ICML*.
- Snell, Jake, Swersky, Kevin, & Zemel, Richard. 2017. Prototypical networks for few-shot learning. *In: NIPS*.
- Vinyals, Oriol, Blundell, Charles, Lillicrap, Tim, Wierstra, Daan, *et al.* . 2016. Matching networks for one shot learning. *In: NIPS*.
- Yu, Chun-Nam John, & Joachims, Thorsten. 2009. Learning structural svms with latent variables. *In: ICML*. ACM.
- Zaheer, Manzil, Kottur, Satwik, Ravanbakhsh, Siamak, Póczos, Barnabas, Salakhutdinov, Ruslan, & Smola, Alexander. 2017. Deep Sets. *arXiv preprint arXiv:1703.06114*.
- Zhu, Jun, & Xing, Eric P. 2009. Maximum entropy discrimination Markov networks. *Journal of Machine Learning Research*, **10**(Nov), 2531–2569.

---

# Densified Winner Take All (WTA) Hashing for Sparse Datasets

---

**Beidi Chen**  
Computer Science Dept.  
Rice University  
beidi.chen@rice.edu

**Anshumali Shrivastava**  
Computer Science Dept.  
Rice University  
anshumali@rice.edu

## Abstract

WTA (Winner Take All) hashing has been successfully applied in many large-scale vision applications. This hashing scheme was tailored to take advantage of the comparative reasoning (or order based information), which showed significant accuracy improvements. In this paper, we identify a subtle issue with WTA, which grows with the sparsity of the datasets. This issue limits the discriminative power of WTA. We then propose a solution to this problem based on the idea of Densification which makes use of 2-universal hash functions in a novel way. Our experiments show that Densified WTA Hashing outperforms Vanilla WTA Hashing both in image retrieval and classification tasks consistently and significantly.

## 1 INTRODUCTION

In many important applications like information retrieval and natural language processing, text documents, and images data are in high-dimensional representations. Such high-dimensionality is usually accompanied by extreme data sparsity due to either a large vocabulary or the use of large image window size. The major reason we find very sparse datasets almost everywhere results from the wide adoption of Bag of Words (BoW) representation for documents and images. In BoW representation, the presence or absence of specific features carries the most information [Chapelle et al., 1999, Jiang et al., 2007], especially with higher order shingles. The popularity of sparse machine learning [Caiafa et al., 2017, Liu and Tsang, 2017, Liu et al., 2017, Liu and Tsang, 2016] and sparse codes [Lee et al., 2006] for image data is another reason for the abundance of sparse datasets in modern applications. In order to get a sense of this extreme

sparsity, the datasets demonstrated in Google’s Machine Learning system SIBYL [Canini et al., 2012] have dimensions in billions and non-zeros in only a few thousands (even hundreds).

With the advent of the Internet and the explosion in volumes of data, almost all machine learning and data mining applications are constrained by their computational requirements. Learning with non-linear kernels, by materializing kernel matrices, which are quadratic in computation and memory, is infeasible [Rahimi and Recht, 2007, Li et al., 2011, Shrivastava, 2015]. Randomized algorithms, especially those based on Locality Sensitive Hashing (LSH) [Indyk and Motwani, 1998], have shown huge promise for reducing computational and memory requirement in these scenarios. These randomized algorithms lead to drastic gains in computation and memory for a small, insignificant, amount of approximations.

LSH-based algorithms are quite popular efficient sub-linear algorithms for near neighbor search [Indyk and Motwani, 1998]. This is because even a simple linear scan for near neighbor search, over massive datasets, becomes prohibitively expensive [Weber et al., 1998]. There are no options but to use hashing approaches for such scenarios. LSH algorithms can also be used as cheap random kernel features [Li et al., 2011] for training large-scale non-linear SVMs without materializing the expensive kernel matrix, leading to linear time algorithms. Besides, recently a line of work appears to use LSH as samplers in optimization [Chen et al., 2018] and deduplication [Chen et al., 2017] problems. They are embarrassingly parallel, simple and cheap. Owing to these unique advantages, they are heavily used by commercial search industries for truly large-scale data processing systems.

In the last decade, similarities based on relative (or comparative) attributes have gained huge popularity, especially in the vision literature [Parikh and Grauman, 2011]. For such similarities, a well-known hashing

scheme is *Winner Take All (or WTA) hashing* [Yagnik et al., 2011]. **It is one of the fastest known hashing scheme, which is much faster than signed random projection (SRP).** SRP requires one pass over the data vector for computing one hash value. This is expensive because in practice we need hundreds of hash values, which results in hundreds of passes over the data. Similarly, even random projections are significantly slow for many large-scale tasks. On the contrary, WTA can generate multiple hashes in one pass. It is widely known that hashing time is the major bottleneck, both in theory and practice, for the task of image retrieval. This is why Google [Dean et al., 2013] needed WTA for detecting 100,000 objects on a single machine in near-real time with very respectable accuracy.

Large-scale image retrieval, with low-latency constraints, is a reality. We cannot afford to have costly hash functions since even one pass over the data vector for hash computation is prohibitively expensive both for energy and latency. WTA hashing has been quite successfully applied to produce superior results on massive-scale object recognition and information retrieval. This randomized hashing scheme seems quite suitable for taking advantage of multiple partial order statistics rather than total orderings of the input vector’s feature dimensions to produce sparse embedding codes.

Deep Neural Networks are widely-used in vision and speech tasks. While the network architecture sizes grow exponentially larger to adapt data complexity, LSH algorithms are recently adopted to reduce the computation [Spring and Shrivastava, 2017, Vijayanarasimhan et al., 2014]. Moreover hashing cost and quality are the critical bottleneck in making such approaches practical. **Our Contributions:** In this work, we study the applicability of WTA hashing for very sparse datasets. We found that WTA hashes are not very informative for sparse datasets. We further provide a remedy based on the recent idea of *Densification* [Shrivastava and Li, 2014a]. In particular, our contributions can be summarized as follows:

1. We illustrate that the popular WTA hashing scheme starts losing information for very sparse datasets, i.e., most of the hash values for very sparse datasets do not have enough discriminative information.
2. We propose Densified WTA Hashing which combines traditional WTA hashing with the idea of Densification [Shrivastava, 2017]. We show that the idea of densification provably fixes the issue of WTA for sparse datasets. Our proposal makes novel use of 2-universal hashing, introduced in Section 4.1, and requires minimal modifications to the original WTA hashing. Furthermore, for dense

datasets, our proposal is equivalent to the original WTA hashes and thus a smooth generalization of WTA for sparse datasets.

3. We show for the first time that the idea of Densification actually leads to significant improvement in the quality of WTA hashing, informative hashes. Previously the idea of Densification was only known to speed up hash functions without losing quality. Furthermore, this is the first use of densification for non-binary data.
4. We demonstrate the benefits of our proposal by showing significant gains in accuracy compared to WTA on real-world sparse datasets for both retrieval and classification tasks.

## 2 REVIEW WTA HASHING

[Parikh and Grauman, 2011] pointed out the importance of relative attributes in the vision community. It suggested that for a given vector  $x$ , the information that the attribute  $x_i$  is dominant over some other attribute  $x_j$  has stronger discriminative powers compared to other features. It was further shown in [Yagnik et al., 2011] that comparative reasoning (or order information) among attributes is a very informative feature and similarities based on such comparisons lead to superior performances compared to widely adopted measures like  $L_2$  distances. However, kernel based (or similarity based) learning is computationally slow. To mitigate this problem, WTA (Winner Takes ALL) Hashing was proposed. The simplicity, scalability, and power of WTA hashing were quite appealing and it has been successfully used by commercial big-data companies to scale up the task of object detection significantly [Dean et al., 2013].

WTA hashing generates a set of random samples of  $K$  attributes, using a random permutation  $\Theta$ , and stores the index of the attribute with the maximum weight. It can be implemented in three lines with Matlab:

```
function [maxval, c] = wta(X, K)
theta = randperm(size(X, 2))
[maxval, c] = max(X(:, theta(1:K)), [], 2)
```

### 2.1 KEY WTA NOTATIONS

We denote  $\Theta(x)$  to be the  $K$  random samples from  $x$  sampled using permutation  $\Theta$ . For convenience, we drop the dependence on  $K$  as it will remain a fixed constant.  $\mathcal{H}_{wta}(\Theta(x))$  indicates the corresponding WTA hash. We will also drop  $\Theta$  and use  $\mathcal{H}_{wta}(x)$  when it is clear.

As illustrated in the example shown in Table 1, the original input vectors  $x_1, x_2, x_3, x_4$  are applied with random

Table 1: WTA Hashing Example with four input vectors  $x_1, x_2, x_3, x_4$ ,  $K = 3$  and one permutation  $\Theta = 4, 1, 2$

	$x_1$	$x_2$	$x_3$	$x_4$
$\mathbf{x}$	10, 12, 9, 23	8, 9, 1, 12	9, 2, 6, 1	3, 5, 1, 7
$\Theta(\mathbf{x})$	23, 10, 12	12, 8, 9	1, 9, 2	7, 3, 5
$\mathcal{H}_{\text{wta}}(\mathbf{x})$	1	1	2	1

permutation  $\Theta = (4, 1, 2, 3)$  and first  $K = 3$  attributes of the permuted vectors are selected (random sample of size 3), e.g. Vector (a) = [10, 12, 9, 23] will sample [23, 10, 12]. Then the index of the maximum attribute in every transformed vector is stored separately, e.g. 1 for (a), to contribute to the final WTA hash codes. If there are  $n$  such hashes codes for one input vector, **we define Bin  $i$  as the space to store the hash code generated from the  $i^{\text{th}}$  set of  $K$  samples.**

It was shown that WTA hashing scheme has locality sensitive hashing property. It implies that collision probability under this scheme, i.e. for given vectors  $x$  and  $y$ ,  $Pr(\mathcal{H}_{\text{wta}}(x) = \mathcal{H}_{\text{wta}}(y)) = \mathbb{E}[\mathbf{I}_{\mathcal{H}_{\text{wta}}(x)=\mathcal{H}_{\text{wta}}(y)}]$  is some desirable order based similarity measure. It was later shown that for  $K = 2$  this similarity is the well known Kendall Tau [Ziegler et al., 2012].

### 3 SPARSE DATASETS AND ISSUES WITH WTA HASHING

WTA hashing and the idea of comparative reasoning is quite appealing and intuitive. In this section, we delve deeper and show a critical issue with WTA hashing. We show that for very sparse datasets, which are common in practice [Li et al., 2011], WTA-based hashes are not very informative and deviate from the "relative attribute" intuition. We use the equivalence between hashing and the kernel view to illustrate this issue. With every hashing scheme  $\mathcal{H}$  is an associated positive definite kernel given by the collision probability  $Pr(\mathcal{H}(x) = \mathcal{H}(y)) = \mathbb{E}[\mathbf{I}_{\mathcal{H}(x)=\mathcal{H}(y)}]$ . For large-scale learning, as shown in [Yagnik et al., 2011], we can convert these hashes into random kernel features [Rahimi and Recht, 2007] by converting hash values to indicator vectors.

#### 3.1 SPARSITY MAKES WTA UNINFORMATIVE

Define the sparsity of a dataset  $X$  with  $n$  samples, with each sample of dimension  $d$ , as

$$S_x = \frac{\sum_{i=1}^n \sum_{j=1}^d [1\{X_{ij} = 0\}]}{n \times d} \quad (1)$$

Note that  $[1\{X_{ij} = 0\}]$  is an indicator for the event  $X_{ij} = 0$ .  $S_x$  is also the probability that  $Pr(X_{ij} = 0)$ . We will show that the kernel associated with WTA hashing becomes uninformative as the sparsity increases.

Consider the example that is shown in Table 2. Given very sparse input vectors  $x_1, x_2$ , we generate six WTA hashes with  $K = 3$ . In order to do this, we sample  $K = 3$  attributes six different times so that each different bin is generated using a different permutation. Due to sparsity, many of these bins contain all zeros. We can see that in all the bins except Bin 5,  $\mathcal{H}_{\text{wta}}(x_1)$  and  $\mathcal{H}_{\text{wta}}(x_2)$  collide and therefore the estimated collision probability, from the hashes, is roughly  $\frac{5}{6}$  indicating high similarity (1 is maximum). This seems misleading.

Due to sparsity, it is very likely that for a given  $x$ , all the sampled attributes  $\Theta(x)$  are zeros for some samples. We represent this situation by  $\Theta(x) = E$  (Empty). Consider Bin 1, 4 and 6, they collide only because they are all zeros. Note, WTA treats all empty bins as collisions and two empty bins will always lead to a hash collision. Sparse datasets are common with Bag-of-Words (or token-based) representation. Empty Bins (1, 4 and 6) indicate the absence of the randomly chosen  $K$  tokens which is not a strong indicator of similarity. In BoW analogy, if two documents concurrently lack the words "Hashing", "Winner" and "Take", it does not indicate strong similarity given the large vocabulary and the sparse nature of the dataset. In sparse BoW representation, the absence of features is not informative but only the presence of features is important. Thus, whenever the bins in both input vectors, under considerations for WTA, are empty, we observe undesirable collisions.

However, it is also problematic if we treat empty ones as mismatches. For two identical sparse vector, ideally they should always collide as they are identical. But if we treat zeros as mismatches, then even identical vectors would have low collision probability. If hashes do not collide, it is an indicator that the input vectors are not similar. Preventing empty bins from colliding will treat sparsity as dissimilarity, which is again undesirable. Thus, there is no straightforward fix to this problem.

If we further observe Bin 3, the collision is even worse because it is meaningless that an empty Bin of  $x_2$  collides with a non-empty bin of  $x_1$ , simply because the max value in  $x_1$  happens to be at index 1. This is actually a spurious collision and can be easily eliminated if we assign special values to all empty bins. Therefore, from the analysis, we ignore this easily fixable but spurious collision.



Table 2: WTA with input vectors  $x_1, x_2$  and six bins generated with six permutations. E denoted an empty sampling. WTA treats E and E as a match of hash values, which artificially inflates the similarity perceived by the hashes.

	$x_1$	<b>0, 0, 5, 0, 0, 7, 6, 0, 0</b>				
	$x_2$	<b>0, 0, 1, 0, 0, 0, 0, 0, 0</b>				
	<b>Bin 1</b>	<b>Bin 2</b>	<b>Bin 3</b>	<b>Bin 4</b>	<b>Bin 5</b>	<b>Bin 6</b>
$\Theta$	<b>2, 1, 8</b>	<b>5, 3, 9</b>	<b>6, 2, 4</b>	<b>8, 9, 1</b>	<b>1, 7, 3</b>	<b>2, 4, 5</b>
$\Theta(x_1)$	<b>0, 0, 0 (E)</b>	<b>0, 5, 0</b>	<b>7, 0, 0</b>	<b>0, 0, 0 (E)</b>	<b>0, 6, 5</b>	<b>0, 0, 0 (E)</b>
$\Theta(x_2)$	<b>0, 0, 0 (E)</b>	<b>0, 1, 0</b>	<b>0, 0, 0 (E)</b>	<b>0, 0, 0 (E)</b>	<b>0, 0, 1</b>	<b>0, 0, 0 (E)</b>
$\mathcal{H}_{wta}(x_1)$	<b>1 (E)</b>	<b>2</b>	<b>1</b>	<b>1 (E)</b>	<b>2</b>	<b>1 (E)</b>
$\mathcal{H}_{wta}(x_2)$	<b>1 (E)</b>	<b>2</b>	<b>1 (E)</b>	<b>1 (E)</b>	<b>3</b>	<b>1 (E)</b>

In Bin 2, neither  $\Theta(x_1)$  nor  $\Theta(x_2)$  are E, so those are informative collisions. This is in line with the original motivation of WTA. Owing to the presence of empty bins, sparsity dominates the hash representations of  $x_1, x_2$  and leads to high undesirable similarity. We can not simply ignore empty values because different vectors will have different occurrences of empty bins. Please refer [Shrivastava and Li, 2014a] to see in details why there is no way to ignore empty values in indexing.

Formally, given vectors  $x_1, x_2$  and a permutation  $\Theta$ , define the indicator vector for empty sampling of both  $x_1$  and  $x_2$ :

$$I_{empty} = \begin{cases} 1 & \Theta(x_1) = \Theta(x_2) = E \\ 0 & otherwise \end{cases} \quad (2)$$

Note if any of the  $\Theta(x_1)$  is not empty then  $I_{empty} = 0$ . Based on this indicator variable, we can define empty and non-empty collisions as:

$$\begin{aligned} k_{bad}(x_1, x_2) &= Pr(\mathcal{H}_{wta}(x_1) = \mathcal{H}_{wta}(x_2) | I_{empty} = 1) \\ k_{good}(x_1, x_2) &= Pr(\mathcal{H}_{wta}(x_1) = \mathcal{H}_{wta}(x_2) | I_{empty} = 0). \end{aligned}$$

As argued,  $k_{bad}(x_1, x_2)$  is not an informative kernel for very sparse datasets. Using these quantities we can formally write the WTA kernel as

$$\begin{aligned} k_{wta}(x_1, x_2) &= Pr(\mathcal{H}_{wta}(x_1) = \mathcal{H}_{wta}(x_2)) \\ &= ak_{bad}(x_1, x_2) + (1 - a)k_{good}(x_1, x_2), \end{aligned} \quad (3)$$

where  $a$  is the probability of  $I_{empty} = 1$ . Clearly, for very sparse datasets  $a$  will be high and hence  $k_{bad}(x_1, x_2)$  dominates the WTA kernel making it less discriminative.

## 4 OUR PROPOSAL: DENSIFIED WTA HASHING

### 4.1 2-UNIVERSAL HASHING

**Definition 1.** A randomized function  $h_u : [l] \rightarrow [k]$  is 2-universal if,  $\forall i, j \in [l]$  with  $i \neq j$ , we have the following

property for any  $z_1, z_2 \in [k]$

$$Pr(h_u(i) = z_1 \text{ and } h_u(j) = z_2) = \frac{1}{k^2}. \quad (4)$$

A simple universal hash function example would be, for random number  $a$  and  $b$  and a prime number  $p \leq k$ , compute:  $h_u(x) = (ax + b \bmod p) \bmod k$ .

### 4.2 PROPOSAL

In [Shrivastava and Li, 2014b] the authors proposed the idea of Densification of hashes for obtaining a one-pass hashing scheme which has the same collision probability as the traditional minwise hashing. The idea was to reassign empty bins, having all zero values, by borrowing values from nearest non-empty bins added with some constant offset. Furthermore, [Shrivastava, 2017] showed a better densification schema with optimal variance. Motivated by this idea, we propose a similar re-assignment of empty bins generated from WTA. We will show that the modified WTA, which we call "Densified WTA" (DWTA) hashing, produces the right kernel. This is little surprising because *Densification* was used in the literature to speed up the hashing scheme with the same old property. Here we rather show a first example where densification improves the hashing scheme by making it more informative. This is also the first use of densification over non-binary data.

Vanilla WTA assigns all empty bins a constant value of 1. Using densification, we assign new random values to all the empty bins. For a given data vector  $x$ , we first generate a set of WTA hashes and place them one after the other (See Table 3).

The overall procedure of Densification for reassigning the empty bins is shown in Algorithm 1. We do not touch non-empty bins, as we know that WTA hashes are informative enough. Thus, if a bin is non-empty, its WTA hash value is the DWTA hash value. The key idea in this algorithm is that when a bin  $i$  is empty, instead of assigning it with a constant 1 like what WTA does, it chose

Table 3: Example densification of WTA hashes shown in Table 2. All the hash values of empty bins are reassigned (shown in red) by the values of the mapped (using  $h_u(\cdot, \cdot)$ ) and lookup table in Table 4) non-empty bins with offset shown by the arrow. This unusual procedure actually is the right fix for WTA as shown by Theorem 1

$\mathcal{H}_{Dwta}(x_1)$	$1+3*C$	2	1	$2+1*C$	2	$2+2*C$
$\mathcal{H}_{Dwta}(x_2)$	$3+3*C$	2	$2+3*C$	$3+4*C$	3	$2+2*C$

Table 4: Results of empty bins re-assignment mapping in Table 2 running Algorithm 1.  $i$  and  $attempt$  are the two arguments for some 2-universal hash function and  $map$  represents the non-empty bin  $i$  is mapped to.

	$i$	$attempt$	$map$
$x_1$	1	3	3
	4	1	5
	6	2	2
$x_2$	1	3	5
	3	3	2
	4	4	5
	6	2	2

some non-empty bin randomly using a 2-universal hash function,  $h_u$  and use the value of the chosen non-empty bin with some appropriate offset that ensures no spurious collisions. The 2-universal hash function takes in two arguments: 1) the index of the current empty bin and 2) the number of attempts to reach the first non-empty bin. The first argument is to ensure that DWTA will produce good kernels defined in Section 3. Specifically, for instance, in Table 2, Bin 1s are both empty for  $x_1$  and  $x_2$ . The ideal collision probability of such empty bins should be the same as that of two non-empty bins, derived in Equation 3. The second argument,  $attempts$ , is to prevent infinite cycles during the process of reaching the non-empty bin. For instance, when we compute the non-empty bin mapping for Bin  $i$ , if  $h_u$  only takes in  $i$  as an argument and  $i = h_u(i)$ , then the algorithm would run into an infinite loop. However, with such monotonically increasing  $attempts$ , even under the same  $i$ , the sequence of hash values generated from  $h_u$  will not run into infinite cycles. Another scenario that can test the randomness of our algorithm is when  $j = h_u(i, attempt)$  and bin  $i$  and  $j$  are both empty. Under such circumstance, bin  $i$  and  $j$  are not guaranteed to be re-assigned with the value of the same bin because the re-assignments are independent due to 2-universality of  $h_u(\cdot, \cdot)$ .

For each empty bin  $i$ , we locate a random (but consistently chosen) non-empty bin  $j$  according to a 2-universal hash function, call it  $h_u$ . Formally,

$$\mathcal{H}_{Dwta}[i] = \mathcal{H}_{wta}[j] + attempt * C. \quad (5)$$

---

#### Algorithm 1 Densified WTA Hashing

---

**input**  $n$  hashes  $\mathcal{H}_{wta}[]$  generated from WTA Hashing  
**input**  $h_u(\cdot, \cdot)$ , constant  $C$   
Initialize  $\mathcal{H}_{Dwta}[] = 0$   
**for**  $i = 1$  to  $n$  **do do**  
  **if**  $\mathcal{H}_{wta}[i] \neq E$  **then**  
     $\mathcal{H}_{Dwta}[i] = \mathcal{H}_{wta}[i]$   
  **else**  
     $attempt = 1$   
     $next = h_u(i, attempt)$   
    **while**  $\mathcal{H}_{wta}[next] = E$  **do**  
       $attempt++$   
       $next = h_u(i, attempt)$   
    **end while**  
     $\mathcal{H}_{Dwta}[i] = \mathcal{H}_{wta}[next] + attempt * C$   
  **end if**  
**end for**  
**return**  $\mathcal{H}_{Dwta}[]$

---

Then the newly assigned value to the empty bin  $i$  is exactly the value of  $j$  with some appropriate offset. The offset is mainly the number of attempts such process make before termination, multiplying by some constant  $C > K$ . Table 3 gives a toy example of how Algorithm 1 works on table 2. For  $x_1$ , from Table 4, the mapped non-empty bin for Bin 1 with map function  $h_u$  is 3 and Bin 3's hash value is 1. The total attempts made for reaching Bin 2 is 3. Therefore, according to equation 5, the new hash value of Bin 1 would be  $1 + 3 * C$ . Similarly, Bin 4 is assigned with  $2 + 1 * C$  and Bin 6 is assigned with  $2 + 2 * C$ . Reassignments in the same manner happen to  $x_2$  but since it is more sparse than  $x_1$ , more bins are filled with new hash values. Recall in Issues with WTA Hashing Section, we discuss that the collisions between  $\mathcal{H}_{wta}(x_1)$  and  $\mathcal{H}_{wta}(x_2)$  happened in Bin 1, 4 and 6. After densification, there is no collision in Bin 1 and 4. Therefore after densification the hash collision similarity comes down to  $\frac{2}{6} = 0.33$ .

Formally, let us assume that we want to generate  $n$  hash values.  $\Theta_i(x)$  denote bin  $i$ . Let  $h_u(i, attempt)$  be the first number in the process described in Algorithm 1 such that  $\Theta_{h_u(i, attempt)}(x) \neq E$ . We can define the Densified

Table 5: Each entry displays the Sparsity of VOC2010, LabelMe-12-50k and MSRc datasets in 1000 BoW, 5000 BoW and 10000 BoW representation. Sparsity shows the Raw Data sparsity of original BoW vectors and Empty Codes shows the ratio of empty hash codes in resulting WTA Hashing encoding (empty codes means empty sampling). By increasing dictionary size, Sparsity naturally goes up in all three datasets.

	1000 BoW (%)		5000 BoW (%)		10000 BoW (%)	
	Sparsity	Empty Codes	Sparsity	Empty Codes	Sparsity	Empty Codes
<b>VOC2010</b>	68.63	23.84	88.18	61.39	92.87	74.81
<b>LabelMe-12-50k</b>	58.07	13.63	82.93	48.18	89.49	64.43
<b>MSRc</b>	69.46	24.66	86.83	56.60	91.54	70.07

WTA,  $\mathcal{H}_{Dwta}$ , as follows

$$\mathcal{H}_{Dwta}(\Theta_i(x)) = \begin{cases} \mathcal{H}_{wta}(\Theta_i(x)) & \text{if } \Theta_i(x) \neq E \\ \mathcal{H}_{wta}(\Theta_{h_u(i, attempt)}(x)) + attempt * C & \text{otherwise.} \end{cases}$$

Based on this definition, we now show our main result that  $\mathcal{H}_{wta}$  precisely fixes the issue of empty bins and get rid of the bad kernels. Since the result holds for any bin, we will drop the subscript  $i$ . Formally,

**Theorem 1.** *For any given  $x$  and  $y$ , the collision probability of "Densified WTA"  $\mathcal{H}_{Dwta}$  satisfies:*

$$\begin{aligned} Pr(\mathcal{H}_{Dwta}(x_1) = \mathcal{H}_{Dwta}(x_2)) &= k_{good}(x_1, x_2) \\ &= k_{Dwta}(x_1, x_2), \end{aligned} \quad (7)$$

**Proof:** See supplementary material.  $\square$

From Theorem 1, it is clear that the new kernel is precisely the good kernel  $k_{good}(x_1, x_2)$  with no contribution of  $k_{bad}(x_1, x_2)$  in  $k_{Dwta}(x_1, x_2)$ , irrespective of the sparsity.

### 4.3 COST OF DENSIFICATION

We can see that we incur an additional cost of densification over the generated WTA hashes. The cost comes from, as shown in Algorithm 1, if the bin is empty, it requires an additional while loop. Let  $n$  be the total number of bins in  $\mathcal{H}_{Dwta}(\Theta(x))$  and  $n_{NE}$  be the number of non-empty bins. The probability of terminating the while loop in one iteration is  $p = \frac{n_{NE}}{n}$ . Therefore the expected iterations each while loop need to run before termination will be  $\frac{1}{p}$ . The computation is negligible because it only involves  $\frac{1}{p}$  hash lookups for every empty bin. We will show in Section 5.4 that this negligible cost leads to huge performance gains in practice. This we believe is one of the many examples where a careful analysis and some mathematics goes a long way in designing simple and significantly better algorithms.

### 4.4 DEALING WITH LARGE HASH VALUES

It can be seen from Equation 6 that the value of  $\mathcal{H}_{Dwta}(\Theta_i(x))$  can become large due to the term  $attempt * C$ . It turns out that this is not a problem. There is a significant amount of literature to reduce the final range of hashing scheme [Li and König, 2011]. The idea is to randomly shrink the range at an insignificant cost of small constant random collisions. We found that if we want to constrain the final hash value to a range  $R$  simply taking mod  $R$  of the final hash value suffices in practice. This is what we use during evaluations.

## 5 EXPERIMENTS

In this section, we compare the performance of Densified WTA hashing with Vanilla WTA on two tasks: 1) Image retrieval and 2) classification. *The experiments do not compare with other hashing algorithms because the goal of this paper is solving the problem of WTA while maintaining its superiority over other methods mentioned in the introduction section.* They are important tasks of evaluating the performance of Hashing algorithms, because hashing has received increasing interests in efficient large-scale image retrieval with the rapid growth of web images and the classification accuracy can quantify the discriminative power in hashes.

### 5.1 DATASETS AND BASELINES

We use three popular publicly available image datasets, including VOC2010 [Everingham and Winn, 2010], LabelMe-12-50k [Russell et al., 2008] and MSRc [msr, 2004]:

- The VOC2010 database contains a total of 10103 annotated images of twenty classes, including people, animals, vehicles and indoors. The data has been split into 50% for training and 50% for testing. One image could belong to different classes.
- The LabelMe-12-50k dataset consists of 50,000

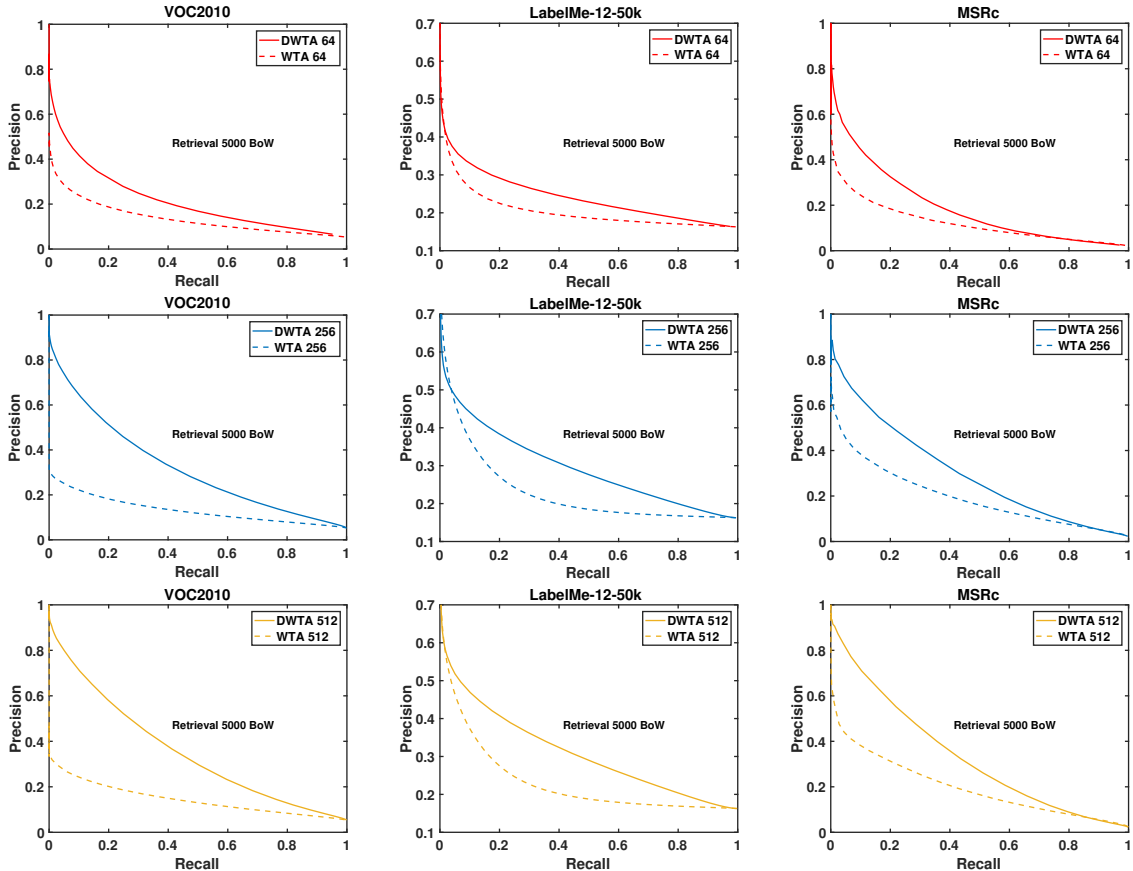


Figure 1: Precision and Recall curves comparing the retrieval performance of Densified WTA vs. WTA on VOC2010, LabelMe-12-50k and MSRC datasets for 1000, 5000 and 10000 BoW feature representations. The semi-dotted lines are the vanilla WTA hashes and bold lines are our proposed Densified WTA Hashes. Different colors represent different number of BoW. We only show 5000 BoW with 64, 256 and 512 hashes (number of hashes used for ranking). Densified WTA significantly outperforms the corresponding WTA consistently.

JPEG images of twelve classes, 80% for training and 20% for testing. They are  $256 \times 256$ -pixels pictures extracted from LabelMe.

- The MSRC is a database of thousands of labeled, high-resolution (680x480 pixels) images of eighteen classes.

The authors of WTA paper used LabelMe for retrieval tasks and VOC2010 datasets for classification tasks. We demonstrate both retrieval and classification on both of the datasets as well as a new MSRC dataset. As described in Section of Large Hash Values, to reduce the space of Densified WTA Hashing, we apply *mod* operation on hash values of all bins as a fix. Table 5 summarizes the sparsity of Raw Data, input Bag of Words, and the ratio of Empty Hash codes, the resulting codes after applying WTA Hashing to input Bag of Words vectors. We can see that when the number of BoW increases, sparsity, highest in 10000 BoW, also goes up in all three datasets.

Note here, we are doing the same tasks as WTA paper, but we do not apply exactly same settings and the sparsities of BoW would thereby be different (they did not reveal sparsity of their datasets as well). Therefore, we do not expect the same results on VOC2010 dataset due to the sparsity difference.

## 5.2 IMAGE RETRIEVAL

We now compare the performance of our Densified WTA codes with Vanilla WTA by replicating the retrieval experiments and studying the standard precision-recall curves. This is our main task of performance comparison because like we mentioned in the Introduction section, WTA is quite appealing for information retrieval. We re-stress that WTA (and our DTWA) are the fastest known hashing scheme, significantly faster than plain random projections. Furthermore hashing cost is a critical bottleneck in large-scale retrieval system.

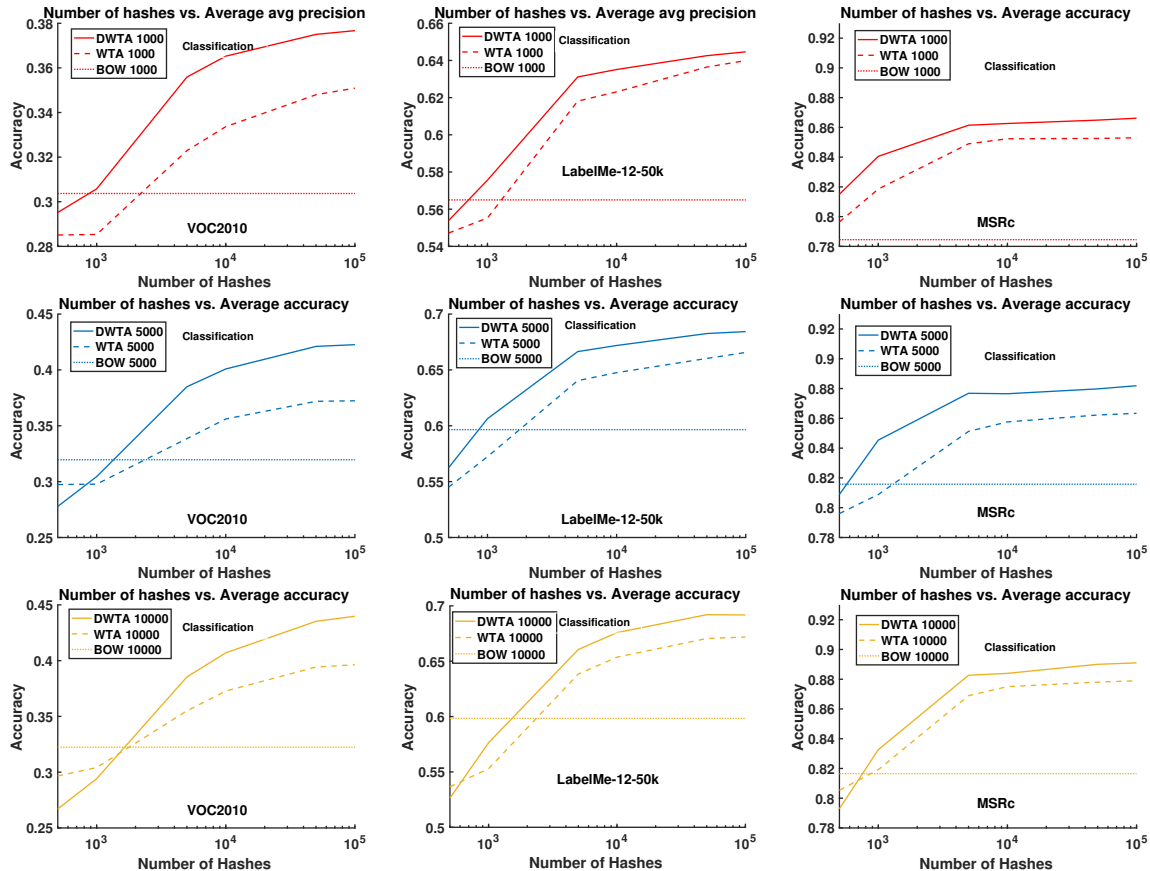


Figure 2: Densified WTA vs. WTA on the task of Image Classification on three different vision datasets. We used 1000, 5000 and 10000 BoW representation of the images. The y-axis is the mean accuracy and the x-axis is the number of hashes used as features. The horizontal lines (dotted) are classification based just on the BoW features. The semi-dotted lines are the vanilla WTA hashes and bold lines are our proposed Densified WTA Hashes. The colors represent which BoW was used as features. Densified WTA significantly outperforms the corresponding WTA consistently for all the choices.

For each query image, the nearest-neighbors of each test data were ranked among training data based on the Hamming distance of the hash codes. Since we had labeled datasets, all the images with the same label as the query were treated as the gold standard neighbors. Note, as mentioned in our proposal, WTA and Densified WTA leads to two different similarity measures (or kernel). Therefore, this experiment is comparing which among these two kernels agrees with the ground truth labels.

Replicating the setting of the original WTA paper, we first generated standard BoW of local descriptors, computed from the images, using the publicly available code [Vedaldi and Fulkerson, 2010]. BoW was generated by extracting local descriptors from the dense grid over each image and quantizing them using K-means. We used DSIFT [Kokkinos et al., 2012] as our descriptor measuring gradient at each key point pixel. The gradient was represented by a single 128-dimensional vector,

stacked by a three-dimensional ( $8 \times 4 \times 4$ ) elementary feature vector formed by the pixel location ( $4 \times 4$ ) and the gradient orientation (8). In this experiment, we consider BoW with 1000, 5000, and 10,000 bins to demonstrate the effect of sparsity. We then generated WTA and Densified WTA hashes from these images and produce feature vectors as suggested in the WTA paper. For the feature generation, we used the fixed recommended setting of  $K = 4$  for all the datasets which was picked using the same method described in [Yagnik et al., 2011] and best for WTA. The precision and recall curves for the rankings based on different hash codes are shown in Figure 1. We show plots for 64, 256 and 512 hash codes of 1000, 5000 and 10000 BoW representations (9 curves for each dataset per hashing scheme). To average out the randomness of both Densified and original WTA hashing, every curve on the graphs is averaged from 10 runs.

Densified WTA hashes lead to notably better precision-

recall compared to Vanilla WTA on all combinations irrespective of the choices of the dataset. As with classification, an increase in BoW leads to larger gap due to increases in sparsity. This again validates our claims. It is exciting to see that a small but principle modification to WTA Hashing can lead to drastic benefits.

### 5.3 CLASSIFICATION

Our motivation for comparing two Hashing algorithms using classification accuracy is to quantify the discriminative power in hashes. We use Densified WTA codes to do the classification task on the VOC2010, LabelMe-12-50k and MSRC datasets. We don't compare with those state-of-the-art methods like a particular type of nonlinear Mercer kernels, e.g. the intersection kernel or the Chi-square kernel [Yang et al., 2009] in classifying these datasets. Instead, we apply Densified WTA and original WTA hashes to a baseline method, sparse BoW of local descriptors and passing to linear SVM classifier, to show that the Densified WTA achieve superior improvement on classification tasks on sparse data.

To compute the classification performance we ran a simple SVM on BoW features, WTA hashed features, and Densified WTA hashed features. We varied the number of hash features over a range of values:  $5 \times 10^2$ ,  $1 \times 10^3$ ,  $5 \times 10^3$ ,  $1 \times 10^4$ ,  $5 \times 10^4$ ,  $1 \times 10^5$ . We again choose  $K=4$ . The  $C$  parameter of SVM was tuned using cross-validation, for every individual run, to ensure the best possible performance on every combination of the number of features and the hashing scheme. This ensures fairness of the comparisons.

Figure 2 compares the mean average precision of classification tasks using Densified WTA codes, WTA codes and basic sparse BoW on three datasets. The baseline, mean average precision for the three BoWs with different bins is shown by dashed straight lines. The mean average precision for WTA feature vectors is shown by dot-dashed curves and for Densified WTA feature vectors is shown by dot-dashed curves. We observe that as stated in WTA paper, precision increases when original BoW bin number increases or the number of codes increases with WTA beating BoW in each case. These observations are in line with the original WTA paper. We followed the experiment pipeline from the WTA paper, while generating BoW using standard package [Vedaldi and Fulkerson, 2010]. It is not surprising to see exactly the same trends in classification results with a difference in relative values.

The Densified WTA consistently outperforms Vanilla WTA significantly on all the three datasets, irrespective of the choice of BoW or the number of hashes. More-

Table 6: Average running time comparison among DWTA, and Sparse Random Projection for three datasets with 10000 BoW representation and 512 hashes.

	SRP (ms)	DWTA (ms)
<b>VOC2010</b>	8.578	0.032
<b>LabelMe-12-50k</b>	8.62	0.046
<b>MSRC</b>	8.609	0.04

over, the performance gap increases with the number of BoW. The increase in BoW rises sparsity of the dataset and hence this trend validates our hypothesis and theory in this paper. The gains over WTA are significant and our results clearly push the boundary of classification performance with hashing-based kernels significantly outperforming BoW. Note that increasing BoW from 5000 to 10000 leads to no gains in accuracy. But with hashing, especially Densified WTA, the gains keep climbing.

### 5.4 RUNNING TIME OF DWTA HASHING

As mentioned in Section 4.3, DWTA hashing only induces negligible cost of densification. We implemented DWTA and another popular algorithm for sparse data, Sparse Random Projection [Achlioptas, 2001] (SRP) and empirically show the average running time comparison of both algorithms for each data point in Table 6. We used 10000 BoW representation for three datasets and 512 hashes. The results clearly exhibited the advantage of DWTA over FRP in running time which further proves the superiority and Practicality of DWTA in general.

## 6 CONCLUSIONS

In this work, we revisited the problem of WTA Hashing for very sparse datasets which are ubiquitous in large-scale applications. We found a particular issue with WTA hashing in this regime which makes them uninformative with an increase in sparsity. We provide a principled solution to this problem using the novel 2-universal hashing for "Densification". Our solutions leverage the theoretical benefits of rank correlation methods and at the same time successfully resolves the concern of uninformative hash values produced by WTA Hashing for data with high sparsity. Evaluation results shown confirm the superior performance of Densified WTA Hashing on both image retrieval and classification task.

## ACKNOWLEDGEMENTS

This work was supported by National Science Foundation IIS-1652131, RI-1718478, AFOSR-YIP FA9550-18-1-0152 and Amazon Research Award.

## References

- (2004). Microsoft research cambridge object recognition image database. Microsoft Research.
- Achlioptas, D. (2001). Database-friendly random projections. In *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 274–281. ACM.
- Caiafa, C. F., Sporns, O., Saykin, A., and Pestilli, F. (2017). Unified representation of tractography and diffusion-weighted mri data using sparse multidimensional arrays. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 4340–4351. Curran Associates, Inc.
- Canini, K., Chandra, T., Ie, E., McFadden, J., Goldman, K., Gunter, M., Harmsen, J., LeFevre, K., Lepikhin, D., Llinares, T., et al. (2012). Sibyl: A system for large scale supervised machine learning. *Technical Talk*.
- Chapelle, O., Haffner, P., and Vapnik, V. N. (1999). Support vector machines for histogram-based image classification. *Neural Networks, IEEE Transactions on*, 10(5):1055–1064.
- Chen, B., Shrivastava, A., and Steorts, R. C. (2017). Unique entity estimation with application to the syrian conflict. *arXiv preprint arXiv:1710.02690*.
- Chen, B., Xu, Y., and Shrivastava, A. (2018). Lsh-sampling breaks the computational chicken-and-egg loop in adaptive stochastic gradient estimation.
- Dean, T., Ruzon, M., Segal, M., Shlens, J., Vijayanarasimhan, S., and Yagnik, J. (2013). Fast, accurate detection of 100,000 object classes on a single machine. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1814–1821.
- Everingham, M. and Winn, J. (2010). The pascal visual object classes challenge 2010 (voc2010) development kit.
- Indyk, P. and Motwani, R. (1998). Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613. ACM.
- Jiang, A., Bohossian, V., and Bruck, J. (2007). Floating codes for joint information storage in write asymmetric memories. In *Information Theory, 2007. ISIT 2007. IEEE International Symposium on*, pages 1166–1170. IEEE.
- Kokkinos, I., Bronstein, M., and Yuille, A. (2012). Dense scale invariant descriptors for images and surfaces.
- Lee, H., Battle, A., Raina, R., and Ng, A. Y. (2006). Efficient sparse coding algorithms. In *Advances in neural information processing systems*, pages 801–808.
- Li, P. and König, A. C. (2011). Theory and applications b-bit minwise hashing. *Commun. ACM*.
- Li, P., Shrivastava, A., Moore, J. L., and König, A. C. (2011). Hashing algorithms for large-scale learning. In *Advances in neural information processing systems*, pages 2672–2680.
- Liu, W., Shen, X., and Tsang, I. (2017). Sparse embedded k-means clustering. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 3319–3327. Curran Associates, Inc.
- Liu, W. and Tsang, I. W. (2016). Sparse perceptron decision tree for millions of dimensions. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI’16*, pages 1881–1887. AAAI Press.
- Liu, W. and Tsang, I. W. (2017). Making decision trees feasible in ultrahigh feature and label dimensions. *Journal of Machine Learning Research*, 18(81):1–36.
- Parikh, D. and Grauman, K. (2011). Relative attributes. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 503–510. IEEE.
- Rahimi, A. and Recht, B. (2007). Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184.
- Russell, B. C., Torralba, A., Murphy, K. P., and Freeman, W. T. (2008). Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1-3):157–173.
- Shrivastava, A. (2015). *Probabilistic Hashing Techniques For Big Data*. PhD thesis, Cornell University.
- Shrivastava, A. (2017). Optimal densification for fast and accurate minwise hashing. *arXiv preprint arXiv:1703.04664*.
- Shrivastava, A. and Li, P. (2014a). Densifying one permutation hashing via rotation for fast near neighbor search. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 557–565.
- Shrivastava, A. and Li, P. (2014b). Improved densification of one permutation hashing. In *UAI, Quebec, CA*.

Spring, R. and Shrivastava, A. (2017). Scalable and sustainable deep learning via randomized hashing. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 445–454. ACM.

Vedaldi, A. and Fulkerson, B. (2010). Vlfeat: An open and portable library of computer vision algorithms. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1469–1472. ACM.

Vijayanarasimhan, S., Shlens, J., Monga, R., and Yagnik, J. (2014). Deep networks with large output spaces. *arXiv preprint arXiv:1412.7479*.

Weber, R., Schek, H.-J., and Blott, S. (1998). A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proceedings of the 24rd International Conference on Very Large Data Bases, VLDB '98*, pages 194–205, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Yagnik, J., Strelow, D., Ross, D. A., and Lin, R.-s. (2011). The power of comparative reasoning. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2431–2438. IEEE.

Yang, J., Yu, K., Gong, Y., and Huang, T. (2009). Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1794–1801. IEEE.

Ziegler, A., Christiansen, E., Kriegman, D., and Belongie, S. (2012). Locally uniform comparison image descriptor. In *Neural Information Processing Systems (NIPS)*, pages 1–9, Lake Tahoe, NV.



---

# Lifted Marginal MAP Inference

---

Vishal Sharma<sup>1</sup>, Noman Ahmed Sheikh<sup>2</sup>, Happy Mittal<sup>1</sup>, Vibhav Gogate<sup>3</sup> and Parag Singla<sup>1</sup>

<sup>1</sup>IIT Delhi, {vishal.sharma, happy.mittal, parags}@cse.iitd.ac.in

<sup>2</sup>IIT Delhi, nomanahmedsheikh@outlook.com

<sup>3</sup>UT Dallas, vgogate@hlt.utdallas.edu

## Abstract

Lifted inference reduces the complexity of inference in relational probabilistic models by identifying groups of constants (or atoms) which behave symmetric to each other. A number of techniques have been proposed in the literature for lifting marginal as well MAP inference. We present the first application of lifting rules for marginal-MAP (MMAP), an important inference problem in models having latent (random) variables. Our main contribution is two fold: (1) we define a new equivalence class of (logical) variables, called Single Occurrence for MAX (SOM), and show that solution lies at extreme with respect to the SOM variables, i.e., predicate groundings differing only in the instantiation of the SOM variables take the same truth value (2) we define a sub-class *SOM-R* (SOM Reduce) and exploit properties of extreme assignments to show that MMAP inference can be performed by reducing the domain of SOM-R variables to a single constant. We refer to our lifting technique as the *SOM-R* rule for lifted MMAP. Combined with existing rules such as decomposer and binomial, this results in a powerful framework for lifted MMAP. Experiments on three benchmark domains show significant gains in both time and memory compared to ground inference as well as lifted approaches not using SOM-R.

## 1 INTRODUCTION

Several real world applications such as those in NLP, vision and biology need to handle non-i.i.d. data as well as represent uncertainty. Relational Probabilistic models (Getoor and Taskar 2007) such as Markov logic networks (Domingos and Lowd 2009) combine the power

of relational representations with statistical models to achieve this objective. The naïve approach to inference in these domains grounds the relational network into a propositional one and then applies existing inference techniques. This can often result in sub-optimal performance for a large number of applications since inference is performed oblivious to the underlying network structure.

Lifted inference (Kimmig, Mihalkova, and Getoor 2015) overcomes this shortcoming by collectively reasoning about groups of constants (atoms) which are identical to each other. Starting with the work of Poole (Poole 2003), a number of lifting techniques which lift propositional inference to the first-order level have been proposed in literature. For instance, for marginal inference, exact algorithms such as variable elimination and AND/OR search and approximate algorithms such as belief propagation and MCMC sampling have been lifted to the first-order level (cf. (de Salvo Braz, Amir, and Roth 2005; Gogate and Domingos 2011; G. Van den Broeck et al. 2011; Kersting, Ahmadi, and Natarajan 2009; Singla and Domingos 2008; Niepert 2012; Venugopal and Gogate 2012)). More recently, there has been increasing interest in lifting MAP inference (both exact and approximate) (Sarkhel et al. 2014; Mittal et al. 2014; Mladenov, Kersting, and Globerson 2014). Some recent work has looked at the problem of approximate lifting i.e., combining together those constants (atoms) which are similar but not necessarily identical (Van den Broeck and Darwiche 2013; Singla, Nath, and Domingos 2014; Sarkhel, Singla, and Gogate 2015).

Despite a large body of work on lifted inference, to the best of our knowledge, there is no work on lifted algorithms for solving marginal maximum-a-posteriori (MMAP) queries. MMAP inference is ubiquitous in real-world domains, especially those having latent (random) variables. It is well known that in many real-world domains, the use of latent (random) variables significantly improves the prediction accuracy (Maaten, Welling, and

Saul 2011). Moreover, the problem also shows up in the context of SRL domains in tasks such as plan and activity recognition (Singla and Mooney 2011). Therefore, efficient lifted methods for solving the MMAP problem are quite desirable.

MMAP inference is much harder than marginal (sum) and MAP (max) inference because sum and max operators do not commute. In particular, latent (random) variables need to be marginalized out before MAP assignment can be computed over the query (random) variables and as a result MMAP is NP-hard even on tree graphical models (Park 2002). Popular approaches for solving MMAP include variational algorithms (Liu and Ihler 2013), AND/OR search (Marinescu, Dechter, and Ihler 2014) and parity solvers (Xue et al. 2016).

In this paper, we propose the first ever lifting algorithm for MMAP by extending the class of lifting rules (Jha et al. 2010; Gogate and Domingos 2011; Mittal et al. 2014). As our first contribution, we define a new equivalence class of (logical) variables called *Single Occurrence for MAX (SOM)*. We show that the MMAP solution lies at extreme with respect to the SOM variables, i.e., predicate groundings which differ only in the instantiation of the SOM variables take the same truth (true/false) value in the MMAP assignment. The proof is fairly involved due to the presence of both MAX and SUM operations in MMAP, and involves a series of problem transformations followed by exploiting the convexity of the resulting function.

As our second contribution, we define a sub-class of SOM, referred to as *SOM-R (SOM Reduce)*. Using the properties of extreme assignments, we show that the MMAP solution can be computed by reducing the domain of SOM-R variables to a single constant. We refer to this as *SOM-R rule for lifted MMAP*. SOM-R rule is often applicable when none of the other rules are, and can result in significant savings since inference complexity is exponential in the domain size in the worst case.

Finally, we show how to combine SOM-R rule along with other lifting rules e.g., binomial and decomposer, resulting in a powerful algorithmic framework for lifted MMAP inference. Our experiments on three different benchmark domains clearly demonstrate that our lifting technique can result in orders of magnitude savings in both time and memory compared to ground inference as well as vanilla lifting (not using the SOM-R rule).

## 2 BACKGROUND

**First-Order Logic:** The language of first-order logic (Russell and Norvig 2010) consists of *constant*, *variable*, *predicate*, and *function* symbols. A *term* is a variable, constant or is obtained by application of a func-

tion to a tuple of terms. Variables in first-order logic are often referred to as *logical variables*. We will simply refer to them as variables, henceforth. A *predicate* defines a relation over the set of its arguments. An *atom* is obtained by applying a predicate symbol to the corresponding arguments. A *ground atom* is an atom having no variables in it. Formulas are obtained by combining predicates using a set operators:  $\wedge$  (and),  $\vee$  (or) and  $\neg$  (not). Variables in a formula can be universally or existentially quantified using the operators  $\forall$  and  $\exists$ , respectively. A first-order theory (knowledge base) is a set of formulas. We will restrict our attention to function free finite first-order logic with Herbrand interpretation (Russell and Norvig 2010) and universally quantified variables. In the process of (partially) grounding a theory, we replace all (some) of the universally quantified variables with the possible constants in the domain. In the following, we will use capital letters (e.g.,  $X, Y$  etc.) to denote logical variables and small case letters to denote constants. We will use  $\Delta_X = \{x_1, x_2, \dots, x_m\}$  denotes the domain of variable  $X$ .

**Markov Logic:** A Markov logic network (Domingos and Lowd 2009) (MLN)  $M$  is defined as a set of pairs  $\{f_i, w_i\}_{i=1}^n$  where  $f_i$  is a formula in first-order logic and  $w_i$  is the weight of  $f_i$ . We will use  $F(M)$  to denote the set of all the formulas in MLN. Let  $\mathcal{X}$  denote the set of all the logical variables appearing in MLN. An MLN can be seen as a template for constructing ground Markov networks. Given the domain  $\Delta_X$  for every variable  $X \in \mathcal{X}$ , the ground network constructed by MLN has a node for every ground atom and a feature for every ground formula. Let  $\mathcal{T}$  denote the set of all the predicates appearing in  $M$ . We will use  $\mathcal{T}_g$  to denote all the ground atoms corresponding to the set  $\mathcal{T}$  and  $t$  to denote an assignment, i.e. a vector of true/false values, to  $\mathcal{T}_g$ . The distribution specified by an MLN is given as:

$$P(\mathcal{T}_g = t) = \frac{1}{Z} e^{\sum_{i=1}^n \sum_{j=1}^{m_i} w_i f_{ij}(t)} \quad (1)$$

where  $m_i$  denotes the number of groundings of the  $i^{th}$  formula.  $f_{ij}$  represents the feature corresponding to the  $j^{th}$  grounding of the  $i^{th}$  formula. The feature is on if the corresponding formula is satisfied under the assignment  $t$  off otherwise.  $Z$  is the normalization constant. Equivalently, in the potential function representation, the distribution can be written as:

$$P(t) = \frac{1}{Z} \prod_{i=1}^n \prod_{j=1}^{m_i} \phi_{ij}(t) \quad (2)$$

where there is a potential  $\phi_{ij}$  for each  $f_{ij}$  such that  $\phi_{ij}(t) = e^{w_i f_{ij}(t)}$ .

**Marginal MAP (MMAP):** Let the set of all predicates  $\mathcal{T}$  be divided into two disjoint subsets  $\mathcal{Q}$  and  $\mathcal{S}$ , referred

to as MAX and SUM predicates, respectively. Let  $q$  (resp.  $s$ ) denote an assignment to all the groundings of the predicates in  $\mathcal{Q}$  (resp.  $\mathcal{S}$ ). Note that  $\mathcal{T} = \mathcal{Q} \cup \mathcal{S}$ , and given assignment  $t$  to  $\mathcal{T}$ ,  $t = q \cup s$ . Then, the *marginal-MAP* (MMAP) problem for MLNs can be defined as:

$$\arg \max_q \sum_s \prod_{i=1}^n \prod_{j=1}^{m_i} \phi_{ij}(q, s) = \arg \max_q W_M(q) \quad (3)$$

$$\text{where, } W_M(q) = \sum_s \prod_{i=1}^n \prod_{j=1}^{m_i} \phi_{ij}(q, s)$$

$W_M(q)$  is referred to as the MMAP objective function for the MLN  $M$ , and its solution  $q^* = \arg \max_q W_M(q)$  is referred as the MMAP solution. Note that we can get rid of  $Z$  in equation 3, since we are only interested in finding the maximizing assignment and  $Z$  is a constant.

**Preliminaries:** We will assume that our MLN is in Normal Form (Mittal et al. 2014) i.e., (a) no constants appear in any of the formulae (b) if  $X$  and  $Y$  appear at the same predicate position in one or more formulae, then  $\Delta_X = \Delta_Y$ . Any MLN can be converted into normal form by a series of mechanical operations. We will also assume that formulas are standardized apart i.e., we rename the variables such that the sets of variables appearing in two different formulae are disjoint with each other. We define an equivalence relation  $\sim$  over the set of variables such that  $X \sim Y$  if (a)  $X$  and  $Y$  appear at the same predicate position OR (b)  $\exists Z$  such that  $X \sim Z$  and  $Y \sim Z$ . We will use  $\tilde{X}$  to denote the equivalence class corresponding to variable  $X$ . Variables in the same equivalence class must have the same domain due to the normal form assumption. We will use  $\Delta_{\tilde{X}}$  to refer to the domain of the variables belonging to  $\tilde{X}$ .

Finally, though our exposition in this work is in terms of MLNs, our ideas can easily be generalized to other representations such as weighted parfactors (de Salvo Braz, Amir, and Roth 2005) and probabilistic knowledge bases (Gogate and Domingos 2011).

### 3 SINGLE OCCURRENCE FOR MMAP

#### 3.1 Motivation

In this work, we are interested in lifting the marginal-MAP (MMAP) problem. Since MMAP is a problem harder than both marginal and MAP inference, a natural question to examine would be if existing lifting techniques for MAP and marginal inference can be extended to the case of MMAP. Or further still, if additional rules can be discovered for lifting the MMAP problem. Whereas many of the existing rules such as decomposer and binomial<sup>1</sup> (Jha et al. 2010; Mittal et al. 2015) extend

<sup>1</sup>applicable when the binomial predicate belongs to MAX

in a straightforward manner for MMAP, unfortunately the SO rule (Mittal et al. 2014), which is a powerful rule for MAP inference, is not directly applicable.

In response, we propose a new rule, referred to as Single Occurrence for MAX Reduce (SOM-R), which is applicable for MMAP inference. We first define a variable equivalence class, referred to as SOM, which requires that (1) no two variables in the class appear in the same formula (2) at least one of the variables in the class appears in a MAX predicate. We further define a sub-class of SOM, referred to as SOM-R, which imposes a third condition (3) either all the SUM predicates in the theory contain a SOM variable or none of them does. Our SOM-R rule states that domain of SOM-R variables can be reduced to a single constant for MMAP inference. Consider the following example MLN, henceforth referred to as  $M_1$ :

$$w_1 : Frnds(X, Y) \wedge Parent(Z, X) \Rightarrow Knows(Z, Y)$$

$$w_2 : Knows(U, V)$$

$$SUM : Parent \quad MAX : Frnds, Knows$$

The equivalence classes in this example are given by  $\{X\}$ ,  $\{Y, V\}$  and  $\{Z, U\}$ . It is easy to see that each of these equivalence classes satisfy the three conditions above and hence, SOM-R rule can be applied over them. This makes the MMAP inference problem independent of the size of the domain and hence, it can be solved in  $O(1)$  time. Ground inference has to deal with  $O(m^2)$  number of ground atoms resulting in  $O(\exp(cm^2))$  complexity in the worst case<sup>2</sup>, where  $c$  is a constant. Further, in the absence of the SOM-R rule, none of the existing lifting rules apply and one has to resort to partial grounding again resulting in worst case exponential complexity.

We note that conditions for identifying SOM and SOM-R specifically make use of the structure of the MMAP problem. Whereas condition 1 is same as Mittal et al.'s SO condition, condition 2 requires the variables in the SOM class to belong to a MAX predicate. Condition 3 (for SOM-R) further refines the SOM conditions so that domain reduction can be applied.

We prove the correctness of our result in two phases. First, we show that SOM equivalence class implies that MMAP solution lies at extreme, meaning that predicate groundings differing only in the instantiation of the SOM class take the same truth value. Second, for the sub-class SOM-R, we further show that domain can be reduced to a single constant for MMAP. Here, we rely on the properties of extreme assignments.

Our proof strategy makes use of a series of problem

<sup>2</sup>Inference complexity is exponential in the number of ground atoms. Here, we assume  $|\Delta_X| = |\Delta_Y| = |\Delta_Z| = m$

transformations followed by using the convexity of the resulting function. These algebraic manipulations are essential to prove the correctness of our result, and are some of the important contributions of our paper. Next, we describe each step in detail. The proofs of theorems (and lemmas) marked with (\*) are in the supplement.

### 3.2 SOM implies Extreme Solution

We introduce some important definitions. We will assume that we are given an MLN  $M$ . Further, we are interested in solving an MMAP problem over  $M$  where the set of MAX predicates is given by  $\mathcal{Q}$ .

**Definition 1.** (Single Occurrence for MAX) We say that a variable equivalence class  $\tilde{X}$  is Single Occurrence for MAX (SOM) if (a)  $\forall i, f_i \in F(M)$ , there is at most one variable from the set  $\tilde{X}$  occurring in  $f_i$  (b) there exists a variable  $X \in \tilde{X}$  and a predicate  $P \in \mathcal{Q}$ , such that  $X$  appears in  $P$ .

Next, we define the notion of an extreme assignment.

**Definition 2.** (Extreme Assignment) Let  $\tilde{X}$  be a variable equivalence class. An assignment  $q$  to MAX predicates  $\mathcal{Q}$  lies at extreme (with respect to  $\tilde{X}$ ), if  $\forall P \in \mathcal{Q}$ , all the groundings of  $P$  with the same instantiation to variables  $\mathcal{X} - \tilde{X}$ , take the same value in  $q$ .

In  $M_1$ , an extreme assignment with respect to variable equivalence class  $\{Y, V\}$  will assign the same truth value to the ground atoms  $Knows(z, y_1)$  and  $Knows(z, y_2)$ ,  $\forall z \in \Delta_Z$  and  $\forall y_1, y_2 \in \Delta_Y$ . We next define the notion of an MLN variablized with respect to a variable equivalence class.

**Definition 3.** (Variablized MLN) Let  $\tilde{X}$  be an equivalence class. Let  $M_{\tilde{X}}$  be the MLN obtained by instantiating (grounding) the variables in the set  $\mathcal{X} - \tilde{X}$ . We say that  $M_{\tilde{X}}$  is variablized (only) with respect to the set  $\tilde{X}$ .

For instance in  $M_1$ , variablizing with respect to the equivalence class  $\{Y, V\}$  results in MLN with formulas similar to:

$$\begin{aligned} w_1 &: Frnds(x, Y) \wedge Parent(z, x) \Rightarrow Knows(z, Y) \\ w_2 &: Knows(u, V) \end{aligned}$$

where  $x, z$  and  $u$  are constants belonging to respective domains.  $Frnds(x, Y)$ ,  $Knows(z, Y)$  and  $Knows(u, V)$  can be treated as unary predicates over the equivalence class  $\{Y, V\}$  since  $x, z$  and  $u$  are constants. Similarly,  $Parent(z, x)$  can be treated as a propositional predicate.

It is important to note that,  $M_{\tilde{X}}$  represents the same distribution as  $M$ . Further,  $M_{\tilde{X}}$  can be converted back into normal form by introducing a new predicate for every combination of constants appearing in a predicate. We now define one of the main theorems of this paper.

**Theorem 1.** Let  $M$  be an MLN and let  $\tilde{X}$  be a SOM equivalence class. Then, an MMAP solution for  $M$  lies at extreme with respect to  $\tilde{X}$ .

We will prove the above theorem by defining a series of problem transformations. In the following, we will work with MLN  $M$  and  $\tilde{X}$  as a SOM variable equivalence class. We will use  $\mathcal{Q}$  and  $\mathcal{S}$  to denote set of MAX and SUM predicates, respectively.  $q$  and  $s$  will denote the assignments to respective predicate groundings (see Background (section 2)).

#### 3.2.1 Problem Transformation (PT) 1

**Objective PT1:** Convert MMAP objective into a form which only has unary and propositional predicates.

**Lemma 1.** Let  $M_{\tilde{X}}$  denote the MLN variablized with respect to SOM equivalence class  $\tilde{X}$ . Then,  $M_{\tilde{X}}$  contains only unary and propositional predicates. Further, the MMAP objective can be written as:

$$\arg \max_q W_M(q) = \arg \max_q W_{M_{\tilde{X}}}(q)$$

The proof that  $M_{\tilde{X}}$  only has unary and propositional predicates follows immediately from the definition of  $M_{\tilde{X}}$  (defn. 3) and the fact that  $\tilde{X}$  is SOM. Further, since  $M$  and  $M_{\tilde{X}}$  define the same distribution, we have the equivalence of the MMAP objectives. Since,  $M_{\tilde{X}}$  only has unary and propositional predicates, we will split the assignment  $q$  to groundings of  $\mathcal{Q}$  into  $(q_u, q_p)$  where  $q_u$  and  $q_p$  denote the assignments to groundings of unary and propositional predicates, respectively. Similarly, for assignment  $s$  to groundings of  $\mathcal{S}$ , we split  $s$  as  $(s_u, s_p)$ .

#### 3.2.2 Problem Transformation 2

**Objective PT2:** In the MMAP objective, get rid of propositional MAX predicates.

**Lemma 2.\*** Consider the MMAP problem over  $M_{\tilde{X}}$ . Let  $q_p$  be some assignment to propositional MAX predicates. Let  $M'_{\tilde{X}}$  be an MLN obtained by substituting the truth value in  $q_p$  for propositional predicates. Then, if  $M'_{\tilde{X}}$  has a solution at extreme for all possible assignments of the form  $q_p$  then,  $M_{\tilde{X}}$  also has a solution at extreme.

Therefore, in order to prove the extrema property for  $M_{\tilde{X}}$ , it is sufficient to prove it for a generic MLN  $M'_{\tilde{X}}$ , i.e., without making any assumptions on the form of  $q_p$ .

For ease of notation, we will drop the prime in  $M'_{\tilde{X}}$  and simply refer to it as  $M_{\tilde{X}}$ . Therefore, we need to show that the solution to the following problem lies at extreme:

$$\arg \max_{q_u} W_{M_{\tilde{X}}}(q_u)$$

where the propositional MAX predicates have been gotten rid of in  $M_{\tilde{X}}$ .

### 3.2.3 Problem Transformation 3

**Objective PT3:** In the MMAP objective, get rid of unary SUM predicates using inversion elimination (de Salvo Braz, Amir, and Roth 2005).

First, we note that the MMAP objective:

$$W_{M_{\tilde{X}}}(q_u) = \sum_{s_p, s_u} \prod_{i=1}^n \prod_{j=1}^{m_i} \phi_{ij}(q_u, s_p, s_u)$$

can be equivalently written as:

$$W_{M_{\tilde{X}}}(q_u) = \sum_{s_p, s_u} \prod_{i=1}^n \prod_{j=1}^m \phi'_{ij}(q_u, s_p, s_u)$$

where  $m = |\Delta_{\tilde{X}}|$ .  $\phi'_{ij}(q_u, s_p, s_u) = \phi_{ij}(q_u, s_p, s_u)$  if  $f_i$  contains a variable from  $\tilde{X}$ , else  $\phi'_{ij}(q_u, s_p, s_u) = \phi_{ij}(q_u, s_p, s_u)^{\frac{1}{m}}$  otherwise. It is easy to see this equivalence since the only variables in the theory are from the class  $\tilde{X}$ . When  $f_i$  contains a variable from  $\tilde{X}$ , it has exactly  $m_i = m$  groundings since  $\tilde{X}$  is SOM. On the other hand, if  $f_i$  does not contain a variable from  $\tilde{X}$ , it only contains propositional predicates. Then we raise it to power  $\frac{1}{m}$ , and then multiply  $m$  times in the latter expression to get an equivalent form.

Next, we use inversion elimination (de Salvo Braz, Amir, and Roth 2005) to get rid of unary SUM predicates.

**Lemma 3.** *MMAP problem over  $M_{\tilde{X}}$  can be written as:*

$$\arg \max_{q_u} W_{M_{\tilde{X}}}(q_u) = \arg \max_{q_u} \sum_{s_p} \prod_{j=1}^m \Theta_j(q_u, s_p)$$

where  $\Theta_j$  is a function of unary MAX and propositional SUM predicates groundings  $q_u$  and  $s_p$ , respectively.

*Proof.* We can write the MMAP objective  $W_{M_{\tilde{X}}}(q_u)$  as:

$$\begin{aligned} &= \sum_{s_p, s_u} \prod_{i=1}^n \prod_{j=1}^m \phi'_{ij}(q_u, s_p, s_u) \\ &= \sum_{s_p, s_u} \prod_{j=1}^m \prod_{i=1}^n \phi'_{ij}(q_u, s_p, s_u) \\ &= \sum_{s_p, s_u} \prod_{j=1}^m \Phi_j(q_u, s_p, s_u) \\ &= \sum_{s_p} \sum_{s_{u_1}, s_{u_2}, \dots, s_{u_m}} \prod_{j=1}^m \Phi_j(q_u, s_p, s_{u_j}) \end{aligned}$$

(apply inversion elimination)

$$\begin{aligned} &= \sum_{s_p} \prod_{j=1}^m \sum_{s_{u_j}} \Phi_j(q_u, s_p, s_{u_j}) \\ &= \sum_{s_p} \prod_{j=1}^m \Theta_j(q_u, s_p) \end{aligned}$$

**Proof Explanation:** Second equality is obtained by interchanging the two products. Third equality is obtained by defining  $\prod_i \phi'_{ij}(q_u, s_p, s_u) = \Phi_j(q_u, s_p, s_u)$ . In fourth equality, we have made explicit the dependence of  $\Phi_j$  on  $s_{u_j}$  i.e. the groundings corresponding to the  $j^{\text{th}}$  constant.

**Inversion Elimination** (de Salvo Braz, Amir, and Roth 2006): Since  $\Phi_j$  only depends on  $s_{u_j}$  (among  $s_u$ ) groundings, we can use inversion elimination to invert the sum over  $s_{u_j}$  and product over  $j$  in the fifth equality.

**Final Expression:** We define  $\Theta_j(q_u, s_p) = \sum_{s_{u_j}} \Phi_j(q_u, s_p, s_u)$ .

Note that, at this point, we have only propositional SUM and unary MAX predicates in the transformed MMAP objective.

### 3.2.4 Problem Transformation 4

**Objective PT4:** Exploit symmetry of the potential functions in the MMAP objective.

We rename  $q_u$  to  $q$  and  $s_p$  to  $s$  for ease of notation in Lemma 3. The MMAP objective can be written as:

$$W_{M_{\tilde{X}}}(q) = \sum_s \prod_{j=1}^m \Theta_j(q_j, s) \quad (4)$$

Here,  $q = (q_1, q_2, \dots, q_m)$  and  $q_j$  represents the assignment to the unary MAX predicate groundings corresponding to constant  $j$ . In the expression above, we have made explicit the dependence of  $\Theta_j$  on  $q_j$ . We make the following two observations.

1) Due to the normal form assumption, all the groundings of a first-order logic formula behave identical to each other (up to renaming of constants). Hence, the resulting potential function  $\Theta_j$ 's are also identical to each other.

2) If there are  $r$  unary MAX predicates in  $M_{\tilde{X}}$ , then each  $q_j$  can take  $R = 2^r$  possible values<sup>3</sup>.

Therefore, the value of the product  $\prod_{j=1}^m \Theta_j(q, s)$  in the RHS of Equation 4 depends only on the number of different types of values  $q_j$ 's take in  $q$  (and not on which  $q_j$  takes which value). Let  $\{v_1, v_2, \dots, v_R\}$  denote the set of  $R$  different values that  $q_j$ 's can take. Given a value  $v_l$ , let  $N_l$  denote the number of times  $v_l$  appears in  $q$ . Next, we state the following lemma.

**Lemma 4.** *The MMAP problem can be written as:*

$$\arg \max_q W_{M_{\tilde{X}}}(q) = \arg \max_{N_1, N_2, \dots, N_R} \sum_s \prod_{l=1}^R f_l(s)^{N_l}$$

subject to the constraints that  $\forall l, N_l \geq 0, N_l \in \mathbb{Z}$  and  $\sum_l N_l = m$ . Here,  $f_l(s) = \Theta_j(v_l, s)$ .

<sup>3</sup>since there are  $r$  predicate groundings for each  $j$  and each is Boolean valued

*Proof.* Proof follows from the fact that  $\Theta_j$ 's are symmetric to each other and that the  $q_j$ 's take a total of  $m$  possible (non-unique) assignments since  $\Delta_{\tilde{X}} = m$ .

We say that an assignment  $N_1, N_2, \dots, N_R$  subject to the constraints:  $\forall l, N_l \geq 0$  and  $\sum_l N_l = m$  is at *extreme* if  $\exists l$  such that  $N_l = m$ . Note that for  $R \geq 2$ , extreme assignment also implies that  $\exists l, N_l = 0$ . We have the following lemma.

**Lemma 5.** \* The solution to the MMAP formulation  $\arg \max_q W_{M_{\tilde{X}}}(q)$  lies at extreme iff solution to its equivalent formulation:

$$\arg \max_{N_1, N_2, \dots, N_R} \sum_s \prod_{l=1}^R f_l(s)^{N_l}$$

subject to the constraints  $\forall l, N_l \geq 0, N_l \in \mathbb{Z}$  and  $\sum_l N_l = m$  lies at extreme.

### 3.2.5 Proving Extreme

**Lemma 6.** Consider the optimization problem:

$$\arg \max_{N_1, N_2, \dots, N_R} \sum_s g(s) \times \prod_{l=1}^R f_l(s)^{N_l}$$

subject to the constraints  $N_l \geq 0, \sum_l N_l = m$ .  $g(s)$  is an arbitrary real-valued function independent of  $l$ . The solution of this optimization problem lies at extreme.

*Proof.* Note that it suffices to prove this theorem assuming  $N_l$ 's are real-valued. If the solution is at extreme with real-valued  $N_l$ 's, it must also be at extreme when  $N_l$ 's are further constrained to be integer valued. We will use induction on  $R$  to prove the result. Consider base case of  $R = 2$ , the function becomes  $\arg \max_{N_1} \sum_s f_1(s)^{N_1} f_2(s)^{m-N_1} \times g(s)$ . This function is convex and has its maximum value at  $N_1 = m$  or  $N_1 = 0$  (see supplement for a proof).

Assuming that the induction hypothesis holds for  $R = k$ . We need to show for the case when  $R = k + 1$ . We will prove it by contradiction. Assume that the solution to this problem does not lie at extreme. Then, in this solution, it must be the case that  $N_l \neq 0, \forall l$ . If not, we can then reduce the problem to a  $k$  sized one and apply our induction hypothesis to get an extreme solution. Also, clearly  $N_l < m, \forall l$ . Let  $N_{k+1}$  has the optimal value of  $N_{k+1}^*$  in this solution. Then, substituting the optimal value of this component in the expression, we can get the optimal value for  $(N_1, N_2, \dots, N_k)$  by solving  $\arg \max_{N_1, N_2, \dots, N_k} \sum_s g'(s) \times \prod_{l=1}^k f_l(s)^{N_l}$ , subject to  $\sum_{l=1}^k N_l = m - N_{k+1}^*$ . Here,  $g'(s) = g(s) \times f_{k+1}(s)^{N_{k+1}^*}$ . Using the induction hypothesis, the solution for this must be at extreme, i.e.  $\exists l, N_l = 0$  since  $k \geq 2$ . This is a contradiction.

**Corollary 1.** The solution to the optimization problem

$$\arg \max_{N_1, N_2, \dots, N_R} \sum_s \prod_{l=1}^R f_l(s)^{N_l}$$

subject to the constraints  $\forall l, N_l \geq 0, N_l \in \mathbb{Z}$  and  $\sum_l N_l = m$  lies at extreme.

**Theorem 1 (Proof):** Corollary 1 combined with Lemma 5, Lemma 4, Lemma 3, Lemma 2 and Lemma 1 proves the theorem.

### 3.3 SOM-R Rule for lifted MMAP

We will first define the SOM-R (SOM Reduce) equivalence class which is a sub-class of SOM. Following our notation, we will use  $\mathcal{Q}$  and  $\mathcal{S}$  to denote the set of MAX and SUM predicates, respectively in the MMAP problem.

**Definition 4.** We say that an equivalence class of variables  $\tilde{X}$  is SOM-R if (a)  $\tilde{X}$  is SOM (b)  $\forall P \in \mathcal{S}, P$  contains a variable from  $\tilde{X}$  OR  $\forall P \in \mathcal{S}, P$  does not have a variable from  $\tilde{X}$ .

Note that if  $|\mathcal{S}| = 1$ , then any SOM equivalence class is also necessarily SOM-R. Next, we exploit the properties of extreme assignments to show that domain of SOM-R variables can be reduced to a single constant for MMAP inference. We start with the definition of a reduced MLN.

**Definition 5.** (Reduced MLN) Let  $\{(f_i, w_i)\}_{i=1}^n$  denote the set of (weighted) formulas in  $M$ . Let  $\tilde{X}$  be a SOM-R equivalence class with  $|\Delta_{\tilde{X}}| = m$ . We construct a reduced MLN  $M^r$  by considering the following 2 cases:

CASE 1:  $\forall P \in \mathcal{S}, P$  contains a variable from  $\tilde{X}$

- $\forall f_i \in F(M)$  containing a variable  $X \in \tilde{X}$ , add  $(f_i, w_i)$  to  $M^r$ .
- $\forall f_i \in F(M)$  not containing a variable  $X \in \tilde{X}$ , add  $(f_i, \frac{1}{m} \times w_i)$  to  $M^r$ .

CASE 2:  $\forall P \in \mathcal{S}, P$  does not contain a variable from  $\tilde{X}$

- $\forall f_i \in F(M)$  containing a variable  $X \in \tilde{X}$ , add  $(f_i, w_i \times m)$  to  $M^r$ .
- $\forall f_i \in F(M)$  not containing a variable  $X \in \tilde{X}$ , add  $(f_i, w_i)$  to  $M^r$ .

In each case, we reduce the domain of  $\tilde{X}$  to a single constant in  $M^r$ .

We are ready to state our SOM-R rule for lifted MMAP.

**Theorem 2.** (SOM-R Rule for MMAP) Let  $\tilde{X}$  be a SOM-R equivalence class. Let  $M^r$  be the reduced MLN in which domain of  $\tilde{X}$  has been reduced to single constant. Then, MMAP problem can be equivalently solved over  $M^r$ .

*Proof.* Let  $\mathcal{Q}$  denote the set of MAX predicates in the problem. We prove the above theorem in two parts. In Lemma 7 below, we show that for every extreme assignment (with respect to  $\tilde{X}$ )  $q$  to groundings of  $\mathcal{Q}$  in  $M$ , there is a corresponding extreme assignment  $q^r$  in  $M^r$  (and vice-versa). In Lemma 8, we show that given two extreme assignments,  $q$  and  $q^r$  for the respective MLNs, the MMAP value at  $q$  (in  $M_{\tilde{X}}$ ) is a monotonically increasing function of the MMAP value at  $q^r$  (in  $M_{\tilde{X}}^r$ ). These two facts combined with the fact that MMAP solution to the original problem is at extreme (using Theorem 1) prove the desired result. Next we prove each result in turn.

**Lemma 7.** *Let  $\mathbf{q}$  (resp.  $\mathbf{q}^r$ ) denote the sets of extreme assignments to the groundings of  $\mathcal{Q}$  in  $M$  (resp.  $M^r$ ). There exists a one to one to mapping between  $\mathbf{q}$  and  $\mathbf{q}^r$ .*

*Proof.* Instead of directly working with  $M$  and  $M^r$ , we will instead prove this lemma for the corresponding variablized MLNs  $M_{\tilde{X}}$  and  $M_{\tilde{X}}^r$ . This can be done since the process of variablization preserves the distribution as well as the set of extreme assignments. Let  $q$  denote an extreme assignment to MAX predicates in  $M_{\tilde{X}}$ . We will construct a corresponding assignment  $q^r$  for MAX predicate in  $M_{\tilde{X}}^r$ . Since  $\tilde{X}$  is SOM-R,  $M_{\tilde{X}}$  has only unary and propositional predicates, whereas  $M_{\tilde{X}}^r$  is full ground since the domain of  $\tilde{X}$  is reduced to a single constant.

First, let us consider a propositional MAX predicate  $P$  in  $M_{\tilde{X}}$ . Since  $P$  is ground both in  $M$  and  $M^r$ , we can assign the value of  $P$  in  $q^r$  to be same as  $q$ . Next, let us consider a unary predicate  $P$ . Let the assignments to the  $m$  groundings of  $P$  in  $q$  be given by the set  $\{q_{P_j}\}$  where  $1 \leq j \leq m$ . Since  $q$  is extreme, each element in the set  $\{q_{P_j}\}$  takes the same truth value. We can simply assign this value to the ground appearance of  $P$  in  $M_{\tilde{X}}$ . Hence, we get a mapping from  $q$  to  $q^r$ . It is easy to see that we can get a reverse mapping from  $q^r$  to  $q$  in a similar manner. Hence, proved.

Next, we state the relationship between the MMAP values obtained by the extreme assignments in  $M$  and  $M^r$ .

**Lemma 8.\*** *Let  $M$  be an MLN and  $M^r$  be the reduced MLN with respect to the SOM-R equivalence class  $\tilde{X}$ . Let  $q$  and  $q^r$  denote two corresponding extreme assignments in  $M$  and  $M^r$ , respectively. Then,  $\exists$  a monotonically increasing function  $g$  such that  $W_M(q) = g(W_{M^r}(q^r))$ .*

The proof of Lemma 8 exploits inversion elimination and symmetry of potential functions over a variablized MLN similar to their use in Section 3.2. These combined with Lemma 7 become our key insights for reducing the complexity of MMAP inference significantly compared to existing methods (see supplement for details).

**Corollary 2.** *SOM-R rule for MMAP problem subsumes SO rule for MAP problem given by Mittal et al. (2014).*

The corollary follows from the fact that MAP is a special case of MMAP when all the predicates are MAX.

## 4 ALGORITHMIC FRAMEWORK

SOM-R rule can be combined with existing lifted inference rules such as lifted decomposition and conditioning (Jha et al. 2010; Gogate and Domingos 2011) (with minor modifications) to yield a powerful algorithm for solving MMAP (see Algorithm 1). The algorithm takes as input an MLN  $M$ , the set of MAX predicates  $\mathcal{Q}$ , SUM predicates  $\mathcal{S}$  and a ground MMAP solver  $gSol$ . It has six steps. In the first step, the algorithm checks to see if the MLN, along with  $\mathcal{Q}$  and  $\mathcal{S}$  can be partitioned into disjoint MLNs that do not share any ground atoms. If this condition is satisfied, then the MMAP solution can be constructed by solving each component independently and simply concatenating the individual solutions. In the next three steps, we apply the decomposer (Jha et al. 2010), SOM-R (this work) and binomial rules (Jha et al. 2010; Gogate and Domingos 2011) in order. The former two reduce the domain of all logical variables in the equivalence class to a constant and thus yield exponential reductions in complexity. Therefore, they are applied before the binomial rule which creates  $O(m)$  ( $|\Delta_{\tilde{X}}| = m$ ) smaller sub-problems. In the algorithm,  $M^d$  refers to an MLN obtained from  $M$  by setting the domain of  $\tilde{X}$  to a single constant and we assume that  $|\Delta_{\tilde{X}}| = m$ . Similarly,  $M^r$  refers to the MLN obtained from  $M$  by applying the SOM-R rule (see Definition 5).

The binomial rule (steps 4a and 4b) efficiently conditions on the unary predicates and can be applied over the SUM as well as MAX predicates. However, care must be taken to ensure that all MAX predicates are instantiated before the SUM predicates. Therefore, the binomial rule is applied over the SUM predicates only when the MLN has no MAX predicates (Step 4b). In the algorithm,  $M_k$  refers to the MLN obtained from  $M$  by setting exactly  $k$  groundings of  $P$  to true and the remaining to false.

If none of the lifting rules are applicable and the MLN has only ground atom, we return the solution returned by the propositional solver  $gSol$ . Otherwise, if not all predicates are ground, we resort to partial grounding, namely we heuristically ground a logical variable and recurse on the corresponding MLN  $M'$ .

Finally, note that the algorithm returns the exponentiated weight of the MMAP assignment. The assignment can be recovered by tracing the recursion backwards.

**Heuristics:** (a) Binomial: In case of multiple possible binomial applications, we pick the one which results in

---

**Algorithm 1** Lifted MMAP

---

**Input:** MLN  $M, \mathcal{Q}, \mathcal{S}, gSol$ **Output:** MMAP value**Begin:**

```
//1. Disjoint Sub-Theories
if  $M$  can be partitioned into disjoint MLNs  $M_1, \dots, M_t$  that
share no atoms then
    return  $\prod_{i=1}^t$  liftedMMAP( $M_i, \mathcal{Q}_i, \mathcal{S}_i$ )
//2. Decomposer
if there exists a decomposer  $\tilde{X}$  in  $M$  then
    return [liftedMMAP( $M^d, \mathcal{Q}, \mathcal{S}, gSol$ )] $^m$ ;
//3. SOM-R (see Defn. 5)
if there exists a SOM-R class  $\tilde{X}$  in  $M$  then
    return liftedMMAP( $M^r, \mathcal{Q}, \mathcal{S}, gSol$ );
//4a. Binomial over MAX
if there exists a unary predicate  $P \in \mathcal{Q}$  then
    return  $\max_k$  liftedMMAP( $M_k, \mathcal{Q} - \{P\}, \mathcal{S}, gSol$ );
//4b. Binomial over SUM
if  $\mathcal{Q} = \emptyset$  and there exists a unary predicate  $P \in \mathcal{S}$  then
    return  $\sum_{k=0}^m \binom{m}{k}$  liftedMMAP( $M_k, \mathcal{Q}, \mathcal{S} - \{P\}, gSol$ );
//5. Check if fully Ground
if  $M$  is fully Ground then
    return apply( $M', \mathcal{Q}, \mathcal{S}, gSol$ );
else
    //6. Partial Grounding
     $M' =$  Heuristically ground an equivalence class  $\tilde{X}$  in  $M$ 
    return liftedMMAP( $M', \mathcal{Q}, \mathcal{S}, gSol$ );
```

**End.**

---

the application of other lifting rules (in the priority order described above) using a one step look ahead. In case of a tie, we pick the one with maximum domain size.

(b) Partial Grounding: We pick the equivalence class which results in further application of lifting rules (in the priority order) using a one step look ahead. In case of a tie, we pick the one which has smallest domain size.

## 5 EXPERIMENTS

The goal of our experiments is two fold. First, we would like to examine the efficacy of lifting for MMAP. Second, we would like to analyze the contribution of SOM-R rule in lifting. Towards this end, we compare the following three algorithms: (1) Ground: ground inference with no lifting whatsoever (2) Lifted-Basic: lifted inference without use of the SOM-R rule<sup>4</sup> (3) Lifted-SOM-R: using all our lifting rules including SOM-R. For ground inference, we use a publicly available<sup>5</sup> base (exact) solver built on top of And/Or search developed by Marinescu et al. (2014).

We experiment with three benchmark MLNs: (1) Stu-

---

<sup>4</sup>We use the rules described in Algorithm 1. For Lifted-Basic, too many applications of the binomial rule led to blow up. So, we restricted the algorithm to a single binomial application and before any partial grounding. Lifted-SOM-R had no such issues.

<sup>5</sup><https://github.com/radum2275/merlin>

dent (Sarkhel et al. 2014) (2) IMDB (Mittal et al. 2016) (3) Friends & Smokers (FS) (Domingos and Lowd 2009). All the datasets are described in the lower part of Figure 1 along with the MAP predicates used in each case; the remaining predicates are treated as marginal predicates. Weights of the formulas were manually set.

We compare the performance of the three algorithms on two different metrics: (a) time taken for inference (b) memory used. We used a time-out of 30 minutes for each run. Memory was measured in terms of the number of formulas in the ground network in each case. We do not compare the solution quality since all the algorithms are guaranteed to produce MMAP assignments with same (optimal) probability. All the experiments were run on a 2.20 GHz Xeon(R) E5-2660 v2 server with 10 cores and 62 GB RAM.

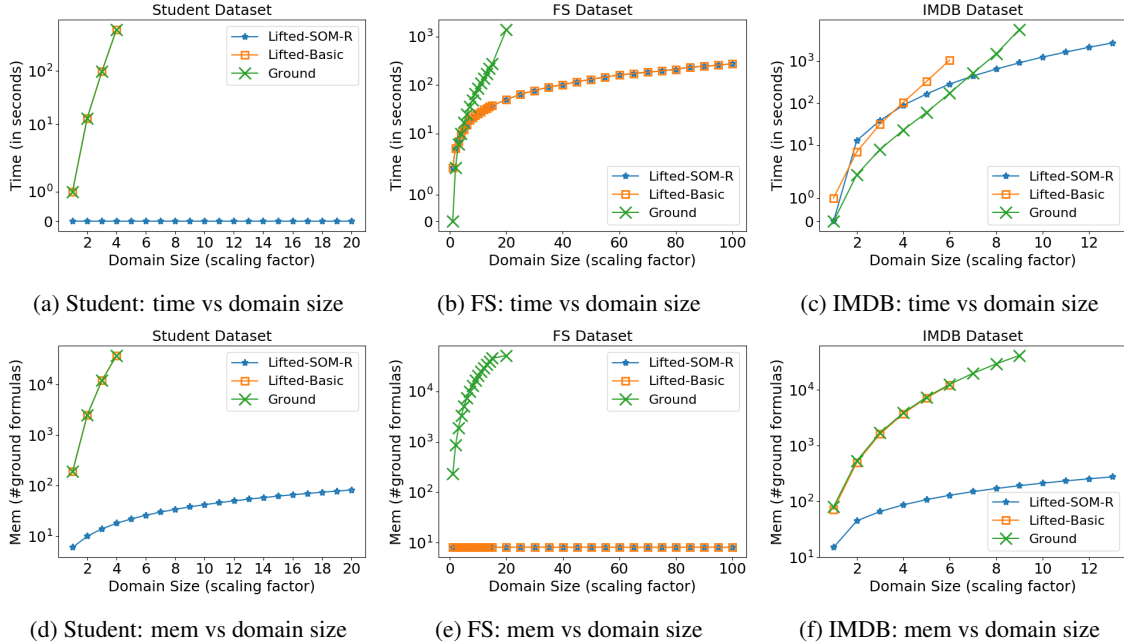
**Results:** For each of the graphs in Figure 1, we plot time (memory) on y-axis (log-scale) and domain size on x-axis. Time is measured in seconds. Since we are primarily concerned about the scaling behavior, we use the number of ground formulae as a proxy for the actual memory usage. Domain size is measured as a function of a scaling factor, which is the number by which (all of) the starting domain sizes are multiplied. We refer to domain descriptions (Figure 1) for the starting sizes.

Figures 1a and 1d compare the performance of the three algorithms on the Student dataset. None of the lifting rules apply for Lifted-Basic. Hence, its performance is identical to Ground. For Lifted-SOM-R, all the variables (except teacher(T)) can be reduced to a single constant, resulting in significant reduction in the size of the ground theory. Lifted-SOM-R is orders of magnitude better than Ground and Lifted-Basic for both time and memory.

Figures 1b and 1e compare the three algorithms on the FS dataset. Here, Lifted-Basic performs identical to Lifted-SOM-R. This is because binomial rule applies in the beginning on Smokes, following which theory decomposes. We never need to apply SOM-R rule on this domain. Both Lifted-SOM-R and Lifted-Basic perform significantly better than Ground on this domain (in both time and memory).

IMDB dataset (Figures 1c and 1f) presents a particularly interesting case of interspersed application of rules. For Lifted-SOM-R, SOM-R rule applies on movie(M) variables, simplifying the theory following which binomial rule can be applied on Mov, Dir and Act predicates. Theory decomposes after these binomial applications. For Lifted-Basic, though binomial rule can be applied on Dir, Act the movie variables still remain, eventually requiring for partial grounding. Surprisingly, Ground does slightly better than both the lifted approaches for





Student (Sarkhel et al. 2014)	IMDB (Mittal et al. 2016)
Teaches(T, C) $\wedge$ Takes(S, C) $\Rightarrow$ JobOffer(S, M)	WorksWith(P1,P2) $\Rightarrow$ Act(P1); WorksWith(P1,P2) $\Rightarrow$ Dir(P2);
<b>MAP Predicate:</b> Takes(S, C), JobOffer(S, M)	Dir(P1) $\wedge$ Act(P2) $\wedge$ Mov(M,P1) $\wedge$ Mov(M,P2) $\Rightarrow$ WorksWith(P2,P1);
<b>size:</b> teachr(T):2,course(C):3,comp(M):4,stud(S):6	Dir(P1) $\wedge$ Act(P2) $\wedge$ Mov(M,P2) $\wedge$ WorksWith(P2,P1) $\Rightarrow$ Mov(M,P1);
<b>FS (Domingos and Lowd 2009)</b>	Dir(P1) $\wedge$ Act(P2) $\wedge$ Mov(M,P1) $\wedge$ WorksWith(P2,P1) $\Rightarrow$ Mov(M,P2);
Smokes(P) $\Rightarrow$ Cancer(P);	Dir(P1) $\wedge$ Act(P2) $\Rightarrow$ WorksWith(P2,P1);
Smokes(P1) $\wedge$ Friend(P1, P2) $\Rightarrow$ Smokes(P2);	<b>MAP Predicates:</b> Act(P), Dir(P), Mov(M,P)
<b>MAP Predicates:</b> Smokes(P), Cancer(P)	<b>size:</b> person(P):3, movie(M):2
<b>size:</b> person(P):5	

Figure 1: Results and rules of Student, FS and IMDB datasets. "size" gives initial domain sizes for each case.

smaller domains for time. This is due to the overhead of solving multiple sub-problems in binomial without much gain since domains are quite small. Lifted-SOM-R has a much better scaling behavior for larger domains. It also needs significantly less memory compared to both other approaches.

In none of the above cases, Lifted-SOM-R has to ever partially ground the theory making a very strong case for using Lifted-SOM-R for MMAP inference in many practical applications. Overall, our experiments clearly demonstrate the utility of SOM-R in the scenarios where other lifting rules fail to scale.

## 6 CONCLUSION

We present the first lifting technique for MMAP. Our main contribution is the SOM-R rule, which states that the domain of a class of equivalence variables, referred to as SOM-R, can be reduced to a single constant for the purpose of MMAP inference. We prove the correctness of our rule through a series of problem transformations followed by the properties of what we refer to as extreme

assignments. Our experiments clearly demonstrate the efficacy of our approach on benchmark domains. Directions for future work include coming up with additional lifting rules, approximate lifting and lifting in presence of constraints (Mittal et al. 2015), all in the context of MMAP, and experimenting with a wider set of domains.

## Acknowledgements

Happy Mittal is supported by the TCS Research Scholars Program. Vibhav Gogate and Parag Singla are supported by the DARPA Explainable Artificial Intelligence (XAI) Program with number N66001-17-2-4032. Parag Singla is supported by IBM Shared University Research Award and the Visvesvaraya Young Faculty Research Fellowship by the Govt. of India. Vibhav Gogate is supported by the National Science Foundation grants IIS-1652835 and IIS-1528037. Any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views or official policies, either expressed or implied, of the funding agencies.

## References

- de Salvo Braz, R.; Amir, E.; and Roth, D. 2005. Lifted first-order probabilistic inference. In *Proc. of IJCAI-05*, 1319–1325.
- de Salvo Braz, R.; Amir, E.; and Roth, D. 2006. MPE and partial inversion in lifted probabilistic variable elimination. In *Proc. of AAAI-06*, 1123–1130.
- Domingos, P., and Lowd, D. 2009. *Markov Logic: An Interface Layer for Artificial Intelligence*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.
- G. Van den Broeck; Taghipour, N.; Meert, W.; Davis, J.; and Raedt, L. D. 2011. Lifted probabilistic inference by first-order knowledge compilation. In *Proc. of IJCAI-11*, 2178–2185.
- Getoor, L., and Taskar, B., eds. 2007. *Introduction to Statistical Relational Learning*. MIT Press.
- Gogate, V., and Domingos, P. 2011. Probabilistic theorem proving. In *Proc. of UAI-11*, 256–265.
- Jha, A. K.; Gogate, V.; Meliou, A.; and Suciu, D. 2010. Lifted inference seen from the other side : The tractable features. In *Proc. of NIPS-10*, 973–981.
- Kersting, K.; Ahmadi, B.; and Natarajan, S. 2009. Counting belief propagation. In *Proc. of UAI-09*, 277–284.
- Kimmig, A.; Mihalkova, L.; and Getoor, L. 2015. Lifted graphical models: a survey. *Machine Learning* 99(1):1–45.
- Liu, Q., and Ihler, A. T. 2013. Variational algorithms for marginal MAP. *Journal of Machine Learning Research* 14(1):3165–3200.
- Maaten, L.; Welling, M.; and Saul, L. K. 2011. Hidden-unit conditional random fields. In *Proc. of AISTATS-11*, 479–488.
- Marinescu, R.; Dechter, R.; and Ihler, A. T. 2014. And/or search for marginal MAP. In *Proc. of UAI-14*, 563–572.
- Mittal, H.; Goyal, P.; Gogate, V.; and Singla, P. 2014. New rules for domain independent lifted MAP inference. In *Proc. of NIPS-14*, 649–657.
- Mittal, H.; Mahajan, A.; Gogate, V.; and Singla, P. 2015. Lifted inference rules with constraints. In *Proc. of NIPS-15*, 3519–3527.
- Mittal, H.; Singh, S. S.; Gogate, V.; and Singla, P. 2016. Fine grained weight learning in markov logic networks. In *Proc. of IJCAI-16 Wkshp. on Statistical Relational AI*.
- Mladenov, M.; Kersting, K.; and Globerson, A. 2014. Efficient lifting of map lp relaxations using k-locality. In *Proc. of AISTATS-14*, 623–632.
- Niepert, M. 2012. Markov chains on orbits of permutation groups. In *Proc. of UAI-14*, 624–633.
- Park, J. 2002. MAP Complexity Results and Approximation Methods. In *Proc. of UAI-02*, 388–396.
- Poole, D. 2003. First-order probabilistic inference. In *Proc. of IJCAI-03*, 985–991.
- Russell, S. J., and Norvig, P. 2010. *Artificial Intelligence - A Modern Approach*. Pearson Education.
- Sarkhel, S.; Venugopal, D.; Singla, P.; and Gogate, V. 2014. Lifted MAP inference for Markov logic networks. In *Proc. of AISTATS-14*, 895–903.
- Sarkhel, S.; Singla, P.; and Gogate, V. G. 2015. Fast lifted MAP inference via partitioning. In *Proc. of NIPS-15*, 3240–3248.
- Singla, P., and Domingos, P. 2008. Lifted first-order belief propagation. In *Proc. of AAAI-08*, 1094–1099.
- Singla, P., and Mooney, R. 2011. Abductive Markov logic for plan recognition. In *Proc. of AAAI-11*, 1069–1075.
- Singla, P.; Nath, A.; and Domingos, P. 2014. Approximate lifted belief propagation. In *Proc. of AAAI-14*, 2497–2504.
- Van den Broeck, G., and Darwiche, A. 2013. On the complexity and approximation of binary evidence in lifted inference. In *Proc. of NIPS-13*, 2868–2876.
- Venugopal, D., and Gogate, V. 2012. On lifting the Gibbs sampling algorithm. In *Proc. of NIPS-12*, 1664–1672.
- Xue, Y.; Ermon, S.; Gomes, C. P.; Selman, B.; et al. 2016. Solving marginal MAP problems with NP oracles and parity constraints. In *Proc. of NIPS-16*, 1127–1135.

---

# PAC-Reasoning in Relational Domains

---

**Ondřej Kuželka**  
Department of CS  
KU Leuven  
Leuven, Belgium

**Yuyi Wang**  
Disco Group  
ETH Zurich  
Zurich, Switzerland

**Jesse Davis**  
Department of CS  
KU Leuven  
Leuven, Belgium

**Steven Schockaert**  
School of CS & Informatics  
Cardiff University  
Cardiff, UK

## Abstract

We consider the problem of predicting plausible missing facts in relational data, given a set of imperfect logical rules. In particular, our aim is to provide bounds on the (expected) number of incorrect inferences that are made in this way. Since for classical inference it is in general impossible to bound this number in a non-trivial way, we consider two inference relations that weaken, but remain close in spirit to classical inference.

## 1 INTRODUCTION

In this paper we study several forms of logical inference for predicting plausible missing facts in relational data. While a variety of approaches have already been studied for this task, ranging from (relational versions of) probabilistic graphical models [19, 4] to neural-network architectures [24, 20] and graph-based methods [15, 16], logic-based inference has several advantages over these other forms of inference. For example, logic-based inference is explainable: there is a proof for any derived statement, which can, in principle, be shown to a human user. It is also more transparent than most other methods, in the sense that a knowledge base as a whole can be understood and modified by domain experts. On the other hand, classic logical inference can be very brittle when some of the rules which are used are imperfect, or some of the initial facts may be incorrect.

Statistical relational learning approaches, such as Markov logic networks [19] or probabilistic logic programming [4], offer a solution to this latter problem, but they require learning a joint probability distribution over the set of possible worlds. This distribution is typically estimated from one or several large examples using maximum likelihood, which essentially corresponds to finding

a maximum-entropy distribution given by a set of sufficient statistics. However, there are usually no guarantees on the learned distributions beyond guarantees for the sufficient statistics (see, e.g., [12]), which means that we do not have much control over the quality of the predictions. Moreover, these models are not easy to modify, and are not always easy to explain because the way in which probabilities are computed can simply be too complex.

In this paper we focus on forms of inference that stay as close to classical logic as possible while not breaking completely when the given theory happens to be “mildly” inconsistent with the data. This problem of reasoning under inconsistency has a long tradition in the field of artificial intelligence, with common solutions including the use of paraconsistent logics [3, 18], belief revision [8] (and related inconsistency repair mechanisms [11]), and argumentation-based inference [7, 2]. In contrast to these approaches, however, our specific aim is to study forms of inference that can allow us to bound the (expected) number of mistakes that are made. To this end, we introduce two inference relations called *k-entailment* and *voting entailment*, both of which are close to classical logic, and in particular do not require rules to be weighted. We define them such that errors produced by imperfect rules would not necessarily propagate too much in the given relational data.

As our main contribution, we are able to show that in a relational learning scenario from [12], in which a (large) training example and a test example are sampled from a hidden relational structure, there are non-trivial PAC-type bounds on the number of errors that a theory learned on the training example produces on the test example. From this perspective, our work can also be seen as a relational-learning counterpart of PAC semantics [23].

**Technical contributions.** The results presented in this paper rest mainly on the following two technical contributions: (i) the introduction of bounds on the worst case behavior of the considered inference relations, and (ii)

new concentration inequalities for sampling from relational data without replacement that allow us to bound the (expected) test error as a function of the training error, in the spirit of classical PAC-learning results [22].

## 2 PRELIMINARIES

In this paper we consider a function-free first-order logic language  $\mathcal{L}$ , which is built from a set of constants  $Const$ , variables  $Var$ , and predicates  $Rel = \bigcup_i Rel_i$ , where  $Rel_i$  contains the predicates of arity  $i$ . We assume an untyped language. For  $a_1, \dots, a_k \in Const \cup Var$  and  $R \in Rel_k$ , we call  $R(a_1, \dots, a_k)$  an atom. If  $a_1, \dots, a_k \in Const$ , this atom is called ground. A literal is an atom or its negation. The formula  $\alpha_0$  is called a grounding of  $\alpha$  if  $\alpha_0$  can be obtained by replacing each variable in  $\alpha$  with a constant from  $Const$ . A formula is called closed if all variables are bound by a quantifier. A possible world  $\omega$  is defined as a set of ground atoms. The satisfaction relation  $\models$  is defined in the usual way. A substitution is a mapping from variables to terms.

## 3 PROBLEM SETTING

First we describe the learning setting considered in this paper. It follows the setting from [12], which was used to study the estimation of relational marginals.

An example is a pair  $(\mathcal{A}, \mathcal{C})$ , with  $\mathcal{C}$  a set of constants and  $\mathcal{A}$  a set of ground atoms which only use constants from  $\mathcal{C}$ . An example is intended to provide a complete description of the world, hence any ground atom over  $\mathcal{C}$  which is not contained in  $\mathcal{A}$  is implicitly assumed to be false. Note that this is why we have to explicitly specify  $\mathcal{C}$ , as opposed to simply considering the set of constants appearing in  $\mathcal{A}$ .

In practice, we usually only have partial information about some example of interest. The problems we consider in this paper relate to how we can then reason about the probability that a given ground atom is true (i.e. belongs to the example). To estimate such probabilities, we assume that we are given a fragment of the example, which we can use as training data. Specifically, let  $\Upsilon = (\mathcal{A}, \mathcal{C})$  be an example and  $\mathcal{S} \subseteq \mathcal{C}$ . The fragment  $\Upsilon\langle\mathcal{S}\rangle = (\mathcal{B}, \mathcal{S})$  is defined as the restriction of  $\Upsilon$  to the constants in  $\mathcal{S}$ , i.e.  $\mathcal{B}$  is the set of all atoms from  $\mathcal{A}$  which only contain constants from  $\mathcal{S}$ . In a given example, any closed formula  $\alpha$  is either true or false. To assign probabilities to formulas in a meaningful way, we consider how often the formula is satisfied in small fragments of the given example.

**Definition 1** (Probability of a formula [12]). *Let  $\Upsilon = (\mathcal{A}, \mathcal{C})$  be an example and  $k \in \mathbb{N}$ . For a closed formula  $\alpha$*

*without constants, we define its probability as follows<sup>1</sup>:*

$$Q_{\Upsilon,k}(\alpha) = P_{\mathcal{S} \sim \text{Unif}(\mathcal{C},k)} [\Upsilon\langle\mathcal{S}\rangle \models \alpha]$$

*where  $\text{Unif}(\mathcal{C}, k)$  denotes uniform distribution on size- $k$  subsets of  $\mathcal{C}$ .*

Clearly  $Q_{\Upsilon,k}(\alpha) = \frac{1}{|\mathcal{C}_k|} \cdot \sum_{\mathcal{S} \in \mathcal{C}_k} \mathbb{1}(\Upsilon\langle\mathcal{S}\rangle \models \alpha)$  where  $\mathcal{C}_k$  is the set of all size- $k$  subsets of  $\mathcal{C}$ .

The above definition is also extended straightforwardly to probabilities of sets of formulas (which we will also call *theories* interchangeably). If  $\Phi$  is a set of formulas, we set  $Q_{\Upsilon,k}(\Phi) = Q_{\Upsilon,k}(\bigwedge \Phi)$  where  $\bigwedge \Phi$  denotes the conjunction of all formulas in  $\Phi$ .

**Example 1.** *Let  $sm/1$  be a unary predicate denoting that someone is a smoker, e.g.  $sm(alice)$  means that  $alice$  is a smoker. Let us have an example  $\Upsilon = (\{fr(alice, bob), sm(alice), sm(eve)\}, \{alice, bob, eve\})$ , and formulas  $\alpha = \forall X : sm(X)$  and  $\beta = \exists X, Y : fr(X, Y)$ . Then, for instance,  $Q_{\Upsilon,1}(\alpha) = 2/3$ ,  $Q_{\Upsilon,2}(\alpha) = 1/3$  and  $Q_{\Upsilon,2}(\beta) = 1/3$ .*

**Definition 2** (Masking). *A masking process is a function  $\kappa$  from examples to ground conjunctions that assigns to any  $\Upsilon = (\mathcal{A}, \mathcal{C})$  a conjunction of ground literals  $\beta$  such that  $\Upsilon \models \beta$ . We also define  $\kappa(\Upsilon)\langle\mathcal{S}\rangle$  to be the conjunction consisting of all literals from  $\kappa(\Upsilon)$  that contain only constants from  $\mathcal{S}$ .*

Unlike examples, masked examples only encode partial information about the world. This is why they are encoded using conjunctions of literals, so we can explicitly encode which atoms we know to be false.

**Example 2.** *Let  $\Upsilon = \{sm(alice), fr(alice, bob), \{alice, bob\}\}$ . Then a masking process  $\kappa$  may, for instance, yield  $\kappa(\Upsilon) = \neg sm(bob) \wedge sm(alice)$ . In this case  $\kappa(\Upsilon)$  retains the information that  $alice$  is a smoker and  $bob$  is not, but it no longer contains any information about their friendship relation.*

Next we introduce the statistical setting considered in this paper.

**Definition 3** (Learning setting). *Let  $\aleph = (\mathcal{A}_{\aleph}, \mathcal{C}_{\aleph})$  be an example and  $\kappa$  be a masking function. Let  $\mathcal{C}_{\Upsilon} \subseteq \mathcal{C}_{\aleph}$  and  $\mathcal{C}_{\Gamma} \subseteq \mathcal{C}_{\aleph}$  be uniformly sampled subsets of size  $n$  and  $u$ , respectively. We call  $\Upsilon = \aleph\langle\mathcal{C}_{\Upsilon}\rangle$  the training example and  $\Gamma = \aleph\langle\mathcal{C}_{\Gamma}\rangle$  the test example. We assume that the learner receives  $\Upsilon$  in the training phase and  $\kappa(\Gamma)$  in the test phase.*

With slight abuse of terminology, we will sometimes say that  $\Upsilon$  and  $\Gamma$  are sampled from  $\aleph$ .

<sup>1</sup>We will use  $Q$  for probabilities of formulas as defined in this section, to avoid confusion with other ‘‘probabilities’’ we deal with in the text.

In addition to the training example  $\Upsilon$  and masked test example  $\kappa(\Gamma)$ , we will assume that we are given a set of formulas  $\Phi$  (which we will also refer to as rules). Our main focus will be on how these formulas can be used to recover as much of  $\Gamma$  as possible. Rather than specifying a loss function that should be minimized, we want to find a form of inference which allows us to provide bounds on the (expected) number of incorrect literals that can be inferred from  $\{\kappa(\Gamma)\} \cup \Phi$ . Note that in this case, the training example  $\Upsilon$  is used to estimate the accuracy of the set of formulas. We also analyze the case where the rules are learned from the training example  $\Upsilon$  (in the spirit of classical PAC-learning results).

Among others, the setting from Definition 3 is close to how Markov logic networks are typically used. For instance, when training Markov logic networks, one typically starts with a training example that contains all facts (i.e. nothing is unknown about the training set), on which a model is trained. This model is then used to predict unknown facts about a test example. However, unlike for Markov logic networks, we do not attempt to learn a probability distribution. It was shown in [14] that models based on classic logical inference, like those considered in this paper, work well in practice for relational inference from evidence sets containing a *small* number of constants (domain elements). Thus, such models are also of considerable practical interest.

## 4 REASONING WITH INACCURATE RULES

When reasoning with imperfect rules, using classical inference can have drastic consequences, as we will illustrate in Section 4.1. Even a single mistake can lead to many errors, since an incorrectly derived literal can be used as the basis for further inferences. This means that classical inference is not suitable for the considered setting, even in cases where the given rules have perfect accuracy on the training example. Intuitively, to allow for any meaningful bounds to be derived, we need to prevent arbitrarily long chains of inference. To this end, we propose and motivate the use of a restricted form of inference, called  $k$ -entailment, in Section 4.2. A further restriction on inferences, based on a form of voting, is subsequently discussed in Section 4.3. In Section 5 we will then show which bounds can be derived for these two restricted forms of inference.

### 4.1 WHEN CLASSICAL REASONING LEADS TO ERRORS

The next example, which is related to label propagation as studied e.g. in [26], shows that classic logical reasoning

on the obtained relational sample may produce *many* mistakes even when all the available rules are very *accurate*.

**Example 3.** Let  $k = 2$ ,  $\Gamma = \{\{rare(c_1)\}, \{c_1, c_2, \dots, c_{1000000}\}, \text{ and } \alpha = \forall X, Y : rare(X) \Rightarrow rare(Y)\}$ . While the rule does not intuitively make sense, its accuracy is actually very high  $Q_{\Gamma,k}(\alpha) = 1 - 999999/(0.5 \cdot 1000000 \cdot 999999) = 0.999998$ . When we apply this rule with the evidence  $rare(c_1)$ , we derive  $rare(c_2), \dots, rare(c_{1000000})$ , all of which are incorrect (i.e. not included in  $\Gamma$ ).

Note that in this paper, we are interested in worst-case behavior, in the sense that the masking process which is used may be seen as adversarial. The next example further illustrates how adversarial masking processes can lead to problems, even for rules with near-perfect accuracy.

**Example 4.** Let  $k = 2$ ,  $\Gamma = \{\{rare(c_1), e(c_1, c_2), e(c_2, c_3), \dots, e(c_{999999}, c_{1000000})\}, \{c_1, c_2, \dots, c_{1000000}\}, \text{ and } \alpha = \forall X, Y : rare(X) \wedge e(X, Y) \Rightarrow rare(Y)\}$ . In this case, there is only one size- $k$  subset of  $C_{\Gamma}$  where the formula  $\alpha$  does not hold, so the accuracy is even higher than in the previous example. Yet the adversarial masking process can select evidence consisting of all true positive literals from  $\Gamma$ , i.e. the evidence will consist of the  $rare(c_1)$  literal and all the  $e/2$  literals from  $\Gamma$ . Then the set of errors that are made when using the formula  $\alpha$  will be the same as in Example 3, despite the fact that the rule is almost perfect on  $\Gamma$ .

Note that in the examples above, we had perfect knowledge of the accuracy of the rule  $\alpha$  on the test example (i.e. we knew the value of  $Q_{\Gamma,k}(\alpha)$ ). In practice, this accuracy needs to be estimated from the training example. In such cases, it can thus happen that a rule  $\alpha$  has accuracy 1 on the training example  $\Upsilon$ , but still produces many errors on  $\kappa(\Gamma)$ . We will provide PAC-type bounds for this setting with estimated accuracies in Sections 5. First, however, in Section 4.2 and 4.3 we will look at how bounds can be provided on the number of incorrectly derived literals in the case where  $Q_{\Gamma,k}(\alpha)$  is known. As the above examples illustrate, to obtain reasonable bounds, we will need to consider forms of inference which are weaker than classical entailment.

### 4.2 BOUNDED REASONING USING $k$ -ENTAILMENT

We saw that even for formulas which hold for almost all subsets of  $\Gamma$ , the result of using them for inference can be quite disastrous. This was to a large extent due to the fact that we had inference chains involving a large number of domain elements (constants). This observation suggests a natural way to restrict the kinds of inferences that can be made when imperfect rules are involved.

**Definition 4** (*k*-entailment). *Let  $k$  be a non-negative integer,  $\Upsilon = (\mathcal{A}, \mathcal{C})$  be an example,  $\kappa$  be a masking process, and  $\Phi$  be a set of closed formulas. We say that a ground formula  $\varphi$  is *k*-entailed by  $\Phi$  and  $\kappa(\Upsilon)$ , denoted  $\{\kappa(\Upsilon)\} \cup \Phi \models_k \varphi$ , if there is a  $\mathcal{C}' \subseteq \mathcal{C}$  such that  $|\mathcal{C}'| \leq k$ ,  $\text{const}(\varphi) \subseteq \mathcal{C}'$ ,  $\{\kappa(\Upsilon)\langle \mathcal{C}' \rangle\} \cup \Phi$  is consistent and  $\{\kappa(\Upsilon)\langle \mathcal{C}' \rangle\} \cup \Phi \models \varphi$ .*

In other words, a formula  $\phi$  is *k*-entailed by  $\Upsilon$  and  $\Phi$  if it can be proved using  $\Phi$  together with a fragment of  $\kappa(\Upsilon)$  induced by no more than  $k$  constants, with the additional condition that  $\Phi$  and this fragment are not contradictory.

**Example 5.** *Let*

$$\begin{aligned} \Upsilon &= (\{fr(\text{alice}, \text{bob}), sm(\text{alice})\}, \{\text{alice}, \text{bob}, \text{eve}\}) \\ \kappa(\Upsilon) &= fr(\text{alice} \wedge \text{bob}) \wedge sm(\text{alice}) \\ \Phi &= \{\forall X, Y : fr(X, Y) \wedge sm(X) \Rightarrow sm(Y)\}. \end{aligned}$$

*Then  $\varphi = sm(\text{bob})$  is 2-entailed from  $\kappa(\Upsilon)$  and  $\Phi$  but not 1-entailed.*

Note that, in the setting of Example 4, *k*-entailment would make at most  $k - 1$  mistakes. However, 2-entailment would already produce many mistakes in the case of Example 3. So there are cases where *k*-entailment produces fewer errors than classical logic entailment but, quite naturally, also cases where both produce the same number of errors. Importantly, however, for *k*-entailment, we can obtain non-trivial bounds on the number of errors.

Next we state two lemmas that follow immediately from the respective definitions.

**Lemma 1.** *Let  $\Upsilon = (\mathcal{A}, \mathcal{C})$  be an example,  $\Phi$  be a set of constant-free formulas and  $\kappa$  be a masking function. Let  $\mathcal{C}_k$  be the set of all size- $k$  subsets of  $\mathcal{C}$ . Let  $\mathcal{H}_{\mathcal{X}}$  denote the set of all ground literals which can be derived using *k*-entailment from  $\{\kappa(\Upsilon)\} \cup \Phi$  and only contain constants from  $\mathcal{X}$ . Then  $\mathcal{H}_{\mathcal{C}} = \bigcup_{S \in \mathcal{C}_k} \mathcal{H}_S$ .*

**Lemma 2.** *When  $\Gamma \langle S \rangle \models \Phi$  then all ground literals that only contain constants from  $S$  and that are entailed by  $\{\kappa(\Gamma \langle S \rangle)\} \cup \Phi$  must be true in  $\Gamma \langle S \rangle$ .*

We now provide a bound on the number of ground literals wrongly *k*-entailed by a given  $\Phi$ , assuming that we know its accuracy  $Q_{\Gamma, k}(\Phi)$  on the example  $\Gamma$ .

**Proposition 6.** *Let  $\Gamma = (\mathcal{A}, \mathcal{C})$  be an example,  $\Phi$  be a set of constant-free formulas and  $\kappa$  be a masking process. Next let  $\mathcal{F}(\Gamma)$  be the set of all ground literals of a predicate  $p/a$ ,  $a \leq k$ , which are *k*-entailed by  $\{\kappa(\Gamma)\} \cup \Phi$  but are false in  $\Gamma$ . Then*

$$|\mathcal{F}(\Gamma)| \leq (1 - Q_{\Gamma, k}(\Phi)) |\mathcal{C}|^k k^a.$$

*Proof.* First, we note that the number of size-*k* subsets is bounded by  $|\mathcal{C}|^k$  and the number of different ground

*p/a* atoms in each of these subsets is  $k^a$ . It follows from Lemma 2 and Lemma 1 that for any literal  $\delta \in \mathcal{F}$  there must be a size-*k* set  $S \subseteq \mathcal{C}$  such that  $\Gamma \langle S \rangle \not\models \delta$ . The number of all such  $S$ 's that satisfy  $\Gamma \langle S \rangle \not\models \delta$  is bounded by  $(1 - Q_{\Gamma, k}(\Phi)) |\mathcal{C}|^k$ . Hence, we have  $|\mathcal{F}(\Gamma)| \leq (1 - Q_{\Gamma, k}(\Phi)) |\mathcal{C}|^k k^a$ .  $\square$

We can notice that when we increase the domain size  $|\mathcal{C}|$ , keeping  $Q_{\Gamma, k}(\Phi)$  fixed and non-zero, the bound eventually becomes vacuous for predicates whose arity  $a$  is strictly smaller than  $k$ . This is because the number of all ground literals grows only as  $|\mathcal{C}|^a$  whereas the bound grows as  $|\mathcal{C}|^k$ . However, if  $a = k$ , the bound stays fixed when we increase the domain size. We will come back to consequences of this fact in Section 6.

### 4.3 BOUNDED REASONING USING VOTING

To further restrict the set of entailed ground literals, we next introduce *voting entailment*.

**Definition 5** (Voting Entailment). *Let  $k$  be an integer and  $\gamma \in [0; 1]$ . Let  $\Upsilon = (\mathcal{A}, \mathcal{C})$  be an example,  $\Phi$  be a set of constant-free formulas, and  $\kappa$  be a masking process. A ground literal  $l$  of arity  $a$ ,  $a \leq k$ , is said to be entailed from  $\Phi$  and  $\kappa(\Upsilon)$  by voting with parameters  $k$  and  $\gamma$  if there are at least  $\max\{1, \gamma \cdot |\mathcal{C}|^{k-a}\}$  size- $k$  sets  $S \subseteq \mathcal{C}$  such that  $l$  is *k*-entailed by  $\kappa(\Upsilon)\langle S \rangle$ .*

The next example illustrates the use of voting entailment.

**Example 7.** *Let  $\Upsilon = (\mathcal{A}, \mathcal{C})$ , where  $\mathcal{C} = \{\text{alice}, \text{bob}, \text{eve}\}$ , and let  $\kappa(\Upsilon) = fr(\text{alice}, \text{bob}) \wedge fr(\text{eve}, \text{bob}) \wedge sm(\text{eve})$ . Next, let  $\Phi = \{\forall X, Y : fr(X, Y) \wedge sm(X) \Rightarrow sm(Y)\}$ . Then  $sm(\text{bob})$  is entailed from  $\Phi$  and  $\kappa(\Upsilon)$  by voting with the parameters  $k = 2$  and  $\gamma = 2/3$ , as  $\gamma \cdot |\mathcal{C}|^{k-a} = 2/3 \cdot 3^{2-1} = 2$  and there are two size-2 subsets of  $\mathcal{C}$  that 2-entail  $sm(\text{bob})$ .*

We now show how the bound from Proposition 6 can be strengthened in the case of voting entailment.

**Proposition 8.** *Let  $k$  be an integer and  $\gamma \in [0; 1]$ . Let  $\Gamma = (\mathcal{A}, \mathcal{C})$  be an example,  $\Phi$  be a set of constant-free formulas, and  $\kappa$  be a masking process. Let  $\mathcal{F}(\Gamma)$  be the set of all ground literals of a predicate  $p/a$ ,  $a \leq k$ , that are entailed by voting from  $\{\kappa(\Gamma)\} \cup \Phi$  with parameters  $k$  and  $\gamma$  but are false in  $\Gamma$ . If  $\gamma \cdot |\mathcal{C}|^{k-a} \geq 1$  then*

$$|\mathcal{F}(\Gamma)| \leq (1 - Q_{\Gamma, k}(\Phi)) \frac{|\mathcal{C}|^a k^a}{\gamma}$$

*and otherwise*

$$|\mathcal{F}(\Gamma)| \leq (1 - Q_{\Gamma, k}(\Phi)) |\mathcal{C}|^k k^a.$$

*Proof.* First we define the number of ‘‘votes’’ for a ground literal  $l$  as

$$\#\_{\kappa(\Gamma), \Phi}(l) = |\{S \subseteq \mathcal{C} \mid |S| = k, \{\kappa(\Gamma)\langle S \rangle\} \cup \Phi \models_k l\}|.$$

Let  $L$  be the set of all ground  $p/a$  literals  $l$  such that  $\Gamma \models \neg l$ . Then, since any size- $k$  subset of  $\mathcal{C}$  can only contribute  $k^a$  votes to literals based on the predicate  $p/a$ , we have

$$\sum_{l \in L} \#\kappa(\Gamma, \Phi)(l) \leq (1 - Q_{\Gamma, k}(\Phi)) |\mathcal{C}|^k k^a.$$

Hence  $|\mathcal{F}(\Gamma)| \leq \frac{(1 - Q_{\Gamma, k}(\Phi)) |\mathcal{C}|^k k^a}{\max\{1, \gamma \cdot |\mathcal{C}|^{k-a}\}}$ . If  $\gamma \cdot |\mathcal{C}|^{k-a} \geq 1$  then  $|\mathcal{F}(\Gamma)| \leq (1 - Q_{\Gamma, k}(\Phi)) \frac{|\mathcal{C}|^a k^a}{\gamma}$ . The case when  $\gamma \cdot |\mathcal{C}|^{k-a} < 1$  follows from Theorem 6.  $\square$

Unlike for  $k$ -entailment, the fraction of “wrong” ground  $p/a$  literals entailed by voting entailment does not grow with an increasing domain size as long as  $\gamma \cdot |\mathcal{C}|^{k-a} \geq 1$ .

## 5 PROBABILISTIC BOUNDS

We now turn to the setting where the accuracy of the formulas needs to be estimated from a training example  $\Upsilon$ . More generally, we also cover the case where the formulas themselves are learned from the training example. In such cases, to account for over-fitting, we need to consider the (size of the) hypothesis class that was used for learning these formulas. Specifically, we prove probabilistic bounds for variants of the following learning problem. We are given a hypothesis set  $\mathcal{H}$  of constant-free theories, and we want to compute bounds on the number of incorrectly predicted literals which simultaneously hold for all  $\Phi \in \mathcal{H}$  (as a function of  $Q_{\Upsilon, k}(\Phi)$ ) with probability at least  $1 - \delta$ , where  $\delta$  is a confidence parameter. Note that the case where the theory  $\Phi$  is given, rather than learned, corresponds to  $\mathcal{H} = \{\Phi\}$ .

We start by proving general concentration inequalities in Section 5.1 which we then use to prove bounds for  $k$ -entailment. These bounds are studied for the realizable case in Section 5.2 and for the general case in Section 5.3. Bounds for voting entailment are studied in Section 5.4

### 5.1 CONCENTRATION INEQUALITIES

We will need to bound the difference between the “accuracy” of given sets of logic formulas  $\Phi$  on the training sample  $\Upsilon$  and their accuracy on a test sample  $\Gamma$  (i.e. the difference between  $Q_{\Upsilon, k}(\Phi)$  and  $Q_{\Gamma, k}(\Phi)$ ). To prove the concentration inequalities in this section, we will utilize the following lemma.

**Lemma 3** (Kuzelka et al. [12]). *Let  $\aleph = (\mathcal{A}_{\aleph}, \mathcal{C}_{\aleph})$  be an example. Let  $0 \leq n \leq |\mathcal{C}_{\aleph}|$  and  $0 \leq k \leq n$  be integers. Let  $\mathbf{X} = (\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_{\lfloor \frac{n}{k} \rfloor})$  be a vector of subsets of  $\mathcal{C}_{\aleph}$ , each sampled uniformly and independently of the others from all size- $k$  subsets of  $\mathcal{C}_{\aleph}$ . Next let  $\mathcal{C}_{\Upsilon}$  be sampled uniformly from all size- $n$  subsets of  $\mathcal{C}_{\aleph}$ . Finally,*

*let  $\mathcal{I}' = \{1, 2, \dots, |\mathcal{C}_{\aleph}|\}$  and let  $\mathbf{Y} = (\mathcal{S}'_1, \mathcal{S}'_2, \dots, \mathcal{S}'_{\lfloor \frac{n}{k} \rfloor})$  be a vector sampled by the following process:*

1. *Sample subsets  $\mathcal{I}'_1, \dots, \mathcal{I}'_{\lfloor \frac{n}{k} \rfloor}$  of size  $k$  from  $\mathcal{I}'$ .*
2. *Sample an injective function  $g : \bigcup_{i=1}^{\lfloor \frac{n}{k} \rfloor} \mathcal{I}'_i \rightarrow \mathcal{C}_{\Upsilon}$  uniformly from all such functions.*
3. *Define  $\mathcal{S}'_i = g(\mathcal{I}'_i)$  for all  $0 \leq i \leq \lfloor \frac{n}{k} \rfloor$ .*

*Then  $\mathbf{X}$  and  $\mathbf{Y}$  have the same distribution.*

The next example illustrates the intuition behind the proof of this lemma, which can be found in [12].

**Example 9.** *Let  $\mathcal{C}_{\aleph} = \{1, 2, \dots, 10^6\}$ . Let us sample  $\lfloor m/k \rfloor$  size- $k$  subsets of  $\mathcal{C}_{\aleph}$  uniformly. If this was the process that generates the data from which we estimate parameters, we could readily apply Hoeffding’s inequality to get the confidence bounds. However, in typical SRL settings (e.g. with MLNs), we are given a complete example on some set of constants (objects), rather than a set of small sampled fragments. So we instead need to assume that the whole training example is sampled at once, uniformly from all size- $m$  subsets of  $\mathcal{C}_{\aleph}$ . However, when we then estimate the probabilities of formulas from this example, we cannot use Hoeffding’s bound or any other bound expecting independent samples. What we can do<sup>2</sup> is to mimic sampling from  $\mathcal{C}_{\aleph}$  by sampling from an auxiliary set of constants of the same size as  $\mathcal{C}_{\aleph}$  and then specialising these constants to constants from a sampled size- $m$  subset. Hence the first  $\lfloor m/k \rfloor$  sampled sets will be distributed exactly as the first  $\lfloor m/k \rfloor$  subsets sampled i.i.d. directly from  $\mathcal{C}_{\aleph}$ .*

Lemma 3 was used in [12] to prove a bound on expected error. Here we extend that result and use Lemma 3 to prove the concentration inequalities stated in the next two theorems.

**Theorem 10.** *Let  $\aleph = (\mathcal{A}_{\aleph}, \mathcal{C}_{\aleph})$  be an example and let  $0 \leq n \leq |\mathcal{C}_{\aleph}|$  and  $0 \leq k \leq n$  be integers. Let  $\mathcal{C}_{\Upsilon}$  be sampled uniformly from all size- $n$  subsets of  $\mathcal{C}_{\aleph}$  and let  $\Upsilon = \aleph(\mathcal{C}_{\Upsilon})$ . Let  $\alpha$  be a closed and constant-free formula and let  $\mathcal{C}_k$  denote all size- $k$  subsets of  $\mathcal{C}_{\Upsilon}$ . Let  $\hat{A}_{\Upsilon} = Q_{\Upsilon, k}(\alpha)$  and let  $A_{\aleph} = Q_{\aleph, k}(\alpha)$ . Then we have  $P[\hat{A}_{\Upsilon} - A_{\aleph} \geq \varepsilon] \leq \exp(-2 \lfloor \frac{n}{k} \rfloor \varepsilon^2)$ ,  $P[A_{\aleph} - \hat{A}_{\Upsilon} \geq \varepsilon] \leq \exp(-2 \lfloor \frac{n}{k} \rfloor \varepsilon^2)$ , and  $P\left[|\hat{A}_{\Upsilon} - A_{\aleph}| \geq \varepsilon\right] \leq 2 \exp(-2 \lfloor \frac{n}{k} \rfloor \varepsilon^2)$ .*

*Proof.* First we define an auxiliary estimator  $\tilde{A}_{\Upsilon}^{(q)}$ . Let  $\mathbf{Y}^{(q)}$  be a vector of  $\lfloor n/k \rfloor \cdot q$  size- $k$  subsets of  $\mathcal{C}_{\Upsilon}$  where

<sup>2</sup>Note that we do not need to do this in practice which will follow from Theorem 10; we only need this mimicking process to prove that theorem.

the subsets of  $\mathcal{C}_\Upsilon$  in each of the  $q$  non-overlapping size- $\lfloor n/k \rfloor$  segments  $\mathbf{Y}_1^{(q)}, \mathbf{Y}_2^{(q)}, \dots, \mathbf{Y}_q^{(q)}$  of  $\mathbf{Y}^{(q)}$  are sampled in the same way as the elements of the vector  $\mathbf{Y}$  in Lemma 3, all with the same  $\mathcal{C}_\Upsilon$  (i.e.  $\mathbf{Y}^{(q)}$  is the concatenation of the vectors  $\mathbf{Y}_1^{(q)}, \mathbf{Y}_2^{(q)}, \dots, \mathbf{Y}_q^{(q)}$ ). Let us define  $\tilde{A}_\Upsilon^{(q)} = \frac{1}{q \cdot \lfloor n/k \rfloor} \sum_{\mathcal{S} \in \mathbf{Y}^{(q)}} \mathbb{1}(\Upsilon(\mathcal{S}) \models \alpha)$ . We can rewrite  $\tilde{A}_\Upsilon^{(q)}$  as  $\tilde{A}_\Upsilon^{(q)} = \frac{1}{q} \sum_{i=1}^q \frac{1}{\lfloor n/k \rfloor} \sum_{\mathcal{S} \in \mathbf{Y}_i^{(q)}} \mathbb{1}(\Upsilon(\mathcal{S}) \models \alpha)$ .

Then we can use the following trick (Hoeffding [9], Section 5) based on application of Jensen's inequality and Markov's inequality: If  $T = a_1 \cdot T_1 + a_2 \cdot T_2 + \dots + a_q \cdot T_n$ , where  $a_i \geq 0$  and  $\sum_{i=1}^q a_i = 1$ , then, for any  $h > 0$ ,  $P[T \geq \varepsilon] \leq \sum_{i=1}^q a_i \cdot \mathbb{E}[\exp(h(T_i - \varepsilon))]$ . Note that the  $T_i$ 's do not have to be independent. Next, using Hoeffding's lemma (Lemma 1 in [9]), if  $a_i = 1/q$  and each of the terms  $T_i$  is a sum of independent random zero-mean variables  $X_j^{(i)}$  such that  $P[a \leq X_j^{(i)} \leq b] = 1$  and  $b - a \leq 1$ , then we get:

$$\begin{aligned} P[T \geq \varepsilon] &\leq \sum_{i=1}^q \frac{1}{q} \cdot \mathbb{E}[\exp(h(T_i - \varepsilon))] \\ &\leq e^{-h\varepsilon} \exp\left(\frac{m \cdot h^2}{8}\right) = \exp\left(-h\varepsilon + \frac{m \cdot h^2}{8}\right) \end{aligned}$$

where  $m$  denotes the number of summands of  $T_i$  (which, in our case, is the same for all  $T_i$ 's). Note that this function achieves its minimum at  $h = \frac{4\varepsilon}{m}$ . We set  $T_i := \sum_{\mathcal{S} \in \mathbf{Y}_i^{(q)}} (\mathbb{1}(\Upsilon(\mathcal{S}) \models \alpha) - A_\aleph)$  (note that  $\mathbb{E}[T_i] = 0$  and  $m = \lfloor n/k \rfloor$ ). Thus, we get  $P[\lfloor \frac{n}{k} \rfloor \cdot (\tilde{A}_\Upsilon^{(q)} - A_\aleph) \geq \varepsilon] \leq \exp(-2\varepsilon^2 / \lfloor \frac{n}{k} \rfloor)$ , and finally

$$P[\tilde{A}_\Upsilon^{(q)} - A_\aleph \geq \varepsilon] \leq \exp\left(-2 \left\lfloor \frac{n}{k} \right\rfloor \varepsilon^2\right),$$

symmetrically also  $P[A_\aleph - \tilde{A}_\Upsilon^{(q)} \geq \varepsilon] \leq \exp(-2 \lfloor \frac{n}{k} \rfloor \varepsilon^2)$ , and, using union bound, we get

$$P[|\tilde{A}_\Upsilon^{(q)} - A_\aleph| \geq \varepsilon] \leq 2 \exp\left(-2 \left\lfloor \frac{n}{k} \right\rfloor \varepsilon^2\right).$$

It follows from the strong law of large numbers (which holds for any  $\Upsilon$ ) that  $P[\lim_{q \rightarrow \infty} \tilde{A}_\Upsilon^{(q)} = \hat{A}_\Upsilon] = 1$ . Since  $q$  was arbitrary, the statement of the proposition follows.  $\square$

As the next theorem shows, the above result can be generalized to the case where we need to bound the difference between the estimations obtained from two samples.

**Theorem 11.** *Let  $\aleph = (\mathcal{A}_\aleph, \mathcal{C}_\aleph)$  be an example and let  $0 \leq n, u \leq |\mathcal{C}_\aleph|$  and  $0 \leq k \leq n$  be integers. Let  $\mathcal{C}_\Upsilon$  and  $\mathcal{C}_\Gamma$  be sampled uniformly from all size- $n$  and size- $u$  subsets of  $\mathcal{C}_\aleph$  and let  $\Upsilon = \aleph(\mathcal{C}_\Upsilon), \Gamma = \aleph(\mathcal{C}_\Gamma)$ . Let  $\alpha$  be a closed and constant-free formula. Let  $\hat{A}_\Upsilon =$*

$Q_{\Upsilon, k}(\alpha), \hat{A}_\Gamma = Q_{\Gamma, k}(\alpha)$ , and let  $A_\aleph = Q_{\aleph, k}(\alpha)$ . Then we have  $P[|\hat{A}_\Upsilon - \hat{A}_\Gamma| \geq \varepsilon] \leq \exp\left(\frac{-2\varepsilon^2}{1/\lfloor n/k \rfloor + 1/\lfloor u/k \rfloor}\right)$ , and  $P[|\hat{A}_\Upsilon - \hat{A}_\Gamma| \geq \varepsilon] \leq 2 \exp\left(\frac{-2\varepsilon^2}{1/\lfloor n/k \rfloor + 1/\lfloor u/k \rfloor}\right)$ .

*Proof.* See the appendix.  $\square$

We note that the concentration inequality derived in Theorem 10 improves upon a concentration inequality derived in [17] (Chapter 10) that contains  $n/k^2$  (in our notation) instead of  $\lfloor n/k \rfloor$  in the exponential.<sup>3</sup>

Next we prove an inequality for the special case where the probability of a formula  $\alpha$  on  $\Upsilon$  is 0. Since we can also take negations of formulas, this theorem will be useful to prove bounds for formulas that are perfectly accurate on training data. As the following theorem shows, in this case we obtain stronger guarantees, where we have  $\varepsilon$  instead of  $\varepsilon^2$  in the exponential.

**Theorem 12.** *Let  $\aleph = (\mathcal{A}_\aleph, \mathcal{C}_\aleph)$  be an example and let  $0 \leq n \leq |\mathcal{C}_\aleph|$  and  $0 \leq k \leq n$  be integers. Let  $\mathcal{C}_\Upsilon$  be sampled uniformly from all size- $n$  subsets of  $\mathcal{C}_\aleph$  and let  $\Upsilon = \aleph(\mathcal{C}_\Upsilon)$ . Let  $\alpha$  be a closed and constant-free formula and let  $\mathcal{C}_k$  denote all size- $k$  subsets of  $\mathcal{C}_\Upsilon$ . Let  $\hat{A}_\Upsilon = Q_{\Upsilon, k}(\alpha)$  and let  $A_\aleph = Q_{\aleph, k}(\alpha) \geq \varepsilon$ . Then we have*

$$P[\hat{A}_\Upsilon = 0] \leq \exp(-\lfloor n/k \rfloor \varepsilon).$$

*Proof.* Let  $\mathbf{Y}$  be sampled as in Lemma 3 (i.e.  $\mathbf{Y}$  is sampled only using  $\Upsilon$  and not directly  $\aleph$ ). Then using Lemma 3 we know that the elements of  $\mathbf{Y}$  are distributed like  $\lfloor n/k \rfloor$  independent samples (size- $k$  subsets) from  $\mathcal{C}_\aleph$ . Hence we can bound the probability  $P[A_\Upsilon = 0] \leq (1 - \varepsilon)^{\lfloor n/k \rfloor} \leq \exp(-\lfloor n/k \rfloor \varepsilon)$ . Obviously, adding the rest of the information from size- $k$  subsets of  $\mathcal{C}_\Upsilon$  that are not contained in  $\mathbf{Y}$  cannot increase the bound.  $\square$

## 5.2 ZERO TRAINING ERROR CASE

We start by proving a bound for the realizable (i.e. zero training error) case.

**Theorem 13.** *Let  $\aleph, \Upsilon, \Gamma, n, u$  and  $\kappa$  be as in Definition 3 (i.e.  $\Upsilon$  and  $\Gamma$  are sampled from  $\aleph$  and  $n, u$  are sizes of  $\Upsilon$ 's and  $\Gamma$ 's domains). Let  $\mathcal{H}$  be a finite hypothesis class of constant-free formulas. Let  $\mathcal{F}(\Gamma, \Phi)$  denote the set of all ground literals of a predicate  $p/a$  that are  $k$ -entailed by  $\{\kappa(\Gamma)\} \cup \Phi$  but are false in  $\Gamma$ .<sup>4</sup> With probability at*

<sup>3</sup>This is essentially due to the fact that we use Hoeffding's decomposition whereas Lovasz relies on Azuma's inequality, leading to a looser bound compared to our bound.

<sup>4</sup>Note that here, as well as in the rest of the theorems in the paper,  $\mathcal{F}(\Gamma, \Phi)$  is a set-valued random variable.



least  $1 - \delta$ , the following holds for all  $\Phi \in \mathcal{H}$  that satisfy  $Q_{\Upsilon,k}(\Phi) = 1$ :

$$\mathbb{E}[|\mathcal{F}(\Gamma, \Phi)|] \leq \frac{\ln |\mathcal{H}| + \ln 1/\delta}{\lfloor n/k \rfloor} u^k k^a.$$

*Proof.* It follows from the linearity of expectation and from Proposition 6 that, for any  $\Phi$ ,  $\mathbb{E}[|\mathcal{F}(\Gamma, \Phi)|] \leq (1 - Q_{\aleph,k}(\Phi))u^k k^a$ . Next, it follows from Theorem 12 and from the union bound taken over all  $\Phi \in \mathcal{H}$  that the probability that there exists  $\Phi \in \mathcal{H}$  such that  $Q_{\Upsilon,k}(\Phi) = 1$  and  $\varepsilon \leq 1 - Q_{\aleph,k}(\Phi)$  is at most  $|\mathcal{H}| \cdot \exp(-\lfloor n/k \rfloor \varepsilon)$ . If  $\varepsilon \geq \frac{\ln |\mathcal{H}| + \ln 1/\delta}{\lfloor n/k \rfloor}$  then  $|\mathcal{H}| \cdot \exp(-\lfloor n/k \rfloor \varepsilon) \leq \delta$ . Hence, with probability at least  $1 - \delta$ , the following holds for all  $\Phi \in \mathcal{H}$  such that  $Q_{\Upsilon,k}(\Phi) = 1$ :  $\mathbb{E}[|\mathcal{F}(\Gamma, \Phi)|] \leq \frac{\ln |\mathcal{H}| + \ln 1/\delta}{\lfloor n/k \rfloor} u^k k^a$ .  $\square$

### 5.3 GENERAL CASE

Next we prove a bound for the general case when the training error is non-zero.

**Theorem 14.** *Let  $\aleph, \Upsilon, \Gamma, n, u$  and  $\kappa$  be as in Definition 3 (i.e.  $\Upsilon$  and  $\Gamma$  are sampled from  $\aleph$  and  $n, u$  are sizes of  $\Upsilon$ 's and  $\Gamma$ 's domains). Let  $\mathcal{H}$  be a finite hypothesis class of constant-free formulas. Let  $\mathcal{F}(\Gamma, \Phi)$  denote the set of all ground literals of a predicate  $p/a$  that are  $k$ -entailed by  $\{\kappa(\Gamma)\} \cup \Phi$  but are false in  $\Gamma$ . With probability at least  $1 - \delta$ , for all  $\Phi \in \mathcal{H}$ :*

$$\mathbb{E}[|\mathcal{F}(\Gamma, \Phi)|] \leq \left( 1 - Q_{\Upsilon,k}(\Phi) + \sqrt{\frac{\ln \left( \frac{|\mathcal{H}|}{\delta} \right)}{2\lfloor n/k \rfloor}} \right) u^k k^a.$$

*Proof.* First, as in the proof of Theorem 13, we find that, for any  $\Phi \in \mathcal{H}$ ,  $\mathbb{E}[|\mathcal{F}(\Gamma)|] \leq (1 - Q_{\aleph,k}(\Phi))u^k k^a$ . Next, it follows from Theorem 10 and from union bound that  $P[\exists \Phi \in \mathcal{H} : Q_{\Upsilon,k}(\Phi) - Q_{\aleph,k}(\Phi) \geq \varepsilon] \leq |\mathcal{H}| \exp(-2\lfloor n/k \rfloor \varepsilon^2)$ . It follows that

$$P\left[\exists \Phi \in \mathcal{H} : Q_{\Upsilon,k}(\Phi) \geq Q_{\aleph,k}(\alpha) + \sqrt{\frac{\ln (|\mathcal{H}|/\delta)}{2\lfloor n/k \rfloor}}\right] \leq \delta.$$

The theorem then follows straightforwardly from the above and from Proposition 6.  $\square$

The previous two theorems provided bounds on the expected number of errors on the sampled test examples. The next theorem is different in that it provides a bound on the actual number of errors.

**Theorem 15.** *Let  $\aleph, \Upsilon, \Gamma$ , and  $\kappa$  be as in Definition 3 (i.e.  $\Upsilon$  and  $\Gamma$  are sampled from  $\aleph$  and  $n, u$  are sizes of  $\Upsilon$ 's and  $\Gamma$ 's domains). Let  $\mathcal{H}$  be a finite hypothesis class of constant-free formulas. Let  $\mathcal{F}(\Gamma, \Phi)$  denote the set of*

*all ground literals of a predicate  $p/a$  that are  $k$ -entailed by  $\{\kappa(\Gamma)\} \cup \Phi$  but are false in  $\Gamma$ . With probability at least  $1 - \delta$ , for all  $\Phi \in \mathcal{H}$ :*

$$|\mathcal{F}(\Gamma, \Phi)| \leq \left( 1 - Q_{\Upsilon,k}(\Phi) + \sqrt{\frac{(\lfloor n/k \rfloor + \lfloor u/k \rfloor) \ln (2|\mathcal{H}|/\delta)}{2\lfloor n/k \rfloor \lfloor u/k \rfloor}} \right) u^k k^a \leq \left( 1 - Q_{\Upsilon,k}(\Phi) + \sqrt{\frac{\ln (2|\mathcal{H}|/\delta)}{\min\{\lfloor n/k \rfloor, \lfloor u/k \rfloor\}}} \right) u^k k^a.$$

*Proof.* Let us denote  $\hat{A} = Q_{\Upsilon,k}(\Phi)$ ,  $\hat{B} = Q_{\Gamma,k}(\Phi)$ . Using Theorem 11 and the union bound over  $\Phi \in \mathcal{H}$ , we get

$$P[\exists \Phi \in \mathcal{H} : |\hat{A} - \hat{B}| \geq \varepsilon] \leq 2|\mathcal{H}| \exp\left(\frac{-2\varepsilon^2 \lfloor n/k \rfloor \lfloor u/k \rfloor}{\lfloor n/k \rfloor + \lfloor u/k \rfloor}\right).$$

Solving the above for  $\varepsilon$  that achieves the  $1 - \delta$  bound, we obtain that, with probability at least  $1 - \delta$ , we have for all  $\Phi \in \mathcal{H}$ :  $|\hat{A} - \hat{B}| \leq \sqrt{\frac{(\lfloor n/k \rfloor + \lfloor u/k \rfloor) \ln (2|\mathcal{H}|/\delta)}{2\lfloor n/k \rfloor \lfloor u/k \rfloor}}$ . Hence, with probability at least  $1 - \delta$ , for all  $\Phi \in \mathcal{H}$  it holds  $1 - Q_{\Gamma,k}(\Phi) \leq 1 - Q_{\Upsilon,k}(\Phi) + \sqrt{\frac{(\lfloor n/k \rfloor + \lfloor u/k \rfloor) \ln (2|\mathcal{H}|/\delta)}{2\lfloor n/k \rfloor \lfloor u/k \rfloor}}$ . The validity of the theorem then follows from the above and from Proposition 6 and the fact that  $\frac{ab}{a+b} \geq \frac{\min(a,b)}{2}$  for any nonnegative  $a$  and  $b$ .  $\square$

### 5.4 BOUNDS FOR VOTING ENTAILMENT

Next we prove a bound for voting entailment, which, unsurprisingly, is tighter than the respective bound for  $k$ -entailment.

**Theorem 16.** *Let  $k$  be an integer and  $\gamma \in [0; 1]$ . Let further  $\aleph, \Upsilon, \Gamma$  and  $\kappa$  be as in Definition 3 (i.e.  $\Upsilon$  and  $\Gamma$  are sampled from  $\aleph$  and  $n, u$  are sizes of  $\Upsilon$ 's and  $\Gamma$ 's domains). Let  $\mathcal{H}$  be a finite hypothesis class of constant-free formulas. Let  $\mathcal{F}(\Gamma, \Phi)$  denote the set of all ground literals of a predicate  $p/a$  that are entailed by voting from  $\{\kappa(\Gamma)\} \cup \Phi$  with parameters  $k$  and  $\gamma$  but are false in  $\Gamma$ . Then, with probability at least  $1 - \delta$ , for all  $\Phi \in \mathcal{H}$ :*

$$|\mathcal{F}(\Gamma)| \leq \left( 1 - Q_{\Upsilon,k}(\Phi) + \sqrt{\frac{\ln (2|\mathcal{H}|/\delta)}{\min\{\lfloor u/k \rfloor, \lfloor n/k \rfloor\}}} \right) \frac{u^a k^a}{\gamma}.$$

*Proof.* This follows from the same reasoning as in the proof of Theorem 15, which gives us the bound on the difference of  $Q_{\Upsilon,k}(\Phi)$  and  $Q_{\Gamma,k}(\Phi)$ , combined with Theorem 8.  $\square$

**Remark 17.** *The fraction of “wrong” ground  $p/a$  literals does not grow with increasing test-set size ( $u$ ), since, by rewriting the bound from Theorem 16, we get, with probability at least  $1 - \delta$ , for all  $\Phi \in \mathcal{H}$ :*

$$\frac{|\mathcal{F}(\Gamma)|}{u^a} \leq \left( 1 - Q_{\Upsilon, k}(\Phi) + \sqrt{\frac{\ln(2|\mathcal{H}|/\delta)}{\min\{\lfloor u/k \rfloor, \lfloor n/k \rfloor\}}} \right) \frac{k^a}{\gamma}.$$

We note here that one can also easily obtain counterparts of Theorems 13 and 14 for voting entailment.

## 6 SUMMARY OF RESULTS

In this section we discuss positive and negative results that follow from the theorems presented in the preceding sections. Here, bounds are considered vacuous if they are not lower than the total number of ground literals. We first focus on  $k$ -entailment in Sections 6.1–6.3, and then discuss the results for voting entailment in Section 6.4. Finally, we also make a connection to MAP-entailment in Section 6.5.

### 6.1 SMALL TEST EXAMPLES

One case where we have non-vacuous bounds for the *expected* number of incorrectly predicted literals with  $k$ -entailment is when the domain of the test examples  $\Gamma$  is small. Naturally a necessary condition is also that the given (or learned) theory  $\Phi$  is sufficiently accurate. The only way to be confident that  $\Phi$  is indeed sufficiently accurate, given that this accuracy needs to be estimated, is by estimating it on a sufficiently large training example. This is essentially what Theorems 13 and 14 imply.

Interestingly, this finding agrees with some experimental observations in the literature. For instance, it has been observed in [14] that classical reasoning in a relational setting close to ours worked well for small-size test-set evidence but was not competitive with other methods for larger evidence sizes. The analysis in the present paper thus sheds light on experimental observations like these.

Note that the bounds from Theorems 13 and 14 are for the *expected* value of the number of errors. Bounds on the *actual* number of errors are provided in Theorem 15. In this case, to obtain non-vacuous bounds, we also need to require that the domain of the test example  $\Gamma$  be sufficiently large. This is not unexpected, however, as it is a known property of statistical bounds for transductive settings (see e.g., [21]) that the size of the test set affects confidence bounds, similarly to how the size of the  $\Gamma$ 's domain affects the bound in Theorem 15.

### 6.2 PREDICATES OF ARITY $k$

Another case where we have non-vacuous bounds for  $k$ -entailment is when the arity of the predicted literals is equal to the parameter  $k$ . In this case both the bounds for the expected error and for the actual error  $|\mathcal{F}(\Gamma, \Phi)|$  are non-vacuous. This means that our results cover important special cases. One such special case is classical attribute-value learning when  $k = 1$  and we represent attributes by unary predicates. Another case is link prediction when  $k = 2$  and higher-arity versions thereof. In link prediction, we have rules such as, for instance,  $\forall X, Y : \text{CoensFan}(X) \wedge \text{CoensFilm}(Y) \Rightarrow \text{likes}(X, Y)$ .

### 6.3 REALIZABLE SETTING

We can get stronger guarantees when the given (or learned) theory  $\Phi$  has zero training error. Keeping the fraction of the domain-sizes  $|\mathcal{C}_\Gamma|^{k-a}/|\mathcal{C}_\Upsilon|$  small, Theorem 13 implies non-vacuous bounds for predicates of arity  $a$  for any size of the domain of  $\Gamma$ . Intuitively, this means that we can use theories that are completely accurate on training data for inference using  $k$ -entailment. However, the required size of the domain of the training example  $\Upsilon$ , to guarantee that we will not produce too many errors, grows exponentially with  $k$  (for a fixed arity  $a$ ) and polynomially with  $|\mathcal{C}_\Gamma|$ .

### 6.4 VOTING

When using voting entailment, we can always obtain non-trivial bounds by making  $\gamma$  large; obviously this comes at the price of making the inferences more cautious. Voting entailment is a natural inference method in domains where one proof is not enough, i.e. where the support from several proofs is needed before we can be sufficiently confident in the conclusion; an example of such a domain is the well-known *smokers* domain, where knowing that one friend smokes does not provide enough evidence to conclude that somebody smokes; only if we have evidence of several smoker friends is the conclusion warranted that this person smokes.

### 6.5 RELATIONSHIP TO MAP INFERENCE

A popular approach to collective classification in relational domains is MAP-inference in Markov logic networks. Therefore a natural question is how this approach performs in our setting. Perhaps surprisingly, it might produce as many errors as classical logic reasoning in the examples from Section 4.1, if the Markov logic network contains the same rules, all with positive weights, as we had in these examples. This is because MAP-inference will predict the same literals as classical logical inference

when the rules from the Markov logic network are consistent with the given evidence. Thus, we can see that our guarantees for both  $k$ -entailment and voting entailment are better than guarantees one could get for MAP-inference. This is also in agreement with the well-known observations that, for instance, in the smokers domain, MAP inference often predicts everyone to be a smoker or everyone to be a non-smoker if there is only a small amount of evidence.

## 7 RELATED WORK

Our main inspiration comes from the works on PAC-semantics by Valiant [23] and Juba [10]. Our work differs mainly in the fact that we have one large relational structure  $\aleph$ , and a training example  $\Upsilon$  and a test example  $\Gamma$ , both sampled from  $\aleph$ , whereas it is assumed in these existing approaches that learning examples are sampled i.i.d. from some distribution. This has two important consequences. First, they could use statistical techniques developed for i.i.d. data whereas we had to first derive concentration inequalities for sampling without replacement in the relational setting. Second, since they only needed to bound the error on the independently sampled examples, they did not have to consider the number of incorrectly inferred facts. In contrast, in the relational setting that we considered here, the number of errors made on one relational example is the quantity that needs to be bounded. It follows that completely different techniques are needed in our case. Another difference is that, in their case, the training examples are also masked. In principle, we could modify our results to accommodate for masked examples by replacing “accurate” formulas by sufficiently-often “witnessed” formulas (see [10] for a definition).

Dhurandhar and Dobra [5] derived Hoeffding-type inequalities for classifiers trained with relational data, but these inequalities, which are based on the restriction on the independent interactions of data points, cannot be applied to solve the problems considered in the present paper. Certain other statistical properties of learning have also been studied for SRL models. For instance, Xiang and Neville [25] studied consistency of estimation. However, guaranteeing convergence to the correct distribution does not mean that the model would not generate many errors when used, e.g., for MAP-inference. In [26], they further studied errors in label propagation in collective classification. In their setting, however, the relational graph is fixed and one only predicts labels of vertices exploiting the relational structure for making the predictions. Here we also note that it is not always possible or desirable in practice to sample sets of domain elements uniformly as we assumed to be the case in our analysis.

Other sampling designs for relational data were studied, e.g. in [1]. A study of PAC guarantees for such other sampling designs is left as a topic for future work.

There have also been works studying restricted forms of inference in a purely logical context, e.g. [6]. It is an interesting question for future work to find out which existing restricted inference systems would lead to non-vacuous error bounds in the relational setting.

## 8 CONCLUSIONS

We have studied the problem of predicting plausible missing facts in relational data, given a set of imperfect logical rules, in a PAC reasoning setting. As for the considered inference methods, one of our main objectives was for the inference methods to stay close to classical logic. The first inference method,  $k$ -entailment, is a restricted form of classical logic inference and hence satisfies this objective. The second inference method, voting entailment, is based on a form of voting that combines results from inferences made by  $k$ -entailment on subsets of the relational data. Importantly, the voting is not weighted which makes voting entailment easier to understand. We were able to obtain non-trivial bounds for the number of literals incorrectly predicted by a learned (or given) theory for both  $k$ -entailment and voting entailment. Probably the most useful results of our analysis lie in the identification of cases where the bounds for learning and reasoning in relational data are non-vacuous, which we discussed in detail in Section 6.

There are many interesting directions in which one could extend the results presented in this paper. For instance, as practical means to improve the explainability of inferences made by voting entailment, we could first find representatives of isomorphism classes of “proofs” that are aggregated by voting entailment, and only show these to the user. Another direction is to extend the notion of implicit learning from [10] into the relational setting. It would also be interesting to exploit explicit sparsity constraints and to study other sampling designs, although that might also turn out to be analytically less tractable than the setting considered in the present paper. Finally, although all bounds presented in this paper assume finite hypothesis classes, we note that it is also possible to extend our results to infinite hypothesis classes [13].

### Acknowledgments

OK’s work was partially supported by the Research Foundation - Flanders (project G.0428.15). SS is supported by ERC Starting Grant 637277. JD is partially supported by the KU Leuven Research Fund (C14/17/070,C22/15/015,C32/17/036), and FWO-Vlaanderen (SBO-150033).

## References

- [1] Nesreen K Ahmed, Jennifer Neville, and Ramana Rao Kompella. Network sampling designs for relational classification. In *ICWSM*, 2012.
- [2] Leila Amgoud. Postulates for logic-based argumentation systems. *International Journal of Approximate Reasoning*, 55(9):2028–2048, 2014.
- [3] Newton CA Da Costa et al. On the theory of inconsistent formal systems. *Notre dame journal of formal logic*, 15(4):497–510, 1974.
- [4] Luc De Raedt, Angelika Kimmig, and Hannu Toivonen. ProbLog: a probabilistic prolog and its application in link discovery. In *Proceedings of the 20th international joint conference on Artificial intelligence*, pages 2468–2473. Morgan Kaufmann Publishers Inc., 2007.
- [5] Amit Dhurandhar and Alin Dobra. Distribution-free bounds for relational classification. *Knowledge and information systems*, 31(1):55–78, 2012.
- [6] Marcello DAgostino, Marcelo Finger, and Dov Gabbay. Semantics and proof-theory of depth bounded boolean logics. *Theoretical Computer Science*, 480:43–68, 2013.
- [7] Morten Elvang-Gøransson and Anthony Hunter. Argumentative logics: Reasoning with classically inconsistent information. *Data & Knowledge Engineering*, 16(2):125–145, 1995.
- [8] P. Gärdenfors. *Knowledge in Flux*. MIT Press, 1988.
- [9] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.
- [10] Brendan Juba. Implicit learning of common sense for reasoning. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages 939–946, 2013.
- [11] Sébastien Konieczny and Ramón Pino Pérez. Merging information under constraints: a logical framework. *Journal of Logic and computation*, 12(5):773–808, 2002.
- [12] Ondřej Kuželka, Yuyi Wang, Jesse Davis, and Steven Schockaert. Relational marginal problems: Theory and estimation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*. AAAI Press, 2018.
- [13] Ondřej Kuželka, Yuyi Wang, and Steven Schockaert. VC-dimension based generalization bounds for relational learning. In *European Conference on Machine Learning and Knowledge Discovery in Databases*, 2018.
- [14] Ondřej Kuželka, Jesse Davis, and Steven Schockaert. Induction of interpretable possibilistic logic theories from relational data. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 1153–1159. AAAI Press, 2017.
- [15] Ni Lao, Tom Mitchell, and William W Cohen. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 529–539, 2011.
- [16] Qiao Liu, Liuyi Jiang, Minghao Han, Yao Liu, and Zhiguang Qin. Hierarchical random walk inference in knowledge graphs. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 445–454, 2016.
- [17] László Lovász. *Large networks and graph limits*, volume 60. American Mathematical Soc., 2012.
- [18] Graham Priest. The logic of paradox. *Journal of Philosophical logic*, 8(1):219–241, 1979.
- [19] Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, 2006.
- [20] Tim Rocktäschel and Sebastian Riedel. End-to-end differentiable proving. In *Proceedings of the Annual Conference on Neural Information Processing Systems*, pages 3791–3803, 2017.
- [21] Ilya Tolstikhin and David Lopez-Paz. Minimax lower bounds for realizable transductive classification. *arXiv preprint arXiv:1602.03027*, 2016.
- [22] Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.
- [23] Leslie G. Valiant. Knowledge infusion. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 1546–1551. AAAI Press, 2006.
- [24] Gustav Šourek, Suresh Manandhar, Filip Železný, Steven Schockaert, and Ondřej Kuželka. Learning predictive categories using lifted relational neural networks. In *Proceedings of the 26th International Conference on Inductive Logic Programming*, pages 108–119, 2016.
- [25] Rongjing Xiang and Jennifer Neville. Relational learning with one network: An asymptotic analysis. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 779–788, 2011.
- [26] Rongjing Xiang and Jennifer Neville. Understanding propagation error and its effect on collective classification. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 834–843. IEEE, 2011.

---

# Pure Exploration of Multi-Armed Bandits with Heavy-Tailed Payoffs

---

Xiaotian Yu, Han Shao, Michael R. Lyu, Irwin King

<sup>1</sup>Department of Computer Science and Engineering

The Chinese University of Hong Kong, Shatin, N.T., Hong Kong

<sup>2</sup>Shenzhen Key Laboratory of Rich Media Big Data Analytics and Applications,  
Shenzhen Research Institute, The Chinese University of Hong Kong, Shenzhen, China  
{xtyu, hshao, lyu, king}@cse.cuhk.edu.hk

## Abstract

Inspired by heavy-tailed distributions in practical scenarios, we investigate the problem on pure exploration of Multi-Armed Bandits (MAB) with heavy-tailed payoffs by breaking the assumption of payoffs with sub-Gaussian noises in MAB, and assuming that stochastic payoffs from bandits are with finite  $p$ -th moments, where  $p \in (1, +\infty)$ . The main contributions in this paper are three-fold. First, we technically analyze tail probabilities of empirical average and truncated empirical average (TEA) for estimating expected payoffs in sequential decisions with heavy-tailed noises via martingales. Second, we propose two effective bandit algorithms based on different prior information (i.e., fixed confidence or fixed budget) for pure exploration of MAB generating payoffs with finite  $p$ -th moments. Third, we derive theoretical guarantees for the proposed two bandit algorithms, and demonstrate the effectiveness of two algorithms in pure exploration of MAB with heavy-tailed payoffs in synthetic data and real-world financial data.

## 1 INTRODUCTION

The prevailing decision-making model named Multi-Armed Bandits (MAB) elegantly characterizes a wide class of practical problems on sequential learning with partial feedbacks, which was first formally proposed and investigated in (Robbins, 1952). In general, a predominant characteristic of MAB is a trade-off between exploration and exploitation for sequential decisions, which has been frequently encountered in scientific research and various industrial applications, e.g., resource allocation, online advertising and personalized recommendations (Auer et al., 2002; Bubeck et al., 2012; Chu et al., 2011; Lattimore et al., 2015; Wu et al., 2016).

Most algorithms in MAB are primarily developed to maximize cumulative payoffs during a number of rounds for sequential decisions. Recently, there have been interesting investigations on various variants of the traditional MAB model, such as linear bandits (Auer, 2002; Yu et al., 2017b; Zhao and King, 2016), pure exploration of MAB (Audibert and Bubeck, 2010), risk-averse MAB (Sani et al., 2012; Yu et al., 2017a), cascading bandits (Kveton et al., 2015) and clustering bandits (Korda et al., 2016; Li et al., 2016).

One non-trivial branch of MAB is pure exploration, where the goal is to find the optimal arm in a given decision-arm set at the end of exploration. In this case, there is no explicit trade-off between exploration and exploitation for sequential decisions, which means that the exploration phase and the exploitation phase are separated. The problem of pure exploration is motivated by real scenarios which prefer to identify an optimal arm instead of maximizing cumulative payoffs. Recent advances in pure exploration of MAB have found potential applications in many practical domains including communication networks and commercialized products (Audibert and Bubeck, 2010; Chen et al., 2014).

In previous studies on pure exploration of MAB, a common assumption is that noises in observed payoffs are sub-Gaussian. The sub-Gaussian assumption encompasses cases of all bounded payoffs and many unbounded payoffs in MAB, e.g., payoffs of an arm following a Gaussian distribution. However, there exist non-sub-Gaussian noises in observed payoffs for bandits, e.g., high-probability extreme payoffs in sequential decisions which are called heavy-tailed payoffs. A practical motivation example for MAB with heavy-tailed payoffs is the distribution of delays in end-to-end network routing (Liebeherr et al., 2012). Pure exploration of MAB with heavy-tailed payoffs is important, especially for identifications of the potential optimal investment target for practical financial applications. It is worth mentioning that the case of maximizing cumulative payoffs

of MAB with heavy tails has been extensively investigated in (Bubeck et al., 2013a; Carpentier and Valko, 2014; Lattimore, 2017; Medina and Yang, 2016; Vakili et al., 2013). In (Bubeck et al., 2013a), the setting of sequential payoffs with bounded  $p$ -th moments was investigated for regret minimization in MAB, where  $p \in (1, 2]$ . Vakili et al. (Vakili et al., 2013) introduced bounded  $p$ -th moments with the support over  $(1, +\infty)$ , and provided a complete regret guarantee in MAB. In (Medina and Yang, 2016), regret guarantee in linear bandits with heavy-tailed payoffs was investigated, which is still scaled by parameters of bounded moments. Recently, payoffs in bandits with bounded kurtosis were discussed in (Lattimore, 2017).

In this paper, we investigate the problem on pure exploration of MAB with heavy-tailed payoffs characterized by the bound of  $p$ -th moments. It is surprising to find that less effort has been devoted to pure exploration of MAB with heavy-tailed payoffs. Compared with previous work on pure exploration of MAB, the problem of best arm identification with heavy-tailed payoffs has three challenges. The first challenge is the estimate of expected payoffs of an arm in MAB. It might not be sufficient to adopt an empirical average (EA) of observed payoffs with heavy-tailed noises for estimating a true mean. The second challenge is the probability of error for the estimate of expected payoffs, which affects performance of bandit algorithms in pure exploration of MAB. The third challenge is to develop effective bandit algorithms with theoretical guarantees for pure exploration of MAB with heavy-tailed stochastic payoffs.

To solve the above three challenges, we need to introduce a general assumption that stochastic payoffs in MAB are with finite  $p$ -th moments, where  $p \in (1, +\infty)$ . Note that the case of  $p \in (1, 2]$  is weaker than the classic assumption of payoffs with sub-Gaussian noises in MAB. Then, under the assumption of finite  $p$ -th moments, we present theoretical behaviours of empirical average, and analyze the estimate of truncated empirical average (TEA). Based on different prior information, i.e., fixed confidence or fixed budget, we propose two bandit algorithms in pure exploration of bandits with heavy-tailed payoffs. Finally, based on synthetic data with noises from standard *Student's t-distribution* and real-world financial data, we demonstrate the effectiveness of the proposed bandit algorithms. To the best of our knowledge, this is the first systematic investigation on pure exploration of MAB with heavy-tailed payoffs. For reading convenience, we list contributions of this paper below.

- We technically analyze tail probabilities of EA and TEA to estimate true mean of arms in MAB with the general assumption of conditionally independent payoffs.

- We propose two bandit algorithms for pure exploration of MAB with heavy-tailed stochastic payoffs characterized by finite  $p$ -th moments, where  $p \in (1, +\infty)$ .
- We derive theoretical results of the proposed bandit algorithms, as well as demonstrating effectiveness of two algorithms via synthetic data and real-world financial data.

## 2 PRELIMINARIES

In this section, we first present related notations and definitions in this paper. Then, we present assumptions and the problem definition for pure exploration of MAB with heavy-tailed payoffs.

### 2.1 NOTATIONS

Let  $\mathcal{A}$  be a bandit algorithm for pure exploration of MAB, which contains  $K$  arms at the beginning of exploration. For pure exploration, let  $\text{Opt}$  be the true optimal arm among  $K$  arms, where  $\text{Opt} \in [K]$  with  $[K] \triangleq \{1, 2, \dots, K\}$ . The total number of sequential rounds for  $\mathcal{A}$  to play bandits is  $T$ , which is also called as sample complexity. The confidence parameter is denoted by  $\delta \in (0, 1)$ , which means that, with probability at least  $1 - \delta$ ,  $\mathcal{A}$  generates an output optimal arm  $\text{Out}$  equivalent to  $\text{Opt}$ , where  $\text{Out} \in [K]$ . In other words, it happens with a small probability  $\delta$  that  $\text{Opt} \neq \text{Out}$ , and  $\delta$  can be also called the probability of error.

There are two settings based on different prior information given at the beginning of exploration, i.e., fixed confidence or fixed budget. For the setting of fixed confidence,  $\mathcal{A}$  receives the information of  $\delta$  at the beginning, and  $\mathcal{A}$  generates  $\text{Out}$  when a certain condition related to  $\delta$  is satisfied. For the setting of fixed budget,  $\mathcal{A}$  receives the information of  $T$  at the beginning, and  $\mathcal{A}$  generates  $\text{Out}$  at the end of  $T$ .

We present the learning process on pure exploration of MAB as follows. For  $t = 1, 2, \dots, T$ ,  $\mathcal{A}$  decides to play an arm  $a_t \in [K]$  with historical information of  $\{a_1, \pi_1(a_1), \dots, a_{t-1}, \pi_{t-1}(a_{t-1})\}$ . Then,  $\mathcal{A}$  observes a stochastic payoff  $\pi_t(a_t) \in \mathbb{R}$  with respect to  $a_t$ , of which the expectation conditional on  $\mathcal{F}_{t-1}$  is  $\mu(a_t)$  with  $\mathcal{F}_{t-1} \triangleq \{a_1, \pi_1(a_1), \dots, a_{t-1}, \pi_{t-1}(a_{t-1}), a_t\}$  and  $\mathcal{F}_0$  being an empty set. Based on  $\pi_t(a_t)$ ,  $\mathcal{A}$  updates parameters to proceed with the exploration at  $t + 1$ . We store time index  $t$  of playing arm  $a_t$  in  $\Phi(a_t)$ , which is a set with increasing integers.

Given an event  $\mathcal{E}$  and a random variable  $\xi$ , let  $\mathbb{P}[\mathcal{E}]$  be the probability of  $\mathcal{E}$  and  $\mathbb{E}[\xi]$  be the expectation of  $\xi$ . For  $x \in \mathbb{R}$ , we denote by  $|x|$  the absolute value of  $x$ , and for a set  $S$ , we denote by  $|S|$  the cardinality of  $S$ . For an event  $\mathcal{E}$ , let  $\mathbf{1}_{[\mathcal{E}]}$  be the indicator function of  $\mathcal{E}$ .

**Definition 1.** (*Heavy-tailed payoffs in MAB*) Given MAB with  $K$  arms, let  $\pi(k)$  be a stochastic payoff drawn from any arm  $k \in [K]$ . For  $t = 1, \dots, T$ , conditional on  $\mathcal{F}_{t-1}$ , MAB has heavy-tailed payoffs with the  $p$ -th raw moment bounded by  $B$ , or the  $p$ -th central moment bounded by  $C$ , where  $p \in (1, +\infty)$ ,  $B, C \in (0, +\infty)$  and  $k \in [K]$ .

## 2.2 PROBLEM DEFINITION

It is general to assume that payoffs during sequential decisions contain noises in many practical scenarios. We list the assumptions in this paper for pure exploration of MAB with heavy-tailed payoffs as follows.

1. Assume that  $\text{Opt} \triangleq \arg \max_{k \in [K]} \mu(k)$  is unique for pure exploration of MAB with  $K$  arms.
2. Assume that MAB has heavy-tailed payoffs with the  $p$ -th raw or central moment conditional on  $\mathcal{F}_{t-1}$  bounded by  $B$  or  $C$ , for  $t = 1, \dots, T$ .
3. Assume that the sequence of stochastic payoffs from arm  $k \in [K]$  has noises with zero mean conditional on  $\mathcal{F}_{t-1}$  in pure exploration of MAB. For any time instant  $t \in [T]$  and the selected arm  $a_t$ , we define random noise of a true payoff as  $\xi_t(a_t) \triangleq \pi_t(a_t) - \mu(a_t)$ , and assume  $\mathbb{E}[\xi_t(a_t) | \mathcal{F}_{t-1}] = 0$ .

Now we present a problem definition for pure exploration of MAB as follows. Given  $K$  arms satisfying Assumptions 1–3, the problem in this paper is to develop a bandit algorithm  $\mathcal{A}$  generating an arm  $\text{Out}_T \in [K]$  after  $T$  pullings of bandits such that  $\mathbb{P}[\text{Out}_T \neq \text{Opt}] \leq \delta$ , where  $\delta \in (0, 1)$ .

We discuss theoretical guarantees in two settings for best arm identification of bandits. One is to derive the theoretical guarantee of  $T$  by fixing the value of  $\delta$ , which is called fixed confidence. The other is to derive the theoretical guarantee of  $\delta$  by fixing the value of  $T$ , which is called fixed budget.

For simplicity of notations, we enumerate the arms according to their expected payoffs as a sequence of  $\mu(1) > \mu(2) \geq \dots \geq \mu(K)$ . In the ranked sequence, we know that  $\text{Opt} = 1$ . Note that the ranking operation does not affect our theoretical guarantees. For any arm  $k \neq \text{Opt}$  and  $k \in [K]$ , we define the sub-optimality as  $\Delta_k \triangleq \mu(\text{Opt}) - \mu(k)$ , which leads to a sequence of sub-optimality as  $\{\Delta_k\}_{k=2}^K$ . To obtain  $K$  terms in sub-optimality, which helps theoretical analyses, we further define  $\Delta_1 \triangleq \Delta_2$ . Inspired by (Audibert and Bubeck, 2010), we define the hardness for pure exploration of MAB with heavy-tailed payoffs by quantities as

$$H_2^p \triangleq \max_{k \in [K]} k^{p-1} \Delta_k^{-p}. \quad (1)$$

## 3 RELATED WORK

Pure exploration in MAB, aiming at finding the optimal arm after exploration among a given decision-arm set, has become an attracting branch in the decision-making domain (Audibert and Bubeck, 2010; Bubeck et al., 2009; Chen et al., 2014; Gabillon et al., 2012, 2016; Jamieson and Nowak, 2014). It has been pointed out that pure exploration in MAB has many applications, such as communication networks and online advertising.

For pure exploration of MAB with payoffs under sub-Gaussian noises, theoretical guarantees have been well studied. Specifically, in the setting of fixed confidence, the first distribution-dependent lower bound of sample complexity was developed in (Mannor and Tsitsiklis, 2004), which is  $\sum_{k \in [K]} \Delta_k^{-2}$ . Even-Dar et al. (2002) originally proposed a bandit algorithm via successive elimination for bounded payoffs with an upper bound of sample complexity matching the lower bound up to a multiplicative logarithmic factor. Karnin et al. (2013) proposed an improved bandit algorithm, which achieves an upper bound of sample complexity matching the lower bound up to a multiplicative doubly-logarithmic factor. Jamieson et al. (2014) proved that it is necessary to have a multiplicative doubly-logarithmic factor in the distribution-dependent lower bound of sample complexity. Jamieson et al. also developed a bandit algorithm via the law of iterated logarithm for pure exploration of MAB, which achieved the optimal sample complexity of the problem.

In the setting of fixed budget with payoffs under sub-Gaussian noises, (Audibert and Bubeck, 2010) developed a distribution-dependent lower bound of probability of error, and provided two algorithms, which achieve optimal probability of error up to logarithmic factors. Gabillon et al. (2012) proposed a unified algorithm for fixed budget and fixed confidence, which discusses  $\epsilon$ -optimal learning in best arm identification of MAB. Karnin et al. (2013) proposed a bandit algorithm via sequential halving to improve probability of error by a multiplicative constant. It is worth mentioning that (Kaufmann et al., 2016) investigated best arm identification of MAB under Gaussian or Bernoulli assumption, and provided lower bounds in terms of Kullback-Leibler divergence. We also notice that there are extensions of best arm identification of MAB, which is multiple-arm identification (Bubeck et al., 2013b; Chen et al., 2014).

To the best of our knowledge, there is no investigation on pure exploration of MAB without the strict assumption of payoffs under sub-Gaussian noises. There are some potential reasons for this fact. One main reason can be that, without sub-Gaussian noises, the tail probabilities of estimates for expected payoffs can be heavy

Table 1: Comparisons on distributional assumptions and theoretical guarantees in pure exploration of MAB. Note we omit constant factors in the following inequalities, and  $H_1$ ,  $H_2$  and  $H_3$  can refer to the corresponding work.

setting	work	assumption on payoffs	algorithm	theoretical guarantee
fixed $\delta$	Even-Dar et al. (2002)	bounded payoffs in $[0, 1]$	SE	$\mathbb{P} \left[ T \leq \sum_{k=1}^K \Delta_k^{-2} \log \left( \frac{K}{\delta \Delta_k} \right) \right] \geq 1 - \delta$
			ME	$\mathbb{P} \left[ T \leq \frac{K}{\epsilon^2} \log \left( \frac{1}{\delta} \right) \right] \geq 1 - \delta$
	Karnin et al. (2013)	bounded payoffs in $[0, 1]$	EGE	$\mathbb{P} \left[ T \leq \sum_{k=1}^K \Delta_k^{-2} \log \left( \frac{1}{\delta} \log \left( \frac{1}{\Delta_k} \right) \right) \right] \geq 1 - \delta$
	Jamieson et al. (2014)	sub-Gaussian noise	LILUCB	$\mathbb{P} \left[ T \leq H_1 \log \left( \frac{1}{\delta} \right) + H_3 \right] \geq 1 - 4\sqrt{c\delta} - 4c\delta$
	Kaufmann et al. (2016)	two-armed Gaussian bandits	$\alpha$ -E	$\mathbb{P} \left[ T \leq \frac{(\sigma_1 + \sigma_2)^2}{(\mu_1 - \mu_2)^2} \log \left( \frac{1}{\delta} \right) \right] \geq 1 - \delta$
	our work	finite $p$ -th moments	SE- $\delta$ (EA)	$\mathbb{P} \left[ T \leq \sum_{k=1}^K \left( \frac{2^{2p+1} K C}{\Delta_k^p \delta} \right)^{\frac{1}{p-1}} \right] \geq 1 - \delta$
	with $p \in (1, 2]$	SE- $\delta$ (TEA)	$\mathbb{P} \left[ T \leq \sum_{k=1}^K \left( \frac{20B^p}{\Delta_k} \right)^{\frac{p}{p-1}} \log \left( \frac{2K}{\delta} \right) \right] \geq 1 - \delta$	
fixed $T$	Audibert and Bubeck (2010)	bounded payoffs in $[0, 1]$	UCB-E	$\mathbb{P}[\text{Out} \neq \text{Opt}] \leq TK \exp \left( -\frac{T-K}{H_1} \right)$
			SR	$\mathbb{P}[\text{Out} \neq \text{Opt}] \leq K(K-1) \exp \left( -\frac{T-K}{\log(K)H_2} \right)$
	Gabillon et al. (2012)	bounded payoffs in $[0, b]$	UGapEb	$\mathbb{P}[\mu_{\text{Out}} - \mu_{\text{Opt}} \geq \epsilon] \leq TK \exp \left( -\frac{T-K}{H_\epsilon} \right)$
	Karnin et al. (2013)	bounded payoffs in $[0, 1]$	SH	$\mathbb{P}[\text{Out} \neq \text{Opt}] \leq \log(K) \exp \left( -\frac{T}{\log(K)H_2} \right)$
	Kaufmann et al. (2016)	two-armed Gaussian bandits	SS	$\mathbb{P}[\text{Out} \neq \text{Opt}] \leq \exp \left( -\frac{(\mu_1 - \mu_2)^2 T}{2(\sigma_1 + \sigma_2)^2} \right)$
	our work	finite $p$ -th moments	SE- $T$ (EA)	$\mathbb{P}[\text{Out} \neq \text{Opt}] \leq 2^{p+1} CK(K-1)H_2^p \left( \frac{K}{T-K} \right)^{p-1}$
	with $p \in (1, 2]$	SE- $T$ (TEA)	$\mathbb{P}[\text{Out} \neq \text{Opt}] \leq 2K(K-1) \exp \left( -\frac{(T-K)\bar{B}_1}{K \underline{\Delta}^{p/(1-p)}} \right)$	

because Chernoff-Hoeffding inequalities of estimates do not hold in general. The failure of Chernoff-Hoeffding inequalities of estimates is a big challenge in pure exploration of MAB. In this paper, we investigate theoretical performance of pure exploration of MAB with heavy-tailed stochastic payoffs characterized by finite  $p$ -th moments, where  $p \in (1, +\infty)$ . We will put more efforts on  $p \in (1, 2]$  because the case of  $p \in (2, +\infty)$  enjoys a similar format of  $p = 2$ . To compare our work with prior studies, we list the distributional assumptions and theoretical guarantees in pure exploration of MAB in Table 1. Finally, it is worth mentioning that the case of maximizing expected cumulative payoffs of MAB with heavy tails has been extensively investigated in (Bubeck et al., 2013a; Carpentier and Valko, 2014; Medina and Yang, 2016; Vakili et al., 2013).

## 4 ALGORITHMS AND ANALYSES

In this section, we first investigate two estimates, i.e., EA and TEA, for expected payoffs of bandits, and derive tail probabilities for EA and TEA under sequential payoffs. Then, we develop two bandit algorithms for best arm identification of MAB in the spirit of successive elimination (SE) and successive rejects (SR). In particular, SE is for the setting of fixed confidence and SR is for the setting of fixed budget. Finally, we derive theoretic

cal guarantees for each bandit algorithm, where we take advantage of EA or TEA.

### 4.1 EMPIRICAL ESTIMATES

In SE and SR, it is common for  $\mathcal{A}$  to maintain a subset of arms  $S_t \subseteq [K]$  at time  $t = 1, 2, \dots$  and  $\mathcal{A}$  will output an arm when a certain condition is satisfied, e.g.,  $|S_t| = 1$  in the setting of fixed confidence. Similar to the most frequently used estimates for expected payoffs in MAB, we consider the following EA to estimate expected payoffs for any arm  $k \in S_t$ :

$$\hat{\mu}_t(k) \triangleq \frac{1}{s_{t,k}} \sum_{i \in \Phi(k)} \pi_i(k), \quad (2)$$

where  $s_{t,k} \triangleq |\Phi(k)|$  at time  $t$ . Note that the number of elements in  $\Phi(k)$  will increase or hold with time evolution, and the elements in  $\Phi(k)$  may not successively increase. We also investigate the following estimator TEA for any arm  $k \in S_t$ :

$$\hat{\mu}_t^\dagger(k) \triangleq \frac{1}{s_{t,k}} \sum_{i \in \Phi(k)} \pi_i(k) \mathbb{1}_{[\pi_i(k) \leq b_i]}, \quad (3)$$

where  $b_i > 0$  is a truncating parameter, and  $b_i$  will be completely discussed in the ensuing theoretical analyses.

We do not discuss the estimator called median of means (MoM) shown in (Bubeck et al., 2013a), because theo-



retical guarantees of MoM enjoy similar formats to those of TEA. Before we prove concentration inequalities for estimates via martingales, we have results as below.

**Proposition 1.** (Dharmadhikari et al., 1968; von Bahr et al., 1965) Let  $\{\nu_i\}_{i=1}^t$  be random variables satisfying  $\mathbb{E}[|\nu_i|^p] \leq C$  and  $\mathbb{E}[\nu_i|\mathcal{F}_{i-1}] = 0$ . If  $p \in (1, 2]$ , then we have  $\mathbb{E}\left[\left|\sum_{i=1}^t \nu_i\right|^p\right] \leq 2tC$ . If  $p \in (2, +\infty)$ , then we have  $\mathbb{E}\left[\left|\sum_{i=1}^t \nu_i\right|^p\right] \leq C_p C t^{p/2}$ , where  $C_p \triangleq (8(p-1)\max(1, 2^{p-3}))^p$ .

**Proposition 2.** (Seldin et al., 2012) Let  $\{\nu_i\}_{i=1}^t$  be random variables satisfying  $|\nu_i| \leq b_i$  with  $\{b_i\}_{i=1}^t$  being a non-decreasing sequence,  $\mathbb{E}[\nu_i|\mathcal{F}_{i-1}] = 0$  and  $\mathbb{E}[\nu_i^2|\mathcal{F}_{i-1}]$  is bounded. Then, with probability  $1 - \delta$ , we have  $\left|\sum_{i=1}^t \nu_i\right| \leq b_t \log(2/\delta) + V_t/b_t$ , and  $V_t = \sum_{i=1}^t \mathbb{E}[\nu_i^2|\mathcal{F}_{i-1}]$ .

**Lemma 1.** In pure exploration of MAB with  $K$  arms, for any  $t \in [T]$  and any arm  $k \in S_t$ , with probability  $1 - \delta$

- for EA, we have

$$\begin{cases} |\hat{\mu}_t(k) - \mu(k)| \leq \left(\frac{2C}{s_{t,k}^{p-1}\delta}\right)^{\frac{1}{p}}, & 1 < p \leq 2, \\ |\hat{\mu}_t(k) - \mu(k)| \leq \left(\frac{C_p C}{s_{t,k}^{p/2}\delta}\right)^{\frac{1}{p}}, & p > 2; \end{cases}$$

- for TEA, we have

$$\begin{cases} |\hat{\mu}_t^\dagger(k) - \mu(k)| \leq 5B^{\frac{1}{p}} \left(\frac{\log(2/\delta)}{s_{t,k}}\right)^{\frac{p-1}{p}}, & 1 < p \leq 2, \\ |\hat{\mu}_t^\dagger(k) - \mu(k)| \leq 5B^{\frac{1}{p}} \left(\frac{\log(2/\delta)}{s_{t,k}}\right)^{\frac{1}{2}}, & p > 2. \end{cases}$$

*Proof.* We first prove the results with the estimator  $\hat{\mu}_t(k)$  with  $k \in S_t$ . By Chebyshev's inequality, we have

$$\begin{aligned} \mathbb{P}[|\hat{\mu}_t(k) - \mu(k)| \geq \delta] &\leq \frac{\mathbb{E}[|\hat{\mu}_t(k) - \mu(k)|^p]}{\delta^p} \\ &= \frac{\mathbb{E}\left[\left|\sum_{i \in \Phi(k)} \pi_i(k) - \mu(k)\right|^p\right]}{s_{t,k}^p \delta^p}, \end{aligned} \quad (4)$$

where  $\delta \in (0, 1)$  and  $s_{t,k}$  is fixed at time  $t$ .

Based on Assumption 2, we have  $\mathbb{E}[|\xi_i(k)|^p] \leq C$  and  $\mathbb{E}[\xi_i(k)|\mathcal{F}_{i-1}] = 0$  for any  $i \in \Phi(k)$  at  $t$ . For  $p \in (1, 2]$ ,

$$\mathbb{P}[|\hat{\mu}_t(k) - \mu(k)| \geq \delta] \leq \frac{\mathbb{E}\left[\left|\sum_{i \in \Phi(k)} \xi_i\right|^p\right]}{s_{t,k}^p \delta^p} \leq \frac{2C}{s_{t,k}^{p-1} \delta^p},$$

where we adopt Proposition 1. Thus, for any arm  $k \in S_t$ , with probability at least  $1 - \delta$

$$|\hat{\mu}_t(k) - \mu(k)| \leq \left(\frac{2C}{s_{t,k}^{p-1}\delta}\right)^{\frac{1}{p}}. \quad (5)$$

For  $p \in (2, +\infty)$ , we have

$$\mathbb{P}[|\hat{\mu}_t(k) - \mu(k)| \geq \delta] \leq \frac{C_p C}{s_{t,k}^{p/2} \delta^p}, \quad (6)$$

where we adopt Proposition 1. With probability  $1 - \delta$

$$|\hat{\mu}_t(k) - \mu(k)| \leq \left(\frac{C_p C}{s_{t,k}^{p/2} \delta}\right)^{\frac{1}{p}}. \quad (7)$$

Now we prove the results with the estimator  $\hat{\mu}_t^\dagger(k)$ , where  $k \in S_t$ . Considering  $b_i$  in Eq. (3), we define  $\mu_i^\dagger(k) \triangleq \mathbb{E}[\pi_i(k)\mathbb{1}_{[|\pi_i(k)| \leq b_i]}|\mathcal{F}_{i-1}]$ , and  $\zeta_i(k) \triangleq \mu_i^\dagger(k) - \pi_i(k)\mathbb{1}_{[|\pi_i(k)| \leq b_i]}$ , for any  $i \in \Phi(k)$ . We have  $|\zeta_i(k)| \leq 2b_i$ ,  $\mathbb{E}[\zeta_i(k)|\mathcal{F}_{i-1}] = 0$  and  $\mathbb{E}[\pi_i(k)\mathbb{1}_{[|\pi_i(k)| > b_i]}|\mathcal{F}_{i-1}] \leq B/b_i^{p-1}$ . Besides, we also have

$$\begin{aligned} \mu(k) - \hat{\mu}_t^\dagger(k) &= \frac{1}{s_{t,k}} \sum_{i \in \Phi(k)} [\mu(k) - \mu_i^\dagger(k)] \\ &\quad + \frac{1}{s_{t,k}} \sum_{i \in \Phi(k)} [\mu_i^\dagger(k) - \pi_i(k)\mathbb{1}_{[|\pi_i(k)| \leq b_i]}] \\ &= \frac{1}{s_{t,k}} \sum_{i \in \Phi(k)} (\mathbb{E}[\pi_i(k)\mathbb{1}_{[|\pi_i(k)| > b_i]}|\mathcal{F}_{i-1}] + \zeta_i(k)), \end{aligned}$$

which implies the inequality of  $\mu(k) - \hat{\mu}_t^\dagger(k) \leq \frac{1}{s_{t,k}} \sum_{i \in \Phi(k)} \left(\frac{B}{b_i^{p-1}} + \zeta_i(k)\right)$ . For  $p \in (1, 2]$ , we have  $\mathbb{E}[\zeta_i^2(k)|\mathcal{F}_{i-1}] \leq \mathbb{E}[\pi_i^2(k)\mathbb{1}_{[|\pi_i(k)| \leq b_i]}|\mathcal{F}_{i-1}] \leq \frac{B}{b_i^{p-2}}$ .

Based on Proposition 2, with probability at least  $1 - \delta$

$$\begin{aligned} \left|\sum_{i \in \Phi(k)} \zeta_i(k)\right| &\leq 2b_t \log(2/\delta) + \frac{1}{2b_t} \sum_{i \in \Phi(k)} \mathbb{E}[\zeta_i^2(k)|\mathcal{F}_{i-1}] \\ &\leq 2b_t \log(2/\delta) + s_{t,k} \frac{B}{2b_t^{p-1}}, \end{aligned} \quad (8)$$

where we adopt the design of  $\{b_i\}_{i \in \Phi(k)}$  as a non-decreasing sequence, i.e.,  $b_1 \leq b_2 \leq \dots \leq b_t$ . Thus, by setting  $b_t = \left(\frac{B s_{t,k}}{\log(2/\delta)}\right)^{\frac{1}{p}}$ , with probability at least  $1 - \delta$ , we have

$$|\hat{\mu}_t^\dagger(k) - \mu(k)| \leq 5B^{\frac{1}{p}} \left(\frac{\log(2/\delta)}{s_{t,k}}\right)^{\frac{p-1}{p}}, \quad (9)$$

where we adopt the fact of

$$\frac{1}{s_{t,k}} \sum_{i \in \Phi(k)} \frac{B}{b_i^{p-1}} \leq 2B^{\frac{1}{p}} \left(\frac{\log(2/\delta)}{s_{t,k}}\right)^{\frac{p-1}{p}}. \quad (10)$$

For  $p \in (2, +\infty)$ , by Jensen's inequality, we have

$$\mathbb{E}[\zeta_t^2(k)|\mathcal{F}_{i-1}] \leq B^{\frac{2}{p}}. \quad (11)$$

By converting the condition in  $p \in (2, +\infty)$  to the condition in  $p = 2$  with Jensen's inequality and using Eq. (9), with probability at least  $1 - \delta$ , we have

$$|\hat{\mu}_t^\dagger(k) - \mu(k)| \leq 5B^{\frac{1}{p}} \left( \frac{\log(2/\delta)}{S_{t,k}} \right)^{\frac{1}{2}}, \quad (12)$$

which completes the proof.  $\square$

**Remark 1.** In (Bubeck et al., 2013a; Vakili et al., 2013), the Bernstein inequality without martingales is adopted with an implicit assumption of sampling payoffs of an arm being independent of sequential decisions, which is informal. By contrast, in Lemma 1, conditional on  $\mathcal{F}_{t-1}$ , the subset  $S_t$  is fixed, and we adopt Bernstein inequality with martingales. Thus, we break the assumption of independent payoffs in previous work, and prove formal theoretical results of tail probabilities of estimators EA and TEA. Note that the superiority of martingales in sequential decisions has been fully discussed in (Zhao et al., 2016).

**Remark 2.** The concentration results with martingales in Lemma 1 for  $p \in (1, +\infty)$  can also be applied into regret minimization of heavy-tailed payoffs and other applications in sequential decisions. In particular, we observe that the concentration inequality of  $p = 2$  recovers that of payoffs under sub-Gaussian noises. Besides, when  $p > 2$ , the concentration results indicate constant variations with respect to  $B$ . Note that, in Lemma 1, we analyze concentration results when  $p > 2$ , which has not been analyzed in (Bubeck et al., 2013a). Compared to (Vakili et al., 2013), the concentration result in our work for TEA when  $p > 2$  enjoys a constant improvement. Since the case of  $p \in (2, +\infty)$  can be resolved by  $p = 2$ , we will focus on  $p \in (1, 2]$  in bandit algorithms for pure exploration of MAB with heavy-tailed payoffs.

## 4.2 FIXED CONFIDENCE

In this subsection, we present a bandit algorithm for pure exploration of MAB with heavy-tailed payoffs under a fixed confidence. Then, we derive upper bounds of sample complexity of the bandit algorithms.

### 4.2.1 Description of SE- $\delta$

In fixed confidence, we design our bandit algorithm for pure exploration of MAB with heavy-tailed payoffs based on the idea of SE, which is inspired by (Even-Dar et al., 2002). For SE- $\delta$ (EA), the algorithmic procedures are almost the same as that in (Even-Dar et al., 2002), which are omitted here. For SE- $\delta$ (TEA),  $\mathcal{A}$  will output an arm **Out** when  $|S_t| = 1$  with computation details shown

---

### Algorithm 1 Successive Elimination- $\delta$ (SE- $\delta$ (TEA))

---

```

1: input:  $\delta, K, p, B$ 
2: initialization:  $\hat{\mu}_1^\dagger(k) \leftarrow 0$  for any arm  $k \in [K]$ ,  $S_1 \leftarrow [K]$ , and  $b_1 \leftarrow 0$ 
3:  $t \leftarrow 1$   $\triangleright$  begin to explore arms in  $[K]$ 
4: while  $|S_t| > 1$  do
5:    $c_t \leftarrow 5B^{\frac{1}{p}} \left( \frac{\log(2K/\delta)}{t} \right)^{\frac{p-1}{p}}$   $\triangleright$  update confidence bound
6:    $b_t \leftarrow \left( \frac{Bt}{\log(2K/\delta)} \right)^{\frac{1}{p}}$   $\triangleright$  update truncating parameter
7:   for  $k \in S_t$  do
8:     play arm  $k$  and observe a payoff  $\pi_t(k)$ 
9:      $\hat{\mu}_t^\dagger(k) \leftarrow \frac{1}{t} \sum_{i=1}^t \pi_i(k) \mathbb{1}_{\{|\pi_i(k)| \leq b_i\}}$   $\triangleright$  calculate TEA
10:  end for
11:   $a_t \leftarrow \arg \max_{k \in [K]} \hat{\mu}_t^\dagger(k)$   $\triangleright$  choose the best arm at  $t$ 
12:   $S_{t+1} \leftarrow \emptyset$   $\triangleright$  create a new arm set for  $t+1$ 
13:  for  $k \in S_t$  do
14:    if  $\hat{\mu}_t^\dagger(a_t) - \hat{\mu}_t^\dagger(k) \leq 2c_t$  then
15:       $S_{t+1} \leftarrow S_{t+1} + \{k\}$   $\triangleright$  add arm  $k$  to  $S_{t+1}$ 
16:    end if
17:  end for
18:   $t \leftarrow t+1$   $\triangleright$  update time index
19: end while
20: Out  $\leftarrow S_t[0]$   $\triangleright$  assign the first entry of  $S_t$  to Out
21: return: Out

```

---

in Algorithm 1, where  $\delta$  is a given parameter. The idea is to eliminate the arm which has the farthest deviation compared with the empirical best arm in  $S_t$ .

### 4.2.2 Theoretical Guarantee of SE- $\delta$

We derive upper bounds of sample complexity of SE- $\delta$  with estimators of EA and TEA. Note that  $T$  is the time complexity of SE- $\delta$ .

**Theorem 1.** For pure exploration in MAB with  $K$  arms, with probability at least  $1 - \delta$ , Algorithm SE- $\delta$  identifies the optimal arm  $Opt$  with sample complexity as

- for SE- $\delta$ (EA)

$$T \leq \sum_{k=1}^K \left( \frac{2^{2p+1} K C}{\Delta_k^p \delta} \right)^{\frac{1}{p-1}};$$

- for SE- $\delta$ (TEA)

$$T \leq \sum_{k=1}^K \left( \frac{20B^{\frac{1}{p}}}{\Delta_k} \right)^{\frac{p}{p-1}} \log \left( \frac{2K}{\delta} \right),$$

where  $p \in (1, 2]$ .

*Proof.* We first consider EA in Eq. (2) for estimating the expected payoffs in MAB. For  $p \in (1, 2]$ , for any arm  $k \in S_t$ , we have

$$\mathbb{P}[|\hat{\mu}_t(k) - \mu(k)| \geq \delta] \leq \frac{2C}{t^{p-1} \delta^p}, \quad (13)$$

where we adopt  $s_{t,k} = t$  in SE- $\delta$ (EA). We notice the inherent characteristic of SE that, for any arm  $k \in S_t$ , we have  $\Phi(k) = \{1, 2, \dots, t\}$ .

Based on Lemma 1, for  $t = 1, 2, \dots$ , with probability at least  $1 - \delta/K$ , the following event holds

$$\mathcal{E}_t \triangleq \{k \in S_t, |\hat{\mu}_t(k) - \mu(k)| \leq c_{t_k}\},$$

where  $c_{t_k} = \left(2KC/(t_k^{p-1}\delta)\right)^{\frac{1}{p}}$  is a confidence interval. To eliminate a sub-optimal arm  $k$ , we need to play any arm  $k \in [K] \setminus \text{Opt}$  with  $t_k$  times such that

$$\hat{\Delta}_k \triangleq \hat{\mu}_{t_k}(\text{Opt}) - \hat{\mu}_{t_k}(k) \geq 2c_{t_k}. \quad (14)$$

Based on Lemma 1, with a high probability, we have

$$\hat{\Delta}_k \geq \mu(\text{Opt}) - c_{t_k} - (\mu(k) + c_{t_k}) = \Delta_k - 2c_{t_k},$$

where  $c_{t_k}$  is a confidence interval. To satisfy Eq. (14), we are ready to set

$$\Delta_k - 2c_{t_k} \geq 2c_{t_k}. \quad (15)$$

To solve the above inequality, we are ready to have that  $t_k = \left(\frac{2^{2p+1}KC}{\Delta_k^p \delta}\right)^{\frac{1}{p-1}}$  is sufficient. The total sample complexity is  $T = t_2 + \sum_{k=2}^K t_k$ , because the number of pulling the optimal arm  $t_1 = t_2$ . This implies, with probability at least  $1 - \delta$ , we have

$$T \leq \sum_{k=1}^K \left(\frac{2^{2p+1}KC}{\Delta_k^p \delta}\right)^{\frac{1}{p-1}}. \quad (16)$$

Now we consider TEA in Eq. (3) for estimating the expected payoffs in MAB. Similarly, for  $p \in (1, 2]$ , with probability at least  $1 - \delta$ , we have

$$T \leq \sum_{k=1}^K \left(\frac{20B^{\frac{1}{p}}}{\Delta_k}\right)^{\frac{p}{p-1}} \log\left(\frac{2K}{\delta}\right), \quad (17)$$

which completes the proof.  $\square$

### 4.3 FIXED BUDGET

In this subsection, we present a bandit algorithm for pure exploration of MAB with heavy-tailed payoffs under a fixed budget. Then, we derive upper bounds of probability of error for the bandit algorithms.

#### 4.3.1 Description of SR- $T$

For SR- $T$ (EA), we omit the algorithm because it is almost the same as that in (Audibert and Bubeck, 2010). For SR- $T$ (TEA), we design a bandit algorithm for pure exploration of MAB with heavy-tailed payoffs based on the idea of SR, with computation details shown in Algorithm 2, where  $T$  is a given parameter. The high-level

---

#### Algorithm 2 Successive Rejects- $T$ (SR- $T$ (TEA))

---

```

1: input  $T, K, p, B, \underline{\Delta} > 0$ 
2: initialization:  $\hat{\mu}^\dagger(k) \leftarrow 0$  for any arm  $k \in [K]$ ,  $S_1 \leftarrow [K]$ ,  $n_0 \leftarrow 0$ ,  $b \leftarrow 0$  and  $\bar{K} \leftarrow \sum_{i=1}^K \frac{1}{i}$ 
3:  $b \leftarrow \left(\frac{3Bp}{\underline{\Delta}}\right)^{\frac{1}{p-1}}$   $\triangleright$  calculate truncating parameter
4: for  $k \in S_1$  do
5:    $\Phi(k) \leftarrow \emptyset$   $\triangleright$  construct sets to store time index
6: end for
7: for  $k \in [K-1]$  do
8:    $n_k \leftarrow \lceil \frac{T-K}{K(K+1-k)} \rceil$   $\triangleright$  calculate  $n_k$  at stage  $k$ 
9:    $n \leftarrow n_k - n_{k-1}$   $\triangleright$  calculate the number of times to pull arms
10:  for  $y \in S_k$  do
11:    for  $i \in [n]$  do
12:       $t \leftarrow t + 1$ 
13:      play arm  $y$ , and observe a payoff  $\pi_t(y)$ 
14:       $\Phi(y) \leftarrow \Phi(y) + \{t\}$   $\triangleright$  store time index for arm  $y$ 
15:    end for
16:     $\hat{\mu}_k^\dagger(y) \leftarrow \frac{1}{|\Phi(y)|} \sum_{i \in \Phi(y)} \pi_i(y) \mathbb{1}_{|\pi_i(y)| \leq b}$ 
17:  end for
18:   $a_k \leftarrow \arg \min_{y \in S_k} \hat{\mu}_k^\dagger(y)$   $\triangleright$  choose the worst arm at  $k$ 
19:   $S_{k+1} \leftarrow S_k - \{a_k\}$   $\triangleright$  successively reject arm  $a_k$ 
20: end for
21:  $\text{Out} \leftarrow S_K[0]$   $\triangleright$  assign the first entry of  $S_K$  to  $\text{Out}$ 
22: return:  $\text{Out}$ 

```

---

idea is to conduct non-uniform pulling of arms by  $K-1$  phases, and SR- $T$  rejects a worst empirical arm for each phase. The reject operation is based on EA or TEA, and we distinguish the two cases by SR- $T$ (EA) and SR- $T$ (TEA).

For simplicity, we show SR- $T$ (TEA) in Algorithm 2, where  $\underline{\Delta} > 0$  is a design parameter for the estimator of TEA. The design parameter  $\underline{\Delta}$  helps to calculate the truncating parameter  $b$  in SR- $T$ (TEA). Usually, we set  $\underline{\Delta} \leq \Delta_k$  for any  $k \in [K]$ .

#### 4.3.2 Theoretical Guarantee of SR- $T$

We derive upper bounds of probability of error for SR- $T$  with estimators of EA and TEA. We have the following theorem for SR- $T$ .

**Theorem 2.** *For pure exploration in MAB with  $K$  arms, if Algorithm SR- $T$  is run with a fixed budget  $T$ , we have probability of error for  $p \in (1, 2]$  as*

- for SR- $T$ (EA)

$$\mathbb{P}[\text{Out} \neq \text{Opt}] \leq 2^{p+1}CK(K-1)H_2^p \left(\frac{\bar{K}}{T-K}\right)^{p-1};$$

- for SR- $T$ (TEA)

$$\mathbb{P}[\text{Out} \neq \text{Opt}] \leq 2K(K-1) \exp\left(-\frac{(T-K)\bar{B}_1}{\bar{K}K\underline{\Delta}^{p/(1-p)}}\right),$$

$$\text{where } \bar{B}_1 = \frac{p-1}{4(2^p 3Bp^p)^{\frac{1}{p-1}}}.$$

*Proof.* We first consider EA in Eq. (2) for estimating the expected payoffs in MAB. For  $p \in (1, 2]$ , we have

$$\begin{aligned} \mathbb{P}[\text{Out} \neq \text{Opt}] &\leq \sum_{k=1}^{K-1} \sum_{i=K+1-k}^K \mathbb{P}[\hat{\mu}_k(\text{Opt}) \leq \hat{\mu}_k(i)] \\ &\leq \sum_{k=1}^{K-1} \sum_{i=K+1-k}^K \mathbb{P}[\hat{\mu}_k(i) - \mu(i) + \mu(\text{Opt}) - \hat{\mu}_k(\text{Opt}) \geq \Delta_i] \\ &\leq \sum_{k=1}^{K-1} \sum_{i=K+1-k}^K \frac{4C}{n_i^{p-1} \left(\frac{\Delta_i}{2}\right)^p} \\ &\leq \sum_{k=1}^{K-1} \frac{2^{p+2} C_k}{n_k^{p-1} \Delta_{K+1-k}^p}, \end{aligned} \quad (18)$$

where the inequality of Eq. (18) is due to the results in Lemma 1 by setting  $s_{t,k} = n_k$ . Besides, we notice that

$$n_k^{p-1} \Delta_{K+1-k}^p \geq \frac{1}{H_2^p} \left( \frac{T-K}{\bar{K}} \right)^{p-1},$$

which implies that

$$\mathbb{P}[\text{Out} \neq \text{Opt}] \leq 2^{p+1} C K (K-1) H_2^p \left( \frac{\bar{K}}{T-K} \right)^{p-1}.$$

Now we consider TEA in Eq. (3) for estimating the expected payoffs in MAB. By considering the design of  $b$  in SR- $T$ (TEA), we have a similar result of Lemma 1. Then, for  $p \in (1, 2]$ , we have probability of error as

$$\begin{aligned} \mathbb{P}[\text{Out} \neq \text{Opt}] &\leq \sum_{k=1}^{K-1} \sum_{i=K+1-k}^K \mathbb{P}[\hat{\mu}_k^\dagger(\text{Opt}) \leq \hat{\mu}_k^\dagger(i)] \\ &\leq \sum_{k=1}^{K-1} \sum_{i=K+1-k}^K \mathbb{P}[\hat{\mu}_k^\dagger(i) - \mu(i) + \mu(\text{Opt}) - \hat{\mu}_k^\dagger(\text{Opt}) \geq \Delta] \\ &\leq 2K(K-1) \exp\left(-\frac{(T-K)\bar{B}_1}{\bar{K} K \underline{\Delta}^{p/(1-p)}}\right), \end{aligned} \quad (20)$$

which completes the proof.  $\square$

## 5 EXPERIMENTS

In this section, we conduct experiments via synthetic and real-world data to evaluate the performance of the proposed bandit algorithms. We run experiments in a personal computer with Intel CPU@3.70GHz and 16GB memory. For the setting of fixed confidence, we compare the sample complexities of SE- $\delta$ (EA) and SE- $\delta$ (TEA). For the setting of fixed budget, we compare the error probabilities of SR- $T$ (EA) and SR- $T$ (TEA).

### 5.1 SYNTHETIC DATA AND RESULTS

For verifications, we adopt two synthetic data (named as S1-S2) in the experiments, of which statistics are shown in Table 2. The data are generated from *Student's t-distribution* with 3 degrees of freedom. In experiments, we run multiple epochs for each dataset, with each epoch containing ten independent experiments for

Table 2: Statistics of used synthetic data.

dataset	#arms	$\{\mu(k)\}$	heavy-tailed $\{p, B, C\}$
S1	10	one arm is 2.0 and nine arms are over [0.7, 1.5] with a uniform gap	{2, 7, 3}
S2	10	one arm is 2.0 and nine arms are over [1.0, 1.8] with a uniform gap	{2, 7, 3}

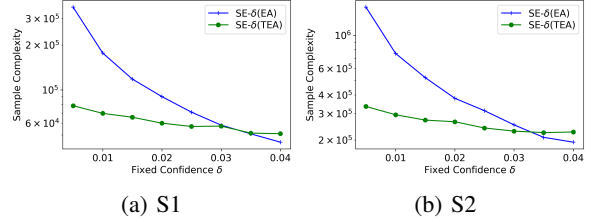


Figure 1: Sample complexity for SE- $\delta$  in pure exploration of MAB with heavy-tailed payoffs.

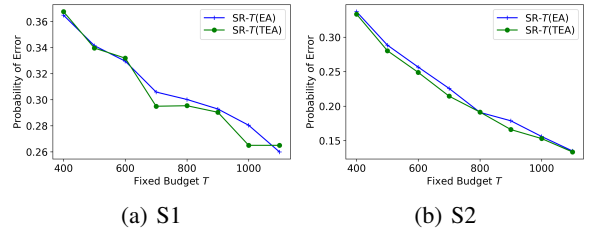


Figure 2: Probability of error for SR- $T$  in pure exploration of MAB with heavy-tailed payoffs.

best arm identification of MAB. Besides, we set the value of fixed confidence from 0.005 to 0.040 with a uniform gap of 0.005. We set the value of fixed budget from 400 to 1100 with a uniform gap of 100.

We show experimental results in Figures 1 and 2, where both proposed algorithms are effective for pure exploration of MAB with heavy-tailed payoffs. In particular, in fixed-confidence setting, sample complexity decreases with increasing value of  $\delta$ . In fixed-budget setting, probability of error converges to zero with increasing value of  $T$ . Besides, for fixed-confidence setting, SE- $\delta$ (TEA) beats SE- $\delta$ (EA) in both datasets with small  $\delta$  due to a better control of confidence interval. The experimental results also reflect that the concentration properties of EA are much weaker than those of TEA. For fixed-budget setting, SR- $T$ (TEA) is comparable to SR- $T$ (EA) due to the selection of truncating parameter.

### 5.2 FINANCIAL DATA AND RESULTS

It has been pointed out that financial data show the inherent characteristic of heavy tails (Panahi, 2016). We choose a financial application of identifying the most profitable cryptocurrency in a given pool of digital cur-

Table 3: Statistical property of ten selected cryptocurrencies with hourly returns from Feb. 3rd, 2018 to Apr. 27th, 2018. KS-test1 denotes Kolmogrov-Smirnov (KS) test with a null hypothesis that real data follow a Gaussian distribution. KS-test2 denotes KS test with a null hypothesis that real data follow a *Student’s t-distribution*.

symbol	empirical statistics (mean $\times 10^3$ , variance $\times 10^3$ )	KS-test1 (statistic, $\bar{p}$ -value)	KS-test2 (statistic, $\bar{p}$ -value)
BTC	(0.36, 0.54)	(0.08, 0.005)	(0.05, 0.20)
ETC	(0.29, 1.03)	(0.07, 0.02)	(0.03, 0.89)
XRP	(0.33, 0.94)	(0.09, 0.0004)	(0.03, 0.61)
BCH	(0.78, 0.92)	(0.08, 0.001)	(0.03, 0.64)
<b>EOS</b>	<b>(1.56, 1.18)</b>	(0.09, 0.0002)	(0.03, 0.88)
LTC	(0.68, 0.86)	(0.10, 0.0002)	(0.04, 0.49)
ADA	(0.02, 1.22)	(0.07, 0.03)	(0.02, 0.99)
XLM	(0.62, 0.12)	(0.07, 0.02)	(0.03, 0.80)
IOT	(0.68, 0.11)	(0.07, 0.02)	(0.04, 0.57)
NEO	(-0.31, 1.26)	(0.10, 0.0002)	(0.04, 0.53)

Table 4: Estimated parameters for ten cryptocurrencies.

symbol	degree of freedom	$(p, B, C)$ in experiments
BTC	3.50	$(2.1577 \times 10^{-3}, 1.575 \times 10^{-3})$
ETC	3.81	
XRP	2.53	
BCH	3.00	
EOS	2.90	
LTC	2.75	
ADA	3.55	
XLM	3.81	
IOT	4.66	
NEO	3.13	

rencies. The identification for the most profitable cryptocurrency among the top ten cryptocurrency in terms of market value is motivated by the practical scenario that an investor would like to invest a fixed budget of money in a cryptocurrency and get return as much as possible.

For experiments, we get hourly price data of the ten selected cryptocurrencies<sup>1</sup>, and show the statistics of real data in Table 3. In the table, we conduct a statistical analysis in hindsight with hourly returns of cryptocurrency from February 3rd, 2018 to April 27th, 2018. From the table, we find that the optimal option in hindsight is EOS in terms of the maximal empirical mean of hourly payoffs. Besides, we conduct Kolmogrov-Smirnov (KS) test to fit real data of a cryptocurrency to a distribution. In particular, via KS test, we know that the null hypothesis of real data following a Gaussian distribution is rejected, because  $\bar{p}$ -value is smaller than a significant level of 0.05. We observe that real data of cryptocurrency are likely to follow a *Student’s t-distribution* via KS test in Table 3.

<sup>1</sup><https://www.cryptocompare.com/>

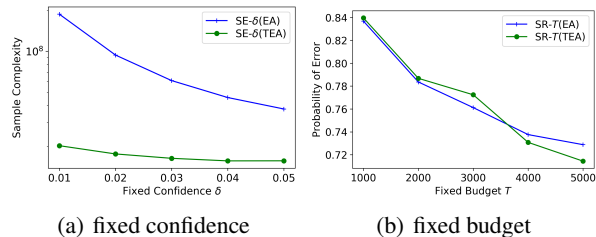


Figure 3: Pure exploration of cryptocurrency.

With the above statistical analyses, we can fit real data of cryptocurrency to a *Student’s t-distribution*, and obtain distribution parameters shown in Table 4. Based on the property of *Student’s t-distribution*, we can set  $p = 2$ , and estimate  $B$  and  $C$  as shown in the table.

Via a similar setting to that of synthetic data, we show the results on pure exploration of top ten cryptocurrencies in Figure 3. Note that, due to limitation of data points in the setting of fixed confidence, we generate payoffs from *Student’s t-distributions* fitting to real data. But in the setting of fixed budget, we adopt exactly real financial data. We have similar observations as those in synthetic data. It is worth mentioning that, TEA algorithm outperforms EA algorithm in fixed-confidence setting when the value of  $\delta$  is small. Besides, TEA is comparable to EA in fixed-budget setting because the truncating parameter in Algorithm 2 only has budget information and does not increase with the number of samples. Overall, with synthetic and real-world data, we have verified the effectiveness of our two algorithms.

## 6 CONCLUSION

In this paper, we broke the assumption of payoffs under sub-Gaussian noises in pure exploration of MAB, and investigated best arm identification of MAB with a general assumption that the  $p$ -th moments of stochastic payoffs are bounded, where  $p \in (1, +\infty)$ . We have technically analyzed tail probabilities of empirical average and truncated empirical average for estimating expected payoffs in sequential decisions. Besides, we proposed two bandit algorithms for pure exploration of MAB with heavy-tailed payoffs based on SE and SR. Finally, we derived theoretical guarantees of the proposed bandit algorithms, and demonstrated the effectiveness of bandit algorithms in pure exploration of MAB with heavy-tailed payoffs.

## Acknowledgments

The work described in this paper was partially supported by the Research Grants Council of the Hong Kong Special Administrative Region, China (No. CUHK 14208815 and No. CUHK 14210717 of the General Research Fund), and Microsoft Research Asia (2018 Microsoft Research Asia Collaborative Research Award).

## References

- J.-Y. Audibert and S. Bubeck. Best arm identification in multi-armed bandits. In *Proceedings of COLT*, pages 41–53, 2010.
- P. Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3:397–422, 2002.
- P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.
- S. Bubeck, R. Munos, and G. Stoltz. Pure exploration in multi-armed bandits problems. In *Proceedings of ALT*, pages 23–37, 2009.
- S. Bubeck, N. Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.
- S. Bubeck, N. Cesa-Bianchi, and G. Lugosi. Bandits with heavy tail. *IEEE Transactions on Information Theory*, 59(11):7711–7717, 2013a.
- S. Bubeck, T. Wang, and N. Viswanathan. Multiple identifications in multi-armed bandits. In *Proceedings of ICML*, pages 258–265, 2013b.
- A. Carpentier and M. Valko. Extreme bandits. In *Proceedings of NIPS*, pages 1089–1097, 2014.
- S. Chen, T. Lin, I. King, M. R. Lyu, and W. Chen. Combinatorial pure exploration of multi-armed bandits. In *Proceedings of NIPS*, pages 379–387, 2014.
- W. Chu, L. Li, L. Reyzin, and R. E. Schapire. Contextual bandits with linear payoff functions. In *Proceedings of AIS-TATS*, pages 208–214, 2011.
- S. Dharmadhikari, V. Fabian, and K. Jogdeo. Bounds on the moments of martingales. *The Annals of Mathematical Statistics*, pages 1719–1723, 1968.
- E. Even-Dar, S. Mannor, and Y. Mansour. PAC bounds for multi-armed bandit and Markov decision processes. In *Proceedings of COLT*, pages 255–270. Springer, 2002.
- V. Gabillon, M. Ghavamzadeh, and A. Lazaric. Best arm identification: A unified approach to fixed budget and fixed confidence. In *Proceedings of NIPS*, pages 3212–3220, 2012.
- V. Gabillon, A. Lazaric, M. Ghavamzadeh, R. Ortner, and P. Bartlett. Improved learning complexity in combinatorial pure exploration bandits. In *Proceedings of AISTAT*, pages 1004–1012, 2016.
- K. Jamieson and R. Nowak. Best-arm identification algorithms for multi-armed bandits in the fixed confidence setting. In *Proceedings of IEEE CISS*, pages 1–6, 2014.
- K. Jamieson, M. Malloy, R. Nowak, and S. Bubeck. liUCB: An optimal exploration algorithm for multi-armed bandits. In *Proceedings of COLT*, pages 423–439, 2014.
- Z. Karnin, T. Koren, and O. Somekh. Almost optimal exploration in multi-armed bandits. In *Proceedings of ICML*, pages 1238–1246, 2013.
- E. Kaufmann, O. Cappé, and A. Garivier. On the complexity of best-arm identification in multi-armed bandit models. *The Journal of Machine Learning Research*, 17(1):1–42, 2016.
- N. Korda, B. Szörényi, and L. Shuai. Distributed clustering of linear bandits in peer to peer networks. In *Proceedings of ICML*, pages 1301–1309, 2016.
- B. Kveton, C. Szepesvári, Z. Wen, and A. Ashkan. Cascading bandits: Learning to rank in the cascade model. In *Proceedings of ICML*, pages 767–776, 2015.
- T. Lattimore. A scale free algorithm for stochastic bandits with bounded kurtosis. In *Proceedings of NIPS*, pages 1583–1592, 2017.
- T. Lattimore, K. Crammer, and C. Szepesvári. Linear multi-resource allocation with semi-bandit feedback. In *Proceedings of NIPS*, pages 964–972, 2015.
- S. Li, A. Karatzoglou, and C. Gentile. Collaborative filtering bandits. In *Proceedings of ACM SIGIR*, pages 539–548, 2016.
- J. Liebeherr, A. Burchard, and F. Ciucu. Delay bounds in communication networks with heavy-tailed and self-similar traffic. *IEEE Transactions on Information Theory*, 58(2):1010–1024, 2012.
- S. Mannor and J. N. Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *Journal of Machine Learning Research*, 5:623–648, 2004.
- A. M. Medina and S. Yang. No-regret algorithms for heavy-tailed linear bandits. In *Proceedings of ICML*, pages 1642–1650, 2016.
- H. Panahi. Model selection test for the heavy-tailed distributions under censored samples with application in financial data. *International Journal of Financial Studies*, 4(4):24, 2016.
- H. Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.
- A. Sani, A. Lazaric, and R. Munos. Risk-aversion in multi-armed bandits. In *Proceedings of NIPS*, pages 3275–3283, 2012.
- Y. Seldin, F. Laviolette, N. Cesa-Bianchi, J. Shawe-Taylor, and P. Auer. PAC-Bayesian inequalities for martingales. *IEEE Transactions on Information Theory*, 58(12):7086–7093, 2012.
- S. Vakili, K. Liu, and Q. Zhao. Deterministic sequencing of exploration and exploitation for multi-armed bandit problems. *IEEE Journal of Selected Topics in Signal Processing*, 7(5):759–767, 2013.
- B. von Bahr, C.-G. Esseen, et al. Inequalities for the  $r$ -th absolute moment of a sum of random variables,  $1 \leq r \leq 2$ . *The Annals of Mathematical Statistics*, 36(1):299–303, 1965.
- Q. Wu, H. Wang, Q. Gu, and H. Wang. Contextual bandits in a collaborative environment. In *Proceedings of ACM SIGIR*, pages 529–538, 2016.
- X. Yu, I. King, and M. R. Lyu. Risk control of best arm identification in multi-armed bandits via successive rejects. In *Proceedings of IEEE ICDM*, pages 1147–1152, 2017a.
- X. Yu, M. R. Lyu, and I. King. CBRAP: Contextual bandits with random projection. In *Proceedings of AAAI*, pages 2859–2866, 2017b.
- S. Zhao, E. Zhou, A. Sabharwal, and S. Ermon. Adaptive concentration inequalities for sequential decision problems. In *Proceedings of NIPS*, pages 1343–1351, 2016.
- T. Zhao and I. King. Locality-sensitive linear bandit model for online social recommendation. In *Proceedings of ICONIP*, pages 80–90, 2016.

---

# Counterfactual Normalization: Proactively Addressing Dataset Shift Using Causal Mechanisms

---

**Adarsh Subbaswamy**

Department of Computer Science  
Johns Hopkins University  
Baltimore, MD 21218

**Suchi Saria**

Department of Computer Science  
Johns Hopkins University  
Baltimore, MD 21218

## Abstract

Predictive models can fail to generalize from training to deployment environments because of dataset shift, posing a threat to model reliability in practice. As opposed to previous methods which use samples from the target distribution to reactively correct dataset shift, we propose using graphical knowledge of the causal mechanisms relating variables in a prediction problem to proactively remove variables that participate in spurious associations with the prediction target, allowing models to generalize across datasets. To accomplish this, we augment the causal graph with latent counterfactual variables that account for the underlying causal mechanisms, and show how we can estimate these variables. In our experiments we demonstrate that models using good estimates of the latent variables instead of the observed variables transfer better from training to target domains with minimal accuracy loss in the training domain.

## 1 INTRODUCTION

Supervised machine learning is concerned with predicting a target output label  $T$  from input features  $\mathbf{X}$ . Classical learning frameworks assume that training and test data are independently and identically distributed from a fixed distribution  $p(\mathbf{X}, T)$ . When this assumption does not hold, training with classical frameworks can yield models with unreliable and, in the case of safety-critical applications like medicine, dangerous predictions (Dyagilev and Saria, 2015; Caruana et al., 2015; Schulam and Saria, 2017). For example, prediction systems are often deployed in dynamic environments that systematically differ from the one in which the historical training data was collected—a problem known as *dataset shift* which results in poor

generalization. Methods for addressing dataset shift are typically reactive: they use unlabeled data from the target deployment environment during the learning process (see Quionero-Candela et al. (2009) for an overview). However, when the differences in environments are unknown prior to model deployment (e.g., no available data from the target environment or target environments that have not yet been conceived), it is important to understand what aspects of the prediction problem can change and how we can train models that will be robust to these changes. We consider this problem of *proactively addressing dataset shift* in this work.

In particular, we will guard against *spurious associations* between predictors and the target—non-causal marginal relationships that often do not generalize due to shifts in training and test distributions. To illustrate, consider an example prediction problem of medical screening. The features ( $\mathbf{X}$ ) are blood pressure (BP)  $Y$  and congestive heart failure  $C$ . The label we want to predict is whether or not a patient has meningitis  $T$ . Underlying every prediction problem is a directed acyclic graph (DAG), such as the one in Figure 1a, which describes the *causal mechanisms* (general directional knowledge of causes and effects, e.g.,  $C \rightarrow Y$ : heart failure causes low BP) between the variables that hold in all environments. In this graph,  $T$  and  $C$  are not *causally related* to each other:  $C$  is neither a causal ancestor nor a causal descendant of  $T$ . By *d*-separation (Koller and Friedman, 2009), unless we condition on  $Y$ , the two are statistically independent:  $T \perp\!\!\!\perp C$ . However, *selection bias* (Figure 1b) or domain-dependent *confounding by indication* (Figure 1c) can introduce a spurious association:  $T \not\perp\!\!\!\perp C$ . We now define these cases of dataset shift and show how they threaten model reliability.

Selection bias occurs when certain subpopulations (with respect to  $T$  and  $C$ ) are underrepresented in the training data ( $S=1$ ) which can result in inaccurate predictions in the deployment population. For example, suppose patients with heart failure but without meningitis ( $C = 1, T = 0$ ) are underrepresented because they rarely

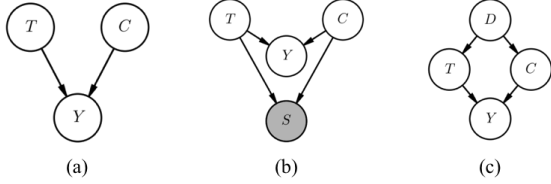


Figure 1: (a) The DAG capturing causal mechanisms for the medical screening example. The features are blood pressure  $Y$  and heart failure  $C$ . The target label  $T$  is meningitis. (b) Selection bias  $S$  is included. (c) Domain-dependent confounding is shown.  $C$  represents narcotics and  $D$  a latent risk factor, brain surgery. Shaded nodes denote observed variables.

visit this hospital since they manage their chronic condition using a high quality local chronic care clinic. This results in a spurious positive association between  $T$  and  $C$ , with the strength of the association depending on the degree of selection bias. Further, the distribution  $p(T|\mathbf{X})$  (i.e.,  $p(T|C, Y)$  in our example) in the deployment population can differ from the distribution in the training population  $p(T|C, Y, S = 1)$ . For the case of the under-represented ( $C = 1, T = 0$ ) subpopulation, the screening model will be poorly calibrated and overestimate the risk of meningitis in patients with the chronic condition (i.e.,  $p(T = 1|C = 1, Y)$  predictions will be too high). These systematic errors on a subpopulation pose a threat to model reliability.

Domain-dependent confounding, shown in Figure 1c, also threatens model reliability. Suppose  $C$  were instead an indicator for narcotic pain medications which lower BP. Doctors sometimes prescribe narcotics after brain surgery ( $D$  in Figure 1c), a risk factor for meningitis that may not be recorded in the data. The policy  $p(C|D)$  doctors use to prescribe narcotics varies between domains (i.e., doctors and hospitals) which also causes  $p(T|Y, C)$  to vary. For example, one hospital may freely prescribe narcotics (resulting in a positive association between  $T$  and  $C$ ) while another hospital may carefully restrict the number of painkiller prescriptions. A model trained on data from the first hospital will overestimate the risk of meningitis in patients treated with narcotics at the second hospital. However, when confounders are observed and differences in policies are known beforehand, adjustments can be made by discounting the policies during learning (e.g., Swaminathan and Joachims (2015); Schulam and Saria (2017)). Otherwise, instead of learning to predict using a domain-specific association between the target and the treatment that will not generalize, we can remove the treatment information from the model or, as we propose, retain relevant information by accounting for the effects of the medication.

In both cases of dataset shift, due to either the *collider*  $S$  (Figure 1b) or the confounder  $D$  (Figure 1c), the graphs contain the spurious marginal association  $T \not\perp C$  that does not generalize across datasets. When we do not have data from the target distribution or the differences in policies across domains are unknown, we propose modifying the graph to contain latent *counterfactual* variables which, when estimated, allow us to remove the variables that participate in spurious associations with  $T$  (such as  $C$  in Figure 1) from the problem. Specifically, if we somehow knew an adjusted value of  $Y$ , denoted  $Y(C = \emptyset)$ —the value of  $Y$  for which the effects of  $C$  were removed (e.g., the blood pressure had the patient not had heart failure or not been given narcotics)—then  $C$  would no longer be causally relevant for predicting  $T$ . This concept is inspired by *potential outcomes* (Neyman, 1923; Rubin, 1974) in causal inference. However, we do not need to assume full knowledge of the causal DAG (which also includes latent factors and intermediate variables), required for the assumptions of causal inference methods. Instead, we only use knowledge of the causal mechanisms between the variables in a prediction problem.

In this paper we make the following contributions. First, we identify variables in a DAG capturing causal mechanisms which make a statistical model *vulnerable* to learning spurious associations that do not generalize across datasets. Second, we define a *node-splitting* operation which modifies the DAG to contain interpretable latent counterfactual variables which render the vulnerable variables irrelevant in the prediction problem. Third, we provide conditions for estimating the latent variables as adjustments of observed features. Fourth, we explain how the proposed method can make a classification problem measurably simpler due to reduced variance of the latent features. On simulated data we evaluate the quality of model predictions when the accuracy of the latent variable estimates changes. Then, on a real world medical classification task, we demonstrate that the proposed method allows us to remove vulnerable variables while preserving relevant information.

## 2 RELATED WORK

**Spurious Associations:** Predictive modeling methods for accounting for spurious associations in data typically require representative unlabeled samples from the test distribution. For example, the classic selection bias paradigm is to detect and correct bias in the training distribution by using unlabeled test samples to estimate the probability of selection in the training data so the training examples can be discounted during learning (see e.g., Heckman (1977); Zadrozny (2004); Huang et al. (2007); Storkey (2009)).

Beyond predictive modeling, previous work has consid-



ered estimation of causal models in the presence of selection bias and confounding. For example, Spirtes et al. (1995) learn the structure of the causal DAG from data affected by selection bias. Others have studied methods and conditions for *identification* of causal effects under spurious associations due to selection bias and confounding (e.g., Bareinboim and Pearl (2012); Bareinboim and Tian (2015); Correa et al. (2018)). Most relevantly, Correa and Bareinboim (2017) determine conditions under which interventional distributions are identified without using external data. Our work is concerned with statistical prediction under selection bias or domain-dependent confounding without external data.

**Transportability:** The goal of an experiment is for the findings to generalize beyond a single study, a concept known as *external validity* (Campbell and Stanley, 1963). Similarly, in causal inference *transportability*, formalized in Pearl and Bareinboim (2011), transfers causal effect estimates from one environment to another. Bareinboim and Pearl (2013) further generalize this to transfer causal knowledge from multiple source domains to a single target domain. Like these works, we assume the structure of the causal mechanism DAG is the same in the source and any relevant target domains. However, rather than transfer causal estimates from source to target, the proposed method learns a single statistical model whose predictions should perform well on the source domain while also generalizing well to new domains.

**Graphical Representations of Counterfactuals:** The node-splitting operation we introduce in Section 3.2.2 is similar to the node-splitting operation in Single World Intervention Graphs (SWIGs) (Richardson and Robins, 2013). However, intervening in a SWIG results in a causal generative graph for a potential outcome with the factual outcome removed from the graph. By contrast, the node-splitting operation of the proposed method results in a modified causal generative graph of the factual outcomes, with new intermediate counterfactual variables. Other graphical representations such as twin networks (Pearl, 2009) and counterfactual graphs (Shpitser and Pearl, 2007) simultaneously represent factual and counterfactual outcomes, rather than the intermediate counterfactuals exploited in this work.

### 3 METHODS

Counterfactual Normalization consists of three steps: identification of variables that are vulnerable to participating in spurious associations with the target that do not generalize across datasets, a node-splitting operation to place latent counterfactual variables onto the causal DAG such that they  $d$ -separate the target from the vulnerable variables, and estimation of the relevant latent variables.

We will first review necessary background about potential outcomes and structural equation models before introducing the method.

#### 3.1 BACKGROUND

##### 3.1.1 Potential Outcomes

The proposed method involves the estimation of counterfactuals, which can be formalized using the Neyman-Rubin potential outcomes framework (Neyman, 1923; Rubin, 1974). For outcome variable  $Y$  and intervention  $A$ , we denote the potential outcome by  $Y(a)$ : the value  $Y$  would have if  $A$  were observed to be  $a$ .

In general, the distributions  $p(Y(a))$  and  $p(Y|A = a)$  are not equal. For this reason, estimation of the distribution of the potential outcomes relies on two assumptions:

**Consistency:** The distribution of the potential outcome under the observed intervention is the same as the distribution of the observed outcome. This implies  $p(Y(a)|A = a) = p(Y|A = a)$ .

**Conditional Ignorability:**  $Y(a) \perp\!\!\!\perp A|X, \forall a \in A$ . There are no unobserved confounders. This implies  $p(Y(a)|X, A = a') = p(Y(a)|X, A = a)$ .

##### 3.1.2 Counterfactuals and SEMs

Shpitser and Pearl (2008) develop a causal hierarchy consisting of three layers of increasing complexity: association, intervention, and counterfactual. Many works in causal inference are concerned with estimating average treatment effects—a task at the intervention layer because it uses information about the interventional distribution  $p(Y(a)|X)$ . In contrast, the proposed method requires counterfactual queries which use the distribution  $p(Y(a)|Y, a', X)$  s.t.  $a \neq a'$ <sup>1</sup>. That is, given that we observed an individual’s outcome to be  $Y$  under intervention  $a'$ , what would the distribution of their outcome have been under a different intervention  $a$ ?

In addition to the assumptions for estimating potential outcomes, computing counterfactual queries requires functional or structural knowledge (Pearl, 2009). We can represent this knowledge using causal structural equation models (SEMs). These models assume variables  $X_i$  are functions of their immediate parents in the generative causal DAG and exogenous noise  $u_i$ :  $X_i = f_i(pa(X_i), u_i)$ . Reasoning counterfactually at the level of an individual unit requires assumptions on the form of the functions  $f_i$  and independence of the  $u_i$ , because typically we are inter-

<sup>1</sup>The distinction is that  $p(Y(a)|X)$  reasons about the effects of causes while  $p(Y(a)|Y, a', X)$  reasons about the causes of effects (see, e.g., Pearl (2015)).

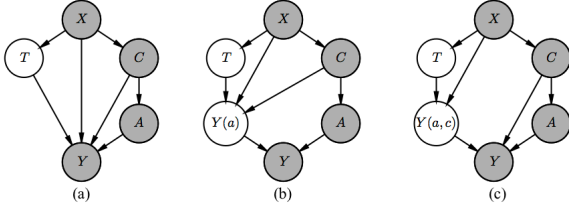


Figure 2: (a) The DAG of causal mechanisms for the medical screening example. (b) The modified DAG after node-splitting yielding the latent signal value under no different treatment  $Y(a)$ . (c) The modified DAG after node-splitting yielding the latent signal value under no treatment and no chronic condition  $Y(a, c)$ .

ested in reasoning about interventions in which the exogenous noise variables remain fixed. We build on this to estimate the latent counterfactual variables.

### 3.2 COUNTERFACTUAL NORMALIZATION

Counterfactual Normalization uses a DAG,  $\mathcal{G}$ , that leverages any prior knowledge of the causal mechanisms relating variables in a prediction problem with target variable  $T$ , assumed to be binary for the purposes of explanation. We further assume that the predictors form a Markov blanket<sup>2</sup> of  $T$  in  $\mathcal{G}$ . To sketch the method, recall that in the example in Figure 1a we identified that  $C$  is vulnerable to participating in a spurious association with  $T$ . To retain generalizable information about  $C$  we will estimate  $Y(C = \emptyset)$ , the counterfactual blood pressure if a patient did not have heart failure or did not receive narcotics. Instead of predicting  $T$  by modeling  $p(T|Y, C)$  which likely will not generalize, we will instead model  $p(T|Y(\emptyset))$  which notably does not contain  $C$  as a feature. To explain the method’s steps in complete detail, we will consider an expanded version of the meningitis example. In Figure 2a we have added a variable  $A$  to represent medications given to the patient, and a variable  $X$  to represent demographic factors (e.g., age).

#### 3.2.1 Identification of Vulnerable Variables

Spurious associations are marginal non-causal associations with  $T$  in the training data. Since we are using the Markov blanket of  $T$  for prediction, a variable  $v \in \mathcal{G}$  makes a model *vulnerable* to learning a spurious association if it is neither an ancestor nor a descendant of the target variable  $T$  while being a member of the Markov

<sup>2</sup>The Markov blanket of a target variable is a set of variables such that, conditioned on the set, the target is independent of all other variables not in the set (Koller and Friedman, 2009). Graphically, these are the target’s parents, children, and other parents of its children.

---

#### Algorithm 1: Node-splitting Operation

---

**Input:** Graph  $\mathcal{G}$ , child of target node  $Y$ , observed parents of  $Y$  to intervene upon  $\mathbf{P}$

**Output:** Modified graph  $\mathcal{G}^*$

1. Insert counterfactual node  $Y(\mathbf{P} = \emptyset)$
  2. Delete edges  $\{x \rightarrow Y : x \in pa(Y) \setminus \mathbf{P}\}$
  3. Insert edges  $\{x \rightarrow Y(\mathbf{P} = \emptyset) : x \in pa(Y) \setminus \mathbf{P}\}$
  4. Insert edge  $Y(\mathbf{P} = \emptyset) \rightarrow Y$
- 

blanket of  $T$ . Thus, vulnerable variables are parents of children of  $T$  that are non-causally associated with the target variable.

In Figure 2(a), the vulnerable variables are  $C$  and  $A$  because they are parents of  $Y$  (a child of  $T$ ) without being descendants or ancestors of  $T$ .

#### 3.2.2 Node-Splitting

To remove vulnerable variables from the Markov blanket of  $T$  we need to create a modified graph  $\mathcal{G}^*$  by adding latent nodes to  $\mathcal{G}$  such that the new nodes and the existing non-vulnerable nodes  $d$ -separate the vulnerable variables from  $T$ . We term the process (shown in Algorithm 1) of generating  $\mathcal{G}^*$  node-splitting.

Consider intervening on treatment ( $A$ ) in Figure 2a. We assume variables are interventionally set to a “null” value (e.g.,  $A = \emptyset$  representing the absence of treatment or  $C = \emptyset$  representing the absence of the chronic condition).  $A$  is a vulnerable variable because it is not causally associated with  $T$  and it is a parent of a child of  $T$ , namely blood pressure ( $Y$ ). The structural equation of blood pressure is  $Y = f_y(T, X, C, A, u_y)$ . Intervening on  $A$  results in the latent variable  $Y(\emptyset) = f_y(T, X, C, A = \emptyset, u_y)$  representing the untreated blood pressure value. Unlike traditional SEM interventions, we retain the factual version of the variables we intervene on in the graph. We visualize this in Figure 2b by placing the resulting latent outcome variable  $Y(a)$  onto the causal graph as a parent of its factual version  $Y$ . The latent version subsumes the parents (in the original graph  $\mathcal{G}$ ) of its factual version that were not intervened upon (e.g.,  $X$  and  $C$ ). Thus, the new latent variable represents the value before the observed effects of the interventional variables occurred. We further assume that the factual outcome can be recovered as some invertible function of the counterfactual outcome and the observed value of the parent, subject to the same values of the exogenous noise variables:  $Y = g_y(Y(\emptyset), A, u_y)$ . As a result, the new graph  $\mathcal{G}^*$  is still a model of the observed data generating process.

The node-splitting operation naturally extends to simultaneous interventions on multiple variables. Figure 2c

shows the modified DAG when  $A$  and  $C$  are simultaneously intervened upon. Importantly, because we intervened on all vulnerable variables, this graph yields the conditional independence:  $T \perp\!\!\!\perp Y, A, C | Y (A = \emptyset, C = \emptyset), X$  in which the vulnerable variables  $A$  and  $C$  are now irrelevant for predicting  $T$  conditioned on the new Markov blanket which contains the latent variable. Thus, to  $d$ -separate the target from the vulnerable variables  $\mathbf{V}$ , we need to compute the latent versions of the shared children of  $T$  and  $\mathbf{V}$  in which we intervene and set  $\mathbf{V} = \emptyset$ .

### 3.2.3 Estimating Latent Variables

Under what conditions can we estimate the latent variables so that we  $d$ -separate the vulnerable variables from the target? First, we need adjusted versions of the assumptions required to estimate the distributions of potential outcomes, namely the previously mentioned conditional ignorability assumption. We assume we can accurately fit SEMs with respect to the available features in  $\mathcal{G}$ . In addition to no unobserved confounders, we also ideally have no unobserved exogenous variables. Enumerating more parents of a variable in its SEM allows us to better fit the equation and reduce the influence of  $u_i$ , the exogenous noise. Additionally, there are structural requirements for the models used to estimate the latent variables because of the underlying prediction problem, which results in an unobserved target variable for test units.

**No Interaction with the Target:** *In the structural equations, the effects of vulnerable variables  $\mathbf{V}$  on children  $Y$  shared with the  $T$  cannot depend on  $T$ .* If this were not the case, then estimating the latent outcome would require knowing the value of  $T$ , defeating the purpose of the prediction problem.

To compute the hypothetical latent variables, we first pick arbitrary forms for the generative structural equations of the children of  $T$  satisfying the invertibility and no interaction requirements and fit them to the factual outcomes data (e.g., using maximum likelihood estimation). Then, we can compute the latent outcome values by performing the interventions on the fitted structural equations. In our experiments in Section 5 we demonstrate how to do this for additive structural equations.

### 3.2.4 Non-Vulnerable Interventions

As we will explain in Section 4, the proposed method can result in a measurably simpler classification problem when the target is binary by decreasing variance in the children of  $T$  due to removing the effects of the vulnerable variables. A natural question is: are we limited to intervening only on the vulnerable variables?

We can intervene on any parent of a child of  $T$  (except

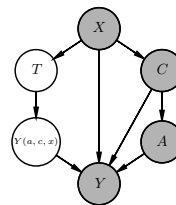


Figure 3: The modified DAG after intervening on  $C$ ,  $A$ , and  $X$ .

for  $T$  itself). However, unless the parent is a vulnerable variable, the parent will still be relevant for predicting  $T$ . This is because we cannot change the value of a parent of  $T$  in the structural equation for  $T$ , because in evaluation data  $T$  is unobserved. In the meningitis example of Figure 2, suppose we intervene on  $X$  (a parent of  $T$  and a parent of a child of  $T$ ) in addition to  $C$  and  $A$ . The resulting DAG after node-splitting is shown in Figure 3. Note that the only parent of  $Y(a, c, x)$  is  $T$  since it is the only parent of  $Y$  in the original DAG that is not intervened upon. Since the edge  $X \rightarrow T$  remains unchanged by the node-splitting operation,  $X$  is still a member of the Markov blanket of  $T$ . Thus, we would predict  $T$  using  $p(T|Y(\emptyset, \emptyset, \emptyset), X)$ . While not pictured, we can also intervene on children of  $T$  that are parents of other children of  $T$ . For example, if we added a variable  $Z$  to Figure 2a with edges  $T \rightarrow Z \rightarrow Y$ , we could intervene on  $Z$ . We would not, however, be able to remove  $Z$  from the Markov blanket of  $T$  because the edge  $T \rightarrow Z$  remains.

Even though these variables remain relevant for predicting  $T$  after intervening on them, there are still potential benefits to removing their effects on the children of  $T$  because it can measurably lower variance in these variables as we now discuss.

## 4 COMPLEXITY METRICS

Beyond guarding against vulnerabilities, what are other benefits of the proposed method? For binary prediction problems, the geometric complexity (on the basis of euclidean distance) of the class boundary of a dataset can decrease when using the latent variables instead of the factual outcome and vulnerable variables. This is similar to the work of Alaa and van der Schaar (2017) who use the smoothness of the treated and untreated response surfaces to quantify the difficulty of a causal inference problem. To measure classifier-independent geometric complexity we will use two types of metrics developed by Ho and Basu (2000, 2002): measures of overlap of individual features and measures of separability of classes.

For measuring feature overlap, we use the maximum

Fisher’s discriminant ratio of the features. For a single feature, this measures the spread of the means for each class ( $\mu_1$  and  $\mu_2$ ) relative to their variances ( $\sigma_1^2$  and  $\sigma_2^2$ ):  $\frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2}$ . Since the proposed method uses latent variables in which we have adjusted for the effects of the vulnerable variables (and any other variables we intervene on), this also removes sources of variance in the outcome. Thus, we expect the variances of each class to reduce resulting in increased feature separability and a corresponding increased Fisher’s discriminant ratio.

One measure of separability of classes is based off of a test (Friedman and Rafsky, 1979) for determining if two samples are from the same distribution. First, compute a minimum spanning tree (MST) that connects all the data points regardless of class. Then, the proportion of nodes which are connected to nodes of a different class is an approximate measure of the proportion of examples on the class boundary. Higher values of this proportion generally indicate a more complex boundary, and thus a more difficult classification problem.

However, this metric is only sensitive to which class neighbors are closer, and not the relative magnitudes of intra-class and interclass distances. Another measure of class separability is the ratio between the average intraclass nearest neighbor distance and the average interclass nearest neighbor distance. This measures the relative magnitudes of the dispersion within classes and the gap between classes. While we do not necessarily expect Counterfactual Normalization to increase the gap between classes, we do expect intraclass distances to decrease because the data units are transformed to have the same value of the vulnerable variables, reducing sources of variance (e.g., less variance in counterfactual untreated BP than in factual BP). While the MST metric may not decrease, we expect the intraclass-interclass distance ratio to decrease.

We can now state more specifically the benefits of the proposed method. Based on the assumptions in Section 3, we know that the vulnerable variables are not causally related to the target variable and that their effects on the outcome variables are not dependent on the target variable. These variables add variance to the prediction problem and, given that we can account for their effects on the children of  $T$ , are irrelevant to it. Thus, the proposed method can directly increase the signal-to-noise ratio of the classification problem. With respect to the geometric complexity of the class boundary, this manifests itself through reductions in the variance within a class, as we will demonstrate in our simulated experiments.

Table 1: Simulated Experiment Results

Method	Source AUROC	Target AUROC
Baseline	0.66	0.67
Baseline (vuln)	0.94	0.87
CFN	0.96	0.96
CFN (vuln)	0.97	0.95

## 5 EXPERIMENTS

The proposed method allows us to learn accurate prediction models that generalize across datasets. We first consider simulated experiments in which we know the true counterfactual outcomes to illustrate how the quality of predictions depends on the accuracy of the counterfactual estimates. Then we apply the method to a real medical classification task and demonstrate how we can use the proposed method to train a model that does not rely on vulnerable variables while retaining relevant information. In all experiments we train models using only source data and evaluate on test data from both the the source and target domains.

### 5.1 SIMULATED EXPERIMENTS

#### 5.1.1 Cross Hospital Transfer

We consider a simulated version of the medical screening problem in Figure 2(a), but remove  $X$  from the graph. We let  $A$  represent the time since treatment and simulate the exponentially decaying effects of the treatment as  $f(A) = 2 \exp(-0.08A)$  where the treatment policy depends on  $C$ . In this example,  $C$  and  $A$  are vulnerable variables.

We simulate data for patients from two hospitals. In the source hospital, we directly introduce a spurious association between  $C$  and  $T$ , which leads to an association between  $A$  and  $T$ . At this hospital shorter times since treatment are correlated with having the target condition. For this hospital the data are generated as follows:

$$\begin{aligned}
 T &\sim \text{Bernoulli}(0.4) \\
 C|T = 1 &\sim \text{Bernoulli}(0.8) \\
 C|T = 0 &\sim \text{Bernoulli}(0.3) \\
 A|C = 1 &\sim 24 * \text{Beta}(0.5, 2.1) \\
 A|C = 0 &\sim 24 * \text{Beta}(0.7, 0.2) \\
 Y &\sim \mathcal{N}(-0.5T + -0.3C + f(A), 0.2^2) \\
 f(A) &= 2 \exp(-0.08A)
 \end{aligned}$$

We remove the spurious correlation between  $T$  and  $C$  in the target hospital:  $p(C = 1|T) = p(C = 1) = 0.75$ . We also change the after-treatment measurement policy parameters to 1.7 and 1.1 such that  $p(A|C) = p(A)$ .

We assume that the  $T$  and  $C$  coefficients (in the struc-

Table 2: Simulated Classification Complexity Metrics

Method	Fisher's	Distance	MST
Baseline (vuln)	0.86	0.11	0.54
CFN	3.51	0.02	0.19

tural equation for  $Y$ ), the treatment response amplitude and timescale parameters, and noise scale parameter are unknown and need to be learned through maximum likelihood estimation, optimized using BFGS (Chong and Zak, 2013). We generate 800 patients from the source hospital, using 600 to learn the parameters and holding out 200 to evaluate performance on the source hospital. We evaluate cross hospital transfer on 600 patients generated from the second hospital.

As we identified in Section 2, the target latent variable is  $Y(A = \emptyset, C = \emptyset)$ : the patient’s blood pressure value if they had not been treated and did not have heart failure. Once the model parameters are learned, computing the latent variable is straightforward due to the additive structural equation of  $Y$ :  $Y_i(A = \emptyset, C = \emptyset) = Y_i - \hat{\beta}C_i - \hat{f}(s_i)^3$  which can be computed for every individual  $i$  at both hospitals without observing  $T$ . We consider counterfactual (CFN)  $p(T|Y(\emptyset, \emptyset))$  and baseline factual models  $p(T|Y)$  and corresponding versions with the vulnerable variables ( $p(T|Y(\emptyset, \emptyset), A, C)$  and  $p(T|Y, A, C)$ ) using logistic regression and measure predictive accuracy with the area under the Receiver Operating Characteristic curve (AUROC).

The results of evaluation on the patients from the source and target are shown in Table 1. The accuracy of the baseline model using the vulnerable variables does not transfer across hospitals. However, simply discarding the vulnerable features results in consistently poor performance at both hospitals. Instead, the counterfactually normalized models both transfer well while maintaining high performance. The latent features also capture most of the relevant information from the vulnerable variables, since adding the vulnerable variables results in marginal improvements at the source hospital.

The increased separability in the latent variables is shown in Figure 4, in which the factual blood pressure distributions (solid lines) contain significant overlap. However, once we normalize the blood pressures for treatment and chronic condition, the separability by class is increased. We also measure the increase through the classification complexity metrics in Table 2, computed using the source hospital training data. The feature with the maximum Fisher’s Discriminant Ratio in the baseline model is  $C$ , but this is much smaller than the ratio for the latent fea-

<sup>3</sup> denotes an estimated value.

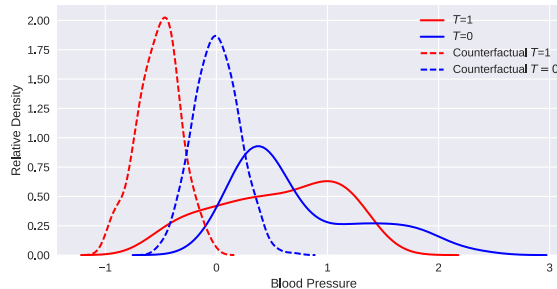


Figure 4: The distribution of factual (solid line) and estimated counterfactual (dashed line) blood pressures at the source hospital in the simulated experiment. It is easier to discriminate  $T$  from counterfactual BP than from observed BP due to decreased overlap in the distributions.

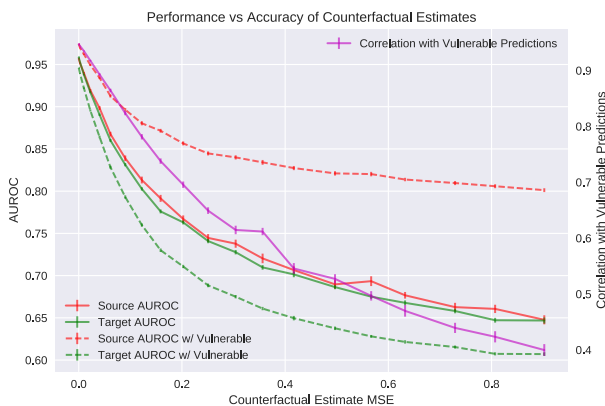


Figure 5: Performance as the accuracy of counterfactual estimates decreases. Secondary y-axis measures correlation between predictions using vulnerable variables and predictions without using them. The error bars denote the standard error of 50 runs.

ture. The large decrease in the MST metric indicates fewer examples lies on the class boundary in the normalized problem, and the decrease in intraclass-interclass is due to a combination of increased separability and reduced intraclass variance of the latent variables visible in the reduced spread of the distributions in Figure 4.

### 5.1.2 Accuracy of Counterfactual Estimates

In this experiment, we examine how the accuracy of counterfactual estimates affects the quality of model predictions. If the counterfactual estimates are accurate, then we expect the conditional independence of the vulnerable variables in the modified DAG to hold. We measure the degree of independence using the correlation between the predictions with ( $p(T|Y(\emptyset, \emptyset))$ ) and without

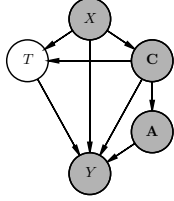


Figure 6: Real data experiment DAG of causal mechanisms. The outcome  $Y$  is INR and the target  $T$  is sepsis.

$(p(T|Y(\emptyset, \emptyset), C, A))$  using vulnerable variables.

We bias the true counterfactual values by adding normally distributed noise of increasing scale. Then, we train the counterfactual logistic regressions (with and without vulnerable variables) to predict  $T$  and evaluate the AUROC on the source and target hospital patients. We vary the standard deviation of the perturbations from 0.05 to 1 in increments of 0.05, repeating the process 50 times for each perturbation.

The results, shown in Figure 5, demonstrate what we expect: as the mean squared error (MSE) of the estimated latent variables increases, predictive performance on both populations worsens and the correlations of the predictions with and without vulnerable variables decreases. Since the model using vulnerable variables is biased by a spurious association that does not transfer (since the noisy adjustment is not capturing the relevant information), that model consistently underperforms at the target hospital. The counterfactual model without the vulnerable variables performs equally well at both hospitals, but the noise removes both the information captured by the adjustment and the information contained in  $Y$  itself.

## 5.2 REAL DATA: SEPSIS CLASSIFICATION

### 5.2.1 Problem and Data Description

We apply the proposed method to the task of detecting sepsis, a deadly response to infection that leads to organ failure. Early detection and intervention has been shown to result in improved mortality outcomes (Kumar et al., 2006) which has resulted in recent applications of machine learning to build predictive models for sepsis (e.g., Henry et al. (2015); Soleimani et al. (2017); Futoma et al. (2017)).

We consider a simple cross-sectional version of the sepsis detection task as follows using electronic health record (EHR) data from our institution’s hospital. Working with a domain expert, we determined the primary factors in the causal mechanism DAG (Figure 6) for the effects of sepsis on a single physiologic signal  $Y$ : the interna-

tional normalized ratio (INR), a measure of the clotting tendency of blood. The target variable  $T$  is whether or not the patient has sepsis due to hematologic dysfunction. We use chronic liver disease and sickle cell disease as conditions  $C$  affecting INR that are risk factors for sepsis (Goyette et al., 2004; Booth et al., 2010). We consider five types of relevant treatments  $A$ : anticoagulants, aspirin, nonsteroidal anti-inflammatory drugs (NSAIDs), plasma transfusions, and platelet transfusions, where  $A_{ij} = 1$  means patient  $i$  has received treatment  $j$  in the last 24 hours. Finally, we include a demographic risk factor, age  $X$ . For each patient, we take the last recorded measurements while only considering data up until the time sepsis is recorded in the EHR for patients with  $T = 1$ .

27,633 patients had at least one INR measurement, 388 of whom had sepsis due to hematologic dysfunction. We introduced spurious correlation through selection bias as follows. First, we took one third of the data as a sample from the original target population for evaluation. Second, we subsample the remaining data such that it only contains patients who are flagged in the EHR for having high INR. Third, we split the subsampled data into a random two thirds/one third train/test splits for training on biased data and evaluating on both the biased and unbiased data to measure transferability. We repeated the three steps 50 times. We normalize INR in all experiments.

### 5.2.2 Experimental Setup

We apply the proposed method by fitting an additive structural equation for  $Y$  using the Bayesian calibration form of Kennedy and O’Hagan (2001):

$$\begin{aligned}
 Y_i &= \beta_0 + \beta_1 T_i + \beta_2^T \mathbf{A}_i + \beta_3^T \mathbf{C}_i + \beta_4 X_i \\
 &\quad + \delta(T_i, \mathbf{A}_i, \mathbf{C}_i, X_i) + \varepsilon \\
 \delta(\cdot) &\sim \mathcal{GP}(0, \gamma^2 K_{rbf}) \\
 \varepsilon &\sim \mathcal{N}(0, \sigma^2)
 \end{aligned}$$

where  $\delta(\cdot)$  is a Gaussian process (GP) prior (with RBF kernel) on the *discrepancy function* since our linear regression model is likely misspecified.

Due to the selection bias in the training data, all patients have high INR making it difficult to calibrate the regression parameters. For this reason we place informative priors on  $\beta_1, \beta_2$ , and  $\beta_3$  using  $\mathcal{N}(1, 0.1)$  for features that increase INR (e.g.,  $T$  and anticoagulants) and  $\mathcal{N}(-1, 0.1)$  for features that decrease INR (e.g., sickle cell disease and plasma transfusions). For full specification of the other priors please consult the supplement. We compute point estimates for the parameters using MAP estimation and the FITC sparse GP (Snelson and Ghahramani, 2006) implementation in PyMC3 (Salvatier et al., 2016).

While the counterfactual  $Y(\mathbf{A} = \emptyset)$  is sufficient for  $d$ -

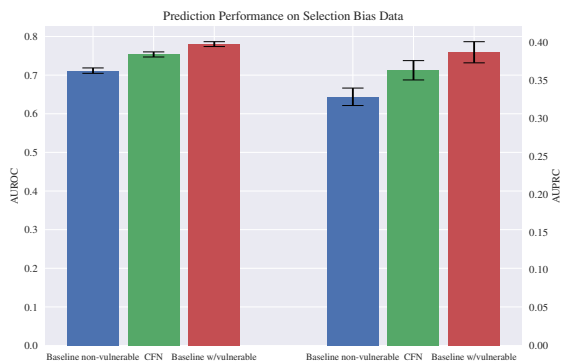


Figure 7: Results for models trained and tested on the selection biased data. In order the average AUROCs are 0.71, 0.75, and 0.78 and the average AUPRCs are 0.34, 0.36, and 0.39. Error bars denote 50 run 95% intervals.

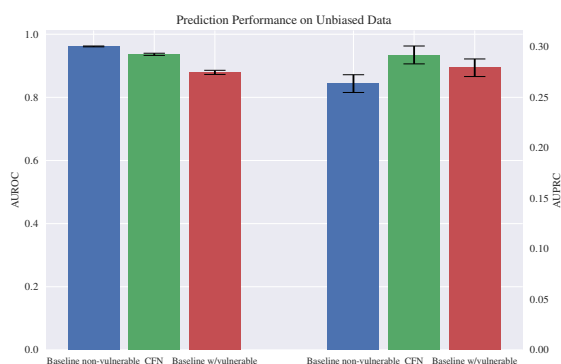


Figure 8: Results for models trained on biased data and tested on unbiased data. In order the average AUROCs are 0.96, 0.94, and 0.88 and the average AUPRCs are 0.26, 0.29, and 0.28. Error bars denote 50 run 95% intervals.

separating  $\mathbf{A}$  from  $T$  in Figure 6 after node splitting, we additionally normalize the effects of  $\mathbf{C}$  and  $X$ :

$$Y_i(\emptyset, \emptyset, \emptyset) = Y_i - \hat{\beta}_2^T \mathbf{A}_i - \hat{\beta}_3^T \mathbf{C}_i - \hat{\beta}_4 X_i \quad (1)$$

We consider three logistic regression models trained on the biased data for predicting  $T$ : a baseline that does not use the vulnerable variables  $p(T|\mathbf{C}, Y, X)$ , a baseline that uses the vulnerable variables  $p(T|\mathbf{A}, \mathbf{C}, Y, X)$ , and a counterfactually normalized model  $p(T|\mathbf{C}, Y(\emptyset, \emptyset, \emptyset), X)$ . We evaluate prediction accuracy on biased and unbiased data using AUROC and the area under the precision-recall curve (AUPRC).

### 5.2.3 Results

The resulting AUCs when predicting on biased data are shown in Figure 7. The counterfactually normalized model (CFN) outperforms the baseline model in which the

vulnerable variables are removed, but performs slightly worse than the normalized model which includes the vulnerable variables. This indicates that the latent variable estimates have captured some, but not all, of the relevant information in the vulnerable variables.

The results when predicting on unbiased data are shown in Figure 8. Since most of the examples in the unbiased data are negative (only 1.4% are positive), the AUPRC is a more interesting measurement because it is sensitive to false positives. As we expect, the baseline model without vulnerable variables has the lowest AUPRC because it has less statistically relevant information to use. Somewhat surprisingly, despite being trained on finite samples of biased data, the model with the vulnerable variables is able to learn a conditional distribution with the vulnerable variables that carries over to the unbiased population. Additionally, the counterfactual model without non-vulnerable variables has similar performance to the vulnerable model with respect to AUPRC indicating that it also captured a relationship of the vulnerable variables that generalizes. These results are encouraging because we were able to learn a counterfactually normalized model that transfers while clearly retaining non-spurious information about the vulnerable variables.

## 6 CONCLUSION

Using properties of DAGs encoding causal mechanisms, we have identified variables in prediction problems that are vulnerable to participating in spurious associations that can cause models to fail to generalize from training to deployment settings. As opposed to previous approaches which rely on unlabeled samples from the target distribution, we proposed a solution which allows us to identify latent variables that, when estimated, can allow a model to generalize by removing the vulnerable variables from the prediction problem. Because of their causal interpretations, we believe these latent variables are more intelligible for human experts than existing adjustment-based methods. For example, we think it is easier to reason about “the blood pressure if the patient had not been treated” than interaction features or kernel embeddings—we would like to test this in a future user study. In our experiments we demonstrated that we can successfully remove vulnerable variables at prediction time with minimal accuracy loss.

### Acknowledgements

The authors would like to thank Katie Henry for her help in developing the sepsis classification DAG.

## References

- Alaa, A. M. and van der Schaar, M. (2017). Bayesian nonparametric causal inference: Information rates and learning algorithms. *arXiv preprint arXiv:1712.08914*.
- Bareinboim, E. and Pearl, J. (2012). Controlling selection bias in causal inference. In *AISTATS*, pages 100–108.
- Bareinboim, E. and Pearl, J. (2013). Meta-transportability of causal effects: A formal approach. In *AISTATS*, pages 134–143.
- Bareinboim, E. and Tian, J. (2015). Recovering causal effects from selection bias. In *AAAI*, pages 3475–3481.
- Booth, C., Inusa, B., and Obaro, S. K. (2010). Infection in sickle cell disease: a review. *International Journal of Infectious Diseases*, 14(1):e2–e12.
- Campbell, D. T. and Stanley, J. C. (1963). Experimental and quasi-experimental designs for research. *Handbook of research on teaching*.
- Caruana, R., Lou, Y., Gehrke, J., Koch, P., Sturm, M., and Elhadad, N. (2015). Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *KDD*, pages 1721–1730. ACM.
- Chong, E. K. and Zak, S. H. (2013). *An introduction to optimization*, volume 76. John Wiley & Sons.
- Correa, J. D. and Bareinboim, E. (2017). Causal effect identification by adjustment under confounding and selection biases. In *AAAI*, pages 3740–3746.
- Correa, J. D., Tian, J., and Bareinboim, E. (2018). Generalized adjustment under confounding and selection biases. In *AAAI*.
- Dyagilev, K. and Saria, S. (2015). Learning (predictive) risk scores in the presence of censoring due to interventions. *Machine Learning*, 102(3):323–348. First Online 2015. Printed Version 2016.
- Friedman, J. H. and Rafsky, L. C. (1979). Multivariate generalizations of the wald-wolfowitz and smirnov two-sample tests. *The Annals of Statistics*, pages 697–717.
- Futoma, J., Hariharan, S., and Heller, K. (2017). Learning to detect sepsis with a multitask gaussian process rnn classifier. In *ICML*.
- Goyette, R. E., Key, N. S., and Ely, E. W. (2004). Hematologic changes in sepsis and their therapeutic implications. In *Seminars in respiratory and critical care medicine*, volume 25, pages 645–659. Thieme Medical Publishers, Inc., NY, USA.
- Heckman, J. J. (1977). Sample selection bias as a specification error (with an application to the estimation of labor supply functions).
- Henry, K. E., Hager, D. N., Pronovost, P. J., and Saria, S. (2015). A targeted real-time early warning score (trewscore) for septic shock. *Science translational medicine*, 7(299):299ra122–299ra122.
- Ho, T. K. and Basu, M. (2000). Measuring the complexity of classification problems. In *Pattern Recognition*, volume 2, pages 43–47. IEEE.
- Ho, T. K. and Basu, M. (2002). Complexity measures of supervised classification problems. *IEEE transactions on pattern analysis and machine intelligence*, 24(3):289–300.
- Huang, J., Gretton, A., Borgwardt, K. M., Schölkopf, B., and Smola, A. J. (2007). Correcting sample selection bias by unlabeled data. In *NIPS*, pages 601–608.
- Kennedy, M. C. and O’Hagan, A. (2001). Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B*, 63(3):425–464.
- Koller, D. and Friedman, N. (2009). *Probabilistic graphical models: principles and techniques*. MIT press.
- Kumar, A., Roberts, D., Wood, K. E., Light, B., Parrillo, J. E., Sharma, S., Suppes, R., Feinstein, D., Zanotti, S., Taiberg, L., et al. (2006). Duration of hypotension before initiation of effective antimicrobial therapy is the critical determinant of survival in human septic shock. *Critical care medicine*, 34(6):1589–1596.
- Neyman, J. (1923). On the application of probability theory to agricultural experiments. essay on principles. *Annals of Agricultural Sciences*, 10:1–51.
- Pearl, J. (2009). *Causality*. Cambridge university press.
- Pearl, J. (2015). Causes of effects and effects of causes. *Sociological Methods & Research*, 44(1):149–164.
- Pearl, J. and Bareinboim, E. (2011). Transportability of causal and statistical relations: a formal approach. In *AAAI*, pages 247–254. AAAI Press.
- Quionero-Candela, J., Sugiyama, M., Schwaighofer, A., and Lawrence, N. D. (2009). *Dataset shift in machine learning*. MIT Press.
- Richardson, T. S. and Robins, J. M. (2013). Single world intervention graphs (swigs): A unification of the counterfactual and graphical approaches to causality. *Center for the Statistics and the Social Sciences, University of Washington Series. Working Paper*, 128(30):2013.
- Rubin, D. B. (1974). Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology*, 66(5):688.
- Salvatier, J., Wiecki, T. V., and Fonnesbeck, C. (2016). Probabilistic programming in python using pymc3. *PeerJ Computer Science*, 2:e55.
- Schulam, P. and Saria, S. (2017). Reliable decision support using counterfactual models. In *NIPS*, pages 1696–1706.



- Shpitser, I. and Pearl, J. (2007). What counterfactuals can be tested. In *UAI*, pages 352–359. AUAI Press.
- Shpitser, I. and Pearl, J. (2008). Complete identification methods for the causal hierarchy. *Journal of Machine Learning Research*, 9(Sep):1941–1979.
- Snelson, E. and Ghahramani, Z. (2006). Sparse gaussian processes using pseudo-inputs. In *NIPS*, pages 1257–1264.
- Soleimani, H., Hensman, J., and Saria, S. (2017). Scalable joint models for reliable uncertainty-aware event prediction. *IEEE transactions on pattern analysis and machine intelligence*.
- Spirtes, P., Meek, C., and Richardson, T. (1995). Causal inference in the presence of latent variables and selection bias. In *UAI*, pages 499–506.
- Storkey, A. (2009). When training and test sets are different: characterizing learning transfer. *Dataset shift in machine learning*, pages 3–28.
- Swaminathan, A. and Joachims, T. (2015). Counterfactual risk minimization: Learning from logged bandit feedback. In *ICML*, pages 814–823.
- Zadrozny, B. (2004). Learning and evaluating classifiers under sample selection bias. In *ICML*, page 114. ACM.

---

# Decentralized Planning for Non-Dedicated Agent Teams with Submodular Rewards in Uncertain Environments

---

**Pritee Agrawal**

Singapore Management University  
Singapore 188065  
priteea.2013@phdis.smu.edu.sg

**Pradeep Varakantham**

Singapore Management University  
Singapore 188065  
pradeepv@smu.edu.sg

**William Yeoh**

Washington University in St. Louis  
St. Louis, MO 63130, USA  
wyeoh@wustl.edu

## Abstract

Decentralized planning under uncertainty for agent teams is a problem of interest in many domains including (but not limited to) disaster rescue, sensor networks and security patrolling. Decentralized MDPs, Dec-MDPs have traditionally been used to represent such decentralized planning under uncertainty problems. However, in many domains, agents may not be dedicated to the team for the entire time horizon. For instance, due to limited availability of resources, it is quite common for police personnel leaving patrolling teams to attend to accidents. Such non-dedication can arise due to the emergence of higher priority tasks or damage to existing agents. However, there is very limited literature dealing with handling of non-dedication in decentralized settings. To that end, we provide a general model to represent problems dealing with cooperative and decentralized planning for non-dedicated agent teams. We also provide two greedy approaches (an offline one and an offline-online one) that are able to deal with agents leaving the team in an effective and efficient way by exploiting the submodularity property. Finally, we demonstrate that our approaches are able to obtain more than 90% of optimal solution quality on benchmark problems from the literature.

## 1 INTRODUCTION

Decentralized planning for a team of agents is required in a wide variety of problems such as target tracking by a team of sensors [Nair *et al.*, 2005; Kumar and Zilberstein, 2011; Chapman and Varakantham, 2014], securing targets from unknown attackers using a team of defenders [Brown *et al.*, 2014; Shieh *et al.*, 2014; Varakantham

*et al.*, 2013], rescuing of victims by a team of robots during disaster [Melo and Veloso, 2011; Varakantham *et al.*, 2009, 2014; Velagapudi *et al.*, 2011] and analysing underwater samples using a team of underwater vehicles [Yin and Tambe, 2011], etc.

These domains have the following common characteristics: (a) A decentralized team of agents (sensors, ambulances, fire-trucks, etc.) that coordinate plans to achieve a goal; (b) There is transition uncertainty in planning problems of individual agents, either due to travelling on roads (due to traffic) or due to physical constraints (sensors, robots, etc.) (c) The agents are independent and collaborate through a global reward (save victims, prevent attacks, etc.); and most importantly (d) The individual agents have a chance of leaving the team at any time step to address a higher priority task. For example, in the case of patrolling, agents (e.g., coast guard boats, traffic police) can be forced to leave their assignment to attend to an accident or incident (e.g., incursion, smuggling, accident).

We are interested in application problems with the above mentioned characteristics and specifically teams in which agents are non-dedicated and may leave the team due to higher priority tasks or damage to agents. Non-dedication has been explored by Agrawal and Varakantham [2017] for centralized planning. Further, Shieh *et al.* [2014] considered non-dedicated teams in decentralized settings, but they provide an exhaustive offline approach that is not scalable. Our contributions differ in providing quick solutions by exploiting reward submodularity for decentralized planning such that remaining agents can reconfigure their policies to attend to tasks of leaving agents.

Submodularity has been exploited by Kumar and Zilberstein [2009]; Satsangi *et al.* [2015] for centralized planning model while Kumar *et al.* [2017] focus on decentralized planning in cooperative teams. Our contributions differ from this line of work in considering non dedicated agent teams with multiple agent exits from the team while still considering joint submodular reward functions for

decentralized planning. Another closely related thread of research is on adaptive submodularity [Golovin and Krause, 2011], where a sequence of decisions are taken by accounting for the observations of past decisions. Our work differs from this thread because we consider a multi-stage submodular problem (which introduces a partition matroid constraint) where at every decision epoch, we have a new submodular problem<sup>1</sup> with fewer number of agents.

To that end, we provide a general model to represent the class of problems dealing with a team of independently collaborating non-dedicated agents. A key contribution of our work lies in establishing connections between non-dedicated agent teams and submodularity. We show that with monotone submodular reward functions subject to the matroid constraint, greedy solutions computed at every decision epoch are still submodular with fewer number of agents and provide an a priori guarantee of at least 50% from the optimal and much better posterior guarantees. Another main contribution includes our two greedy approaches to efficiently deal with agent exits before the end of horizon. In our first approach, we exploit lazy greedy to obtain a unique offline policy for every agent irrespective of the agent exits from the team. The second approach is an offline-online approach where the offline phase creates a fixed number of joint policies to be used in the online phase. Finally, our experiments demonstrate the improved performance of our approaches on benchmark problems from literature.

## 2 BACKGROUND

### 2.1 Monotone Submodularity and Matroids

**Definition 1** Given a finite set,  $\Pi$ , a **submodular function** is a set function,  $g : 2^\Pi \rightarrow \mathbb{R}$ , where  $2^\Pi$  is the power set corresponding to  $\Pi$ . More importantly,  $\forall X, Y \subseteq \Pi$  with  $X \subseteq Y$  and for every  $i \in \Pi \setminus Y$ , we have:

$$g(X \cup i) - g(X) \geq g(Y \cup i) - g(Y)$$

A submodular function  $g$  is **monotone** if  $g(Y) \geq g(X)$  for  $X \subseteq Y$ .

Monotone submodular functions are interesting because maximizing a submodular function to pick a fixed number of elements (say  $k$ ) from the finite set ( $\Pi$ ) while difficult can be approximated efficiently with a strong quality guarantee. Specifically, a greedy algorithm that incrementally generates the solution set by maximizing marginal utility provides solutions that are at least 63% ( $1 - \frac{1}{e}$ ) of the optimal solution.

<sup>1</sup>This is unlike in adaptive submodularity, where there is one submodular problem with updated information on sensor state.

If we have a submodular function under a specific constraint on the finite set ( $\Pi$ ) and the elements that are picked, the constraint is specified using a partition matroid. In this paper, we are also interested in maximizing a submodular function, however, under a specific constraint on the finite set ( $\Pi$ ) and the elements that are picked. Specifically, the constraint is specified using a partition matroid. We provide the formal definitions below:

**Definition 2** For a finite ground set  $\Pi$ , let  $\mathcal{P}$  be a non-empty collection of subsets of  $\Pi$ . The system  $\Gamma = (\Pi, \mathcal{P})$  is a matroid if it satisfies the following two properties:

- *The hereditary property:*  $\mathcal{P}_1 \in \mathcal{P} \wedge \mathcal{P}_2 \subset \mathcal{P}_1 \implies \mathcal{P}_2 \in \mathcal{P}$ . In other words, all the subsets of  $\mathcal{P}_1$  must be in  $\mathcal{P}$ .
- *The exchange property:*  $\forall \mathcal{P}_1, \mathcal{P}_2 \in \mathcal{P} : |\mathcal{P}_1| < |\mathcal{P}_2| \implies \exists x \in \mathcal{P}_2 \setminus \mathcal{P}_1; \mathcal{P}_1 \cup x \in \mathcal{P}$ .

We are specifically interested in a ground set that is partitioned as  $\Pi = \Pi_1 \cup \Pi_2 \cup \dots \cup \Pi_k$ . The family of subsets,  $\mathcal{P} = \{P \subseteq \Pi : \forall i, |P \cap \Pi_i| \leq 1\}$  forms a matroid called a partition matroid. This family of subsets denotes that any solution can include at most one element from each ground set partition where the ground set partitions represent the policy space of each agent and exactly one policy must be picked for each agent.

### 2.2 Submodular TI-Dec-MDP

Submodular Transition Independent Decentralized Markov Decision Process (TI-Dec-MDP) model [Kumar *et al.*, 2017] is characterized by the tuple:  $\langle \mathcal{A}g, S, A, \{P_i\}_{i \in \mathcal{A}g}, R, H, \alpha \rangle$ , where

- $\mathcal{A}g$  is the set of agents.
- $S$  is the factored joint state space.  $S = S_1 \times S_2 \dots S_{|\mathcal{A}g|}$ , where  $S_i$  is the state space corresponding to each individual agent  $i$ . We can also have a global unaffected state feature  $S_u$ .
- $A$  is the joint action space.  $A = \times_{i \in \mathcal{A}g} A_i$ , where  $A_i$  is the action space corresponding to each agent  $i$ .
- $P_i$  is the individual agent transition function.  $P_i(s'_i | s_i, a_i)$  indicates the transition probability of moving from  $s_i$  to  $s'_i$  on taking action  $a_i$ .
- $R$  is the monotone submodular joint reward, with  $R(s, a)$  representing the reward for taking joint action  $a$  in joint state  $s$ . In security domains [Shieh *et al.*, 2014], reward is both monotonically increasing and submodular. It is defined as follows:

$$R(s, a) = \sum_{\tau} y_{\tau} \cdot f_{\tau}(\sigma(s, a, \tau)) \quad (1)$$

$y_{\tau}$  indicates the value of target  $\tau$  and hence is a non-

negative number.  $f_\tau(\cdot)$  is a monotone submodular function referred to as the effectiveness of patrolling a target  $\tau$ . Effectiveness of patrols at a target  $\tau$  depends on the number of agents patrolling the target.  $\sigma(s, a, \tau)$  counts the number of agents at target  $\tau$  if the current joint state is  $s$  and joint action is  $a$ . Let  $\epsilon$  ( $0 < \epsilon \leq 1$ ) represent the effectiveness of one agent visiting a target. Then, the effectiveness of  $\sigma$  agents visiting the same target  $\tau$  in the joint state  $s$  is given by the usual definition of  $f(\cdot)$  for effectiveness parameter  $\epsilon$  is  $f(\sigma) = 1 - (1 - \epsilon)^\sigma$ .

- $H$  is the time horizon and  $\alpha$  is the starting state distribution.

The goal is to obtain a joint policy  $\pi^* = \langle \pi_1, \pi_2, \dots \rangle$  (with one policy,  $\pi_i$  for each agent  $i$ ) that maximizes expected reward or value defined as follows:

$$V(\pi) = \sum_s \alpha(s) \cdot V^H(s, \pi) \quad (2)$$

$$V^t(s, \pi) = R\left(s, \langle \pi_1^t(s_1), \dots, \pi_{|\mathcal{A}g|}^t(s_{|\mathcal{A}g|}) \rangle\right) + \sum_{s'} \left[ \prod_{i \in \mathcal{A}g} P_i\left(s'_i | \pi_i^t(s_i), s_i\right) \right] \cdot V^{t-1}(s', \pi) \quad (3)$$

### 3 SUBMODULAR ND-TI-Dec-MDP

We extend Submodular TI-Dec-MDPs to Non-Dedicated TI-Dec-MDPs (ND-TI-Dec-MDPs) in order to model non-dedicated teams. The model is characterised by the following tuple:

$$\langle \mathcal{A}g, \{\Delta_i\}_{i \in \mathcal{A}g}, S, A, \{P_i\}_{i \in \mathcal{A}g}, R, H, \alpha \rangle$$

The main change to the Submodular TI-Dec-MDP is  $\Delta_i$ .  $\Delta_i$  is the vector of probabilities for agent  $i$  leaving the system at different times. Specifically,  $\Delta_i^t$  represents the probability of agent  $i$  leaving the team at time  $t$  and  $\sum_t \Delta_i^t = 1$ . We use the global state  $S_u$  to represent the dead state (i.e., the state that agents enter when they move out of the system). The individual agent transition function  $P_i^t(s'_i | s_i, a_i)$  is modified to  $P_i^t(s'_i | s_i, a_i, \Delta_i)$  and is described as following:

$$P_i^t(s'_i | s_i, a_i, \Delta_i) = P_i^t(s'_i | s_i, a_i) \cdot (1 - \Delta_i^t) \quad (4)$$

$$P_i^t(S_u | s_i, a_i, \Delta_i) = \Delta_i^t \quad (5)$$

If  $\Delta_i^t = 0$ , it implies that the agent transitions to the expected state according to its transition probability  $P_i^t(s'_i | s_i, a_i)$ . Otherwise, if  $\Delta_i^t \neq 0$ , the transitions depend on the agent's probability of staying in the system (i.e.,  $1 - \Delta_i^t$ ). Furthermore, an agent transitions to the dead state from any other state with probability  $\Delta_i^t$ . Note that once an agent transitions to the dead state  $S_u$ , it stays there

until the end of horizon (i.e.,  $P_i^t(S_u | S_u, a_i, \Delta_i) = 1$ ) irrespective of the action taken. The joint reward function  $R(s, a)$  however remains unchanged since the computation of reward only requires the count of agents present in the joint state  $s$ . In addition, there is no reward associated with agents present in  $S_u$  and we simply have  $R(S_u, a) = 0$ . The goal of submodular ND-TI-Dec-MDP is to obtain a joint policy  $\pi$  that maximizes the expected value  $V^t(s, \pi)$  over all agents with an additional constraint that the agents may leave the team.

#### 3.1 Properties of ND-TI-Dec-MDP

We now describe the important properties of ND-TI-Dec-MDPs with a joint reward function that is monotonically increasing and submodular. Let us first consider the case of a dedicated team where no agent leaves the system (represented as  $\Delta_i^H = 1$  for all agents). In this case, the state of the system is fixed (i.e., no agents leaving) and already known to the decision maker, and hence, the policy of every agent can be determined in advance. However, in a non-dedicated agent team, agents may leave the team midway requiring reconfiguration of the remaining agent policies. The timestep at which an agent leaves the team is referred to as observation timestep,  $t'$  and the set of agents leaving the system at  $t'$  represent the observation  $\psi$ . All the agents that have left until  $t'$  constitute the observation set  $\psi_{t'}$ . The joint policy for a ND-TI-Dec-MDP is a concatenated policy which is formally defined for one observation timestep as following.

**Definition 3** *Policy Concatenation:* Let  $\pi_{\psi_0}$  be the joint policy over all agents until the first observation at time  $t'$  and  $\pi_{\psi_{t'}}$  be the joint policy with observation set  $\psi_{t'}$ . The concatenated policy  $\hat{\pi}$  is represented as:

$$\hat{\pi} = [\pi_{\psi_0}]_{t=0}^{t < t'} + [\pi_{\psi_{t'}}]_{t=t'}^{t=H}$$

**Proposition 1** [Kumar et al., 2017]: For a TI-Dec-MDP,  $V^H(s, \pi)$  is monotonically increasing and submodular if the joint reward,  $R$  is monotonically increasing and submodular.

At  $t = 0$ , ND-TI-Dec-MDP is similar to TI-Dec-MDP and is solved for  $|\mathcal{A}g|$  agents and  $H$  timesteps. The value function,  $V^H(s, \pi)$  is a monotone and submodular being the case of dedicated agent team. Similarly, for every observation timestep  $t'$ , ND-TI-Dec-MDP is solved as a new TI-Dec-MDP problem with  $\mathcal{A}g \setminus \psi_{t'}$  agents and  $H - t'$  timesteps where  $\psi_{t'}$  represents the set of agents that have left until  $t'$ . The value function,  $V^{H-t'}(s, \pi)$  at  $t'$  is also monotonically increasing and submodular. Hence, for a single observation  $\psi_{t'}$ , the joint policy comprises of two components (as per definition 3) where the second

component is guaranteed to be submodular but not the first component. This is because submodularity of the value function  $V^H(\pi)$  holds for  $[t]_0^H$  but for ND-TI-Dec-MDP, we consider only timesteps  $[t]_0^{t'}$  for the first component. Hence, the value function  $V^H(\hat{\pi})$  for ND-TI-Dec-MDP is not guaranteed to be submodular for  $\hat{\pi}$ , however, it is submodular for every TI-Dec-MDP sub-problem.

The goal in ND-TI-Dec-MDPs is to maximize the expected value by obtaining a correct joint policy (i.e., exactly one policy per agent). Formally, the goal is to maximize  $V^H(\pi)$  for every individual TI-Dec-MDP problem given the partition matroid  $\Gamma = (\Pi, I)$  where  $I = \{X \subseteq \Pi : |X \cap \Pi_i| = 1\}$ . Intuitively, the partition matroid enforces that we can only have one policy for each agent.

**Proposition 2** [Fisher et al., 1978]: *Greedy algorithm for maximizing a monotone submodular function subject to a partition matroid yields solutions that are at least 50% of the optimal solution.*

For a non-dedicated agent team, the a priori bounds for every TI-Dec-MDP sub-problem at any  $t'$  is guaranteed to be at least 50% of optimal in the worst case. However, these bounds are quite loose since the solution provided by greedy is much better in most cases. Therefore, we compute online bounds by adding the marginal value of the best policy for every agent in the solution set to provide a tighter upper bound on the optimum. The online bound for a monotonically increasing and submodular value function is represented as below:

**Proposition 3** [Kumar et al., 2017]: *For any joint policy,  $\pi$ :*

$$V(\pi^*) \leq V(\pi) + \sum_{i \in \mathcal{A}g} \delta_i(\pi)$$

where  $\delta_i(\pi) = \max_{\pi_i \in \Pi_i} V(\pi \cup \pi_i) - V(\pi)$

Here,  $\pi^*$  is the optimal joint policy with optimal individual policies for every agent. For any joint policy  $\pi$ , we get an upper bound on the value of the optimal policy by adding the individual policies,  $\pi_i$  that yield best marginal values for each agent. In the context of ND-TI-Dec-MDP, at every observation timestep  $t'$ , we solve a new TI-Dec-MDP problem with  $\mathcal{A}g \setminus \psi_{t'}$  agents and  $H - t'$  timesteps where any policy  $\pi_{\psi_{t'}}$  provides an upper bound on the optimal policy  $\pi_{\psi_{t'}}^*$ . However, any concatenated policy  $\hat{\pi}$  is not guaranteed to provide an upper bound on the optimal concatenated policy  $\hat{\pi}^*$  since submodularity may not hold for  $\pi_{\psi_0}$ . We still compute the online bound for the concatenated policy as following.

$$V(\hat{\pi}^*) \leq \left[ V(\pi_{\psi_0}) + \sum_{i \in \mathcal{A}g} \delta_i(\pi_{\psi_0}) \right]_{t=0}^{t < t'} + \left[ V(\pi_{\psi_{t'}}) + \sum_{i \in \mathcal{A}g \setminus \psi_{t'}} \delta_i(\pi_{\psi_{t'}}) \right]_{t=t'}^{t=H} \quad (6)$$

---

**Algorithm 1** ND-GREEDY ( $\mathcal{A}g, S, A, P, R, H - t', \alpha, \psi_{t'}$ )

---

```

1:  $Z \leftarrow \emptyset$ 
2:  $\pi_i^* \leftarrow \emptyset, \forall i \in \mathcal{A}g \setminus \psi_{t'}$ 
3: repeat
4:   for all  $i \in \mathcal{A}g \setminus \{\psi_{t'} \cup Z\}$  do
5:      $\pi_i^* \leftarrow \max_{\pi_i} V_i(\pi_i, \alpha_i^{t'} | \pi_Z^*)$ 
6:    $\langle i^*, V_{i^*} \rangle \leftarrow \max_{i \in \mathcal{A}g \setminus \psi_{t'} \cup Z} V_i(\pi_i^*, \alpha_i^{t'} | \pi_Z^*)$ 
7:    $Z \leftarrow Z \cup \{i^*\}$ 
8: until  $\mathcal{A}g \setminus \{\psi_{t'} \cup Z\} = \emptyset$ 
9: return  $\{Z, \pi^* \leftarrow \{\pi_i^*\}_{i \in \mathcal{A}g \setminus \psi_{t'}}\}$ 

```

---

where  $\delta_i(\pi_{\psi_t}) = \max_{\pi_i \in \Pi_i} V(\pi_{\psi_t} \cup \pi_i) - V(\pi_{\psi_t})$ ,  $t \in \{0, t'\}$

The expression in the first square bracket bounds the value of the optimal concatenated policy  $V^H(\hat{\pi}^*)$  from  $t = 0$  to  $t \leq t'$  for the policy  $\pi_{\psi_0}$  (however, it is not a guaranteed online bound), while the second expression provides a guaranteed online bound on the value of the optimal concatenated policy from  $t \geq t'$  to  $t = H$ . For our experiments, we compute online bounds for ND-TI-Dec-MDP using Equation 6.

## 4 APPROACHES

In this section, we provide enhancements to the existing approaches in literature along with an offline and an offline-online approach for solving ND-TI-Dec-MDPs. We extend the existing lazy greedy algorithm for TI-Dec-MDPs to provide solutions for non-dedicated agent teams. We further provide a lazy greedy extension for the benchmark heuristics in non-dedicated teams [Agrawal and Varakantham, 2017] to provide bounds on the solution quality of ND-TI-Dec-MDPs.

### 4.1 Greedy and Lazy Greedy

For dedicated agent teams, greedy has been well explored in the context of Dec-MDPs [Shieh et al., 2014; Agrawal et al., 2016; Kumar et al., 2017] while for non-dedicated agent teams, it has been explored only in centralized settings [Agrawal and Varakantham, 2017]. Therefore, we extend the previous work by [Kumar et al., 2017] to provide a lazy greedy extension for non-dedicated teams in decentralized settings.

Algorithm 1 provides the pseudocode for a non-dedicated greedy algorithm that is solved at every observation timestep,  $t'$  where  $|\psi_{t'}|$  agents leave the team and  $H - t'$  timesteps are remaining. The algorithm is initially invoked at the starting timestep (i.e.,  $t = t' = 0$  and  $\psi_{t'=0} = \emptyset$ ) after which it is invoked only for timesteps where  $\psi_{t'} \neq \emptyset$ . ND-Greedy builds the solution set by incrementally adding a policy for every agent that has not

been assigned a policy. Initially, we start with an empty solution set  $Z$  (line 1). At every iteration, for each agent in the set of remaining agents,  $\mathcal{A}_g \setminus \psi_{t'}$  that has not been assigned a policy (line 4), we compute a policy with the highest marginal value given the current solution set (line 5) by constructing and solving an MDP (similar to the TI-Dec-MDP. Among those highest marginal value policies, we choose the one with the highest value and add it to the solution set (lines 6-7). This process is repeated until all  $\mathcal{A}_g \setminus \psi_{t'}$  agents have been assigned a policy to collectively provide the joint policy  $\pi^*$  (Every agent is assigned exactly one policy with the help of partition matroid constraint). Finally, the agents in  $Z$  are present in decreasing order of their marginal values. We refer this solution set  $Z$  as **selection order** of the agents.

ND-Greedy evaluates the marginal value for all the agents at every iteration, thereby affecting the scalability of the algorithm with increasing agents. Interestingly, submodularity of the value function  $V^H(\cdot)$  can be exploited to implement an accelerated version of classical greedy algorithm, otherwise known as Lazy Greedy [Minoux, 1978]. Instead of computing the marginal gain for all agents, lazy greedy allows a lazy evaluation of marginal benefits by storing the upper bounds  $\mu(i)$  on the marginal gain for all agents  $i \in \mathcal{A}_g$  sorted in descending order. This reduces the marginal gain computation as the submodularity of value/objective function guarantees that the marginal gain for an agent is always equal to or lower than the previous iteration. Intuitively, for each iteration, lazy greedy evaluates the agent on the top of the list, say  $i$ , and updates its upper bound,  $\mu(i)$ . If  $\mu(i) \geq \mu(i'), \forall i' \neq i$ , submodularity guarantees that agent  $i$  has the highest marginal gain. Therefore, lazy greedy leads to significant reduction in running times compared to the classical greedy.

**Why is the new policy recomputation needed:** The recomputation of a new joint policy at every observation timestep  $t'$  is important because the contribution of rewards by agents at every timestep may vary. This means that an agent may have higher rewards at earlier timesteps compared to later timesteps. In security games, if the remaining agents continue with their initial policies even after few agents leave, the coverage of important targets may be missed, making the system vulnerable to attacks. This creates an urgency for policy recomputation and therefore, we use lazy greedy to obtain a new selection order for agents by considering the reward contributions from the current timestep to the end of planning horizon. For example, let the selection order of agents at  $t = 0$  be  $[A_2(555), A_3(545), A_1(500), A_4(490)]$  with the reward values for agents specified alongside. Let  $a_1$  leave the system at  $t = 1$ . The total value for agents at  $t = 1$  could be  $[A_2(500), A_3(505), A_4(490)]$  on recomputation of reward for the remaining agents. This creates a change

in order of selection of the agents because the contribution at  $t = 0$  dominated the contribution over remaining timesteps for agents  $A_2$  and  $A_3$ . Hence, the change in order contributes to the change in marginal gain, and therefore, agents must rearrange their policies to adapt to the change in system.

## 4.2 Benchmarking Heuristics

The existing benchmark heuristics for non dedicated agent teams [Agrawal and Varakantham, 2017] are centralized approaches and incapable of computing joint policy and joint reward for the agent team. Hence, we provide a lazy greedy extension for the existing benchmarks to be able to solve ND-TI-Dec-MDPs.

**Ignore the leaving agent, Dec-ILA:** We start with a lazy greedy solution for the dedicated team and whenever agents leave the team, the remaining agents continue with the execution of their existing policies. However, due to the presence of joint reward for the system, we recompute the joint reward over the remaining agents whenever agents leave. This provides a good lower bound on solution quality that has to be achieved. For example, in security games domain, ignoring the targets covered by leaving defender agent is not the best choice since the leaving agent may be protecting a target of high importance. Hence, it is important for the remaining agents to modify their policies to provide an improved coverage to the targets that would become vulnerable to attacks. Similarly, in sensor domain, the sensors in the vicinity of a spoilt sensor should be able to change their policies and sense the target locations assigned to the spoilt sensor for better observation of any spatial phenomenon.

**Offline Optimal, Dec-OPT:** This heuristic assumes that the sample information (details of agents leaving the system) is received beforehand. Mixed integer program provides an optimal solution, but is not a suitable approach for finding the joint policy and the joint reward computation for a decentralized team of heterogeneous agents. Hence, we use lazy greedy for finding the agent policies where the agents are selected sequentially in the decreasing order of their values. Since the agents leaving the system have a shorter timespan compared to non-leaving agents, the marginal gain for such agents will be lowest. Hence, non-leaving agents are provided least preference in the selection process by greedy. Although not an exactly optimal approach, this heuristic provides a good upper bound on the solution quality.

**Online Revamp, Dec-O-Rev:** Similar to Dec-ILA, for this heuristic, we start with the initial lazy greedy solution until one or more agents leave the system. At the observation timestep  $t'$ , the problem is solved again for

---

**Algorithm 2** OFFLINE-GREEDY ( $\xi, \mathcal{A}g, W$ )

---

```
1:  $Z \leftarrow \emptyset, O \leftarrow \emptyset$ 
2:  $V_i \leftarrow 0, \forall i \in \mathcal{A}g$ 
3: for all  $\xi^k \in \xi$  do
4:    $V^k \leftarrow \text{Dec-OPT}(\mathcal{A}g, S, A, P, R, H, \alpha, \xi^k)$ 
5:    $V_i \leftarrow V_i + W^k \cdot V_i^k$ 
6: for all  $i \in \mathcal{A}g$  do
7:    $V_{i^*} \leftarrow \max_{i \in \mathcal{A}g \setminus O} V_i$ 
8:    $O \leftarrow O \cup i^*$ 
9: for all  $o \in O$  do
10:   $\langle \pi_o^*, V_o^* \rangle \leftarrow V_o(\pi_o^*, \alpha_o^0 | \pi_Z^*)$ 
11:   $Z \leftarrow Z \cup \{o\}$ 
12:  $\pi^* \leftarrow \{\pi_o^*\}_{o \in O}$ 
13: return  $\langle \pi^*, O \rangle$ 
```

---

the remaining agents  $\mathcal{A}g \setminus \psi_{t'}$  and remaining timesteps  $H - t'$ . The starting distribution of the remaining agents is recomputed at  $t'$  and is input to the lazy greedy algorithm along with the information of leaving agents,  $\psi_{t'}$ . The new joint policy obtained for the remaining agents is executed by the agent team until there is a change in the system (i.e., an agent leaves the system). Dec-O-Rev provides a good upper bound on the desired performance for our proposed approaches but suffers from some limitations. Although the running time reduction due to lazy greedy is significant compared to classical greedy, the total number of function evaluations with lazy greedy cannot be predicted beforehand to provide the exact running cost. This makes the complete recomputation of selection order at observation timesteps time consuming and difficult to be evaluated on the fly. Secondly, if there is a requirement of recomputation at every timestep  $t$ , revamp would become infeasible since at least  $\mathcal{A}g \setminus \psi_t$  rounds of sequential computation for agents will be required.

### 4.3 Offline-Greedy Approach

Offline-Greedy is a sampling-based approach that computes an offline selection order,  $O$  and a single joint policy  $\pi^*$  over multiple scenarios of agent availability. Since it is impossible to consider all the samples of agent availability on larger problems, we choose a smaller training set for the joint policy computation. The sample set is represented as  $\xi$  and has  $|K|$  samples. Due to repetition of samples, we assign frequency-specific weights  $W^k, \forall k \in K$  and select 20 best samples in decreasing order of weights. Every sample of agent availability,  $\xi^k$  is generated by sampling from a biased coin with probability  $p_i$  independently for every agent  $i$ . At every timestep  $t$ , the coin is tossed to decide whether agent  $i$  leaves or stays in the team depending on the value of associated probability in  $\Delta_i$ . Hence, for every sample  $\xi^k$ , we know the available horizon  $\xi^k(i)$  for every agent  $i$ .

Algorithm 2 provides the pseudocode for Offline-Greedy

---

**Algorithm 3** OFFLINE-ONLINE ( $\mathcal{A}g, N$ )

---

```
1: for all  $n \in N$  do
2:    $Z \leftarrow \emptyset$ 
3:    $\pi_i^n \leftarrow \emptyset, \forall i \in \mathcal{A}g$ 
4:   repeat
5:      $r_i \leftarrow \text{Random}(\mathcal{A}g \setminus Z)$ 
6:      $\pi_{r_i}^n \leftarrow V_{r_i}^n(\pi_{r_i}, \alpha_{r_i}^0 | \pi_Z^n)$ 
7:      $Z \leftarrow Z \cup \{r_i\}$ 
8:   until  $\mathcal{A}g \setminus Z = \emptyset$ 
9:    $\pi^n \leftarrow \{\pi_i^n\}_{i \in \mathcal{A}g}$ 
10:  $\Pi \leftarrow \Pi \cup \{\pi^n\}$ 
11: return  $\Pi$ 
```

---

with the training set  $\xi$ , the agent set  $\mathcal{A}g$  and the vector of frequency weights over all samples  $W$  as inputs. The agent selection set,  $Z$  and the selection order  $O$  are initialized as empty sets and the total value of every agent over all samples  $V_i$  is set to 0 (line 1-2). For every sample  $\xi^k$  in the training set, the available horizon of every agent is already known, and therefore, we use Dec-OPT heuristic to obtain the total value,  $V^k$  for every  $\xi^k \in \xi$  (line 4). The total value for every agent  $V_i$  is computed as the weighted sum of values over the sample set (i.e.,  $W^k \cdot V_i^k$ ) (line 5). The selection order  $O$  is computed by sorting the agents in decreasing order of their values  $V_i$  (line 6-9) such that the agents with higher probability of staying in the system are added before the agents with higher probability of leaving. For all the agents in the selection order, highest marginal value policy for an agent given the current solution set (line 10) is computed by constructing and solving an MDP (similar to TI-Dec-MDP) and the computed agent is then added to the solution set (line 11). Finally, we return the best selection order  $O$  and the offline joint policy  $\pi^*$  over all agents and all training samples.

For every test sample, the agents are assigned their individual policies from the offline joint policy  $\pi^*$ . However, irrespective of the observations obtained at different observation timesteps, the agents continue with their pre-assigned policies while the joint reward is recomputed for the remaining agents. This approach saves the online recomputation of policy at observation timesteps but with a compromise in the solution quality.

### 4.4 Offline-Online Approach

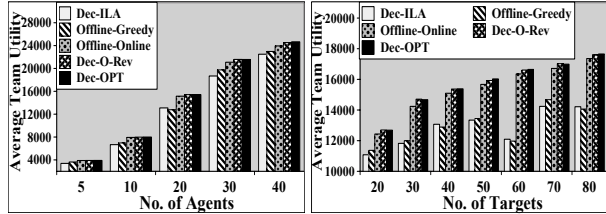
In this section, we present our Offline-Online algorithm which is a randomized greedy algorithm with an offline and an online phase. The offline phase focuses on the generation of multiple agent(s) selection orders to handle the different possibilities of scenarios, while the online phase focuses on choosing the best selection order for remaining agents depending on the current observation (availability of agents). We note that having multiple selection orders is better than having one fixed selection

order for all scenarios (as present in Offline-Greedy) because the total value of a selection order can change at different observation timesteps due to the dominance of rewards in previous timesteps (explained in details in section 4.1). We generate a fixed number of selection orders for the agent set since the total number of orderings possible with  $|\mathcal{A}g|$  agents is  $|\mathcal{A}g|!$  orders which is difficult to maintain with increasing agents. At every decision stage, we choose the best/closest selection order such that the position of the leaving agent is towards the end of the selection order, thereby, avoiding the recomputations. The time complexity of the offline phase is linear in the number of agents (or  $O(|\mathcal{A}g|)$ ) while it takes constant time for the online phase. The main difference with respect to lazy greedy (used in all the above approaches) is that instead of choosing the agent with highest marginal gain at every iteration, we randomly pick an agent and add it to the selection set. However, due to the joint reward computation and the presence of submodular rewards, the total utility always improves with addition of agents iteratively.

Algorithm 3 shows the offline phase of Offline-Online algorithm where the input to the algorithm is the agent count  $|\mathcal{A}g|$ , and the number of selection orders to be generated ( $N$ ). For computing every order  $n$ , we start with an empty agent selection set  $Z$  and add one agent at a time by randomly selecting agents from the set of remaining agents  $\mathcal{A}g \setminus Z$ . The policy and value of every agent is obtained by solving an MDP and is stored in  $\pi^n$ . Finally, we return  $\Pi$  that represents the set of policies for all the  $N$  selection orders.

The online phase of our algorithm does not require any computation and only reacts to a situation by choosing the best order from the set of offline orders for the remaining agents and providing a new policy for every agent from the observation timestep  $t'$  until the end of horizon. The selection criteria for choosing the best order for the defender team whenever any agent leaves the team depends on the number of exact matching and closest matching selection orders. For example, let us assume that there are 4 agents in the system  $\{a_1, a_2, a_3, a_4\}$  and the available set of selection order contains three orders,  $O_1 = \{3, 2, 1, 4\}$ ,  $O_2 = \{4, 2, 3, 1\}$  and  $O_3 = \{3, 2, 4, 1\}$  with total utility of  $\{200, 150, 100\}$  for the orders respectively. Let us consider two case studies:

- **Agent  $a_1$  leaves the system:** In this case,  $O_2$  and  $O_3$  are the best suitable orders since they require no re-evaluation but the order with highest utility is given preference and hence,  $O_2$  is chosen.
- **Agent  $a_3$  leaves the system:** In this case, none of the matches are exact and therefore, we find the closest match. We choose  $O_2$  to assign policies to the remain-



(a)  $\tau = 40, H = 20, \epsilon = 0.7$  (b)  $|\mathcal{A}g| = 20, H = 20, \epsilon = 0.7$

Figure 1: Quality Comparison w.r.t. (a) Agents and (b) Targets

ing agents since it requires minimal updates to agent policies. At the observation timestep, the previous policy of  $a_1$  is replaced by the existing policy of  $a_3$ , but after considering the change in state distribution of the agents since  $a_1$  and  $a_3$  are not guaranteed to be in the same state at the considered timestep. However, due to the replacement of agent policies,  $a_1$  would now become the third agent in the system, assuming the presence of two agents. Policy recomputation is not required because the offline joint policy (of every selection order) computes the  $V^t(s, \pi)$  values for all states at all intermediate timesteps (i.e., joint value after selection of every agent in the selection order). Furthermore, no reward recomputation is required since the joint reward considers only the count of agents (and not the identity of agents) at any state due to the monotone submodular reward structure for the joint reward.

In this manner, the online phase improves the value of solution roughly the same as Dec-O-Rev, but very quickly.

## 5 EXPERIMENTS

We evaluate<sup>2</sup> the performance of our greedy approaches and compare them with the benchmark approaches mentioned in section 4.2 on the security games domain provided by Shieh *et al.* [2014] and the sensor network domain provided by Kumar *et al.* [2017]. The performance is evaluated on the following metrics: (a) solution quality; (b) runtime; (c) quality of online bounds. We generate 1500 samples of agent availability (defenders in security domain and sensors in sensor domain) and divide it into training and testing sets of 1000 and 500 samples, respectively. To obtain a fair comparison over all approaches, we compare the solutions on the same test set.

### 5.1 Security Games Domain

In this domain, there are a set of targets (train stations) on the metro rail network which must be defended by a set of decentralized (yet cooperative) defenders in the presence

<sup>2</sup>All our optimization problems are run on CPLEX v12.7



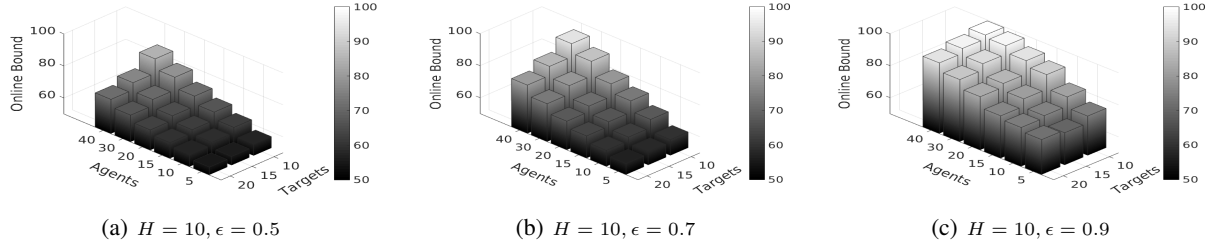


Figure 2: Comparison of Online Bound w.r.t. Effectiveness

of transition uncertainty. We constructed the metro rail graphs by connecting the stations together in lines of length 5 and then randomly adding  $|\tau|/2$  edges between targets, to resemble train systems in the real world with complex loops. The reward is a joint reward which is a function of the number of active defenders and the targets for a joint state  $s$  (see Section 2.2 for details). The test results were averaged over 15 randomly generated metro-based graph networks and the rewards were generated randomly in the range of  $[0,100]$ . We run the scenarios with a probability delay of .2 and a maximum of 5 agents (varies from 10% to 25% across scenarios) with an ability to leave the system, defined by probability vector  $\Delta$ . The defender agents are homogeneous (due to same reward and transition function) but differ from each other in their starting states (generated randomly for every agent) and their capability to leave the system.

**Solution Quality:** We compare different approaches with respect to average team utility in Figure 1(a) as the number of defenders  $|\mathcal{A}g|$  is increased. Specifically, we consider a metro network with targets  $\tau = 40$ , horizon  $H = 20$  and effectiveness parameter  $\epsilon = 0.7$ . Similarly, in Figure 1(b), we vary the targets,  $\tau$  for a fixed number of agents  $|\mathcal{A}g| = 20$ , horizon  $H = 10$  and effectiveness parameter  $\epsilon = 0.7$ . The key observations are summarized as following:

- (1) The average team utility increases with increasing defenders for a fixed number of targets and planning horizon due to the submodular reward structure. Similarly, the team utility increases with increasing targets due to increased number of choices for obtaining better rewards.
- (2) Dec-ILA provides low team utility solutions since the remaining agents continue with existing policies even after agents leave. This impacts the scope of improvement in rewards and is a cause of serious concern in security domain since it allows easy access to an adversary to plan an attack in unprotected areas.
- (3) Offline-Greedy provides similar or better solutions than Dec-ILA. We observe that with fewer agents and targets, and smaller planning horizon, Offline-Greedy performs almost similar to Dec-O-Rev as agent exits are given due importance during the offline policy design but

the performance degrades quickly with increasing count of agents and the planning horizon. In the worst case, the solution quality was seen to be even lower than Dec-ILA. (4) Offline-Online provides a steady performance, almost at par with the upper bound benchmarks (Dec-O-Rev and Dec-OPT) even with increasing problem sizes. Due to the random selection of agents at different observation times and the presence of submodular reward function, in the best case, Offline-Online could provide better team utility than Dec-O-Rev (uses lazy Greedy).

(5) Dec-OPT provides a good upper bound but is not always guaranteed to provide better utility compared to Dec-O-Rev due to the dominance of rewards in earlier timesteps explained in the section 4.1. However, on average, Dec-OPT provides slightly better solution quality compared to Dec-O-Rev after having the knowledge of samples before-hand.

**Solution Runtime:** With respect to runtime, we compare only online runtime since the offline runtimes do not matter. Due to decentralized planning of agents, individual agent planning time varies from 100 ms to 5000 ms from the smallest problem instance (20 targets and 10 timesteps) to the largest instance (40 targets and 20 timesteps). For Dec-O-Rev, due to the use of lazy greedy approach at every timestep of revamp, the revamp time varies from 15 seconds to 1700 seconds depending on the number of defenders and the problem size per defender. Further, there can be multiple revamps for one planning scenario making Dec-O-Rev infeasible for providing new policies quickly. However, our Offline-Online approach uses a proactive offline planning which reduces the online execution time to milliseconds, even in the worst case (although it requires offline training time). Similarly, the online runtime is minimal for Dec-ILA, Dec-OPT and Offline-Greedy.

**Online Bound Comparison:** For the online bound comparison, we use a consistent reward structure for every randomly generated metro network. For every metro network, we generate various scenarios of agents availabilities for different number of defenders and varying effectiveness of defenders. We compute the online bound for every scenario using Equation 6 and average the online

Grid-Size	Sensors,Targets Global States	$\epsilon$		
		.3	.5	.7
5 × 5	5, 1, 10	58.3	64.5	71.5
5 × 5	5, 2, 6*6	58.6	65	71.5
10 × 5	6, 3, 14*10*10	57.4	61.2	64.2
10 × 5	6, 4, 5*5*5*5	57.3	61.6	63.7
10 × 5	6, 5, 6*5*5*5*5	55.7	61.3	65.3
10 × 5	10, 3, 14*10*10	58.5	63.5	69
10 × 5	10, 4, 5*5*5*5	55.5	58.5	61.7
10 × 5	10, 5, 6*5*5*5*5	55.9	61.5	67.7
10 × 10	10, 4, 6*5*5*5	58.7	64.5	71.2
10 × 10	15, 4, 6*5*5*5	58.5	63.8	72.2
10 × 10	20, 4, 6*5*5*5	58.9	65.5	72.7
10 × 10	10, 5, 5*5*5*5*5	55.5	61.2	67.3

Table 1: Online Bound Comparison for Sensor Domain

bounds over all test samples and all randomly generated graphs. This leads to the inference that the online guarantees are significantly better than the a priori guarantees (of 50% from optimal), with the best case of at least 90% from optimal for different values of effectiveness parameter. Figure 2 compares the online (or posterior) quality guarantees obtained by Dec-O-Rev for different values of agents ( $A_g$ ), targets ( $\tau$ ) and effectiveness parameter ( $\epsilon$ ). It shows that the online guarantees improve with increasing agents and decreasing targets over varying effectiveness, with highest guarantee being reported for 10 targets and 40 agents. Further, with increasing effectiveness of agents, the optimal bound increases with highest quality guarantees (up to 99%) observed for  $\epsilon = 0.9$ . To avoid clutter, we do not plot the quality guarantees provided by policies generated using Offline-Online in the same graph. However, Offline-Online fared slightly lower than Dec-O-Rev in terms of guarantees and provided a guarantee that was 0.7% lower than Dec-O-Rev in the best case, while in the worst case, it was 2% lower than Dec-O-Rev.

## 5.2 Sensor Network Domain

We use the similar settings as Kumar *et al.* [2017] for this domain. The environment is modelled as a grid and a submodular reward function with n-ary interactions (any number of sensors can track a target) is used where the reward of tracking a target is dependent on the number of sensors tracking it. The sensors are randomly placed at junctions of cells on the grid and can track four target cells surrounding the sensor. However, due to wear and tear or due to unforeseen conditions, some sensors may get spoilt and the neighbouring sensors must track the targets of damaged sensors to maximize the reward. Therefore, reconfiguration of sensors after one or more sensors are spoilt is important. The targets move stochastically (according to some fixed distribution) in the grid and follow a path of fixed length for movement. The product of path

lengths of all available targets defines the total number of global states for the sensor domain.

**Online Bound Comparison:** Table 1 shows the online guarantees obtained for offline-online by varying the grid-size, number of sensors and their effectiveness, number of targets and the number of global states. We vary the effectiveness parameter from 0.3 to 0.7 and observe that the online bounds vary from 55% to 73% for Offline-Online, while the guarantees provided by Dec-O-Rev were 4% and 1.8% better than Offline-Online in the worst case and best case, respectively. An important observation is that with increasing targets, the number of global states increases exponentially, leading to memory issues. We note that 5 targets for a 10 × 10 grid with every target having a path-length of 5 was very difficult instance to solve with  $5^5$  or 3125 global states. However, increasing the number of sensors with a fixed number of targets was comparatively easier to solve since every sensor agent problem was solved independent of other agents due to the decentralized settings. With respect to runtime, the time taken by any sensor agent for individual planning varies from few milliseconds to 10 seconds with increasing number of targets and the global states. Due to lazy greedy evaluations for Dec-O-Rev, every revamp may take time ranging from less than a minute to 40 minutes depending on the complexity of the problem being solved. This makes the usage of Dec-O-Rev infeasible in online settings. Further, there can be multiple revamps for every scenario to worsen the situation. Similar to the security domain, the performance of Offline-Greedy is very similar to Dec-ILA while Dec-OPT provides results similar to Dec-O-Rev. More interestingly, our Offline-Online approach continues to perform gracefully with increasing number of sensors, targets and the grid size, while taking minimal time (in milliseconds) for solving the largest problem. Finally, we conclude from the experiments that Offline-Online is the best choice considering the trade-off of running time and compromise in solution quality.

## 6 CONCLUSION

In this work, we focussed on cooperative decentralized stochastic planning for non-dedicated agent teams. We provided a general model for decentralized non dedicated agent teams. Our offline greedy based approach provided good results in small instances while our Offline-Online approach provided the best results even in large instances in an effective manner. Finally, our extensive experiments on benchmark problems demonstrate that our Offline-Online approach provides the best solutions that are on par with benchmarks that provide an upper bound on the performance while taking negligible online runtime making it effective even for taking decisions at every step.

## References

- Pritee Agrawal and Pradeep Varakantham. Proactive and reactive coordination of non-dedicated agent teams operating in uncertain environments. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 28–34, 2017.
- Pritee Agrawal, Pradeep Varakantham, and William Yeoh. Scalable greedy algorithms for task/resource constrained multi-agent stochastic planning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.
- Matthew Brown, Sandhya Saisubramanian, Pradeep Varakantham, and Milind Tambe. STREETS: game-theoretic traffic patrolling with exploration and exploitation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 2966–2971, 2014.
- Archie Chapman and Pradeep Varakantham. Marginal contribution stochastic games for dynamic resource allocation. In *Proceedings of the International Conference on Principles and Practice of Multi-Agent Systems (PRIMA)*, pages 333–340, 2014.
- Marshall L Fisher, George L Nemhauser, and Laurence A Wolsey. An analysis of approximations for maximizing submodular set functions- ii. In *Polyhedral combinatorics*, pages 73–87. Springer, 1978.
- Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.
- Akshat Kumar and Shlomo Zilberstein. Event-detecting multi-agent MDPs: Complexity and constant-factor approximation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 201–207, 2009.
- Akshat Kumar and Shlomo Zilberstein. Message-passing algorithms for large structured decentralized POMDPs. In *Proceedings of the Tenth International Conference on Autonomous Agents and Multiagent Systems*, pages 1087–1088, Taipei, Taiwan, 2011.
- Rajiv Ranjan Kumar, Pradeep Varakantham, and Akshat Kumar. Decentralized planning in stochastic environments with submodular rewards. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 3021–3028, 2017.
- Francisco S Melo and Manuela Veloso. Decentralized mdps with sparse interactions. *Artificial Intelligence*, 175(11):1757–1789, 2011.
- Michel Minoux. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques*, pages 234–243. Springer, 1978.
- Ranjit Nair, Pradeep Varakantham, Milind Tambe, and Makoto Yokoo. Networked distributed pomdps: A synthesis of distributed constraint optimization and pomdps. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 5, pages 133–139, 2005.
- Yash Satsangi, Shimon Whiteson, Frans A Oliehoek, et al. Exploiting submodular value functions for faster dynamic sensor selection. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 3356–3363, 2015.
- Eric Anyung Shieh, Albert Xin Jiang, Amulya Yadav, Pradeep Varakantham, and Milind Tambe. Unleashing dec-mdps in security games: Enabling effective defender teamwork. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, pages 819–824, 2014.
- Pradeep Varakantham, Jun-Young Kwak, Matthew Taylor, Janusz Marecki, Paul Scerri, and Milind Tambe. Exploiting coordination locales in distributed POMDPs via social model shaping. In *Proceedings of the International Conference on Planning and Scheduling (ICAPS)*, pages 313–320, 2009.
- Pradeep Varakantham, Hoong Chuin Lau, and Zhi Yuan. Scalable randomized patrolling for securing rapid transit networks. In *Proceedings of the Conference on Innovative Applications of Artificial Intelligence Conference (IAAI)*, 2013.
- Pradeep Varakantham, Yossiri Adulyasak, and Patrick Jaillet. Decentralized stochastic planning with anonymity in interactions. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 2505–2512, 2014.
- Prasanna Velagapudi, Pradeep Varakantham, Paul Scerri, and Katia Sycara. Distributed model shaping for scaling to decentralized POMDPs with hundreds of agents. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 955–962, 2011.
- Zhengyu Yin and Milind Tambe. Continuous time planning for multiagent teams with temporal constraints. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 465, 2011.

---

# A Forest Mixture Bound for Block-Free Parallel Inference

---

Neal G. Lawton and Greg Ver Steeg and Aram Galstyan

Information Sciences Institute  
University of Southern California  
Marina del Rey, CA 90292

## Abstract

Coordinate ascent variational inference is an important algorithm for inference in probabilistic models, but it is slow because it updates only a single variable at a time. Block coordinate methods perform inference faster by updating blocks of variables in parallel. However, the speed and convergence of these algorithms depends on how the variables are partitioned into blocks. In this paper, we give a convergent parallel algorithm for inference in deep exponential families that doesn't require the variables to be partitioned into blocks. We achieve this by lower bounding the ELBO by a new objective we call the *forest mixture bound* (FM bound) that separates the inference problem for variables within a hidden layer. We apply this to the simple case when all random variables are Gaussian and show empirically that the algorithm converges faster for models that are inherently more forest-like.

## 1 INTRODUCTION

Inference in directed models like deep exponential families (DEF's) [Ranganath et al., 2015] is complicated by the “explaining away effect”: for a directed model with observed variables  $\mathbf{x} \in \mathbb{R}^n$  and latent variables  $\mathbf{y} \in \mathbb{R}^m$ , independent “causes”  $y_j$  become dependent given an observed “effect”  $x_i$ . To handle this, the coordinate ascent variational inference (CAVI) algorithm iteratively updates the variational distribution for a single latent variable  $y_j$  while holding the variational distribution for all other latent variables fixed [Blei et al., 2017].

Though the  $y_j$ 's are not conditionally independent given  $\mathbf{x}$  except in exceedingly simple models, in many cases the  $y_j$ 's are *nearly* conditionally independent. Is there a

way to perform parallel inference in such models, or do we have to resort to the serial coordinate algorithm?

Block methods provide one avenue for parallel inference. These algorithms work by first partitioning the latent variables into a collection of blocks, and then iteratively updating a variable from each block in parallel. However, the speed (as in MCMC methods [Terenin et al., 2015]) or convergence (as in Hogwild methods [Recht et al., 2011]) of the resulting algorithm will depend on how the variables are blocked, and finding a good choice of blocking for an arbitrary model can be difficult.

The main contribution of this paper is a novel lower bound on log-likelihood we call the forest mixture bound (FM bound) that separates the problem of inference for each variable in a hidden layer. This allows all the variables in a layer to be updated in parallel, without the use of blocks. We call the resulting parallel inference algorithm the forest mixture algorithm (FM algorithm).

We study in detail the case when all the random variables in the DEF are Gaussian. We then demonstrate on both synthetic and real-world data the proposed method achieves faster convergence compared to existing methods.

## 2 RELATED WORK

**Hogwild Block Methods** There are two types of block methods for inference. The first is Hogwild-type algorithms [Recht et al., 2011][Sa et al., 2016] [Wang and Banerjee, 2014] [Zhao et al., 2014]. After partitioning the variables into blocks, these algorithms iteratively choose a single variable from each block and update as in CAVI, but in parallel [Sa et al., 2016]. These algorithms are guaranteed to converge only in certain cases, e.g., when the blocks are conditionally independent [Johnson et al., 2013].

**Convergent Block Methods** Instead of making CAVI updates in parallel, block algorithms may achieve convergence by making small parallel updates [Sontag and Jaakkola, 2009]. For example, “exact” asynchronous Gibbs sampling randomly rejects each block update according to an MCMC rejection ratio [Terenin et al., 2015]. If the blocks are chosen poorly, the rejection rate will increase and the rate of convergence will decrease [Singh et al., 2017].

In either type of block method, the performance of the algorithm depends on how the variables are blocked. In a distributed computation setting, blocking is necessary since each worker can only store a fraction of all variables in local memory. In this case, the FM bound provides a method for updating variables within a block or worker in parallel, instead of updating only a single variable in each block at a time.

**Amortized Inference** Instead of treating inference as an inverse problem that has to be solved for each observation, VAE’s train inference network (encoder) so the cost of inference is amortized over many observations [Kingma and Welling, 2013]. Once the encoder is trained, inference for any observation can be performed quickly with a single pass through the inference network. Encoder-free methods like ours may still be useful in the case when we have a trained generative model (decoder) but no trained encoder and want to perform inference for only a few samples or, more likely, for when we want to improve the solution produced by the encoder at test time.

**Undirected Models** Besides directed models, there is a wide literature for fast inference in undirected models [Baqué et al., 2016] [Singh et al., 2010]. Note that inference in undirected models like Deep Restricted Boltzmann Machines [Salakhutdinov and Hinton, 2009] can already be parallelized: non-consecutive layers can be updated in parallel in red-black fashion. In fact, the same degree of parallelization can be achieved in a directed model using our technique. While there is also a wide literature on bounding the log-partition function of an *undirected* model [Wainwright et al., 2005], we derive the FM bound by lower bounding the log-partition function of a *directed* model. The technique we use may be applicable to undirected models, but that is not explored in this paper.

**Structure Learning** The FM bound we derive is closely related to an interesting family of models called *forest mixture models*. These models may be applicable to the problem of structure learning, where the task is to infer the graphical structure of the underlying model

from data [Chow and Liu, 1968]. However, in this paper we narrowly focus on the problem of inference in a *given* generative model, not on training a new one.

### 3 PRELIMINARIES

Vector-valued variables are written in bold. The component-wise product of two vectors  $\mathbf{u}$  and  $\mathbf{v}$  is denoted  $\mathbf{u} \odot \mathbf{v}$ . Unless stated otherwise, all expectations, including the variance  $\text{Var}[\cdot]$ , standard deviation  $\text{Std}[\cdot]$ , and conditional entropy  $H(\mathbf{y}|\mathbf{x})$ , are taken with respect to the variational distribution  $q(\mathbf{y}|\mathbf{x})$ , though we sometimes write this explicitly for emphasis.

An *exponential family* of distributions is a family of distributions of the form

$$p(x) = \exp\{g(x) + t(x) \cdot \eta - a(\eta)\} \quad (1)$$

Where  $g$  is the log-base measure,  $t$  are the sufficient statistics,  $\eta$  are the natural parameters, and  $a$  is the log-partition function. When  $\eta$  is a function of another random variable  $\mathbf{y}$ , e.g.,  $\eta = b + \mathbf{w} \cdot \mathbf{y}$ , we will sometimes write  $\eta = \eta(\mathbf{y})$  for emphasis.

We denote the Gaussian probability density function with mean  $\mu$  and variance  $\sigma^2$  as  $\mathcal{N}(\mu, \sigma^2)$ . When we write  $\log p(x) \propto f(x)$ , we mean  $\log p(x) = f(x) + \text{constant}$ .

#### 3.1 FOREST MIXTURE MODELS

Consider a general directed model with a single layer of observed variables  $\mathbf{x} \in \mathbb{R}^n$  and latent variables  $\mathbf{y} \in \mathbb{R}^m$ . The joint distribution  $p(\mathbf{x}, \mathbf{y})$  takes the form

$$p(\mathbf{x}, \mathbf{y}) = \left[ \prod_{j=1}^m p(y_j) \right] \left[ \prod_{i=1}^n p(x_i | \mathbf{y}) \right] \quad (2)$$

A directed model is a *forest model* if each  $x_i$  has exactly one parent in the model’s directed dependency graph; they are so-named because the resulting graphical model is a forest with one tree per latent variable  $y_j$ . These models are particularly simple because the  $y_j$ ’s are conditionally independent given  $\mathbf{x}$ . Let  $\mathbf{e}_i \in I^m$  be the one-hot vector indicating the parent of  $x_i$ , so  $e_{ij} = 1$  if and only if  $y_j$  is the parent of  $x_i$ . Then we can write

$$p(x_i | \mathbf{y}) = \prod_{j=1}^m p(x_i | y_j)^{e_{ij}} \quad (3)$$

Suppose we want to fit a forest model to data, but we don’t know which  $x_i$ ’s should be the children of which  $y_j$ ’s. One way to handle this uncertainty is to treat the  $\mathbf{e}_i$ ’s as independent latent random variable that have to

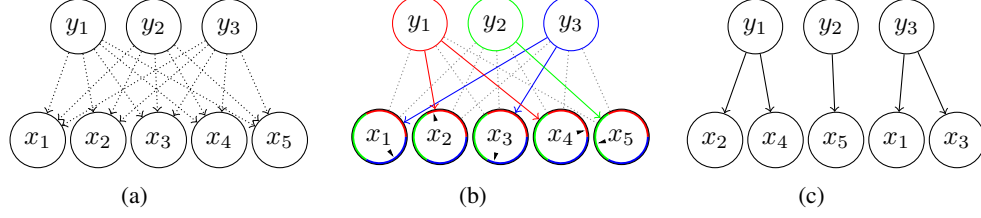


Figure 1: Visualization of sampling from a forest mixture model. (a) In a forest mixture model, the edges between  $\mathbf{x}$  and  $\mathbf{y}$  are unknown random variables. (b) To sample from the model, first the parent of each  $x_i$  is chosen independently at random according to  $p(e_i)$ . In this visualization, each  $p(e_i)$  is uniform over the latent variables. (c) After sampling a forest structure from  $p(e)$ ,  $\mathbf{x}$  and  $\mathbf{y}$  are sampled according to the resulting forest model.

be inferred, just like  $\mathbf{y}$ . To do this, we must first define a prior  $p(e_i)$  for each  $i$ . Given such a prior, the joint distribution over  $\mathbf{x}$ ,  $\mathbf{y}$ , and  $\mathbf{e} \equiv \{e_i\}_{i=1}^n$  is

$$p(\mathbf{x}, \mathbf{y}, \mathbf{e}) = \left[ \prod_{i=1}^n p(e_i) \right] \left[ \prod_{j=1}^m p(y_j) \right] \left[ \prod_{i=1}^n p(x_i | \mathbf{y}, e_i) \right] \quad (4)$$

The resulting model is a *forest mixture model* (FMM): to sample from this model, we first draw a random forest structure by sampling from the prior  $p(e)$ ; then,  $\mathbf{x}$  and  $\mathbf{y}$  are sampled from the selected forest model.

Though the  $y_j$ 's are no longer conditionally independent given  $\mathbf{x}$ , they are independent given  $\mathbf{x}$  and  $\mathbf{e}$ . Similarly, the  $e_i$ 's are conditionally independent given  $\mathbf{x}$  and  $\mathbf{y}$ . To see this, define  $\hat{p}(x_i | y_j) \equiv p(x_i | y_j, e_{ij} = 1)$ . Then the joint distribution can be written

$$p(\mathbf{x}, \mathbf{y}, \mathbf{e}) = \left[ \prod_{i=1}^n p(e_i) \right] \left[ \prod_{j=1}^m p(y_j) \right] \prod_{i=1}^n \prod_{j=1}^m \hat{p}(x_i | y_j)^{e_{ij}} \quad (5)$$

In the next section, we will use the mean-field variational ELBO for this model, which for a given variational distribution  $q(\mathbf{y}, \mathbf{e} | \mathbf{x})$  is

$$\begin{aligned} \log p(\mathbf{x}) &\geq \mathbb{E}[\log p(\mathbf{x} | \mathbf{y}, \mathbf{e})] - D_{KL}(q(\mathbf{y}, \mathbf{e} | \mathbf{x}) \| p(\mathbf{y}, \mathbf{e})) \\ &= \sum_{i=1}^n \sum_{j=1}^m \mathbb{E}[e_{ij}] \mathbb{E}[\log \hat{p}(x_i | y_j)] \\ &\quad - \sum_{j=1}^m D_{KL}(q(y_j | \mathbf{x}) \| p(y_j)) \\ &\quad - \sum_{i=1}^n D_{KL}(q(e_i | \mathbf{x}) \| p(e_i)) \end{aligned} \quad (6)$$

## 4 THE FOREST MIXTURE BOUND

For simplicity, we only consider shallow models in this section. The extension to deep models is straightforward (see Appendix C).

A single-layer *deep exponential family* (DEF) model is a directed model with a single layer of observed variables  $\mathbf{x} \in \mathbb{R}^n$  and hidden variables  $\mathbf{y} \in \mathbb{R}^m$ , where the conditional distribution is in an exponential family. The joint distribution  $p(\mathbf{x}, \mathbf{y})$  takes the form

$$p(\mathbf{x}, \mathbf{y}) = \left[ \prod_{j=1}^m p(y_j) \right] \left[ \prod_{i=1}^n p(x_i | \mathbf{y}) \right] \quad (7)$$

$$p(x_i | \mathbf{y}) = \exp \{g(x_i) + t(x_i)\eta_i(\mathbf{y}) - a(\eta_i(\mathbf{y}))\} \quad (8)$$

Suppose we are given an observation  $\mathbf{x}$  and want to approximately infer the posterior  $p(\mathbf{y} | \mathbf{x})$  by maximizing the variational ELBO, and suppose the  $y_j$ 's are conditionally independent given  $\mathbf{x}$ , so  $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x}) \prod_{j=1}^m p(y_j | \mathbf{x})$ . Then the mean-field variational ELBO is

$$\log p(\mathbf{x}) \geq \max_{q(\mathbf{y} | \mathbf{x})} \mathbb{E}[\log p(\mathbf{x}, \mathbf{y})] + H(\mathbf{y} | \mathbf{x}) \quad (9)$$

$$\equiv \max_{q(\mathbf{y} | \mathbf{x})} \sum_{j=1}^m \mathbb{E}[\log p(y_j | \mathbf{x})] + H(y_j | \mathbf{x}) \quad (10)$$

$$= \sum_{j=1}^m \max_{q(y_j | \mathbf{x})} \mathbb{E}[\log p(y_j | \mathbf{x})] + H(y_j | \mathbf{x}) \quad (11)$$

In the second line,  $\log p(\mathbf{x})$  is constant with respect to  $q(\mathbf{y} | \mathbf{x})$  and can be removed without changing the optimization problem. In this case, the ELBO separates into a sum of terms, each of which involves only a single  $y_j$ . This allows us to optimize the ELBO by updating each  $q(y_j | \mathbf{x})$  independently and in parallel.

In a general DEF, the  $y_j$ 's are not conditionally independent and the objective does not separate. However, without much manipulation, much of the ELBO does separate: for a single-layer DEF, the ELBO can be written

$$\log p(\mathbf{x}) \geq \mathbb{E}[\log p(\mathbf{x}, \mathbf{y})] + H(\mathbf{y}|\mathbf{x}) \quad (12)$$

$$= \sum_{i=1}^n \mathbb{E}[\log p(x_i|\mathbf{y})] + \sum_{j=1}^m \mathbb{E}[\log p(y_j)] + H(y_j|\mathbf{x}) \quad (13)$$

So only the  $\mathbb{E}[\log p(x_i|\mathbf{y})]$  terms aren't separable. However, if  $\eta_i$  is an affine function of  $\mathbf{y}$ , so  $\eta_i \equiv b_i + \mathbf{w}_i \cdot \mathbf{y}$  for some  $b_i \in \mathbb{R}$  and  $\mathbf{w}_i \in \mathbb{R}^m$ , then each  $\mathbb{E}[\log p(x_i|\mathbf{y})]$  term can be expanded

$$\mathbb{E}[\log p(x_i|\mathbf{y})] = g(x_i) + t(x_i)\mathbb{E}[\eta_i] - \mathbb{E}[a(\eta_i)] \quad (14)$$

$$= g(x_i) + t(x_i)(b_i + \mathbf{w}_i \cdot \mathbb{E}[\mathbf{y}]) - \mathbb{E}[a(b_i + \mathbf{w}_i \cdot \mathbf{y})] \quad (15)$$

From this we can see the only term left preventing the entire ELBO from separating is  $\mathbb{E}_{q(\mathbf{y}|\mathbf{x})}[-a(\eta_i(\mathbf{y}))]$ , a high-dimensional expectation of the non-linear log-partition function. The one thing we know about the log-partition function in exponential families is that it's convex. This suggests we use Jensen's inequality to bound  $\mathbb{E}[-a(\eta_i)]$ . Note that using Jensen's to bring the expectation over  $q$  inside  $a$  gives an inequality in the wrong direction because  $-a(\eta_i)$  is concave; to get a lower bound, we need to pull an expectation out from the inside of  $a$ . The derivation of the ELBO gives a hint on how to do this: recall

$$\log p(x) = \log \int p(x, y) dy \quad (16)$$

$$= \log \int \frac{q(y|x)}{q(y|x)} p(x, y) dy \quad (17)$$

$$= \log \mathbb{E}_{q(y|x)} \left[ \frac{p(x, y)}{q(y|x)} \right] \quad (18)$$

$$\geq \mathbb{E}_{q(y|x)} \left[ \log \frac{p(x, y)}{q(y|x)} \right] \quad (19)$$

In the same way, we will introduce a variational or auxiliary distribution inside the concave function  $-a(\eta)$ , then use Jensen's to pull it out. For each  $i$ , introduce an auxiliary discrete distribution over  $m$  categories  $\varepsilon_i \in \Delta^{m-1}$ , so

$$\sum_{j=1}^m \varepsilon_{ij} = 1 \quad \varepsilon_{ij} \geq 0 \quad \forall j \in [m] \quad (20)$$

Injecting this inside the log-partition function gives

$$\mathbb{E}[-a(b_i + \mathbf{w}_i \cdot \mathbf{y})] = \mathbb{E} \left[ -a \left( b_i + \sum_{j=1}^m \varepsilon_{ij} \frac{w_{ij} y_j}{\varepsilon_{ij}} \right) \right] \quad (21)$$

To use Jensen's inequality, we first need to bring  $b_i$  inside the sum, which we can do using  $b_i = \sum_{j=1}^m \varepsilon_{ij} b_i$ . This partitions the bias  $b_i$  into  $m$  parts according to  $\varepsilon_i$ . However, to get a sufficiently tight bound, we'll need to consider more general splittings: introduce another set of auxiliary parameters  $\hat{b}_i \in \mathbb{R}^m$  with the constraint  $b_i = \sum_{j=1}^m \varepsilon_{ij} \hat{b}_{ij}$ . Then

$$\begin{aligned} \mathbb{E}[-a(b_i + \mathbf{w}_i \cdot \mathbf{y})] &= \mathbb{E} \left[ -a \left( \sum_{j=1}^m \varepsilon_{ij} \left( \hat{b}_{ij} + \frac{w_{ij} y_j}{\varepsilon_{ij}} \right) \right) \right] \\ &\geq \sum_{j=1}^m \varepsilon_{ij} \mathbb{E} \left[ -a \left( \hat{b}_{ij} + \frac{w_{ij} y_j}{\varepsilon_{ij}} \right) \right] \end{aligned} \quad (22)$$

Bounding this term for each  $i$  separates the entire ELBO into a sum of terms, each of which involves only a single  $y_j$ . Plugging this in directly to get a final bound on log-likelihood results in an unwieldy expression, so first we will introduce new notation to simplify the bound.

#### 4.1 CONNECTION WITH FMM

To demonstrate the relation of the above bound and forest mixture models, let us define

$$\hat{\eta}_{ij} \equiv \hat{b}_{ij} + \frac{w_{ij} y_j}{\varepsilon_{ij}} \quad (23)$$

$$\hat{p}(x_i|y_j) \equiv \exp \{g(x_i) + t(x_i)\hat{\eta}_{ij} - a(\hat{\eta}_{ij})\} \quad (24)$$

Then  $\eta_i = \sum_{j=1}^m \varepsilon_{ij} \hat{\eta}_{ij}$  and the bound can be rewritten as follows:

$$\mathbb{E}[-a(\eta_i)] \geq \sum_{j=1}^m \varepsilon_{ij} \mathbb{E}[-a(\hat{\eta}_{ij})] \quad (25)$$

This expression can be used to impose bounds on each  $\mathbb{E}[\log p(x_i|\mathbf{y})]$ :

$$\mathbb{E}[\log p(x_i|\mathbf{y})] = g(x_i) + t(x_i)\mathbb{E}[\eta_i] - \mathbb{E}[a(\eta_i)] \quad (26)$$

$$\geq g(x_i) + t(x_i)\mathbb{E}[\eta_i] - \sum_{j=1}^m \varepsilon_{ij} \mathbb{E}[a(\hat{\eta}_{ij})] \quad (27)$$

$$= \sum_{j=1}^m \varepsilon_{ij} (g(x_i) + t(x_i)\mathbb{E}[\hat{\eta}_{ij}] - \mathbb{E}[a(\hat{\eta}_{ij})]) \quad (28)$$

$$= \sum_{j=1}^m \varepsilon_{ij} \mathbb{E}[\log \hat{p}(x_i|y_j)] \quad (29)$$

**input** : An observation  $\mathbf{x} \in \mathbb{R}^n$  and model parameters

$W \in \mathbb{R}^{n \times m}$ ,  $\mathbf{b} \in \mathbb{R}^n$ ,  $\sigma_y^2 \in \mathbb{R}$  and  $\sigma_x^2 \in \mathbb{R}$ .

**output**: The mean-field variational distribution

$$q(\mathbf{y}|\mathbf{x}) \equiv \prod_{j=1}^m q(y_j|\mathbf{x})$$

initialize  $(\mu_0)_j$  and  $(\sigma_0)_j^2$  for each  $j \in [m]$

**for**  $t = 0$  **to**  $T - 1$  **do**

**for**  $i = 1$  **to**  $n$  **do**

**for**  $j = 1$  **to**  $m$  **do**

$$(\varepsilon_t)_{ij} = \frac{|w_{ij}|(\sigma_t)_j}{\sum_{j'=1}^m |w_{ij'}|(\sigma_t)_{j'}^2}$$

$$(\hat{\mathbf{b}}_t)_{ij} = (b_i + \sum_{j=1}^m w_{ij}(\mu_t)_j) - \frac{w_{ij}(\mu_t)_j}{(\varepsilon_t)_{ij}}$$

**end**

**end**

**for**  $j = 1$  **to**  $m$  **do**

$$(\mu_{t+1})_j \equiv \frac{\sum_{i=1}^n w_{ij}(x_i - (\hat{\mathbf{b}}_t)_{ij})}{\frac{\sigma_x^2}{\sigma_y^2} + \sum_{i=1}^n \frac{w_{ij}^2}{(\varepsilon_t)_{ij}}}$$

$$(\sigma_{t+1})_j^2 \equiv \frac{1}{\frac{1}{\sigma_y^2} + \frac{1}{\sigma_x^2} \sum_{i=1}^n \frac{w_{ij}^2}{(\varepsilon_t)_{ij}}}$$

**end**

**end**

**return**  $q(y_j|\mathbf{x}) = \mathcal{N}((\mu_T)_j, (\sigma_T)_j^2)$  for  $j \in [m]$

**Algorithm 1**: The FM algorithm in the Gaussian case.

Finally, plugging the above expression into the ELBO gives

$$\begin{aligned} \log p(x) &\geq \mathbb{E}[\log p(\mathbf{x}, \mathbf{y})] + H(\mathbf{y}|\mathbf{x}) \\ &\geq \sum_{i=1}^n \sum_{j=1}^m \varepsilon_{ij} \mathbb{E}[\log \hat{p}(x_i|y_j)] \\ &\quad - \sum_{j=1}^m D_{KL}(q(\mathbf{y}|\mathbf{x}) \| p(\mathbf{y})) \end{aligned} \quad (30)$$

Comparing (30) with (6) confirms that this bound is *identical* to the ELBO of a forest mixture model with the same  $\hat{p}(x_i, y_j)$  and  $q(y_j|\mathbf{x})$ , with  $q(e_{ij} = 1|\mathbf{x}) = \varepsilon_{ij}$  (so that  $\mathbb{E}[e_{ij}] = \varepsilon_{ij}$ ) and  $p(e_i) = q(e_i|\mathbf{x})$  (so that the second  $KL$  term of the FMM ELBO is zero and disappears entirely). For this reason, we call this bound the *forest mixture bound* (FM bound). Note this bounds the DEF ELBO by the ELBO of each FMM in a large family of FMM's parameterized by  $\varepsilon \equiv \{\varepsilon_i\}_{i=1}^n$  and  $\hat{\mathbf{b}} \equiv \{\hat{\mathbf{b}}_i\}_{i=1}^n$ .

## 5 ALGORITHM

To optimize the FM bound, we propose an alternating maximization algorithm: in the first step, update all  $q(y_j|\mathbf{x})$  in parallel while holding all  $\varepsilon_{ij}$  and  $\hat{\mathbf{b}}_{ij}$  fixed; in the second step, update all  $\varepsilon_{ij}$  and  $\hat{\mathbf{b}}_{ij}$  in parallel while holding all  $q(y_j|\mathbf{x})$  fixed. In this section, we will derive the optimal updates for  $q(y_j|\mathbf{x})$ ,  $\varepsilon_{ij}$ , and  $\hat{\mathbf{b}}_{ij}$  in the case when each  $x_i$  and  $y_j$  are Gaussian with known variance:

$$p(y_j) = \mathcal{N}(0, \sigma_y^2) \quad p(x_i|\mathbf{y}) = \mathcal{N}(\eta_i(\mathbf{y}), \sigma_x^2) \quad (31)$$

We will derive the updates for the auxiliary parameters first since this will help simplify the update for the variational distribution later.

### 5.1 AUXILIARY PARAMETER UPDATES

Maximizing the FM bound over  $\varepsilon$  and  $\hat{\mathbf{b}}$  is equivalent to maximizing  $\mathcal{L}_i \equiv \sum_{j=1}^m \varepsilon_{ij} \mathbb{E}[-a(\hat{\eta}_{ij})]$  over  $\varepsilon_i$  and  $\hat{\mathbf{b}}_i$  for each  $i$ , since these are the only terms in the FM bound that depend on  $\varepsilon$  and  $\hat{\mathbf{b}}$ . In the Gaussian case,  $-a(\hat{\eta}_{ij}) = -\frac{1}{2\sigma_x^2} \hat{\eta}_{ij}^2$  and

$$\mathcal{L}_i = \sum_{j=1}^m \varepsilon_{ij} \mathbb{E} \left[ -\frac{1}{2\sigma_x^2} \hat{\eta}_{ij}^2 \right] \quad (32)$$

$$= -\frac{1}{2\sigma_x^2} \sum_{j=1}^m \varepsilon_{ij} \left( \text{Var}[\hat{\eta}_{ij}] + \mathbb{E}[\hat{\eta}_{ij}]^2 \right) \quad (33)$$

$$= -\frac{1}{2\sigma_x^2} \sum_{j=1}^m \frac{w_{ij}^2 \text{Var}[y_j]}{\varepsilon_{ij}} + \varepsilon_{ij} \left( \hat{\mathbf{b}}_{ij} + \frac{w_{ij} \mathbb{E}[y_j]}{\varepsilon_{ij}} \right)^2 \quad (34)$$

**Theorem 1** Holding  $q(y_j|\mathbf{x})$  constant, the choice of  $\hat{\mathbf{b}}_i$  and  $\varepsilon_i$  that maximizes  $\mathcal{L}_i$  is  $\hat{\mathbf{b}}_i = \hat{\mathbf{b}}_i^*$  and  $\varepsilon_i = \varepsilon_i^*$ , where

$$\hat{\mathbf{b}}_{ij}^* = \mathbb{E}[\eta_i] - \frac{w_{ij} \mathbb{E}[y_j]}{\varepsilon_{ij}^*} \quad \varepsilon_{ij}^* = \frac{|w_{ij}| \text{Std}[y_j]}{\sum_{j'=1}^m |w_{ij'}| \text{Std}[y_{j'}]} \quad (35)$$

For a proof, see Appendix A. Note that these computations can be parallelized across  $i$  and  $j$ .

### 5.2 VARIATIONAL UPDATES

Holding the auxiliary parameters fixed, each variational distribution  $q(y_j|\mathbf{x})$  can be updated in parallel:

**Theorem 2** For a fixed  $\varepsilon$  and  $\hat{\mathbf{b}}$ , the choice for the next variational distribution  $q_{t+1}(y_j|\mathbf{x})$  that maximizes the FM bound is  $q_{t+1}(y_j|\mathbf{x}) = \mathcal{N}((\mu_{t+1}^*)_j, (\sigma_{t+1}^*)_j^2)$ , where

$$(\mu_{t+1}^*)_j \equiv \frac{(\mathbf{x} - \mathbb{E}_{q_t}[\boldsymbol{\eta}]) \cdot \mathbf{w}_j + \mathbb{E}_{q_t}[y_j] \sum_{i=1}^n \frac{w_{ij}^2}{\varepsilon_{ij}}}{\frac{\sigma_x^2}{\sigma_y^2} + \sum_{i=1}^n \frac{w_{ij}^2}{\varepsilon_{ij}}} \quad (36)$$

$$(\sigma_{t+1}^*)_j^2 \equiv \frac{1}{\frac{1}{\sigma_y^2} + \frac{1}{\sigma_x^2} \sum_{i=1}^n \frac{w_{ij}^2}{\varepsilon_{ij}}} \quad (37)$$

For a proof, see Appendix B.



## 6 DISCUSSION

**Tightness** We derived the FM bound by using Jensen’s inequality to lower bound the ELBO. For a given variational distribution  $q$ , the gap between the two bounds is

$$\text{GAP} \equiv \sum_{i=1}^n \mathbb{E}[-a(\eta_i)] - \sum_{i=1}^n \sum_{j=1}^m \varepsilon_{ij} \mathbb{E}[-a(\hat{\eta}_{ij})] \quad (38)$$

In the Gaussian case, for an optimal choice of auxiliary parameters (see Appendix A),

$$\sum_{j=1}^m \varepsilon_{ij} \mathbb{E}[-a(\hat{\eta}_{ij})] = -\frac{1}{2\sigma_x^2} \|\mathbf{w}_i \odot \text{Std}[\mathbf{y}]\|_1^2 - \frac{1}{2\sigma_x^2} \mathbb{E}[\eta_i]^2 \quad (39)$$

$$\mathbb{E}[-a(\eta_i)] = -\frac{1}{2\sigma_x^2} \|\mathbf{w}_i \odot \text{Std}[\mathbf{y}]\|_2^2 - \frac{1}{2\sigma_x^2} \mathbb{E}[\eta_i]^2 \quad (40)$$

$$\text{GAP} = \frac{1}{2\sigma_x^2} \sum_{i=1}^n \|\mathbf{w}_i \odot \text{Std}[\mathbf{y}]\|_1^2 - \|\mathbf{w}_i \odot \text{Std}[\mathbf{y}]\|_2^2 \quad (41)$$

Since  $\sum_{i=1}^n \|\mathbf{w}_i\|_1^2 \geq \sum_{i=1}^n \|\mathbf{w}_i\|_2^2$ , the FM bound imposes a stronger regularization on the variance of the variational distribution compared to the variational ELBO. For this reason, the variational distribution  $q$  that maximizes the FM bound generally has a smaller variance compared to the variational distribution that maximizes the ELBO.

The FM bound tightly bounds the ELBO when  $p$  is a forest model, so that  $w_{ij}$  has exactly one non-zero element in the component  $j(i)$  corresponding to the parent of  $x_i$ . In this case,

$$\|\mathbf{w}_i \odot \text{Std}[\mathbf{y}]\|_1^2 = w_{ij(i)}^2 \text{Var}[y_{j(i)}] = \|\mathbf{w}_i \odot \text{Std}[\mathbf{y}]\|_2^2 \quad (42)$$

The bound is also tight when  $\text{Var}[\mathbf{y}] = 0$ , but in this case both the ELBO and the FM bound yield  $-\infty$  because of the conditional entropy term  $H(\mathbf{y}|\mathbf{x})$ .

**Speed of Convergence** Let’s examine the role of  $\varepsilon$  in the update for  $q(y_j|\mathbf{x})$ . If  $\sum_{j=1}^m \frac{w_{ij}}{\varepsilon_{ij}}$  is large, then  $\mathbb{E}_{q_{t+1}} \approx \mathbb{E}_{q_t}[y_j]$ , and so the FM algorithm makes a small update for  $y_j$ . If  $\sum_{j=1}^m \frac{w_{ij}}{\varepsilon_{ij}}$  is small, then  $\mathbb{E}_{q_{t+1}}[y_j]$  makes a large step in the direction of the residual  $\mathbf{x} - \mathbb{E}[\boldsymbol{\eta}]$ . In fact, if for some  $j$ ,  $\varepsilon_{ij} = 1$  for all  $i$  where  $w_{ij}$  is non-zero, then the FM algorithm updates

$q(y_j|\mathbf{x})$  exactly as CAVI would. In this sense,  $\varepsilon$  acts like an attention parameter that selects which  $q(y_j|\mathbf{x})$  to change and by how much.

If  $p$  is a forest model, then the FM algorithm chooses  $\varepsilon_i$  to be the one-hot vector indicating the parent of  $x_i$ . In this case, the FM algorithm makes coordinate updates for all  $j$  in parallel and converges in one iteration. If  $p$  is forest-like, i.e.,  $|\mathbf{w}_j| \cdot |\mathbf{w}_{j'}|$  is small for  $j \neq j'$ , then  $\varepsilon_i$  is close to one-hot and the FM algorithm makes damped, nearly-CAVI updates in parallel. In this sense, the speed at which the FM algorithm converges depends on how inherently forest-like the model  $p$  is.

## 7 EXPERIMENTS

Recall that we derived the FM bound by lower bounding the ELBO. Algorithms that optimize the ELBO like CAVI will generally provide a superior lower bound on log-likelihood compared to the FM algorithm. For a more fair comparison, we can instead measure how quickly these algorithms converge to the optimal mean. In the Gaussian case, optimizing the mean of the mean-field variational distribution is equivalent to minimizing a ridge regression objective:

$$\frac{1}{2\sigma_x^2} \sum_{i=1}^n (x_i - (b_i + \mathbf{w}_i \cdot \mathbb{E}[\mathbf{y}]))^2 + \frac{1}{2\sigma_y^2} \sum_{j=1}^m \mathbb{E}[y_j]^2 \quad (43)$$

To evaluate each algorithm on the ridge regression problem, we must first choose a  $\mathbf{x}$ ,  $\mathbf{b}$ , and a set of  $w_{ij}$ . All the algorithms we consider in this section are guaranteed to converge to the optimal solution, so we are only interested in comparing how quickly each algorithm converges to that optimal solution. This is measured by recording the objective value achieved by the mean of the variational distribution  $\mathbb{E}_{q_t}[\mathbf{y}]$  in the ridge regression problem across 200 iterations.

In the first experiment, we choose  $\mathbf{x}$  to be a vectorized sample from the MNIST dataset, with pixel values scaled to lie in the interval  $[-1, 1]$ ; we choose  $\mathbf{b}$  to be the average of 1000 randomly chosen MNIST samples; and we construct a synthetic  $w_{ij}$  as follows: given an integer window side length  $s$ , we construct all possible square  $s \times s$  windows of pixels. For windows that overlap the border of the  $28 \times 28$  MNIST image region, we clip the window so that it lies entirely inside the image region, resulting in a rectangular window. For each window, we add a latent variable  $y_j$  to the model and a corresponding  $\mathbf{w}_j$ , where  $w_{ij} = 1$  if pixel  $i$  lies in window  $j$ , and  $w_{ij} = 0$  otherwise. The resulting model is more forest-like for smaller choices of  $s$ : if  $s = 1$ , the windows are

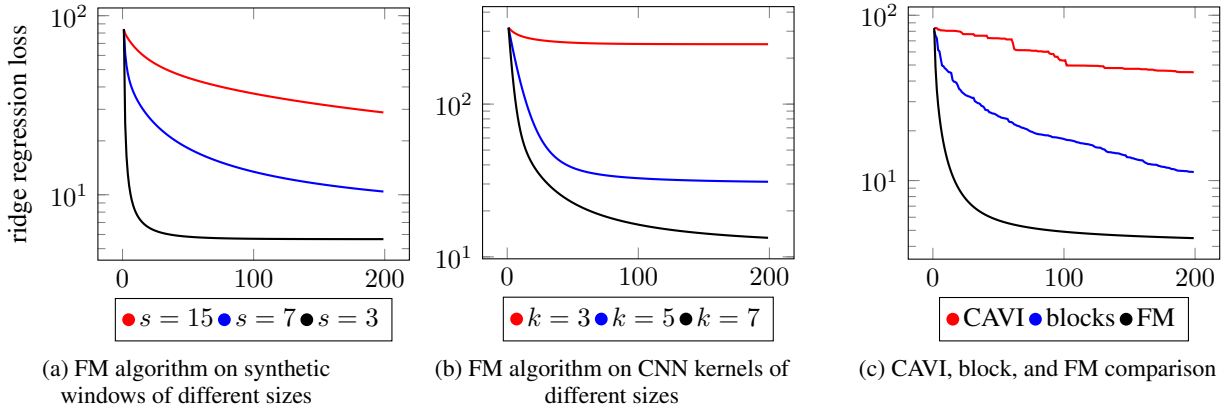


Figure 2: The ridge regression objective over 200 iterations.

disjoint and the graphical model is exactly a forest. Figure 2a demonstrates the rate of convergence of the FM algorithm for various choices of  $s$ . As we expect, the FM algorithm converges faster for more forest-like models, i.e., smaller  $s$ . Note that the objective value achieved by the optimal solution to the ridge regression problem changes as  $w_{ij}$  changes.

The second experiment is similar to the first, except it uses  $\mathbf{x}$  from the CIFAR-10 dataset,  $\mathbf{b} = 0$ , and instead of uniform windows, uses the first layer kernels from a convolutional neural net trained several times changing only the width of the first layer kernels. Figure 2b demonstrates the FM algorithm converges faster for more forest-like models even using real-world data.

Our last experiment compares the convergence of the FM algorithm with CAVI and block coordinate ascent. Here we choose  $\mathbf{x}$  and  $\mathbf{b}$  the same as in the first experiment, but we choose  $w_{ij}$  differently to make blocking the latent variables easy: first we partition the  $28 \times 28$  MNIST image region into 16 regions, each of size  $7 \times 7$ . Then, we construct all possible  $7 \times 7$  windows (as in the first experiment with  $s = 7$ ), then clip them to fit in the first region. This is repeated for each region. If we block the latent variables according to which region the corresponding windows were clipped to, then the blocks will be conditionally independent, since windows clipped to different regions must be disjoint. Blocking in this way guarantees that the block coordinate algorithm will converge to the optimal solution. Figure 2c compares the rate of convergence for CAVI, block coordinate ascent, and the FM algorithm. The figure shows our block-free method can outperform the block coordinate method, even when the blocking is quite good.

## 8 CONCLUSION

In this paper we derived a forest mixture bound on the log-likelihood of deep exponential families. This bound gets around the “explaining away effect” by using a set of auxiliary parameters to separate the problem of inference for each latent variable in the same layer, allowing us to make parallel updates. We then made a deep dive into the simple case where all variables are Gaussian: we derived the exact variable updates, then tested the algorithm on both synthetic and real-world data. Our promising results show that fast, parallel inference in deep exponential families is possible without the use of blocks.

## A AUXILIARY PARAMETER UPDATES

**Proof of Theorem 1:** First, we will find the optimal choice of  $\hat{\mathbf{b}}_i$  for any given  $\varepsilon_i$ . Since  $\hat{\mathbf{b}}_i$  is constrained by  $\sum_{j=1}^m \varepsilon_{ij} \hat{b}_{ij} = b_i$ , let’s first parameterize  $\hat{\mathbf{b}}_i$  by a set of unconstrained parameters: let  $\gamma_i \in \mathbb{R}^m$  and write

$$\hat{b}_{ij} = b_i - \gamma_{ij} + \varepsilon_i \cdot \gamma_i \quad (44)$$

So for any choice of  $\gamma_i$ , the constraint  $b_i = \sum_{j=1}^m \varepsilon_{ij} \hat{b}_{ij}$  is satisfied. Now we can differentiate the bound with respect to  $\gamma_{ij}$ , set to zero and solve. We will need the following partial derivatives:

$$\frac{\partial \hat{b}_{ij}}{\partial \gamma_{ij}} = -1 + \varepsilon_{ij} \quad \frac{\partial \hat{b}_{ij'}}{\partial \gamma_{ij}} = \varepsilon_{ij} \quad \forall j' \neq j \quad (45)$$

Now setting the partial derivative of  $\mathcal{L}_i$  with respect to  $\gamma_{ij}$  to zero,

$$0 = \frac{\partial}{\partial \gamma_{ij}} \mathcal{L}_i = -\frac{1}{\sigma_x^2} \sum_{j'=1}^m \varepsilon_{ij'} \mathbb{E}[\hat{\eta}_{ij'}] \frac{\partial \hat{b}_{ij'}}{\partial \gamma_{ij}} \quad (46)$$

$$= \frac{\varepsilon_{ij}}{\sigma_x^2} \left( \mathbb{E}[\hat{\eta}_{ij}] - \sum_{j'=1}^m \varepsilon_{ij'} \mathbb{E}[\hat{\eta}_{ij'}] \right) \quad (47)$$

The derivative is zero for all  $j$  in particular when the choice of  $\hat{b}_{ij}$  makes  $\mathbb{E}[\hat{\eta}_{ij}]$  constant across  $j$ . We can verify this is satisfied by the choice  $\gamma_{ij} = \frac{w_{ij}}{\varepsilon_{ij}} \mathbb{E}[y_j]$ , which makes  $\hat{b}_{ij} = \hat{b}_{ij}^*$ :

$$\mathbb{E}[\hat{\eta}_{ij}] = \mathbb{E} \left[ \hat{b}_{ij} + \frac{w_{ij}}{\varepsilon_{ij}} y_j \right] \quad (48)$$

$$= \mathbb{E} \left[ \mathbb{E}[\eta_i] - \frac{w_{ij}}{\varepsilon_{ij}} \mathbb{E}[y_j] + \frac{w_{ij}}{\varepsilon_{ij}} y_j \right] \quad (49)$$

$$= \mathbb{E}[\eta_i] \quad (50)$$

Plugging this choice into  $\mathcal{L}_i$  yields

$$\mathcal{L}_i = -\frac{1}{2\sigma_x^2} \sum_{j=1}^m \left( \frac{w_{ij}^2 \text{Var}[y_j]}{\varepsilon_{ij}} + \varepsilon_{ij} \mathbb{E}[\eta_i]^2 \right) \quad (51)$$

$$= -\frac{1}{2\sigma_x^2} \left( \sum_{j=1}^m \frac{w_{ij}^2 \text{Var}[y_j]}{\varepsilon_{ij}} \right) - \frac{1}{2\sigma_x^2} \mathbb{E}[\eta_i]^2 \quad (52)$$

Now let's try to find the optimal choice of  $\varepsilon_{ij}$ . Since  $\varepsilon_{ij}$  is constrained by  $\varepsilon_i \in \Delta^{m-1}$ , we'll also parameterize  $\varepsilon_{ij}$  by a set of unconstrained parameters  $\tau_i \in \mathbb{R}^m$ :

$$\varepsilon_{ij} = \exp\{\tau_{ij}\} / \sum_{j'=1}^m \exp\{\tau_{ij'}\} \quad (53)$$

We will need the following partial derivatives:

$$\frac{\partial \varepsilon_{ij}}{\partial \tau_{ij}} = \varepsilon_{ij}(1 - \varepsilon_{ij}) \quad \frac{\partial \varepsilon_{ij'}}{\partial \tau_{ij}} = -\varepsilon_{ij} \varepsilon_{ij'} \quad \forall j' \neq j \quad (54)$$

Now setting the partial derivative of  $\mathcal{L}_i$  with respect to  $\tau_{ij}$  to zero,

$$0 = \frac{\partial}{\partial \tau_{ij}} \mathcal{L}_i = \frac{1}{2\sigma_x^2} \sum_{j'=1}^m \frac{w_{ij'}^2 \text{Var}[y_{j'}]}{\varepsilon_{ij'}^2} \frac{\partial \varepsilon_{ij'}}{\partial \tau_{ij}} \quad (55)$$

$$= \frac{1}{2\sigma_x^2} \sum_{j'=1}^m \text{Var}[\hat{\eta}_{ij'}] \frac{\partial \varepsilon_{ij'}}{\partial \tau_{ij}} \quad (56)$$

$$= \frac{\varepsilon_{ij}}{2\sigma_x^2} \left( \text{Var}[\hat{\eta}_{ij}] - \sum_{j'=1}^m \varepsilon_{ij'} \text{Var}[\hat{\eta}_{ij'}] \right) \quad (57)$$

The derivative is zero for all  $j$  in particular when the choice of  $\varepsilon_{ij}$  makes  $\text{Var}[\hat{\eta}_{ij}]$  constant across  $j$ . We can verify this is satisfied by the choice  $\tau_{ij} = \log |w_{ij} \text{Std}[y_j]|$ , which makes  $\varepsilon_{ij} = \varepsilon_{ij}^*$ :

$$\text{Var}[\hat{\eta}_{ij}] = \frac{w_{ij}^2 \text{Var}[y_j]}{\varepsilon_{ij}^2} \quad (58)$$

$$= \frac{w_{ij}^2 \text{Var}[y_j]}{w_{ij}^2 \text{Var}[y_j] / \left( \sum_{j'=1}^m |w_{ij'}| \text{Std}[y_{j'}] \right)^2} \quad (59)$$

$$= \|\mathbf{w}_i \odot \text{Std}[\mathbf{y}]\|_1^2 \quad (60)$$

Plugging this choice into  $\mathcal{L}_i$  yields

$$\mathcal{L}_i = -\frac{1}{2\sigma_x^2} \sum_{j=1}^m |w_{ij}| \text{Std}[y_j] \left( \sum_{j'=1}^m |w_{ij'}| \text{Std}[y_{j'}] \right) - \frac{1}{2\sigma_x^2} \mathbb{E}[\eta_i]^2 \quad (61)$$

$$= -\frac{1}{2\sigma_x^2} \left( \sum_{j=1}^m |w_{ij}| \text{Std}[y_j] \right)^2 - \frac{1}{2\sigma_x^2} \mathbb{E}[\eta_i]^2 \quad (62)$$

$$= -\frac{1}{2\sigma_x^2} \|\mathbf{w}_i \odot \text{Std}[\mathbf{y}]\|_1^2 - \frac{1}{2\sigma_x^2} \mathbb{E}[\eta_i]^2 \quad (63)$$

## B VARIATIONAL UPDATES

Proof of **Theorem 2**: First, note that for any DEF, the optimal update equation is as follows:

$$\log q_{t+1}(y_j | \mathbf{x}) \propto \log p(y_j) + \sum_{i=1}^n \varepsilon_{ij} \log \hat{p}_t(x_i | y_j) \quad (64)$$

In the Gaussian case, we have

$$\log p(y_j) \propto -\frac{1}{2\sigma_y^2} y_j^2 \quad (65)$$

$$\log \hat{p}(x_i|y_j) \propto -\frac{1}{2\sigma_x^2} (x_i - \hat{\eta}_{ij})^2 \quad (66)$$

$$\propto -\frac{1}{2\sigma_x^2} \left( x_i - \left( \hat{b}_{ij} + \frac{w_{ij}}{\varepsilon_{ij}} y_j \right) \right)^2 \quad (67)$$

$$\propto \frac{1}{\sigma_x^2} \frac{(x_i - \hat{b}_{ij})w_{ij}}{\varepsilon_{ij}} y_j - \frac{1}{2\sigma_x^2} \frac{w_{ij}^2}{\varepsilon_{ij}^2} y_j^2 \quad (68)$$

Plugging this in yields

$$\log q(y_j|\mathbf{x}) \propto \frac{1}{\sigma_x^2} \left( (\mathbf{x} - \hat{\mathbf{b}}_j) \cdot \mathbf{w}_j \right) y_j - \frac{1}{2} y_j^2 \left( \frac{1}{\sigma_y^2} + \frac{1}{\sigma_x^2} \sum_{i=1}^n \frac{w_{ij}^2}{\varepsilon_{ij}} \right) \quad (69)$$

$$\propto -\frac{1}{\sigma_x^2} \left( (\mathbf{x} - \hat{\mathbf{b}}_j) \cdot \mathbf{w}_j \right) y_j - \frac{1}{2(\sigma_{t+1}^*)^2} y_j^2 \quad (70)$$

$$\propto -\frac{1}{2(\sigma_{t+1}^*)^2} \left( y_j - \frac{\frac{1}{\sigma_x^2} (\mathbf{x} - \hat{\mathbf{b}}_j) \cdot \mathbf{w}_j}{\frac{1}{\sigma_y^2} + \frac{1}{\sigma_x^2} \sum_{i=1}^n \frac{w_{ij}^2}{\varepsilon_{ij}}} \right)^2 \quad (71)$$

$$\propto -\frac{1}{2(\sigma_{t+1}^*)^2} \left( y_j - \frac{(\mathbf{x} - \hat{\mathbf{b}}_j) \cdot \mathbf{w}_j}{\frac{\sigma_x^2}{\sigma_y^2} + \sum_{i=1}^n \frac{w_{ij}^2}{\varepsilon_{ij}}} \right)^2 \quad (72)$$

After substituting  $\hat{b}_{ij} = \mathbb{E}[\eta_i] - \frac{w_{ij}}{\varepsilon_{ij}} \mathbb{E}_{q_t}[y_j]$  and rearranging, we get  $\log q_{t+1}(y_j|\mathbf{x}) \propto \mathcal{N}((\mu_{t+1}^*)_{ij}, (\sigma_{t+1}^*)_{ij}^2)$ .

## C EXTENSION TO DEEP MODELS

A DEF model with observed variables  $\mathbf{y}^{(0)} \in \mathbb{R}^{m_0}$  and  $L$  layers of latent variables  $\{\mathbf{y}^{(\ell)}\}_{\ell=1}^L$  with  $\mathbf{y}^{(\ell)} \in \mathbb{R}^{m_\ell}$  has joint distribution

$$p(\{\mathbf{y}^{(\ell)}\}_{\ell=0}^L) = \left[ \prod_{\ell=0}^{L-1} \prod_{i=1}^{m_\ell} p(y_i^{(\ell)}|\mathbf{y}^{(\ell+1)}) \right] \left[ \prod_{i=1}^{m_L} p(y_i^{(L)}) \right] \quad (73)$$

$$p(y_i^{(\ell)}|\mathbf{y}^{(\ell+1)}) = \exp \left\{ g(y_i^{(\ell)}) + t(y_i^{(\ell)})\eta_i^{(\ell)} - a(\eta_i^{(\ell)}) \right\} \quad (74)$$

$$\eta_i^{(\ell)} \equiv b_i^{(\ell)} + \mathbf{w}_i^{(\ell)} \cdot \mathbf{y}^{(\ell+1)} \quad (75)$$

The ELBO for this model is

$$\log p(\mathbf{y}^{(0)}) \geq \sum_{i=1}^{m_0} \mathbb{E}[\log p(y_i^{(0)}|\mathbf{y}^{(1)})] \quad (76)$$

$$+ \sum_{\ell=1}^{L-1} \sum_{i=1}^{m_\ell} \mathbb{E}[\log p(y_i^{(\ell)}|\mathbf{y}^{(\ell+1)})] + H_q(\mathbf{y}_i^{(\ell)}|\mathbf{y}^{(0)}) \quad (77)$$

$$+ \sum_{i=1}^{m_L} p(y_i^{(L)}) + H_q(\mathbf{y}_i^{(L)}|\mathbf{y}^{(0)}) \quad (78)$$

For each  $\ell \in \{0, \dots, L-1\}$ , introduce the auxiliary parameters  $\{\varepsilon_i^{(\ell)}\}_{i=1}^{m_\ell}$  and  $\{\hat{\mathbf{b}}_i^{(\ell)}\}_{i=1}^{m_\ell}$ , with  $\varepsilon_i^{(\ell)} \in \Delta^{m_{\ell+1}-1}$  and  $\hat{\mathbf{b}}_i^{(\ell)} \in \mathbb{R}^{m_{\ell+1}}$  constrained by  $b_i^{(\ell)} = \sum_{j=1}^{m_{\ell+1}} \varepsilon_{ij}^{(\ell)} \hat{b}_{ij}^{(\ell)}$ . For all  $\ell \in \{0, \dots, L-1\}$ ,  $i \in [m_\ell]$ , and  $j \in [m_{\ell+1}]$ , define

$$\hat{\eta}_{ij}^{(\ell)} \equiv \hat{b}_{ij}^{(\ell)} + \frac{w_{ij}^{(\ell)}}{\varepsilon_{ij}^{(\ell)}} y_j^{(\ell+1)} \quad (79)$$

$$\hat{p}(y_i^{(\ell)}|\mathbf{y}_j^{(\ell+1)}) \equiv \exp\{g(y_i^{(\ell)}) + t(y_i^{(\ell)})\hat{\eta}_{ij}^{(\ell)} - a(\hat{\eta}_{ij}^{(\ell)})\} \quad (80)$$

Then by (25),

$$\mathbb{E}[\log p(y_i^{(\ell)}|\mathbf{y}^{(\ell+1)})] \geq \sum_{j=1}^{m_{\ell+1}} \varepsilon_{ij}^{(\ell)} \mathbb{E}[\log \hat{p}(y_i^{(\ell)}|\mathbf{y}_j^{(\ell+1)})] \quad (81)$$

Plugging this into the ELBO yields

$$\log p(\mathbf{y}^{(0)}) \geq \sum_{i=1}^{m_0} \sum_{j=1}^{m_1} \varepsilon_{ij}^{(0)} \mathbb{E}[\log \hat{p}(y_i^{(0)}|\mathbf{y}_j^{(1)})] \quad (82)$$

$$+ \sum_{\ell=1}^{L-1} \sum_{i=1}^{m_\ell} \sum_{j=1}^{m_{\ell+1}} \varepsilon_{ij}^{(\ell)} \mathbb{E}[\log \hat{p}(y_i^{(\ell)}|\mathbf{y}_j^{(\ell+1)})] + H_q(\mathbf{y}_i^{(\ell)}|\mathbf{y}^{(0)}) \quad (83)$$

$$+ \sum_{i=1}^{m_L} p(y_i^{(L)}) + H_q(\mathbf{y}_i^{(L)}|\mathbf{y}^{(0)}) \quad (84)$$

This objective separates as a sum of terms, each of which involves no more than one latent variable in the same layer. This allows any group of variables forming an independent set in the model graph to be updated in parallel, the same as for undirected models.

## References

- [Baqué et al., 2016] Baqué, P., Bagautdinov, T., Fleuret, F., and Fua, P. (2016). Principled parallel mean-field inference for discrete random fields. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5848–5857.
- [Blei et al., 2017] Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877.
- [Chow and Liu, 1968] Chow, C. and Liu, C. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467.
- [Johnson et al., 2013] Johnson, M., Saunderson, J., and Willsky, A. (2013). Analyzing hogwild parallel Gaussian Gibbs sampling. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26*, pages 2715–2723. Curran Associates, Inc.
- [Kingma and Welling, 2013] Kingma, D. P. and Welling, M. (2013). Auto-Encoding Variational Bayes. *ArXiv e-prints*.
- [Ranganath et al., 2015] Ranganath, R., Tang, L., Charlin, L., and Blei, D. (2015). Deep exponential families. In Lebanon, G. and Vishwanathan, S. V. N., editors, *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, volume 38 of *Proceedings of Machine Learning Research*, pages 762–771, San Diego, California, USA. PMLR.
- [Recht et al., 2011] Recht, B., Re, C., Wright, S., and Niu, F. (2011). Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In Shawe-Taylor, J., Zemel, R. S., Bartlett, P. L., Pereira, F., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 24*, pages 693–701. Curran Associates, Inc.
- [Sa et al., 2016] Sa, C. D., Re, C., and Olukotun, K. (2016). Ensuring rapid mixing and low bias for asynchronous Gibbs sampling. In Balcan, M. F. and Weinberger, K. Q., editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1567–1576, New York, New York, USA. PMLR.
- [Salakhutdinov and Hinton, 2009] Salakhutdinov, R. and Hinton, G. (2009). Deep Boltzmann machines. In van Dyk, D. and Welling, M., editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pages 448–455, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA. PMLR.
- [Singh et al., 2010] Singh, S., Subramanya, A., Pereira, F., and McCallum, A. (2010). Distributed MAP inference for undirected graphical models. In *Neural Information Processing Systems (NIPS) Workshop on Learning on Cores, Clusters, and Clouds (LCCC)*.
- [Singh et al., 2017] Singh, S. S., Lindsten, F., and Moulines, E. (2017). Blocking strategies and stability of particle Gibbs samplers. *Biometrika*, 104(4):953–969.
- [Sontag and Jaakkola, 2009] Sontag, D. and Jaakkola, T. (2009). Tree block coordinate descent for MAP in graphical models. In van Dyk, D. and Welling, M., editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pages 544–551, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA. PMLR.
- [Terenin et al., 2015] Terenin, A., Simpson, D., and Draper, D. (2015). Asynchronous Gibbs sampling. *ArXiv e-prints*.
- [Wainwright et al., 2005] Wainwright, M. J., Jaakkola, T. S., and Willsky, A. S. (2005). A new class of upper bounds on the log partition function. *IEEE Transactions on Information Theory*, 51(7):2313–2335.
- [Wang and Banerjee, 2014] Wang, H. and Banerjee, A. (2014). Randomized block coordinate descent for online and stochastic optimization. *CoRR*, abs/1407.0107.
- [Zhao et al., 2014] Zhao, T., Yu, M., Wang, Y., Arora, R., and Liu, H. (2014). Accelerated mini-batch randomized block coordinate descent method. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 27*, pages 3329–3337. Curran Associates, Inc.

---

# Causal Identification under Markov Equivalence

---

**Amin Jaber**

Computer Science Department  
Purdue University, IN, USA  
jaber0@purdue.edu

**Jiji Zhang**

Philosophy Department  
Lingnan University, NT, HK  
jjizhang@ln.edu.hk

**Elias Bareinboim**

Computer Science Department  
Purdue University, IN, USA  
eb@purdue.edu

## Abstract

Assessing the magnitude of cause-and-effect relations is one of the central challenges found throughout the empirical sciences. The problem of identification of causal effects is concerned with determining whether a causal effect can be computed from a combination of observational data and substantive knowledge about the domain under investigation, which is formally expressed in the form of a causal graph. In many practical settings, however, the knowledge available for the researcher is not strong enough so as to specify a unique causal graph. Another line of investigation attempts to use observational data to learn a qualitative description of the domain called a Markov equivalence class, which is the collection of causal graphs that share the same set of observed features. In this paper, we marry both approaches and study the problem of causal identification from an equivalence class, represented by a partial ancestral graph (PAG). We start by deriving a set of graphical properties of PAGs that are carried over to its induced subgraphs. We then develop an algorithm to compute the effect of an arbitrary set of variables on an arbitrary outcome set. We show that the algorithm is strictly more powerful than the current state of the art found in the literature.

## 1 INTRODUCTION

Science is about explaining the mechanisms underlying a phenomenon that is being investigated. One of the marks imprinted by these mechanisms in reality is cause and effect relationships. Systematically discovering the existence, and magnitude, of causal relations constitutes,

therefore, a central task in scientific domains. The value of inferring causal relationships is also tremendous in other, more practical domains, including, for example, engineering and business, where it is often crucial to understand how to bring about a specific change when a constrained amount of controllability is in place. If our goal is to build AI systems that can act and learn autonomously, formalizing the principles behind causal inference, so that these systems can leverage them, is a fundamental requirement (Pearl and Mackenzie, 2018).

One prominent approach to infer causal relations leverages a combination of substantive knowledge about the domain under investigation, usually encoded in the form of a causal graph, with observational (non-experimental) data (Pearl, 2000; Spirtes et al., 2001; Bareinboim and Pearl, 2016). A sample causal graph is shown in Fig. 1a such that the nodes represent variables, directed edges represent direct causal relation from tails to heads, and bi-directed arcs represent the presence of unobserved (latent) variables that generate a spurious association between the variables, also known as *confounding bias* (Pearl, 1993). The task of determining whether an interventional (experimental) distribution can be computed from a combination of observational and experimental data together with the causal graph is known as the problem of identification of causal effects (identification, for short). For instance, a possible task in this case is to identify the effect of  $do(X=x)$  on  $V_4=v_4$ , i.e.  $P_x(v_4)$ , given the causal graph in Fig. 1a and data from the observational distribution  $P(x, v_1, \dots, v_4)$ .

The problem of identification has been extensively studied in the literature, and a number of criteria have been established (Pearl, 1993; Galles and Pearl, 1995; Kuroki and Miyakawa, 1999; Tian and Pearl, 2002; Huang and Valtorta, 2006; Shpitser and Pearl, 2006; Bareinboim and Pearl, 2012), which include the celebrated back-door criterion and the do-calculus (Pearl, 1995). Despite their power, these techniques require a fully specified causal graph, which is not always available in practical settings.

Another line of investigation attempts to learn a qualitative description of the system, which in the ideal case would lead to the “true” data-generating model, the blueprint underlying the phenomenon being investigated. These efforts could certainly be deemed more “data-driven” and aligned with the zeitgeist in machine learning. In practice, however, it is common that only an equivalence class of causal models can be consistently inferred from observational data (Verma, 1993; Spirtes et al., 2001; Zhang, 2008b). One useful characterization of such an equivalence class comes under the rubric of *partial ancestral graphs (PAGs)*, which will be critical to our work. Fig. 1 shows the PAG (right) that can be inferred from observational data that is consistent with the true causal model (left). The directed edges in a PAG signify ancestral relations (not necessarily direct) and circle marks stand for structural uncertainty.

In this paper, we analyze the marriage of these two lines of investigation, where the structural invariance learned in the equivalence class will be used as input to identify the strength of causal effect relationships, if possible. Identification from an equivalence class is considerably more challenging than from a single diagram due to the structural uncertainty regarding both the direct causal relations among the variables and the presence of latent variables that confounds causal relations between observed variables. Still, there is a growing interest in identifiability results in this setting (Maathuis et al., 2010). Zhang (2007) extended the do-calculus to PAGs. In practice, however, it is in general computationally hard to decide whether there exists (and, if so, find) a sequence of applications of the rules of the generalized calculus to identify the interventional distribution. Perković et al. (2015) generalized the back-door criterion to PAGs, and provided a sound and complete algorithm to find a back-door admissible set, should such a set exist. However, in practice, the back-door criterion is not as powerful as the do-calculus, since no adjustment set exists for many identifiable causal effects. Jaber et al. (2018b) generalized the work of (Tian and Pearl, 2002) and devised a graphical criterion to identify causal effects with singleton interventions in PAGs.<sup>1</sup>

Building on this work, we develop here a decomposition strategy akin to the one introduced in (Tian, 2002) to identify causal effects given a PAG. Our proposed approach is computationally more attractive than the do-calculus as it provides a systematic procedure to identify

<sup>1</sup>Another possible approach is based on SAT (boolean constraint satisfaction) solvers (Hyttinen et al., 2015). Given its somewhat distinct nature, a closer comparison lies outside the scope of this paper. We note, however, that an open research direction would be to translate our systematic approach into logical rules so as to help improving the solver’s scalability.

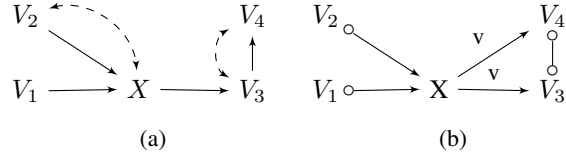


Figure 1: A causal model (left) and the inferred PAG (right).

a causal effect, if identifiable. It is also more powerful than the generalized adjustment criterion, as we show later. More specifically, our main contributions are:

1. We study some critical properties of PAGs and show that they also hold in induced subgraphs of a PAG over an arbitrary subset of nodes. We further study Tian’s *c*-component decomposition and relax it to PAGs (when only partial knowledge about the ancestral relations and *c*-components is available).
2. We formulate a systematic procedure to compute the effect of an arbitrary set of intervention variables on an arbitrary outcome set from a PAG and observational data. We show that this algorithm is strictly more powerful than the adjustment criterion.

## 2 PRELIMINARIES

In this section, we introduce the basic notation and machinery used throughout the paper. Bold capital letters denote sets of variables, while bold lowercase letters stand for particular assignments to those variables.

**Structural Causal Models.** We use the language of Structural Causal Models (SCM) (Pearl, 2000, pp. 204–207) as our basic semantic framework. Formally, an SCM  $M$  is a 4-tuple  $\langle U, V, F, P(u) \rangle$ , where  $U$  is a set of exogenous (latent) variables and  $V$  is a set of endogenous (measured) variables.  $F$  represents a collection of functions  $F = \{f_i\}$  such that each endogenous variable  $V_i \in V$  is determined by a function  $f_i \in F$ , where  $f_i$  is a mapping from the respective domain of  $U_i \cup Pa_i$  to  $V_i$ ,  $U_i \subseteq U$ ,  $Pa_i \subseteq V \setminus V_i$ . The uncertainty is encoded through a probability distribution over the exogenous variables,  $P(u)$ . A causal diagram associated with an SCM encodes the structural relations among  $V \cup U$ , in which an arrow is drawn from each member of  $U_i \cup Pa_i$  to  $V_i$ . We constraint our results to recursive systems, which means that the corresponding diagram will be acyclic. The marginal distribution over the endogenous variables  $P(v)$  is called observational, and factorizes according to the causal diagram, i.e.:

$$P(v) = \sum_u \prod_i P(v_i | pa_i, u_i) P(u)$$

Within the structural semantics, performing an action  $X=x$  is represented through the do-operator,  $do(X=x)$ , which encodes the operation of replacing the original equation for  $X$  by the constant  $x$  and induces a submodel  $M_x$ . The resulting distribution is denoted by  $P_x$ , which is the main target for identification in this paper. For details on structural models, we refer readers to (Pearl, 2000).

**Ancestral Graphs.** We now introduce a graphical representation of equivalence classes of causal diagrams. A *mixed* graph can contain directed ( $\rightarrow$ ) and bi-directed edges ( $\leftrightarrow$ ).  $A$  is a *spouse* of  $B$  if  $A \leftrightarrow B$  is present. An *almost directed cycle* happens when  $A$  is both a spouse and an ancestor of  $B$ . An *inducing path relative to  $\mathbf{L}$*  is a path on which every node  $V \notin \mathbf{L}$  (except for the endpoints) is a collider on the path (i.e., both edges incident to  $V$  are into  $V$ ) and every collider is an ancestor of an endpoint of the path. A mixed graph is *ancestral* if it doesn't contain a directed or almost directed cycle. It is *maximal* if there is no inducing path (relative to the empty set) between any two non-adjacent nodes. A *Maximal Ancestral Graph* (MAG) is a graph that is both ancestral and maximal. MAG models are closed under marginalization (Richardson and Spirtes, 2002).

In general, a causal MAG represents a set of causal models with the same set of observed variables that entail the same independence and ancestral relations among the observed variables. Different MAGs may be Markov equivalent in that they entail the exact same independence model. A partial ancestral graph (PAG) represents an equivalence class of MAGs  $[\mathcal{M}]$ , which shares the same adjacencies as every MAG in  $[\mathcal{M}]$  and displays all and only the invariant edge marks.

**Definition 1 (PAG).** Let  $[\mathcal{M}]$  be the Markov equivalence class of an arbitrary MAG  $\mathcal{M}$ . The PAG for  $[\mathcal{M}]$ ,  $\mathcal{P}$ , is a partial mixed graph such that:

- i.  $\mathcal{P}$  has the same adjacencies as  $\mathcal{M}$  (and hence any member of  $[\mathcal{M}]$ ) does.
- ii. An arrowhead is in  $\mathcal{P}$  iff shared by all MAGs in  $[\mathcal{M}]$ .
- iii. A tail is in  $\mathcal{P}$  iff shared by all MAGs in  $[\mathcal{M}]$ .
- iv. A mark that is neither an arrowhead nor a tail is recorded as a circle.

A PAG is learnable from the conditional independence and dependence relations among the observed variables and the FCI algorithm is a standard method to learn such an object (Zhang, 2008b). In short, a PAG represents an equivalence class of causal models with the same observed variables and independence model.

**Graphical Notions.** Given a DAG, MAG, or PAG, a path between  $X$  and  $Y$  is *potentially directed* (causal)

from  $X$  to  $Y$  if there is no arrowhead on the path pointing towards  $X$ .  $Y$  is called a *possible descendant* of  $X$  and  $X$  a *possible ancestor* of  $Y$  if there is a potentially directed path from  $X$  to  $Y$ . A set  $\mathbf{A}$  is (*descendant*) *ancestral* if no node outside  $\mathbf{A}$  is a possible (descendant) ancestor of any node in  $\mathbf{A}$ .  $Y$  is called a *possible child* of  $X$ , i.e.  $Y \in \text{Ch}(X)$ , and  $X$  a *possible parent* of  $Y$ , i.e.  $X \in \text{Pa}(Y)$ , if they are adjacent and the edge is not into  $X$ . For a set of nodes  $\mathbf{X}$ , we have  $\text{Pa}(\mathbf{X}) = \cup_{X \in \mathbf{X}} \text{Pa}(X)$  and  $\text{Ch}(\mathbf{X}) = \cup_{X \in \mathbf{X}} \text{Ch}(X)$ . Given two sets of nodes  $\mathbf{X}$  and  $\mathbf{Y}$ , a path between them is called *proper* if one of the endpoints is in  $\mathbf{X}$  and the other is in  $\mathbf{Y}$ , and no other node on the path is in  $\mathbf{X}$  or  $\mathbf{Y}$ . For convenience, we use an asterisk (\*) to denote any possible mark of a PAG ( $\circ, >, -$ ) or a MAG ( $>, -$ ). If the edge marks on a path between  $X$  and  $Y$  are all circles, we call the path a *circle path*.

A directed edge  $X \rightarrow Y$  in a MAG or PAG is *visible* if there exists no DAG  $\mathcal{D}(\mathbf{V}, \mathbf{L})$  in the corresponding equivalence class where there is an inducing path between  $X$  and  $Y$  that is into  $X$  relative to  $\mathbf{L}$ . This implies that a visible edge is not confounded ( $X \leftarrow U_i \rightarrow Y$  doesn't exist). Which directed edges are visible is easily decidable by a graphical condition (Zhang, 2008a), so we simply mark visible edges by  $v$ . For brevity, we refer to any edge that is not a visible directed edge as *invisible*.

**Identification Given a Causal DAG.** Tian and Pearl (2002) presented an identification algorithm based on a decomposition strategy of the DAG into a set of so-called *c-components* (confounded components).

**Definition 2 (C-Component).** In a causal DAG, two observed variables are said to be in the same *c-component* if and only if they are connected by a bi-directed path, i.e. a path composed solely of such bi-directed treks as  $V_i \leftarrow U_{ij} \rightarrow V_j$ , where  $U_{ij}$  is an exogenous variable.

For convenience, we often refer to a bi-directed trek like  $V_i \leftarrow U_{ij} \rightarrow V_j$  as a bi-directed edge between  $V_i$  and  $V_j$  (and  $U_{ij}$  is often left implicit). For any set  $\mathbf{C} \subseteq \mathbf{V}$ , we define the quantity  $Q[\mathbf{C}]$  to denote the post-intervention distribution of  $\mathbf{C}$  under an intervention on  $\mathbf{V} \setminus \mathbf{C}$ :

$$Q[\mathbf{C}] = P_{\mathbf{v} \setminus \mathbf{c}}(\mathbf{c}) = \sum_u \prod_{\{i|V_i \in \mathbf{C}\}} P(v_i | pa_i, u_i) P(u)$$

The significance of c-components and their decomposition is evident from (Tian, 2002, Lemmas 10, 11), which are the basis of Tian's identification algorithm.

### 3 REVISIT IDENTIFICATION IN DAGS

We revisit the identification results in DAGs, focusing on Tian's algorithm (Tian, 2002). Our goal here is to have an amenable algorithm that allows the incorporation of the structural uncertainties arising in the equivalence class.



Let  $\mathcal{D}_A$  denote the (induced) subgraph of a DAG  $\mathcal{D}(\mathbf{V}, \mathbf{L})$  over  $\mathbf{A} \subseteq \mathbf{V}$  and the latent parents of  $\mathbf{A}$  (i.e.  $\text{Pa}(\mathbf{A}) \cap \mathbf{L}$ ). The original algorithm (Alg. 5 in (Tian, 2002)) alternately applies Lemmas 10 and 11 in (Tian, 2002) until a solution is derived or a failure condition is triggered. We rewrite this algorithm with a more local, atomic criterion based on the following results.

**Definition 3** (Composite C-Component). *Given a DAG that decomposes into c-components  $S_1, \dots, S_k$ ,  $k \geq 1$ , a composite c-component is the union of one or more of these c-components.*

**Lemma 1.** *Given a DAG  $\mathcal{D}(\mathbf{V}, \mathbf{L})$ ,  $\mathbf{X} \subset \mathbf{T} \subseteq \mathbf{V}$ , and  $P_{\mathbf{V} \setminus \mathbf{T}}$  the interventional distribution of  $\mathbf{V} \setminus \mathbf{T}$  on  $\mathbf{T}$ . Let  $S^{\mathbf{X}}$  denote a composite c-component containing  $\mathbf{X}$  in  $\mathcal{D}_{\mathbf{T}}$ . If  $\mathbf{X}$  is a descendant set in  $\mathcal{D}_{S^{\mathbf{X}}}$ , then  $Q[\mathbf{T} \setminus \mathbf{X}]$  is identifiable and given by*

$$Q[\mathbf{T} \setminus \mathbf{X}] = \frac{P_{\mathbf{V} \setminus \mathbf{T}}}{Q[S^{\mathbf{X}}]} \times \sum_{\mathbf{x}} Q[S^{\mathbf{X}}] \quad (1)$$

*Proof.* By (Tian, 2002, Lemma 11),  $Q[\mathbf{T}]$  decomposes as follows.

$$Q[\mathbf{T}] = Q[\mathbf{T} \setminus S^{\mathbf{X}}] \times Q[S^{\mathbf{X}}] = \frac{Q[\mathbf{T}]}{Q[S^{\mathbf{X}}]} \times Q[S^{\mathbf{X}}]$$

$Q[S^{\mathbf{X}}]$  is computable from  $P_{\mathbf{V} \setminus \mathbf{T}}$  using Lemma 11 in (Tian, 2002), and  $Q[S^{\mathbf{X}} \setminus \mathbf{X}]$  is computable from  $Q[S^{\mathbf{X}}]$  using (Tian, 2002, Lemma 10) as  $\mathbf{X}$  is a descendant set in  $\mathcal{D}_{S^{\mathbf{X}}}$ . Therefore,

$$Q[\mathbf{T} \setminus \mathbf{X}] = \frac{P_{\mathbf{V} \setminus \mathbf{T}}}{Q[S^{\mathbf{X}}]} \cdot Q[S^{\mathbf{X}} \setminus \mathbf{X}] = \frac{P_{\mathbf{V} \setminus \mathbf{T}}}{Q[S^{\mathbf{X}}]} \cdot \sum_{\mathbf{x}} Q[S^{\mathbf{X}}] \quad \square$$

The next result follows directly when  $\mathbf{X}$  is a singleton.

**Corollary 1.** *Given a DAG  $\mathcal{D}(\mathbf{V}, \mathbf{L})$ ,  $X \in \mathbf{T} \subseteq \mathbf{V}$ , and  $P_{\mathbf{V} \setminus \mathbf{T}}$ . If  $X$  is not in the same c-component with a child in  $\mathcal{D}_{\mathbf{T}}$ , then  $Q[\mathbf{T} \setminus \{X\}]$  is identifiable and given by*

$$Q[\mathbf{T} \setminus \{X\}] = \frac{P_{\mathbf{V} \setminus \mathbf{T}}}{Q[S^X]} \times \sum_x Q[S^X] \quad (2)$$

where  $S^X$  is the c-component of  $X$  in  $\mathcal{D}_{\mathbf{T}}$ .

The significance of Corol. 1 stems from the fact that it can be used to rewrite the identification algorithm in a step-wise fashion, which is shown in Algorithm 1. The same is equivalent to the original algorithm since neither one of Lemmas 10 nor 11 in (Tian, 2002) is applicable whenever Corol. 1 is not applicable, which is shown by Lemmas 2 and 3. This result may not be surprising since Corol. 1 follows from the application of these lemmas.

---

**Algorithm 1:** ID( $\mathbf{x}, \mathbf{y}$ ) given DAG  $\mathcal{G}$

---

**input** : two disjoint sets  $\mathbf{X}, \mathbf{Y} \subset \mathbf{V}$

**output:** Expression for  $P_{\mathbf{x}}(\mathbf{y})$  or FAIL

1. Let  $\mathbf{D} = \text{An}(\mathbf{Y})_{\mathcal{G}_{\mathbf{V} \setminus \mathbf{X}}}$
2. Let the c-components of  $\mathcal{G}_{\mathbf{D}}$  be  $\mathbf{D}_i, i = 1, \dots, k$
3.  $P_{\mathbf{x}}(\mathbf{y}) = \sum_{\mathbf{d} \setminus \mathbf{y}} \prod_i \text{Identify}(\mathbf{D}_i, \mathbf{V}, P)$

**Function** Identify ( $\mathbf{C}, \mathbf{T}, Q = Q[\mathbf{T}]$ ) :

```

if  $\mathbf{C} = \mathbf{T}$  then
  | return  $Q[\mathbf{T}]$ ;
end
/* Let  $S^B$  be the c-component of  $\{B\}$  in  $\mathcal{G}_{\mathbf{T}}$  */
if  $\exists B \in \mathbf{T} \setminus \mathbf{C}$  such that  $S^B \cap \text{Ch}(B) = \emptyset$  then
  | Compute  $Q[\mathbf{T} \setminus \{B\}]$  from  $Q$ ; // Corollary 1
  | return Identify( $\mathbf{C}, \mathbf{T} \setminus \{B\}, Q[\mathbf{T} \setminus \{B\}]$ );
else
  | throw FAIL;
end

```

---

**Lemma 2.** *Given a DAG  $\mathcal{D}(\mathbf{V}, \mathbf{L})$ ,  $\mathbf{C} \subset \mathbf{T} \subseteq \mathbf{V}$ . If  $\mathbf{A} = \text{An}(\mathbf{C})_{\mathcal{D}_{\mathbf{T}}} \neq \mathbf{T}$ , then there exist some node  $X \in \mathbf{T} \setminus \mathbf{A}$  such that  $X$  is not in the same c-component with any child in  $\mathcal{D}_{\mathbf{T}}$ .*

*Proof.* If  $\mathbf{A} \neq \mathbf{T}$ , then  $\mathbf{T} \setminus \mathbf{A}$  is a non-empty set where none of the nodes is an ancestor of  $\mathbf{A}$ . Since the graph is acyclic, then at least one node of  $\mathbf{T} \setminus \mathbf{A}$  is with no children. Hence, the above conclusion follows.  $\square$

**Lemma 3.** *Given a DAG  $\mathcal{D}(\mathbf{V}, \mathbf{L})$ ,  $\mathbf{C} \subset \mathbf{T} \subseteq \mathbf{V}$ , and assume  $\mathcal{D}_{\mathbf{C}}$  is a single c-component. If  $\mathcal{D}_{\mathbf{T}}$  partitions into c-components  $\mathbf{S}_1 \dots \mathbf{S}_k$ , where  $k > 1$ , then there exists some node  $X \in \mathbf{S}_i$  such that  $\mathbf{C} \not\subseteq \mathbf{S}_i$  and  $X$  is not in the same c-component with any child in  $\mathcal{D}_{\mathbf{T}}$ .*

*Proof.* Subgraph  $\mathcal{D}_{\mathbf{S}_i}$  is acyclic, so there must exist some node ( $X$ ) that doesn't have any children in  $\mathcal{D}_{\mathbf{S}_i}$ . Since  $\mathbf{S}_i$  is one of the c-components in  $\mathcal{D}_{\mathbf{T}}$ , then  $X$  is not in the same c-component with any of its children in  $\mathcal{D}_{\mathbf{T}}$ .  $\square$

The revised algorithm requires checking an atomic criterion at every instance of the recursive routine Identify. This might not be crucial when the precise causal diagram is known and induced subgraphs preserve a complete graphical characterization of the c-components and the ancestral relations between the nodes. The latter, unfortunately, doesn't hold when the model is an equivalence class represented by a PAG.<sup>2</sup>

<sup>2</sup>We thank a reviewer for bringing to our attention a similar formulation of Alg. 1 (Richardson et al., 2017, Thm. 60).

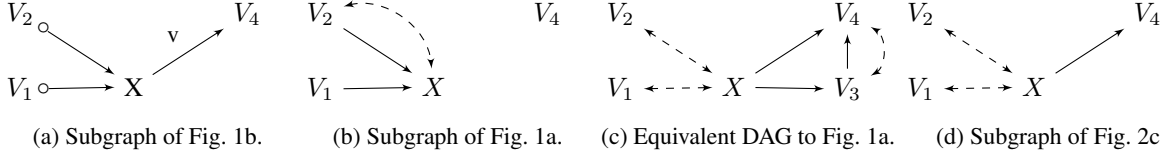


Figure 2: Example for properties discussed in Section 4

## 4 PAG-SUBGRAPH PROPERTIES

Evidently, induced subgraphs of the original causal model play a critical role in identification (cf Alg. 1). It is natural to expect that in the generalized setting we study here, induced subgraphs of the given PAG will also play an important role. An immediate challenge, however, is that a subgraph of a PAG  $\mathcal{P}$  over  $\mathbf{V}$  induced by  $\mathbf{A} \subseteq \mathbf{V}$  is, in general, not a PAG that represents a full Markov equivalence class. In particular, if  $\mathcal{D}(\mathbf{V}, \mathbf{L})$  is a DAG in the equivalence class represented by  $\mathcal{P}$ ,  $\mathcal{P}_{\mathbf{A}}$  is in general not the PAG that represents the equivalence class of  $\mathcal{D}_{\mathbf{A}}$ . To witness, let  $\mathcal{D}$  and  $\mathcal{P}$  denote the DAG and the corresponding PAG in Figure 1, respectively, and let  $\mathbf{A} = \{V_1, V_2, X, V_4\}$ . The induced subgraph of  $\mathcal{P}$  over  $\mathbf{A}$  (Fig. 2a) does not represent the equivalence class of the corresponding induced subgraph of  $\mathcal{D}$  (Fig. 2b). Despite this subtlety, we establish a few facts below showing that for any  $\mathbf{A} \subseteq \mathbf{V}$  and any DAG  $\mathcal{D}$  in the equivalence class represented by  $\mathcal{P}$ , some information about  $\mathcal{D}_{\mathbf{A}}$ , which is particularly relevant to identification, can be read off from  $\mathcal{P}_{\mathbf{A}}$ .

**Proposition 1.** *Let  $\mathcal{P}$  be a PAG over  $\mathbf{V}$ , and  $\mathcal{D}(\mathbf{V}, \mathbf{L})$  be any DAG in the equivalence class represented by  $\mathcal{P}$ . Let  $X \neq Y$  be two nodes in  $\mathbf{A} \subseteq \mathbf{V}$ . If  $X$  is an ancestor of  $Y$  in  $\mathcal{D}_{\mathbf{A}}$ , then  $X$  is a possible ancestor of  $Y$  in  $\mathcal{P}_{\mathbf{A}}$ .*

*Proof.* If  $X$  is an ancestor of  $Y$  in  $\mathcal{D}_{\mathbf{A}}$ , then there is a path  $p$  in  $\mathcal{D}_{\mathbf{A}}$  composed of nodes  $\langle X = V_0, \dots, Y = V_m \rangle$ ,  $m \geq 1$  such that  $V_i \in \mathbf{A}$  and  $V_i \rightarrow V_{i+1}$ ,  $0 \leq i < m$ . Path  $p$  is obviously also present in  $\mathcal{D}$ , and consequently the corresponding MAG  $\mathcal{M}$ . Hence,  $p$  corresponds to a possibly directed path in  $\mathcal{P}$ . Since all the nodes along  $p$  are in  $\mathbf{A}$ , then  $p$  is present in  $\mathcal{P}_{\mathbf{A}}$  and so  $X$  is a possible ancestor of  $Y$  in  $\mathcal{P}_{\mathbf{A}}$ .  $\square$

This simple proposition guarantees that possible-ancestral relationship in  $\mathcal{P}_{\mathbf{A}}$  subsumes ancestral relationship in  $\mathcal{D}_{\mathbf{A}}$  for every  $\mathcal{D}$  in the class represented by  $\mathcal{P}$ . This is illustrated by  $\mathcal{D}_{\mathbf{A}}$  and  $\mathcal{P}_{\mathbf{A}}$  in Figures 2a and 2b.

Given an induced subgraph of a PAG,  $\mathcal{P}_{\mathbf{A}}$ , a directed edge  $X \rightarrow Y$  in  $\mathcal{P}_{\mathbf{A}}$  is said to be *visible* if for every DAG  $\mathcal{D}$  in the class represented by  $\mathcal{P}$ , there is no inducing path in  $\mathcal{D}_{\mathbf{A}}$  between  $X$  and  $Y$  relative to the latent nodes in  $\mathcal{D}_{\mathbf{A}}$  that is into  $X$ .

**Lemma 4.** *Let  $\mathcal{P}$  be a PAG over  $\mathbf{V}$ , and  $\mathcal{P}_{\mathbf{A}}$  be an induced subgraph of  $\mathcal{P}$  over  $\mathbf{A} \subseteq \mathbf{V}$ . For every  $X \rightarrow Y$  in  $\mathcal{P}_{\mathbf{A}}$ , if it is visible in  $\mathcal{P}$ , then it remains visible in  $\mathcal{P}_{\mathbf{A}}$ .*

*Proof.* Let  $\mathcal{D}(\mathbf{V}, \mathbf{L})$  be any causal model in the equivalence class represented by  $\mathcal{P}$ , and let  $X \rightarrow Y$  be a visible edge in  $\mathcal{P}$ ,  $X, Y \in \mathbf{A}$ . Then, there is no inducing path between  $X$  and  $Y$  relative to  $\mathbf{L}$  that is into  $X$  in  $\mathcal{D}$ . It follows that no such inducing path (relative to the latent nodes in  $\mathcal{D}_{\mathbf{A}}$ ) exists in the subgraph  $\mathcal{D}_{\mathbf{A}}$ .  $\square$

Visibility is relevant for identification because it implies absence of confounding, which is the major obstacle to identification. Lemma 4 shows that an edge in an induced subgraph that is visible in the original PAG also implies absence of confounding in the induced subgraphs. Interestingly, note that a directed edge  $X \rightarrow Y$  in  $\mathcal{P}_{\mathbf{A}}$ , visible or not, does not imply that  $X$  is an ancestor of  $Y$  in  $\mathcal{D}_{\mathbf{A}}$  for every  $\mathcal{D}$  in the class represented by  $\mathcal{P}$ . For example,  $X$  is not an ancestor of  $V_4$  in Fig. 2b, even though  $X \rightarrow V_4$  is a visible edge in Fig. 2a.

**Definition 4 (PC-Component).** *In a MAG, a PAG, or any of its induced subgraphs, two nodes  $X$  and  $Y$  are in the same possible c-component (pc-component) if there is a path between the two nodes such that (1) all non-endpoint nodes along the path are colliders, and (2) none of the edges is visible.*

As alluded earlier, a c-component in a causal graph plays a central role in identification. The following proposition establishes a graphical condition in an induced subgraph  $\mathcal{P}_{\mathbf{A}}$  that is necessary for two nodes being in the same c-component in  $\mathcal{D}_{\mathbf{A}}$  for some DAG  $\mathcal{D}$  represented by  $\mathcal{P}$ .

**Proposition 2.** *Let  $\mathcal{P}$  be a PAG over  $\mathbf{V}$ , and  $\mathcal{D}(\mathbf{V}, \mathbf{L})$  be any DAG in the equivalence class represented by  $\mathcal{P}$ . For any  $X, Y \in \mathbf{A} \subseteq \mathbf{V}$ , if  $X$  and  $Y$  are in the same c-component in  $\mathcal{D}_{\mathbf{A}}$ , then  $X$  and  $Y$  are in the same pc-component in  $\mathcal{P}_{\mathbf{A}}$ .*

*Proof Sketch.* If  $X$  and  $Y$  are in the same c-component in  $\mathcal{D}_{\mathbf{A}}$ , then there is a path  $p$  in  $\mathcal{D}_{\mathbf{A}}$  composed of nodes  $\langle X = V_0, \dots, Y = V_m \rangle$ ,  $m \geq 1$ , such that  $V_i \in \mathbf{A}$  and  $V_i \leftarrow L_{i,i+1} \rightarrow V_{i+1}$ ,  $0 \leq i < m$ . We prove that  $X$  and  $Y$  are in the same pc-component in  $\mathcal{M}$ , the MAG of  $\mathcal{D}$  over  $\mathbf{V}$ , due to a path  $p'$  over a subsequence of  $p$ . We then

show that  $X$  and  $Y$  are in the same pc-component in  $\mathcal{P}$ , the PAG of  $\mathcal{M}$ , due to a path  $p^*$  over a subsequence of  $p'$ . Since all the nodes along  $p^*$  are in  $\mathbf{A}$ , then  $p^*$  is present in  $\mathcal{P}_{\mathbf{A}}$ , and so  $X$  and  $Y$  are in the same pc-component in  $\mathcal{P}_{\mathbf{A}}$ . Due to space constraints, the complete proofs are provided in (Jaber et al., 2018a).  $\square$

This result provides a sufficient condition for *not* belonging to the same c-component in any of the relevant causal graphs. In Fig. 2a, for example,  $V_1$  and  $V_4$  or  $X$  and  $V_4$  are not in the same pc-component, which implies by Prop. 2 that they are not in the same c-component in  $\mathcal{D}_{\mathbf{A}}$  for any DAG  $\mathcal{D}$  in the equivalence class represented by the PAG in Fig. 1b.

As a special case of Def. 4, we define the following notion, which will prove useful later on.

**Definition 5** (DC-Component). *In a MAG, a PAG, or any of its induced subgraphs, two nodes  $X$  and  $Y$  are in the same definite c-component (dc-component) if they are connected with a bi-directed path, i.e. a path composed solely of bi-directed edges.*

One challenge with the notion of *pc-component* is that it is not transitive as *c-component* is. Consider the PAG  $V_1 \circ\text{---}\circ V_2 \circ\text{---}\circ V_3$ . Here,  $V_1$  and  $V_2$  are in the same pc-component,  $V_2$  and  $V_3$  are in the same pc-component, however,  $V_1$  and  $V_3$  are not in the same pc-component. Hence, we define a notion that is a transitive closure of the notion of pc-component, which will prove instrumental to our goal.

**Definition 6** (CPC-Component). *Let  $\mathcal{P}$  denote a PAG or a corresponding induced subgraph. Nodes  $X$  and  $Y$  are in the same composite pc-component in  $\mathcal{P}$ , denoted *cpc-component*, if there exist a sequence of nodes  $\langle X = V_0, \dots, Y = V_m \rangle$ ,  $m \geq 1$ , such that  $V_i$  and  $V_{i+1}$  are in the same pc-component,  $0 \leq i < m$ .*

It follows from the above definition that a PAG or an induced subgraph  $\mathcal{P}$  can be decomposed into unique sets of cpc-components. For instance, the cpc-components in Fig. 2a are  $\mathbf{S}_1 = \{V_1, V_2, X\}$  and  $\mathbf{S}_2 = \{V_4\}$ . The significance of a *cpc-component* is that it corresponds to a composite c-component in the relevant causal graphs as shown in the following proposition.

**Proposition 3.** *Let  $\mathcal{P}$  be a PAG over  $\mathbf{V}$ ,  $\mathcal{D}(\mathbf{V}, \mathbf{L})$  be any DAG in the equivalence class represented by  $\mathcal{P}$ , and  $\mathbf{A} \subseteq \mathbf{V}$ . If  $\mathbf{C} \subseteq \mathbf{A}$  is a cpc-component in  $\mathcal{P}_{\mathbf{A}}$ , then  $\mathbf{C}$  is a composite c-component in  $\mathcal{D}_{\mathbf{A}}$ .*

*Proof.* According to Definition 6,  $\mathbf{C}$  includes all the nodes that are in the same pc-component with some node in  $\mathbf{C}$  in  $\mathcal{P}_{\mathbf{A}}$ . It follows from the contrapositive of Prop. 2 that no node outside  $\mathbf{C}$  is in the same c-component with

---

### Algorithm 2: PTO Algorithm

---

**input** : PAG  $\mathcal{P}$  over  $\mathbf{V}$

**output**: PTO over  $\mathcal{P}$

1- Create singleton buckets  $\mathbf{B}_i$  each containing  $V_i \in \mathbf{V}$ .

2- Merge buckets  $\mathbf{B}_i$  and  $\mathbf{B}_j$  if there is a circle edge between them ( $\mathbf{B}_i \ni X \circ\text{---}\circ Y \in \mathbf{B}_j$ ).

3- **while** set of buckets ( $\mathbf{B}$ ) is not empty **do**

    (i) Extract  $\mathbf{B}_i$  with only arrowheads incident on it.

    (ii) Remove edges between  $\mathbf{B}_i$  and other buckets.

**end**

4- The partial order is  $\mathbf{B}_1 < \mathbf{B}_2 < \dots < \mathbf{B}_m$  in reverse order of the bucket extraction. Hence,  $\mathbf{B}_1$  is the last bucket extracted and  $\mathbf{B}_m$  is the first bucket extracted.

---

any node in  $\mathbf{C}$  in  $\mathcal{D}_{\mathbf{A}}$ . Hence, set  $\mathbf{C}$  represents a composite c-component in  $\mathcal{D}_{\mathbf{A}}$  by Definition 3.  $\square$

Recall that the algorithm for identification given a DAG uses a topological order over the nodes. Similarly, the algorithm we design for PAGs will depend on some (partial) topological order. Thanks to the possible presence of circle edges ( $\circ\text{---}\circ$ ) in a PAG, in general, there may be no complete topological order that is valid for all DAGs in the equivalence class. Algorithm 2 presents a procedure to derive a *partial topological order* over the nodes in a PAG, using buckets of nodes that are connected with circle paths (Jaber et al., 2018b). This algorithm remains valid over an induced subgraph of a PAG. To show this, the following lemma is crucial:

**Lemma 5.** *Let  $\mathcal{P}$  be a PAG over  $\mathbf{V}$ , and  $\mathcal{P}_{\mathbf{A}}$  be the induced subgraph over  $\mathbf{A} \subseteq \mathbf{V}$ . For any three nodes  $A, B, C$ , if  $A \ast \rightarrow B \circ\text{---}\ast C$ , then there is an edge between  $A$  and  $C$  with an arrowhead at  $C$ , namely,  $A \ast \rightarrow C$ . Furthermore, if the edge between  $A$  and  $B$  is  $A \rightarrow B$ , then the edge between  $A$  and  $C$  is either  $A \rightarrow C$  or  $A \circ\text{---}\rightarrow C$  (i.e., it is not  $A \leftrightarrow C$ ).*

*Proof.* Lemma 3.3.1 of (Zhang, 2006) establishes the above property for every PAG. By the definition of an induced subgraph, the property is preserved in  $\mathcal{P}_{\mathbf{A}}$ .  $\square$

Thus, a characteristic feature of PAGs carries over to their induced subgraphs. It follows that Algorithm 2 is sound for induced subgraphs as well.

**Proposition 4.** *Let  $\mathcal{P}$  be a PAG over  $\mathbf{V}$ , and let  $\mathcal{P}_{\mathbf{A}}$  be the subgraph of  $\mathcal{P}$  induced by  $\mathbf{A} \subseteq \mathbf{V}$ . Then, Algorithm 2 is sound over  $\mathcal{P}_{\mathbf{A}}$ , in the sense that the partial order is valid with respect to  $\mathcal{D}_{\mathbf{A}}$ , for every DAG  $\mathcal{D}$  in the equivalence class represented by  $\mathcal{P}$ .*

*Proof.* Let  $D$  be any DAG in the equivalence class represented by  $\mathcal{P}$ . By Prop. 1, the possible-ancestral relations in  $\mathcal{P}_A$  subsume those present in  $\mathcal{D}_A$ . Hence, a partial topological order that is valid with respect to  $\mathcal{P}_A$  is valid with respect to  $\mathcal{D}_A$ . The correctness of Alg. 2 with respect to a PAG in (Jaber et al., 2018b) depends only on the property in Lemma 5, a proof of which is given in the Supplementary Materials for completeness. Therefore, thanks to Lemma 5, the algorithm is also sound with respect to an induced subgraph  $\mathcal{P}_A$ .  $\square$

For example, for  $\mathcal{P}_A$  in Fig. 2a, a partial topological order over the nodes is  $V_1 < V_2 < X < V_4$ , which is valid for all the relevant DAGs.

With these results about induced subgraphs of a PAG, we are ready to develop a recursive approach for identification given a PAG, to which we now turn.

## 5 IDENTIFICATION IN PAGS

We start by formally defining the notion of identification given a PAG, which generalizes the model-specific notion (Pearl, 2000, pp. 70).

**Definition 7.** Given a PAG  $\mathcal{P}$  over  $\mathbf{V}$  and a query  $P_{\mathbf{x}}(\mathbf{y})$  where  $\mathbf{X}, \mathbf{Y} \subset \mathbf{V}$ ,  $P_{\mathbf{x}}(\mathbf{y})$  is identifiable given  $\mathcal{P}$  if and only if  $P_{\mathbf{x}}(\mathbf{y})$  is identifiable given every DAG  $\mathcal{D}(\mathbf{V}, \mathbf{L})$  in the Markov equivalence class represented by  $\mathcal{P}$ , and with the same expression.

We first derive an atomic identification criterion analogous to Corollary 1. As seen in the algorithm for constructing a partial order (Alg. 2), a bucket or circle component in a PAG is for our purpose analogous to a single node in a DAG. Therefore, the following criterion targets a bucket  $\mathbf{X}$  rather than a single node.

**Theorem 1.** Given a PAG  $\mathcal{P}$  over  $\mathbf{V}$ , a partial topological order  $\mathbf{B}_1 < \dots < \mathbf{B}_m$  with respect to  $\mathcal{P}$ , a bucket  $\mathbf{X} = \mathbf{B}_j \subset \mathbf{T} \subseteq \mathbf{V}$ , for some  $1 \leq j \leq m$ , and  $P_{\mathbf{v} \setminus \mathbf{t}}$  (i.e.  $Q[\mathbf{T}]$ ),  $Q[\mathbf{T} \setminus \mathbf{X}]$  is identifiable if and only if there does not exist  $X \in \mathbf{X}$  such that  $X$  has a possible child  $C \notin \mathbf{X}$  that is in the same pc-component as  $X$  in  $\mathcal{P}_T$ . If identifiable, then the expression is given by

$$Q[\mathbf{T} \setminus \mathbf{X}] = \frac{P_{\mathbf{v} \setminus \mathbf{t}}}{\prod_{\{i | \mathbf{B}_i \subseteq S^{\mathbf{X}}\}} P_{\mathbf{v} \setminus \mathbf{t}}(\mathbf{B}_i | \mathbf{B}^{(i-1)})} \times \sum_{\mathbf{x}} \prod_{\{i | \mathbf{B}_i \subseteq S^{\mathbf{X}}\}} P_{\mathbf{v} \setminus \mathbf{t}}(\mathbf{B}_i | \mathbf{B}^{(i-1)}), \quad (3)$$

where  $S^{\mathbf{X}} = \bigcup_{X \in \mathbf{X}} S^X$ ,  $S^X$  being the dc-component of  $X$  in  $\mathcal{P}_T$ , and  $\mathbf{B}^{(i-1)}$  denoting the set of nodes preceding bucket  $\mathbf{B}_i$  in the partial order.

*Proof Sketch.* (if) Let  $D$  be any DAG in the equivalence class represented by  $\mathcal{P}$ ,  $\mathcal{D}_T$  be the induced subgraph over  $\mathbf{T}$ , and  $S'$  be the smallest composite c-component containing  $\mathbf{X}$  in  $\mathcal{D}_T$ . We show that  $\mathbf{X}$  is a descendant set in  $\mathcal{D}_{S'}$ . Suppose otherwise for the sake of contradiction. Then, there is a node  $C \in S' \setminus \mathbf{X}$  such that  $C$  is a child of  $X_i$  and is in the same c-component with  $X_j$ , where  $X_i, X_j \in \mathbf{X}$  and possibly  $i = j$ . By Prop. 2,  $X_j$  is in the same pc-component with  $C$  in  $\mathcal{P}_T$ . Let  $T_i$  be the node closest to  $X_j$  along the collider path in  $\mathcal{P}_T$  between  $X_j$  and  $C$  consistent with Def. 4. If the edge between  $X_j$  and  $T_i$  in  $\mathcal{P}_T$  is not into  $X_j$ , then  $X_j$  is in the same pc-component with a possible child as the edge is not visible. This violates the criterion stated in the theorem. Otherwise, the edge is  $X_j \leftrightarrow T_i$  and there exist a bi-directed edge between  $T_i$  and every node in  $\mathbf{X}$  (which follows from Lemma 5). Hence,  $X_i$  is in the same pc-component with a possible child  $C$  in  $\mathcal{P}_T$  (Prop. 1), and the criterion stated in the theorem is violated again. Therefore,  $\mathbf{X}$  is a descendant set in  $\mathcal{D}_{S'}$  and  $Q[\mathbf{T} \setminus \mathbf{X}]$  is identifiable from  $Q[\mathbf{T}]$  by Lemma 1. It remains to show that Eq. 3 is equivalent to Eq. 1 for  $\mathcal{D}$ . The details for this step are left to the Supplementary Material.

(only if) Suppose the criterion in question is not satisfied. Then some  $X_i \in \mathbf{X}$  is in the pc-component with a possible child  $C \notin \mathbf{X}$  in  $\mathcal{P}_T$ . The edge between  $X_i$  and  $C$  is  $X_i \ast \rightarrow C$  as  $C$  is outside of  $\mathbf{X}$ . If the edge is not visible in  $\mathcal{P}_T$ , then this edge is not visible in  $\mathcal{P}$  (Lemma 4). Hence, we can construct a DAG  $\mathcal{D}$  in the equivalence class of  $\mathcal{P}$  where  $C$  is a child of  $X_i$  and the two nodes share a latent variable. The pair of sets  $\mathbf{F} = \{X_i, C\}$  and  $\mathbf{F}' = \{C\}$  form a so-called hedge for  $Q[\mathbf{T} \setminus \mathbf{X}]$  and the effect is not identifiable in  $\mathcal{D}$  (Shpitser and Pearl, 2006, Theorem 4), and hence not identifiable given  $\mathcal{P}$ .

Otherwise,  $X_i \rightarrow C$  is visible in  $\mathcal{P}_T$ . So, there is a collider path between  $X_i$  and  $C$  consistent with Def. 4 such that the two nodes are in the same pc-component. Let  $p = \langle X_i = T_0, T_1, \dots, T_m = C \rangle$  denote the shortest such path in  $\mathcal{P}_T$ . If the edge between  $X_i$  and  $T_1$  is not into  $X_i$ , then  $T_1$  is a child of  $X_i$  and the proof follows as in the previous case. Otherwise, we have  $X_i \leftrightarrow T_1$  and we can show that  $X_i$  is the only node along  $p$  that belongs to  $\mathbf{X}$  (details in the Supplementary Material). In  $\mathcal{P}$ , path  $p$  is present with  $X_i \rightarrow C$  visible. Hence, we can construct a DAG  $\mathcal{D}$  in the equivalence class of  $\mathcal{P}$  such that  $C$  is a child of  $X_i$  and both are in the same c-component through a sequence of bi-directed edges along the corresponding nodes of  $p$ . The pair of sets  $\mathbf{F} = \{X_i, T_1, \dots, T_m = C\}$  and  $\mathbf{F}' = \{T_1, \dots, T_m = C\}$  form a hedge for  $Q[\mathbf{T} \setminus \mathbf{X}]$  and the effect is not identifiable in  $\mathcal{D}$ , and hence it is not identifiable given  $\mathcal{P}$ .  $\square$

Note that the above result simplifies into computing the

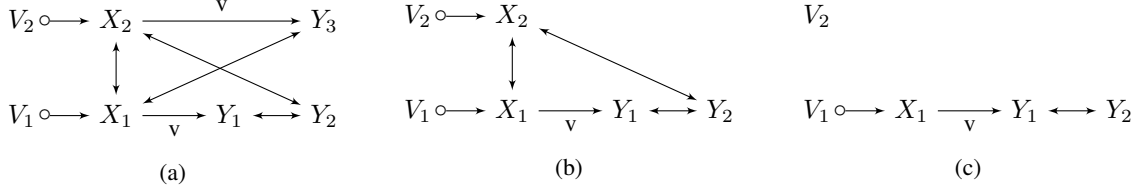


Figure 3: Sample PAG  $\mathcal{P}$  (left) and induced subgraphs used to identify  $Q[\{Y_1, Y_2\}]$ .

interventional distribution  $P_{\mathbf{x}}$  whenever the input distribution is the observational distribution, i.e.  $\mathbf{T} = \mathbf{V}$ . Consider the query  $P_{\mathbf{x}}(\mathbf{v} \setminus \{x\})$  over the PAG in Fig. 1b. The intervention node  $X$  is not in the same pc-component with any of its possible children ( $V_3$  and  $V_4$ ), hence the effect is identifiable and given by

$$\begin{aligned} P_{\mathbf{x}}(\mathbf{v} \setminus \{x\}) &= \frac{P(\mathbf{v})}{P(x|v_1, v_2)} \times \sum_{x'} P(x'|v_1, v_2) \\ &= P(v_1, v_2)P(v_4, v_5|v_1, v_2, x) \end{aligned}$$

Putting these observations together leads to the procedure we call **IDP**, which is shown in Alg. 3. In words, the main idea of **IDP** goes as follows. After receiving the sets  $\mathbf{X}, \mathbf{Y}$ , and a PAG  $\mathcal{P}$ , the algorithm starts the pre-processing steps: First, it computes  $\mathbf{D}$ , the set of possible ancestors of  $\mathbf{Y}$  in  $\mathcal{P}_{\mathbf{V} \setminus \mathbf{X}}$ . Second, it uses  $\mathcal{P}_{\mathbf{D}}$  to partition set  $\mathbf{D}$  into cpc-components. Following the pre-processing stage, the procedure calls the subroutine **Identify** over each cpc-component  $\mathbf{D}_i$  to compute  $Q[\mathbf{D}_i]$  from the observational distribution  $P(\mathbf{V})$ . The recursive routine basically checks for the presence of a bucket  $\mathbf{B}$  in  $\mathcal{P}_{\mathbf{T}}$  that is a subset of the intervention nodes, i.e.  $\mathbf{B} \subseteq \mathbf{T} \setminus \mathbf{C}$ , and satisfies the conditions of Thm. 1. If found, it is able to successfully compute  $Q[\mathbf{T} \setminus \mathbf{B}]$  using Eq. 3, and proceed with a recursive call. Alternatively, if such a bucket doesn't exist in  $\mathcal{P}_{\mathbf{T}}$ , then **IDP** throws a failure condition, since it's unable to identify the query. We show next that this procedure is, indeed, correct.

**Theorem 2.** *Algorithm IDP (Alg.3) is sound.*

*Proof.* Let  $\mathcal{G}(\mathbf{V}, \mathbf{L})$  be any causal graph in the equivalence class of PAG  $\mathcal{P}$  over  $\mathbf{V}$ , and let  $\mathbf{V}' = \mathbf{V} \setminus \mathbf{X}$ . We have

$$P_{\mathbf{x}}(\mathbf{y}) = \sum_{\mathbf{v}' \setminus \mathbf{y}} P_{\mathbf{x}}(\mathbf{v}') = \sum_{\mathbf{v}' \setminus \mathbf{y}} Q[\mathbf{V}'] = \sum_{\mathbf{v}' \setminus \mathbf{d}} \sum_{\mathbf{d} \setminus \mathbf{y}} Q[\mathbf{V}']$$

By definition,  $\mathbf{D}$  is an ancestral set in  $\mathcal{P}_{\mathbf{V}'}$ , and hence it is ancestral in  $\mathcal{G}_{\mathbf{V}'}$  by Prop. 1. So, we have the following by (Tian, 2002, Lemma 10):

$$P_{\mathbf{x}}(\mathbf{y}) = \sum_{\mathbf{d} \setminus \mathbf{y}} \sum_{\mathbf{v}' \setminus \mathbf{d}} Q[\mathbf{V}'] = \sum_{\mathbf{d} \setminus \mathbf{y}} Q[\mathbf{D}] \quad (4)$$

---

**Algorithm 3: IDP**( $\mathbf{x}, \mathbf{y}$ ) given PAG  $\mathcal{P}$

---

**input** : two disjoint sets  $\mathbf{X}, \mathbf{Y} \subset \mathbf{V}$

**output**: Expression for  $P_{\mathbf{x}}(\mathbf{y})$  or FAIL

1. Let  $\mathbf{D} = \text{An}(\mathbf{Y})_{\mathcal{P}_{\mathbf{V} \setminus \mathbf{X}}}$
2. Let the cpc-components of  $\mathcal{P}_{\mathbf{D}}$  be  $\mathbf{D}_i, i = 1, \dots, k$
3.  $P_{\mathbf{x}}(\mathbf{y}) = \sum_{\mathbf{d} \setminus \mathbf{y}} \prod_i \text{Identify}(\mathbf{D}_i, \mathbf{V}, P)$

**Function Identify**( $\mathbf{C}, \mathbf{T}, Q = Q[\mathbf{T}]$ ):

```

if  $\mathbf{C} = \mathbf{T}$  then
  | return  $Q[\mathbf{T}]$ ;
end
/* In  $\mathcal{P}_{\mathbf{T}}$ , let  $\mathbf{B}$  be a bucket, and  $\mathbf{C}^{\mathbf{B}}$  be the
pc-component of  $\mathbf{B}$  */
if  $\exists \mathbf{B} \subseteq \mathbf{T} \setminus \mathbf{C}$  such that  $\mathbf{C}^{\mathbf{B}} \cap \text{Ch}(\mathbf{B}) \subseteq \mathbf{B}$  then
  | Compute  $Q[\mathbf{T} \setminus \mathbf{B}]$  from  $Q$ ; // Theorem 1
  | return  $\text{Identify}(\mathbf{C}, \mathbf{T} \setminus \mathbf{B}, Q[\mathbf{T} \setminus \mathbf{B}])$ ;
else
  | throw FAIL;
end

```

---

Using Prop. 3, each cpc-component in  $\mathcal{P}_{\mathbf{D}}$  corresponds to a composite c-component in  $\mathcal{G}_{\mathbf{D}}$ . Hence, Eq. 4 can be decomposed as follows by (Tian, 2002, Lemma 11).

$$P_{\mathbf{x}}(\mathbf{y}) = \sum_{\mathbf{d} \setminus \mathbf{y}} Q[\mathbf{D}] = \sum_{\mathbf{d} \setminus \mathbf{y}} \prod_i Q[\mathbf{D}_i] \quad (5)$$

Eq. 5 is equivalent to the decomposition we have in step 3 of Alg. 3, where we attempt to compute each  $Q[\mathbf{D}_i]$  from  $P$ . Finally, the correctness of the recursive routine **Identify** follows from that of Theorem 1.  $\square$

## 5.1 ILLUSTRATIVE EXAMPLE

Consider the query  $P_{x_1, x_2}(y_1, y_2, y_3)$  given  $\mathcal{P}$  in Fig. 3a. We have  $\mathbf{D} = \{Y_1, Y_2, Y_3\}$ , and the cpc-components in  $\mathcal{P}_{\mathbf{D}}$  are  $\mathbf{D}_1 = \{Y_1, Y_2\}$  and  $\mathbf{D}_2 = \{Y_3\}$ . Hence, the problem reduces to computing  $Q[\{Y_1, Y_2\}] \cdot Q[\{Y_3\}]$ .

We start with the call **Identify**( $\mathbf{D}_1, \mathbf{V}, P$ ). Consider the singleton bucket  $Y_3$  the pc-component of which includes all the nodes in  $\mathcal{P}$ . This node satisfies the condition in **Identify** as it has no children, and we compute

$Q[\mathbf{V} \setminus \{Y_3\}]$  using Theorem 1.

$$\begin{aligned} Q[\mathbf{V} \setminus \{Y_3\}] &= \frac{P(\mathbf{v})}{P(y_1, y_2, y_3, x_1, x_2 | v_1, v_2)} \times \\ &\quad \sum_{y_3} P(y_1, y_2, y_3, x_1, x_2 | v_1, v_2) \\ &= P(v_1, v_2) \cdot P(y_1, y_2, x_1, x_2 | v_1, v_2) \\ &= P(y_1, y_2, x_1, x_2, v_1, v_2) \end{aligned} \quad (6)$$

In the next recursive call,  $\mathbf{T}_1 = \mathbf{V} \setminus \{Y_3\}$ ,  $P_{y_3}$  corresponds to Eq. 6, and the induced subgraph  $\mathcal{P}_{\mathbf{T}_1}$  is shown in Fig. 3b. Now,  $X_2$  satisfies the criterion and we can compute  $Q[\mathbf{T}_1 \setminus \{X_2\}]$  from  $P_{y_3} = Q[\mathbf{T}_1]$ , i.e.,

$$\begin{aligned} Q[\mathbf{T}_1 \setminus \{X_2\}] &= \frac{P_{y_3}}{P_{y_3}(y_1, y_2, x_1, x_2 | v_1, v_2)} \times \\ &\quad \sum_{x_2} P_{y_3}(y_1, y_2, x_1, x_2 | v_1, v_2) \\ &= P(y_1, y_2, x_1, v_1, v_2) \end{aligned} \quad (7)$$

Let  $\mathbf{T}_2 = \mathbf{T}_1 \setminus \{X_2\}$ , where the induced subgraph  $\mathcal{P}_{\mathbf{T}_2}$  is shown in Fig. 3c. Now,  $X_1$  satisfies the criterion and we can compute  $Q[\mathbf{T}_2 \setminus \{X_1\}]$  from Eq. 7,

$$\begin{aligned} Q[\mathbf{T}_2 \setminus \{X_1\}] &= \frac{P_{y_3, x_2}}{P_{y_3, x_2}(x_1 | v_1, v_2)} \times \\ &\quad \sum_{x_1} P_{y_3, x_2}(x_1 | v_1, v_2) \\ &= \frac{P(v_1, v_2) \cdot P(y_1, y_2, x_1, v_1, v_2)}{P(x_1, v_1, v_2)} \\ &= P(v_1, v_2) \cdot P(y_1, y_2 | x_1, v_1, v_2) \end{aligned}$$

Choosing  $V_1$  and  $V_2$  in the next two recursive calls, we finally obtain the simplified expression:

$$Q[\{Y_1, Y_2\}] = P(y_1, y_2 | x_1)$$

Next, we solve for  $Q[\mathbf{D}_2]$  and we get an expression analogous to that of  $Q[\mathbf{D}_1]$ . Hence, the final solution is:

$$P_{x_1, x_2}(y_1, y_2, y_3) = P(y_1, y_2 | x_1) \times P(y_3 | x_2)$$

## 5.2 COMPARISON TO STATE OF THE ART

In the previous section, we formulated an identification algorithm in PAGs for causal queries of the form  $P_{\mathbf{x}}(\mathbf{y})$ ,  $\mathbf{X}, \mathbf{Y} \subset \mathbf{V}$ . A natural question arises about the expressiveness of the **IDP** in comparison with the state-of-the-art methods. One of the well established results in the literature is the adjustment method (Perković et al., 2015), which is complete whenever an adjustment set exists.

In the sequel, we formally show that the proposed algorithm subsumes the adjustment method.

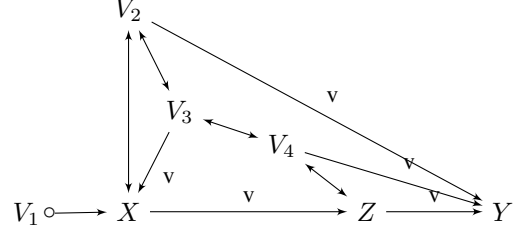


Figure 4: Query  $P_{\mathbf{x}}(\mathbf{y})$  is identifiable by **IDP**.

**Theorem 3.** Let  $\mathcal{P}$  be a PAG over set  $\mathbf{V}$  and let  $P_{\mathbf{x}}(\mathbf{y})$  be a causal query where  $\mathbf{X}, \mathbf{Y} \subset \mathbf{V}$ . If the distribution  $P_{\mathbf{x}}(\mathbf{y})$  is not identifiable using **IDP** (Alg. 3), then the effect is not identifiable using the generalized adjustment criterion in (Perković et al., 2015).

*Proof Sketch.* Whenever **IDP** fails to identify some query, it is due to one of the recursive calls to `Identify`. We use the failing condition inside this call to systematically identify a proper definite status non-causal path from  $\mathbf{X}$  to  $\mathbf{Y}$  in  $\mathcal{P}$  that is m-connecting given set  $\text{Adjust}(\mathbf{X}, \mathbf{Y}, \mathcal{P})$  (Perković et al., 2016, Def. 4.1). As this set fails to satisfy the adjustment criterion, then there exist no adjustment set relative to the pair  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{P}$  (Perković et al., 2016, Cor. 4.4). The details of the proof are left to the Supplementary Material.  $\square$

Based on this result, one may wonder whether these algorithms are, after all, just equivalent. In reality, **IDP** captures strictly more identifiable effects than the adjustment criterion. To witness, consider the PAG in Fig. 4 and note that the causal distribution  $P_{\mathbf{x}}(\mathbf{y})$  is identifiable by **IDP** but not by adjustment in this case.

## 6 CONCLUSION

We studied the problem of identification of interventional distributions in Markov equivalence classes represented by PAGs. We first investigated graphical properties for induced subgraphs of PAGs over an arbitrary subset of nodes with respect to induced subgraphs of DAGs that are in the equivalence class. We believe that these results can be useful to general tasks related to causal inference from equivalence classes. We further developed an identification algorithm in PAGs and proved it to subsume the state-of-the-art adjustment method.

## Acknowledgments

We thank Sanghack Lee and the reviewers for all the feedback provided. Bareinboim and Jaber are supported in parts by grants from NSF IIS-1704352 and IIS-1750807 (CAREER). Zhang is supported in part by the Research Grants Council of Hong Kong under the General Research Fund LU13600715.

## References

- Elias Bareinboim and Judea Pearl. Causal inference by surrogate experiments: z-identifiability. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, 2012.
- Elias Bareinboim and Judea Pearl. Causal inference and the data-fusion problem. *Proc. Natl. Acad. Sci.*, 113: 7345–7352, 2016.
- D. Galles and J. Pearl. Testing identifiability of causal effects. In P. Besnard and S. Hanks, editors, *Uncertainty in Artificial Intelligence 11*, pages 185–195. Morgan Kaufmann, San Francisco, 1995.
- Yimin Huang and Marco Valtorta. Pearl’s calculus of intervention is complete. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, UAI’06, pages 217–224. AUAI Press, 2006.
- Antti Hyttinen, Frederick Eberhardt, and Matti Järvisalo. Do-calculus when the true graph is unknown. In *UAI*, pages 395–404, 2015.
- Amin Jaber, Jiji Zhang, and Elias Bareinboim. Causal identification under Markov equivalence. Technical report, R-35, Purdue AI Lab, Department of Computer Science, Purdue University, 2018a.
- Amin Jaber, Jiji Zhang, and Elias Bareinboim. A graphical criterion for effect identification in equivalence classes of causal diagrams. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI’18, 2018b.
- Manabu Kuroki and Masami Miyakawa. Identifiability criteria for causal effects of joint interventions. *Journal of the Japan Statistical Society*, 29(2):105–117, 1999.
- Marloes H. Maathuis, Diego Colombo, Markus Kalisch, and Peter Bühlmann. Predicting causal effects in large-scale systems from observational data. *Nature Methods*, 7(4):247–248, 2010.
- J. Pearl. Aspects of graphical models connected with causality. In *Proceedings of the 49th Session of the International Statistical Institute*, pages 391–401, Tome LV, Book 1, Florence, Italy, 1993.
- J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, New York, 2000. 2nd edition, 2009.
- Judea Pearl. Causal diagrams for empirical research. *Biometrika*, 82(4):669–688, 1995.
- Judea Pearl and Dana Mackenzie. *The Book of Why: The New Science of Cause and Effect*. Basic Books, 2018. forthcoming.
- Emilija Perković, Johannes Textor, Markus Kalisch, and Marloes H. Maathuis. A complete generalized adjustment criterion. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, pages 682–691, 2015.
- Emilija Perković, Johannes Textor, Markus Kalisch, and Marloes H. Maathuis. Complete graphical characterization and construction of adjustment sets in Markov equivalence classes of ancestral graphs. *arXiv preprint arXiv:1606.06903*, 2016.
- Thomas Richardson and Peter Spirtes. Ancestral graph Markov models. *Annals of Statistics*, pages 962–1030, 2002.
- Thomas S Richardson, Robin J Evans, James M Robins, and Ilya Shpitser. Nested Markov properties for acyclic directed mixed graphs. *arXiv preprint arXiv:1701.06686*, 2017.
- Ilya Shpitser and Judea Pearl. Identification of joint interventional distributions in recursive semi-Markovian causal models. In *Proceedings of the National Conference on Artificial Intelligence*, volume 21, page 1219. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.
- Peter Spirtes, Clark N Glymour, and Richard Scheines. *Causation, prediction, and search*, volume 81. MIT press, 2001.
- Jin Tian. *Studies in causal reasoning and learning*. PhD thesis, University of California, Los Angeles, 2002.
- Jin Tian and Judea Pearl. A general identification condition for causal effects. In *AAAI/IAAI*, pages 567–573, 2002.
- TS Verma. Graphical aspects of causal models. *Technical Report R-191, UCLA*, 1993.
- Jiji Zhang. *Causal inference and reasoning in causally insufficient systems*. PhD thesis, Carnegie Mellon University, 2006.
- Jiji Zhang. Generalized do-calculus with testable causal assumptions. In *International Conference on Artificial Intelligence and Statistics*, pages 667–674, 2007.
- Jiji Zhang. Causal reasoning with ancestral graphs. *Journal of Machine Learning Research*, 9(Jul):1437–1474, 2008a.
- Jiji Zhang. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. *Artificial Intelligence*, 172(16): 1873–1896, 2008b.

---

# The Variational Homoencoder: Learning to learn high capacity generative models from few examples

---

Luke B. Hewitt<sup>12</sup>   Maxwell I. Nye<sup>1</sup>   Andreea Gane<sup>1</sup>   Tommi Jaakkola<sup>1</sup>   Joshua B. Tenenbaum<sup>1</sup>

<sup>1</sup>Massachusetts Institute of Technology

<sup>2</sup>MIT-IBM Watson AI Lab

## Abstract

Hierarchical Bayesian methods can unify many related tasks (e.g.  $k$ -shot classification, conditional and unconditional generation) as inference within a single generative model. However, when this generative model is expressed as a powerful neural network such as a PixelCNN, we show that existing learning techniques typically fail to effectively use latent variables. To address this, we develop a modification of the Variational Autoencoder in which encoded observations are decoded to new elements from the same class. This technique, which we call a *Variational Homoencoder* (VHE), produces a hierarchical latent variable model which better utilises latent variables. We use the VHE framework to learn a hierarchical PixelCNN on the Omniglot dataset, which outperforms all existing models on test set likelihood and achieves strong performance on one-shot generation and classification tasks. We additionally validate the VHE on natural images from the YouTube Faces database. Finally, we develop extensions of the model that apply to richer dataset structures such as factorial and hierarchical categories.

## 1 INTRODUCTION

Learning from few examples is possible only with strong inductive biases. In machine learning these biases can be hand designed, such as a model’s parametrisation, or can be the result of a meta-learning algorithm. Furthermore they may be task-specific, as in discriminative modelling, or may describe the world causally so as to be naturally

reused across many tasks. Recent work has approached one- and few-shot learning from all of these perspectives.

Much research has focused on developing neural architectures for few-shot classification (Koch, 2015; Vinyals et al., 2016; Snell et al., 2017; Santoro et al., 2016). These discriminatively-trained networks take as input a test example and a ‘support set’ of examples from several novel classes, and determine the most likely classification of the test example within the novel classes. A second approach, as explored in Ravi & Larochelle (2016); Finn et al. (2017), is to use only a standard classification network but adapt its parameters to the support examples with a learned initialisation and update rule. In either case, such discriminative models can achieve state-of-the-art few-shot classification performance, although they provide no principled means for transferring knowledge to other tasks.

An alternative approach centers on few-shot learning of *generative* models, from which good classification ought to come for free. Much recent work on meta-learning aims to take one or a few observations from a set  $D$  as input, and produce a distribution over new elements  $p(x|D)$  by some learning procedure, expressed either as a neural network (Rezende et al., 2016; Bartunov & Vetrov, 2016; Reed et al., 2017) or by adapting the parameters of an unconditional model (Reed et al., 2017).

A promising route to learning generative models is hierarchical Bayesian inference, which aims to capture shared structure between instances through *shared* latent variables. A recent example is developed in Lake et al. (2015): a compositional, causal generative model of handwritten characters which achieves state-of-the-art results at few-shot character classification, alphabet classification, and both conditional and unconditional generation. However, this model was hand engineered for the Omniglot domain, and so leaves open the challenge of how to learn such hierarchical Bayesian models using only a generic architecture. The recently proposed *Neu-*

---

A PyTorch implementation of the Variational Homoencoder can be found at [github.com/insperatum/vhe](https://github.com/insperatum/vhe).



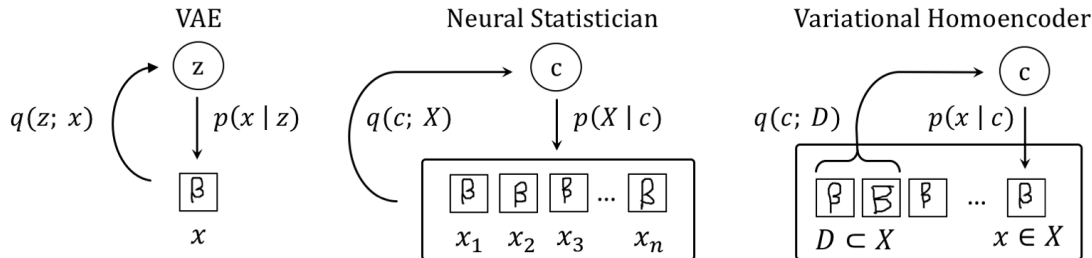


Figure 1: Single step of gradient training in various models. A VAE treats all datapoints as independent, so only a single random element need be encoded (with  $q(z; x)$ ) and decoded (with  $p(x|z)$ ) each step. A Neural Statistician instead feeds a full set of elements  $X$  through both encoder ( $q(c; X)$ ) and decoder ( $p(X|c)$ ) networks, in order to share a latent variable  $c$ . In a VHE, we bound the full likelihood  $p(X)$  using only random subsamples  $D$  and  $x$  for encoding/decoding. Optionally, the decoder  $p(x|c)$  may be defined through a local latent variable  $z$ .

*ral Statistician* (Edwards & Storkey, 2016) offers one means towards this, using amortised variational inference to support learning in a deep hierarchical generative model.

In this work we aim to learn generative models, expressed using high capacity neural network architectures, from just a few examples of a concept. To this end we propose the *Variational Homoencoder* (VHE), combining several advantages of the models described above:

1. Like conditional generative approaches (e.g. Rezende et al. (2016)), we train on a few-shot generation objective which matches how our model may be used at test time. However, by introducing an encoding cost, we simultaneously optimise a likelihood lower bound for a hierarchical generative model, in which structure shared across elements is made explicit by shared latent variables.
2. Edwards & Storkey (2016) has learned hierarchical Bayesian models by applying Variational Autoencoders to sets, such as classes of images. However, their approach requires feeding a full set through the model per gradient step (Figure 1), rendering it intractable to train on very large sets. In practice, they avoid computational limits by training on smaller, random subsets. In a VHE, we instead optimise a likelihood bound for the complete dataset, while *constructing this bound* by subsampling. This approach can not only improve generalisation, but also departs from previous work by extending to models with richer latent structure, for which the joint likelihood cannot be factorised.
3. As with a VAE, the VHE objective includes both an *encoding-* and *reconstruction-* cost. However, by sharing latent variables across a large set of elements, the *encoding cost* per element is reduced

significantly. This facilitates use of powerful autoregressive decoders, which otherwise often suffer from ignoring latent variables (Chen et al., 2016). We demonstrate the significance of this by applying a VHE to the Omniglot dataset. Using a PixelCNN decoder (Oord et al., 2016), our generative model is arguably the first with a general purpose architecture to both attain near state-of-the-art one-shot classification performance and produce high quality samples in one-shot generation.

## 2 BACKGROUND

### 2.1 VARIATIONAL AUTOENCODERS

When dealing with latent variable models of the form  $p(x) = \int_z p(z)p(x|z)dz$ , the integration is necessary for both learning and inference but is often intractable to compute in closed form. *Variational Autoencoders* (VAEs, Kingma & Welling (2013)) provide a method for learning such models by utilising neural-network based approximate posterior inference. Specifically, a VAE comprises a generative network  $p_\theta(z)p_\theta(x|z)$  alongside a separate inference network  $q_\phi(z; x)$ . These are trained jointly to maximise a single objective:

$$\begin{aligned} \mathcal{L}_X(\theta, \phi) = & \sum_{x \in X} \left[ \log p_\theta(x) - \mathbf{D}_{KL}(q_\phi(z; x) \parallel p_\theta(z|x)) \right] \quad (1) \\ = & \sum_{x \in X} \left[ \mathbb{E}_{q_\phi(z; x)} \log p_\theta(x|z) - \mathbf{D}_{KL}(q_\phi(z; x) \parallel p_\theta(z)) \right] \quad (2) \end{aligned}$$

As can be seen from Equation 1, this objective  $\mathcal{L}_X$  is a lower bound on the total log likelihood of the dataset

---

**Algorithm 1:** Minibatch training for the *Variational Homoencoder*. Minibatches are of size  $M$ . Stochastic inference network  $q$  uses subsets of size  $N$ .

---

initialize $(\theta, \phi)$ <b>repeat</b> sample $(x_k, i_k)$ for $k = 1, \dots, M$ sample $D_k \subseteq X_{i_k}$ for $k = 1, \dots, M$ sample $c_k \sim q_\phi(c; D_k)$ for $k = 1, \dots, M$ (optional) sample $z_k \sim q_\phi(z; c_k, x_k)$ for $k = 1, \dots, M$ $\mathbf{g} \approx \frac{1}{M} \sum_k \nabla \mathcal{L}_{\theta, \phi}(x_k; D_k,  X_{i_k} )$ $(\theta, \phi) \leftarrow (\theta, \phi) + \lambda \mathbf{g}$ <b>until</b> convergence of $(\theta, \phi)$	<i>Parameters for decoder <math>p</math> and encoder <math>q</math></i>  <i>Minibatch of elements with corresponding class labels where <math> D_k  = N</math></i>  <i>Reparametrization gradient estimate using <math>\mathbf{c}, \mathbf{z}</math></i> <i>Gradient step, e.g. SGD</i>
--	--

---

$\sum_{x \in X} \log p_\theta(x)$ , while  $q_\phi(z; x)$  is trained to approximate the true posterior  $p_\theta(z|x)$  as accurately as possible. If it could match this distribution exactly then the bound would be tight so that the VAE objective equals the true log likelihood of the data. In practice, the resulting model is typically a compromise between two goals: pulling  $p_\theta$  towards a distribution that assigns high likelihood to the data, but also towards one which allows accurate inference by  $q_\phi$ . Equation 2 provides a formulation for the same objective which can be optimised stochastically, using Monte-Carlo integration to approximate the expectation. For brevity, we will omit subscripts  $\theta, \phi$  for the remainder of this paper.

## 2.2 VARIATIONAL AUTOENCODERS OVER SETS

The *Neural Statistician* (Edwards & Storkey, 2016) is a Variational Autoencoder in which each item to be encoded is itself a set, such as the set  $X^{(i)}$  of all images with a particular class label  $i$ :

$$X^{(i)} = \{x_1^{(i)}, x_2^{(i)}, \dots, x_n^{(i)}\} \quad (3)$$

The generative model for sets,  $p(X)$ , is described by introduction of a corresponding latent variable  $c$ . Given  $c$ , individual  $x \in X$  are conditionally independent:

$$p(X) = \int_c p(c) \prod_{x \in X} p(x|c) dc \quad (4)$$

The likelihood is again intractable to compute, but it can be bounded below via:

$$\log p(X) \geq \mathcal{L}_X = \mathbb{E}_{q(c; X)} \left[ \sum_{x \in X} \log p(x|c) \right] - \mathbf{D}_{KL}(q(c; X) \| p(c)) \quad (5)$$

Unfortunately, calculating the variational lower bound for each set  $X$  requires evaluating both  $q(c; X)$  and  $p(X|c)$ , meaning that the entire set must be passed

through both networks for each gradient update. This can become computationally challenging for classes with hundreds of examples. Instead, previous work (Edwards & Storkey, 2016) ensures that sets used for training are always of small size by maximising a log-likelihood bound for randomly sampled subsets  $D \subset X$ :

$$\mathbb{E}_{D \subset X} \left[ \mathbb{E}_{q(c; D)} \left[ \sum_{x \in D} \log p(x|c) \right] - \mathbf{D}_{KL}(q(c; D) \| p(c)) \right] \quad (6)$$

As we demonstrate in section 4, this subsampling decreases the model’s incentive to capture correlations within a class, reducing utilisation of the latent variables. This poses a significant challenge when scaling up to more powerful generative networks, which require a greater incentive to avoid simply memorising the global distribution. Our work addresses this by replacing the variational lower-bound in Equation 6 with a new objective, which better incentivises the use of latent variables, leading to improved generalisation.

## 3 VARIATIONAL HOMOENCODERS

Rather than bound the likelihood of subsamples  $D$  from a set, as in Edwards & Storkey (2016), we instead use subsampling to construct a lower bound on the complete set  $X$ . We use a constrained variational distribution  $q(c; D), D \subseteq X$  for posterior inference and an unbiased stochastic approximation  $\log p(x|c), x \in X$  for the likelihood. This bound will typically be loose due to stochasticity in sampling  $D$ , and we view this as a regularization strategy: we aim to learn latent representations that are quickly inferable from a small number of instances, and the VHE objective is tailored for this purpose.

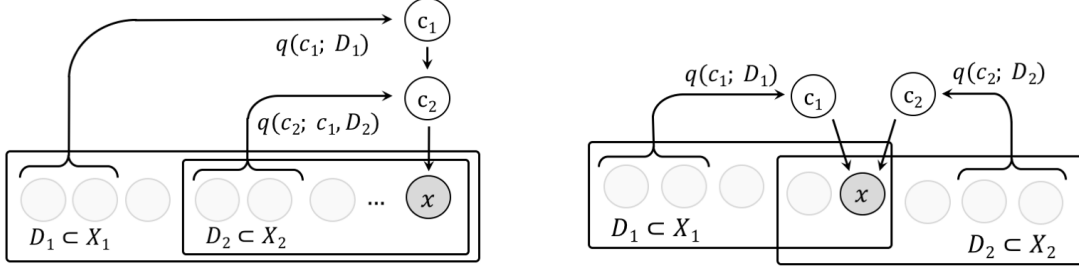


Figure 2: Application of VHE framework to hierarchical (*left*) and factorial (*right*) models. Given an element  $x$  such that  $x \in X_1$  and  $x \in X_2$ , an approximate posterior is constructed for the corresponding shared latent variables  $c_1, c_2$  using subsampled sets  $D_1 \subset X_1, D_2 \subset X_2$ .

### 3.1 STOCHASTIC LOWER BOUND

We would like to learn a generative model for sets  $X$  of the form

$$p(X) = \int p(c) \prod_{x \in X} p(x|c) dc \quad (7)$$

We will refer our full dataset as a union of disjoint sets  $\mathcal{X} = X_1 \sqcup X_2 \sqcup \dots \sqcup X_n$ , and use  $X_{(x)}$  to refer to the set  $X_i \ni x$ . Using the standard consequent of Jensen's inequality, we can lower bound the log-likelihood of each set  $X$  using an arbitrary distribution  $q$ . In particular, we give  $q$  as a fixed function of arbitrary data.

$$\log p(X) \geq \mathbb{E}_{q(c;D)} \log p(X|c) - \mathbf{D}_{KL}[q(c;D) \| p(c)], \quad \forall D \subset X \quad (8)$$

Splitting up individual likelihoods, we may rewrite

$$\log p(X) \geq \mathbb{E}_{q(c;D)} \left[ \sum_{x \in X} \log p(x|c) \right] - \mathbf{D}_{KL}[q(c;D) \| p(c)], \quad \forall D \subset X \quad (9)$$

$$= \sum_{x \in X} \left[ \mathbb{E}_{q(c;D)} \log p(x|c) - \frac{1}{|X|} \mathbf{D}_{KL}[q(c;D) \| p(c)] \right], \quad \forall D \subset X \quad (10)$$

$$\stackrel{\text{def}}{=} \sum_{x \in X} \mathcal{L}(x; D, |X|), \quad \forall D \subset X \quad (11)$$

Finally, we can replace the universal quantification with an expectation under any distribution of  $D$  (e.g. uniform

sampling from  $X$  without replacement):

$$\log p(X) \geq \mathbb{E}_{D \subset X} \sum_{x \in X} \mathcal{L}(x; D, |X|) \quad (12)$$

$$= \sum_{x \in X} \mathbb{E}_{D \subset X} \mathcal{L}(x; D, |X|) \quad (13)$$

$$\log p(\mathcal{X}) \geq \sum_{x \in \mathcal{X}} \mathbb{E}_{D \subset X_{(x)}} \mathcal{L}(x; D, |X_{(x)}|) \quad (14)$$

This formulation suggests a simple modification to the VAE training procedure, as shown in Algorithm 1. At each iteration we select an element  $x$ , use resampled elements  $D \subset X_{(x)}$  to construct the approximate posterior  $q(c; D)$ , and rescale the encoding cost appropriately.

*VHE objective:*

$$\mathbb{E}_{\substack{x \in \mathcal{X} \\ D \subset X_{(x)}}} \left[ \mathbb{E}_{q(c;D)} \log p(x|c) - \frac{1}{|X_{(x)}|} \mathbf{D}_{KL}[q(c;D) \| p(c)] \right] \quad (15)$$

If the generative model  $p(x|c)$  also describes a separate latent variable  $z$  for each element, we may simply introduce a second inference network  $q(z; c, x)$  in order to further bound the reconstruction error of Equation 15:

*VHE objective with per-element latent variables:*

$$\mathbb{E}_{\substack{x \in \mathcal{X} \\ D \subset X_{(x)}}} \left[ \mathbb{E}_{q(c;D)} \left[ \mathbb{E}_{q(z;c,x)} \log p(x|c,z) - \mathbf{D}_{KL}[q(z;c,x) \| p(z|c)] \right] - \frac{1}{|X_{(x)}|} \mathbf{D}_{KL}[q(c;D) \| p(c)] \right] \quad (16)$$

### 3.2 APPLICATION TO STRUCTURED DATASETS

The above derivation applies to a dataset partitioned into *disjoint* subsets  $\mathcal{X} = X_1 \sqcup X_2 \sqcup \dots \sqcup X_n$ , each with a corresponding latent variable  $c_i$ . However, many datasets

offer a richer organisational structure, such as the hierarchical grouping of characters into alphabets (Lake et al., 2015) or the factorial categorisation of rendered faces by identity, pose and lighting (Kulkarni et al., 2015).

Provided that such organisational structure is known in advance, we may generalise the training objective in Equation 14 to include a separate latent variable  $c_i$  for each group  $X_i$  within the dataset, even when these groups overlap. To do this we first rewrite this bound in its most general form, where  $\mathbf{c}$  collects all latent variables:

$$\log p(\mathcal{X}) \geq \mathbb{E}_{Q(\mathbf{c}; \mathbf{D})} \left[ \sum_{x \in X} \log p(x|\mathbf{c}) \right] - D_{KL}[Q(\mathbf{c}; \mathbf{D}) \| P(\mathbf{c})] \quad (17)$$

As shown in Figure 2, a separate  $D_i \subset X_i$  may be subsampled for inference of each latent variable  $c_i$ , so that  $Q(\mathbf{c}) = \prod_i q_i(c_i; D_i)$ . This leads to an analogous training objective (Equation 18), which may be applied to data with factorial or hierarchical category structure. For the hierarchical case, this objective may be further modified to infer layers sequentially, derived in Supplementary Material.

$$\log p(\mathcal{X}) \geq \sum_{x \in \mathcal{X}} \mathbb{E}_{\substack{D_i \subset X_i \\ \text{for each} \\ i: x \in X_i}} \left[ \mathbb{E}_{\substack{q_i(c_i; D_i) \\ \text{for each} \\ i: x \in X_i}} \log p(x|\mathbf{c}) - \sum_{i: x \in X_i} \frac{1}{|X_i|} D_{KL}(q_i(c_i; D_i) \| p(c_i)) \right] \quad (18)$$

### 3.3 POWERFUL DECODER MODELS

As evident in Equation 10, the VHE objective provides a formal motivation for KL rescaling in the variational objective (a common technique to increase use of latent variables in VAEs) by sharing these variables across many elements. This is of particular importance when using autoregressive decoder models, for which a common failure mode is to learn a decoder  $p(x|z)$  with no dependence on the latent space, thus avoiding the encoding cost. In the context of VAEs, this particular issue has been discussed by Chen et al. (2016) who suggest crippling the decoder as a potential remedy.

The same failure mode can occur when training a VAE for sets, particularly when the sets  $D$  are of small size and thus have low total correlation. *Variational Homocoders* suggest a potential remedy to this, encouraging use of the latent space by reusing the same latent variables across a large set  $X$ . This allows a VHE to learn useful representations even with  $|D| = 1$ , while at the

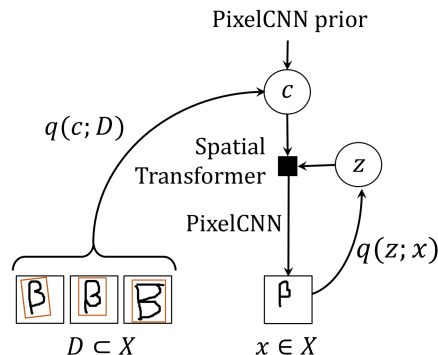


Figure 3: PixelCNN VHE architecture used for Omniglot and Youtube Faces. A spatial transformer network  $q(c; D)$  encodes a subset  $D$  of a character class into a class latent variable  $c$  with the same width and height as the input image. A separate encoder  $q(z; x)$ , parametrized by a convolutional network, encodes position information in the target image into a latent variable  $z$ . A PixelCNN prior is used for  $c$ , and a Gaussian prior for  $z$ . During decoding,  $c$  and  $z$  are combined by a spatial transformer and used to condition a PixelCNN decoder network  $p(x|\text{STN}(c, z))$ .

same time utilising a powerful decoder to achieve highly accurate density estimation. In our experiments, we exploit the VHE’s ability to use powerful decoders: specifically, we learn a generative model with a PixelCNN decoder, which is not possible with previous frameworks.

## 4 EXPERIMENTAL RESULTS

### 4.1 HANDWRITTEN CHARACTER CLASSES

To demonstrate that the VHE objective can facilitate learning with more expressive generative networks, we trained a variety of models on the Omniglot dataset exploring the interaction between model architecture and training objective. We consider two model architectures: a standard deconvolutional network based on Edwards & Storkey (2016), and a hierarchical PixelCNN architecture inspired by the PixelVAE (Gulrajani et al., 2016). For each, we compare models trained with the VHE objective against three alternative objectives.

For our hierarchical PixelCNN architecture (Figure 3) each character class is associated with a spatial latent variable  $c$  (a character ‘template’) with a PixelCNN prior, and each image  $x$  is associated with its own latent variable  $z$  (its ‘position’) with a Gaussian prior. To generate  $x$ , a Spatial Transformer Network (STN) (Jaderberg

Table 1: Comparison of VHE, Neural Statistician, and intermediate objectives with both deconvolutional and PixelCNN architectures. Rescaling encourages use of the latent space, while resampling encourages generalisation from the support set. The VHE is able to utilise the PixelCNN to achieve the highest classification accuracy.

	KL / nats*	Accuracy (5-shot)
<b>Deconvolutional Architecture</b>		
Neural Statistician [3]	31.34	95.6%
Resample (Eq 19)	25.74	94.0%
Rescale (Eq 20)	477.65	95.3%
VHE (resample + rescale, Eq 16)	452.47	95.6%
<b>PixelCNN Architecture</b>		
Neural Statistician	14.90	66.0%
Resample	0.22	4.9%
Rescale	506.48	62.8%
VHE (resample + rescale)	268.37	<b>98.8%</b>

\* $D_{KL}(q(c; D) \parallel p(c))$ , train set

et al., 2015) applies  $z$  to the class template  $c$ , and the result is input to a Gated PixelCNN  $p(x|\text{STN}(c, z))$  (Oord et al., 2016). The position encoder  $q(z; x)$  is given by a CNN, and the class encoder  $q(c; D)$  by an STN averaged over  $D$ . Both produce diagonal Gaussian distributions.

Using both PixelCNN and deconvolutional architectures, we trained models by several objectives. We compare a VHE model against a Neural Statistician baseline, with each trained on sampled subsets  $D \subset X$  with  $|D| = 5$  (as in Edwards & Storkey (2016)). Secondly, since the VHE introduces both data-resampling and KL-rescaling as modifications to this baseline, we separate the contributions of each using two intermediate objectives:

Resample only:

$$\underbrace{\mathbb{E}_{\substack{D \subset X \\ x \in X}}}_{\text{resample decoded element}} \left[ \mathbb{E}_{q(c; D)} \log p(x|c) - \frac{1}{|D|} D_{KL}[q(c; D) \parallel p(c)] \right] \quad (19)$$

Rescale only:

$$\mathbb{E}_{\substack{D \subset X \\ x \in D}} \left[ \mathbb{E}_{q(c; D)} \log p(x|c) - \underbrace{\frac{1}{|X|}}_{\text{rescale KL}} D_{KL}[q(c; D) \parallel p(c)] \right] \quad (20)$$

Table 2: Comparison of classification accuracy with previous work. The VHE objective allows us to use a powerful decoder network, yielding state-of-the-art few-shot classification amongst deep generative models.

	Classification Accuracy (20-way)	
	1-shot	5-shot
<b>Generative models, <math>\log p(X)</math></b>		
Generative Matching Networks [1]	77.0%	91.0%
Neural Statistician [3]	93.2%	98.1%
VHE	<b>95.2%</b>	<b>98.8%</b>
<b>Discriminative models, <math>\log q(y; x, X, Y)</math></b>		
Matching Networks [21]	93.8%	98.7%
Convnet with memory module [9]	95.0%	98.6%
mAP-DLM [20]	95.4%	98.6%
Model-Agnostic Meta-learning [4]	95.8%	<b>98.9%</b>
Prototypical Networks [19]	<b>96.0%</b>	<b>98.9%</b>
(VHE, within-alphabet <sup>1</sup> )	81.3%	90.3%

All models were trained on a random sample of 1200 Omniglot classes using images scaled to 28x28 pixels, dynamically binarised, and augmented by 8 rotations/reflections to produce new classes. We additionally used 20 small random affine transformations to create new instances within each class. Models were optimised using Adam (Kingma & Welling, 2013), and we used training error to select the best parameters from 5 independent training runs. We also implemented the ‘sample dropout’ trick of Edwards & Storkey (2016), but found that this had no effect on performance. At test time we classify an example  $x$  by Monte Carlo estimation of the expected conditional likelihood under the variational posterior  $E_{q(c; D)} p(x|c)$ , with 20 samples from  $q(c; D)$ .  $x$  is then classified to class with support set  $D$  that maximises this expected conditional likelihood.

Table 1 collects classification results of models trained using each of the four alternative training objectives, for both architectures. For a deconvolutional architecture, we find little difference in classification performance between all four training objectives, with the Neural Statistician and VHE models achieving equally high accuracy.

<sup>1</sup>The few-shot classification task defined by Lake et al. (2015) is to identify an image to one of 20 character classes, where *all 20 classes belong to the same (unseen) alphabet*. However, most work since has evaluated on an easier one-shot classification task, in which the 20 support characters are drawn from the entire test set (so are typically more dissimilar). We find that our model performs significantly worse on the within-alphabet variant, and so include results to facilitate future comparison on this more challenging task. Attaining near-human classification accuracy on this variant remains an open challenge for neural network models.



Figure 4: 5-shot samples generated by each model (more in Supplement). With a PixelCNN architecture, both Neural Statistician and Resample objectives lead to underutilisation of the latent space, producing unfaithful samples.

For the hierarchical PixelCNN architecture, however, significant differences arise between training objectives. In this case, a Neural Statistician learns a strong global distribution over images but makes only minimal use of latent variables  $c$ . This means that, despite the use of a higher capacity model, classification accuracy is much poorer (66%) than that achieved using a deconvolutional architecture. For the same reason, conditional samples display an improved sharpness but are no longer identifiable to the cue images on which they were conditioned (Figure 4). Our careful training suggests that this is not an optimisation difficulty but is core to the objective, as discussed in Chen et al. (2016).

By contrast, a VHE is able to gain a large benefit from the hierarchical PixelCNN architecture, with a 3-fold reduction in classification error (5-shot accuracy 98.8%) and conditional samples which are simultaneously sharp and identifiable (Figure 4). This improvement is in part achieved by increased utilisation of the latent space, due to rescaling of the KL divergence term in the objective. However, our results show that this common technique is insufficient when used alone, leading to overfitting to cue images with an equally severe impairment of classification performance (accuracy 62.8%). Rather, we find that KL-rescaling and data resampling must be used together in order for the benefit of the powerful PixelCNN architecture to be realised.

Table 2 lists the classification accuracy achieved by VHEs with both  $|D| = 1$  and  $|D| = 5$ , as compared to existing deep learning approaches. We find that both networks are not only state-of-the-art amongst deep generative models, but also competitive against the best discriminative models trained directly for few-shot classification. Unlike these discriminative models, a VHE is also able to generate new images of a character in one shot, producing samples which are both realistic and faithful to the class of the cue image (Figure 5).

As our goal is to model shared structure across images, we evaluate generative performance using joint log like-

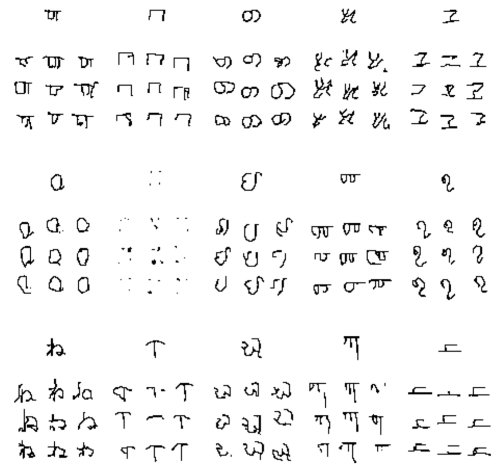


Figure 5: One-shot same-class samples generated by our model. Cue images were sampled from previously unseen classes.

lihood of the entire Omniglot test set (rather than separately across images). From this perspective, a single element VAE will perform poorly as it treats all datapoints as independent, optimising a sum over log likelihoods for each element. By sharing latents across elements of the same class, a VHE can improve upon this considerably.

For likelihood evaluation, our most appropriate comparison is with Generative Matching Networks (Bartunov & Vetrov, 2016) as they also model dependencies within a class. Thus, we trained models under the same train/test split as them, with no data augmentation. We evaluate the joint log likelihood of full character classes from the test set, normalised by the number of elements, using importance weighting with  $k=500$  samples from  $q(c; X)$ . As can be seen in Tables 3 and 4, our hierarchical PixelCNN architecture is able to achieve state-of-the-art log likelihood results only when trained using the VHE objective.

Table 3: Joint NLL of Omniglot test set, compared across architectures and objectives.

Test NLL per image	
<b>Deconvolutional Architecture</b>	
NS [3]	102.84 nats
Resample	110.30 nats
Rescale	109.01 nats
<i>VHE (resample + rescale)</i>	104.67 nats
<b>PixelCNN Architecture</b>	
NS	73.50 nats
Resample	66.42 nats
Rescale	71.37 nats
<i>VHE (resample + rescale)</i>	<b>61.22 nats</b>

## 4.2 YOUTUBE FACES

To confirm that our approach can be used to produce naturalistic images, we compare VHE and Neural Statistician models trained on images from the YouTube Faces Database (Wolf et al., 2011), comprising 3,425 videos of 1,595 celebrities downloaded from YouTube. For our experiments, we use the aligned and cropped to face version, additionally cropping each image by 50% in both height and width, and rescale to 40x40 pixels. Our training, validation, and test sets consist of one video per person and 48 images per video. We use 954 videos for the training set and 641 videos for the test set.

We consider two architectures: the hierarchical PixelCNN network used for Omniglot experiments, and the deconvolution network used to model faces in Edwards & Storkey (2016). As above, we train each model using both VHE and NS objectives with  $|D| = 5$ .

Table 5: Classification results for YouTube faces dataset. The VHE PixelCNN utilises the latent space most effectively, and therefore achieves the highest few-shot classification accuracy and test image NLL.

	Test NLL per image	Accuracy (200-way)	
		1-shot	5-shot
<b>Deconvolutional</b>			
Neural Statistician	12512.4	39.2%	49.0%
<i>VHE</i>	12717.6	37.2%	44.8%
<b>PixelCNN</b>			
Neural Statistician	4229.8	92.1%	98.5%
<i>VHE</i>	<b>4091.3</b>	<b>92.5%</b>	<b>98.9%</b>

<sup>2</sup>We thank the authors of Bartunov & Vetrov (2016) for providing us with this comparison.

Table 4: Comparison of deep generative models by joint NLL of Omniglot test set.

Test NLL per image	
<b>Independent models</b>	$\frac{1}{n} \log \prod_i p(x_i)$
DRAW [5]	< 96.5 nats
Conv DRAW [6]	< 91.0 nats
VLAE [2]	89.83 nats
<b>Conditional models</b>	$\frac{1}{n} \log \prod_i p(x_i   x_{1:i-1})$
Generative Matching Networks [1]	62.42 nats <sup>2</sup>
<b>Shared-latent models</b>	$\frac{1}{n} \log \mathbb{E}_{p(c)} \prod_i p(x_i   c)$
<i>Variational Homoencoder</i>	<b>61.22 nats</b>

Classification results for trained models are shown in Table 5, and conditionally generated samples in Figure 6. As with Omniglot experiments, we find that the VHE objective improves use of the hidden layer  $c$ , leading to more accurate classification and conditional generation than the Neural Statistician. While the deconvolutional architecture is capable of producing realistic images (see Edwards & Storkey (2016)), our results show that it is not powerful enough to perform accurate few-shot classification. On the other hand, the PixelCNN architecture trained using the Neural Statistician objective achieves accurate few-shot classification, but generates poor images. The only network able to produce realistic images *and* perform accurate classification is the PixelCNN trained using our VHE objective.

## 4.3 MODELLING RICH CATEGORY STRUCTURE

To demonstrate how the VHE framework may apply to models with richer category structure, we built both a hierarchical and a factorial VHE (Figure 2) using simple modifications to the above architectures. For the hierarchical VHE, we extended the deconvolutional model with an extra latent layer  $a$  using the same encoder and decoder architecture as  $c$ . This was used to encode alphabet level structure for the Omniglot dataset, learning a generative model for alphabets of the form

$$p(\mathcal{A}) = \int p(a) \prod_{X_i \in \mathcal{A}} \int p(c_i | a) \prod_{x_{ij} \in X_i} p(x_{ij} | c_i, a) dc_i da \quad (21)$$

Again, we trained this model using a single objective, using separately resampled subsets  $D^a$  and  $D^c$  to infer each latent variable (see Supplement). We then tested

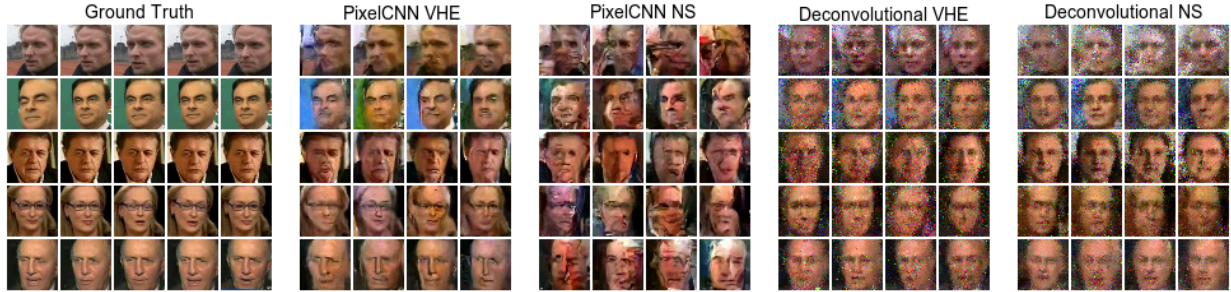


Figure 6: 5-shot samples of YouTube faces generated using both PixelCNN and deconvolutional architectures. Note that, for accurate comparison, we *sample* images from the decoder rather than taking the conditional mode as is common. For the deconvolutional models, this leads to images which appear more noisy than shown in previous work.

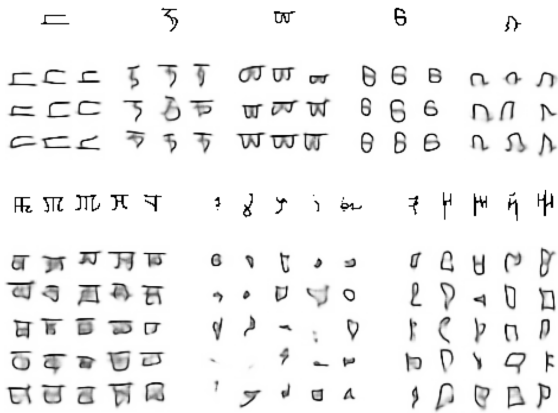


Figure 7: Conditional samples from character (top) and alphabet (bottom) levels of the same hierarchical model.

our model at both one-shot character generation and 5-shot alphabet generation, using samples from previously unseen alphabets. Our single trained model is able to learn structure at both layers of abstraction (Figure 7)

For the factorial VHE, we extended the Omniglot dataset by assigning each image to one of 30 randomly generated styles (independent of its character class), modifying both the colour and pen stroke characteristics of each image. We then extended the PixelCNN model to include a 6-dimensional latent variable  $s$  to represent the *style* of an image, alongside the existing  $c$  to represent the *character*. We used a CNN for style encoder  $q(s; D^s)$ , and for each image location we condition the PixelCNN decoder using the outer product  $s \otimes c_{ij}$ .

We then test this model on a *style transfer* task by feeding separate images into the character encoder  $q(c; D^c)$  and style encoder  $q(s; D^s)$ , then rendering a new image from the inferred  $(c, s)$  pair. We find that synthesised samples are faithful to the respective character and style of both support images (Figure 8), demonstrating the ability of a



Figure 8: Previously unseen characters redrawn with both the colour and stroke width of a second character. For each group, the top two images denote the content (left) and style (right).

factorial VHE to successfully disentangle these two image factors using separate latent variables.

## 5 CONCLUSION

We introduce the *Variational Homoencoder*: a hierarchical Bayesian approach to learning expressive generative models from few examples. We test the VHE by training a hierarchical PixelCNN on the Omniglot dataset, and achieve state-of-the-art results: our model is arguably the first which uses a general purpose architecture to both produce high quality samples and attain near state-of-the-art one-shot classification performance. We further validate our approach on a dataset of face images, and find that the VHE significantly improves the visual quality and classification accuracy achievable with a PixelCNN decoder. Finally, we show that the VHE framework extends naturally to models with richer latent structure, which we see as a promising direction for future work.



## References

- [1] Sergey Bartunov and Dmitry P Vetrov. Fast adaptation in generative models with generative matching networks. *arXiv preprint arXiv:1612.02192*, 2016.
- [2] Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational lossy auto-encoder. *arXiv preprint arXiv:1611.02731*, 2016.
- [3] Harrison Edwards and Amos Storkey. Towards a neural statistician. *arXiv preprint arXiv:1606.02185*, 2016.
- [4] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.
- [5] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.
- [6] Karol Gregor, Frederic Besse, Danilo Jimenez Rezende, Ivo Danihelka, and Daan Wierstra. Towards conceptual compression. In *Advances In Neural Information Processing Systems*, pp. 3549–3557, 2016.
- [7] Ishaan Gulrajani, Kundan Kumar, Faruk Ahmed, Adrien Ali Taïga, Francesco Visin, David Vazquez, and Aaron Courville. Pixelvae: A latent variable model for natural images. *arXiv preprint arXiv:1611.05013*, 2016.
- [8] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pp. 2017–2025, 2015.
- [9] Łukasz Kaiser, Ofir Nachum, Aurko Roy, and Samy Bengio. Learning to remember rare events. *arXiv preprint arXiv:1703.03129*, 2017.
- [10] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- [11] Gregory Koch. *Siamese neural networks for one-shot image recognition*. PhD thesis, University of Toronto, 2015.
- [12] Tejas D Kulkarni, William F Whitney, Pushmeet Kohli, and Josh Tenenbaum. Deep convolutional inverse graphics network. In *Advances in Neural Information Processing Systems*, pp. 2539–2547, 2015.
- [13] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350 (6266):1332–1338, 2015.
- [14] Aaron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. Conditional image generation with pixelcnn decoders. *arXiv preprint arXiv:1606.05328*, 2016.
- [15] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. 2016.
- [16] Scott Reed, Yutian Chen, Thomas Paine, Aaron van den Oord, SM Eslami, Danilo Rezende, Oriol Vinyals, and Nando de Freitas. Few-shot autoregressive density estimation: Towards learning to learn distributions. *arXiv preprint arXiv:1710.10304*, 2017.
- [17] Danilo Rezende, Ivo Danihelka, Karol Gregor, Daan Wierstra, et al. One-shot generalization in deep generative models. In *Proceedings of The 33rd International Conference on Machine Learning*, pp. 1521–1529, 2016.
- [18] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. One-shot learning with memory-augmented neural networks. *arXiv preprint arXiv:1605.06065*, 2016.
- [19] Jake Snell, Kevin Swersky, and Richard S Zemel. Prototypical networks for few-shot learning. *arXiv preprint arXiv:1703.05175*, 2017.
- [20] Eleni Triantafillou, Richard Zemel, and Raquel Urtasun. Few-shot learning through an information retrieval lens. In *Advances in Neural Information Processing Systems*, pp. 2252–2262, 2017.
- [21] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, pp. 3630–3638, 2016.
- [22] Lior Wolf, Tal Hassner, and Itay Maoz. Face recognition in unconstrained videos with matched background similarity. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 529–534. IEEE, 2011.

---

# Probabilistic Collaborative Representation Learning for Personalized Item Recommendation

---

Aghiles Salah and Hady W. Lauw  
School of Information Systems  
Singapore Management University, Singapore  
{asalah, hadywlauw}@smu.edu.sg

## Abstract

We present Probabilistic Collaborative Representation Learning (PCRL), a new generative model of user preferences and item contexts. The latter builds on the assumption that relationships among items within contexts (e.g., browsing session, shopping cart, etc.) may underlie various aspects that guide the choices people make. Intuitively, PCRL seeks representations of items reflecting various regularities between them that might be useful at explaining user preferences. Formally, it relies on Bayesian Poisson Factorization to model user-item interactions, and uses a multilayered latent variable architecture to learn representations of items from their contexts. PCRL seamlessly integrates both tasks within a joint framework. However, inference and learning under the proposed model are challenging due to several sources of intractability. Relying on the recent advances in approximate inference/learning, we derive an efficient variational algorithm to estimate our model from observations. We further conduct experiments on several real-world datasets to showcase the benefits of the proposed model.

## 1 INTRODUCTION

With pervasive digitization of marketplaces and services, we now make most of our consumption choices online. Relieved from the inventory limitation of a physical storefront, online providers are able to offer a mind-boggling array of choices numbering in the thousands to millions. To help users in navigating this sea of choices, modern applications rely heavily on recommender systems to deliver a personalized ranking or selection of items to each user according to her preferences.

There are various approaches to recommender systems, including memory-based and model-based approaches (Sarwar et al., 2001). At the heart of the more prevalent model-based approach is learning a latent representation for every user and every item. Such a latent representation places a user or an item in the “feature” space of preferences, such that when two related items share similar representations or “features”, a user who prefers one likely also prefers the other. Further recommendation predictions are based on these latent representations.

Much of the previous work seek to learn these representations from historical behavioral data, such as ratings, clicks, purchases, etc. (usually organized into a user-item interaction or preference matrix). For instance, the widespread Matrix Factorization (MF) (Mnih and Salakhutdinov, 2008; Hu et al., 2008; Koren et al., 2009) derives user and item latent representations in the form of low dimensional vectors by decomposing the preference matrix. The bilinear combination of user and item’s latent factors can be used to predict unknown preferences.

The limitation of learning these representations from historical behaviors is the sparsity of such data. The long-tail effect (Park and Tuzhilin, 2008) means that most items have been adopted by few users. Moreover, given the rapid expansion of catalogues, there are continually new items with scant record of historical consumption. One consequence of this sparsity is that closely related items may not be mapped to the same direction in the latent space, as they might not have been rated by the same users. As such, historical consumption data alone may not suffice for learning effective item representations.

In some real-world scenarios, there may be known some auxiliary information on how items are likely related to one another. For example, a user interested in a particular shirt may also be interested in a matching pair of jeans. Moreover, such relatedness among items may not have to be explicitly stated, and could be implicitly inferred from such indicative events as whether items are placed

within the same shopping carts, are browsed within the same session, etc. Such item-item relationships constitute valuable information that would otherwise not easily be derivable from similarities in product attributes alone. We thus seek to enrich the learned item representation to also incorporate such item-item relationships, to supplement the sparse user-item interactions.

Representation learning (Bengio et al., 2013) is of interest to learn features or representations from different data, such as images, text, etc. Recent techniques rely on deep neural networks to learn compositional representations. While inspired by this promising approach, our work is set apart in that we are interested not only on extracting objective features of items, but more importantly also those that could help describe user preferences effectively. Therefore, instead of relying on representation learning solely or separately, given the efficacy of probabilistic models for collaborative filtering, we propose to conjoin the representation learning from item-item contextual relationships, and collaborative filtering from user-item interactions, within a unified model.

In this paper, we develop Probabilistic Collaborative Representation Learning (PCRL), which seeks to learn item representations both *contextually* based on their relatedness with other items, as well as *collaboratively* based on their interactions/adoptions by users. For the former, PCRL uses a multilayered (hierarchical) latent variable structure, with a Poisson likelihood and Gamma distributed layers, to model the item’s context (e.g., shopping cart, session). For the latter, PCRL relies on Poisson Factorization (PF) for decomposing users’ interactions with items. As shown in (Gopalan et al., 2015), PF realistically models user preferences, fits well to sparse data thanks to the Poisson’s mathematical form, and it substantially outperforms previous state-of-the-art Gaussian likelihoods-based MF models (Mnih and Salakhutdinov, 2008; Shan and Banerjee, 2010; Koren et al., 2009) for item recommendation.

PCRL joins both sources of data through a shared item latent space within a probabilistic generative model. Intuitively, the collaborative PF component can guide the contextual representation learning process to focus on extracting features that are relevant for predicting the preference information. The contextual representation learning component in turn will encourage the PF part to rely on items’ contexts to explain user preferences, which would supplement the lack of user-item interactions.

Exact inference under the PCRL model is very challenging due to various sources of intractability. To overcome this difficulty we rely on recent innovations in approximate inference/learning and derive an efficient variational algorithm to estimate PCRL from observed user

preferences and item contexts. Empirical results on several real-world datasets reflect the benefits of PCRL in terms of both personalized recommendation and item representation learning.

## 2 RELATED WORK

The sparsity of preference data has driven many to extend Matrix Factorization (MF) models (Mnih and Salakhutdinov, 2008; Hu et al., 2008; Koren et al., 2009) beyond user-item interactions, and leverage auxiliary information, such as social networks (Ma et al., 2008; Zhou et al., 2012; Rao et al., 2015), product taxonomy (Koenigstein et al., 2011), item content (Wang and Blei, 2011), etc. However, these are mostly still within the framework of MF. For instance, Collective Matrix Factorization (Singh and Gordon, 2008), which co-factorized multiple data matrices, is a popular approach in the recommendation literature to jointly model several sources of information.

Yet other approaches, similarly to ours, use graphical models to join different modalities. Wang and Blei (2011) developed Collaborative Topic Regression (CTR), which composes a topic model, Latent Dirichlet Allocation (LDA), with probabilistic matrix factorization to model texts (articles) and user (reader) preferences. Along the same line, Wang et al. (2015); Li and She (2017) proposed alternatives to CTR where probabilistic auto-encoder, is substituted for LDA for modeling text.

We focus on incorporating item relatedness, a modality mostly neglected by previous personalized recommendation models. Notable exceptions include CoFactor (Liang et al., 2016) and Matrix Co-Factorization (MCF) (Park et al., 2017), which used the principle of collective MF based on Gaussian likelihoods. In contrast, we build on Bayesian Poisson Factorization (PF), and we further investigate another architecture for leveraging the item’s contexts with new modeling perspectives. In experiments, we compare to the more recent MCF that learns from item network as a baseline. CoFactor learns not from an external auxiliary source, but rather from item-item relations induced from the user-item interactions.

Since (Gopalan et al., 2015), there is a growing body of work on applying PF (Canny, 2004; Cemgil, 2009) to recommender systems. Gopalan et al. (2014a) developed non-parametric PF. Chaney et al. (2015) incorporated social interactions. Charlin et al. (2015) accounted for user and item evolution over time. Notably, Gopalan et al. (2014b) proposed Collaborative Topic Poisson Factorization (CTPF) to model both article contents and reader preferences. In contrast to CTPF that uses PF to model both the user preferences and auxiliary item information (text), we adopt PF for the user-item interactions only,

and we use a multilayered latent variable structure to learn item representations from auxiliary data (item contexts). The benefits of our modeling architecture would be reflected in the experiments with CTPF as a baseline.

### 3 PROBABILISTIC COLLABORATIVE REPRESENTATION LEARNING

The observed data that we would learn from are user preferences and item contexts respectively. The former are organized into a user-item preference matrix of size  $U \times I$ , denoted  $\mathbf{X} = (x_{ui})$ , where  $x_{ui}$  is the integer rating<sup>1</sup> that user  $u$  gave to item  $i$ , or zero if no preference was expressed. The contextual interactions among items are encoded in an item-context matrix  $\mathbf{C} = (c_{ij})$ , of size  $I \times J$ , where  $c_{ij} = 1$  if item  $j$  belongs to the context<sup>2</sup> of  $i$ , and  $c_{ij} = 0$  otherwise. The  $i^{\text{th}}$  row of this matrix is represented by a vector  $\mathbf{c}_i = (c_{i1}, \dots, c_{iJ})^\top$ , where  $\top$  denotes the transpose. We will refer to the set of items  $j$  such that  $c_{ij} > 0$  as the context of item  $i$ .

We now describe *Probabilistic Collaborative Representation Learning* or PCRL, a new probabilistic latent variable model for jointly modeling user preferences and item contexts. The intuition is to learn item representations reflecting various contextual relationships between them that are useful for explaining user preferences. Figure 1 depicts PCRL in plate notation.

**Contextual Representation Learning.** To model representations due to the item contexts (refer to the left portion of Figure 1) we use a multilayer structure similar to Deep Exponential Families (Ranganath et al., 2015). More precisely, PCRL assumes  $L$  layers of hidden variables per item:  $\mathcal{Z}_i = \{\mathbf{z}_{i,1}, \dots, \mathbf{z}_{i,L}\}$ , such that  $\mathbf{z}_{i,\ell} \in \mathbb{R}_+^{K_\ell}$ . For a reason that will be clear shortly, we denote  $\mathbf{z}_{i,L+1} = \beta_i$ . Along with these variables, PCRL has  $L + 1$  layers of latent weights shared across items,  $\mathcal{W} = \{\mathbf{W}_0, \dots, \mathbf{W}_L\}$ , where  $\mathbf{W}_\ell$  is a matrix of size  $K_{\ell+1} \times K_\ell$ , with  $K_0 = J$ , and its  $k^{\text{th}}$  column is denoted by  $\mathbf{w}_{\ell,k}$ . Effectively, each hidden layer models representations for items based on their contexts. Intuitively, a higher layer encodes a higher level of representational abstraction;  $\beta_i$  is the most abstract representation.

The components  $z_{i,\ell,k}$  at each hidden layer are Gamma distributed. Note that this choice is not a limitation of our modeling framework. Depending on specific requirements, other types of  $z_{i,\ell,k}$  are possible, e.g., Gaussian, and these might differ across layers.

<sup>1</sup>Other user-item interactions indicative of preferences are also possible, e.g., number of clicks.

<sup>2</sup>The definition of ‘‘context’’ is scenario-dependent, e.g., another item  $j$  is found in the same shopping cart as item  $i$ .

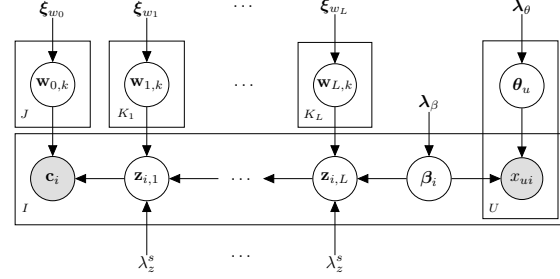


Figure 1: The proposed model PCRL in plate representation,  $\xi$  and  $\lambda = (\lambda^s, \lambda^r)$  stand for Gaussian and Gamma parameters.

To capture various correlations across layers, including negative ones, we let the weights  $\mathbf{W}_\ell$  be real valued with Gaussian priors. These latent variables interact with each other to explain the contextual relationships among items. While several interaction schemes are possible, we mimic neural networks (multilayer perceptron or MLP), and let the mean of the local variable at the current layer to be driven by the current weights and the previous layer as follows:

$$\mathbb{E}(\mathbf{z}_{i,\ell} | \mathbf{W}_\ell, \mathbf{z}_{i,\ell+1}) = a_\ell(\mathbf{z}_{i,\ell+1}^\top \mathbf{W}_\ell) \quad (1)$$

where  $a_\ell(x)$  is a function that maps  $x$  into the right mean space. Following the nomenclature in the neural network literature, we call it the *activation function*.

Conditional on the lowest layer,  $\mathbf{z}_{i,1}$ , the components of the item-context vector  $\mathbf{c}_i$  are independent Poisson variables, i.e.,  $\mathbf{c}_i \sim p(\mathbf{c}_i | \mathbf{z}_{i,1}, \mathcal{W}) = \prod_j p(c_{ij} | \mathbf{z}_{i,1}, \mathcal{W})$ , and

$$p(c_{ij} | \mathcal{Z}, \mathcal{W}, \beta) = \text{Poisson}(\mathbf{z}_{i,1}^\top \mathbf{w}_{0,j}) \quad (2)$$

where  $\mathbf{w}_{0,j}$  denotes the  $j^{\text{th}}$  column of the matrix  $\mathbf{W}_0$ .

**Collaborative Poisson Factorization.** To model user preferences (refer to the right portion of Figure 1), PCRL relies on Poisson factorization, i.e.,

$$x_{ui} | \theta, \beta \sim \text{Poisson}(\theta_u^\top \beta_i), \quad (3)$$

where  $\theta_u^\top \in \mathbb{R}_+^K$  and  $\beta_i^\top \in \mathbb{R}_+^K$  are latent variables referred to as the vectors of user preferences and item attributes respectively. Similar to the original Bayesian Poisson factorization, we let the user preferences  $\theta_{uk}$  and item attributes  $\beta_{ik}$  be Gamma random variables—throughout the paper, we use the shape and rate parameterization of the Gamma distribution.

**Unified Generative Model.** The intuition behind this multilayer architecture and sharing  $\beta$  between the collaborative and contextual parts, is to let the latent variables  $\mathcal{Z}$  and  $\mathcal{W}$ , at the intermediate layers, absorb various item-context patterns encoded in  $\mathbf{C}$ , while encouraging the item latent attributes  $\beta$  to capture only those patterns

which are useful for explaining user preferences. The corresponding generative process is as follows:

1. Draw user preferences:  $\theta_{uk} \sim \text{Gamma}(\lambda_\theta^s, \lambda_\theta^r)$ .
2. For each item  $i$ 
  - (a) Draw its attributes:  $\beta_{ik} \sim \text{Gamma}(\lambda_\beta^s, \lambda_\beta^r)$
  - (b) For each layer  $\ell$ , for  $k \in \{1, \dots, K_\ell\}$ :
    - i. Draw  $\mathbf{w}_{\ell,k} \sim \mathcal{N}(\boldsymbol{\mu}_\ell, \boldsymbol{\sigma}_\ell^2 \mathbf{I}_{K_{\ell+1}})$
    - ii. Draw  $z_{i,\ell,k} \sim \text{Gamma}(\lambda_z^s, \frac{\lambda_z^s}{a_\ell(\mathbf{z}_{\ell+1}^\top \mathbf{w}_{\ell,k})})$
  - (c) For  $j \in \{1, \dots, J\}$ ,
    - i. Draw  $\mathbf{w}_{0,j} \sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\sigma}_0^2 \mathbf{I}_{K_1})$
    - ii. Draw  $c_{ij} \sim \text{Poisson}(a_0(\mathbf{z}_{i,1}^\top \mathbf{w}_{0,j}))$
3. For each user-item pair  $(u, i)$  sample a preference:  $x_{ui} \sim \text{Poisson}(\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i)$ ,

where  $\mathbf{I}_K$  stands for the identity matrix of size  $K$ . In practice, we use the standard multivariate isotropic Gaussian as the prior over each variable  $\mathbf{w}_{\ell,k}$ . Further, for efficiency purposes, we will make the latent variables  $\mathbf{z}_{i,\ell}$  for  $\ell \in \{1, \dots, L\}$  deterministic by taking  $\lambda_z^s$  to infinity.

In principle PCRL should place high probability on item factors  $\boldsymbol{\beta}$  reflecting various item relationship patterns that are useful at explaining user preferences.

**Connections to Existing Models.** In unifying item contexts and user-item preferences, PCRL effectively generalizes and subsumes other more restricted formulations.

For one, as evident from the construction of PCRL, if we remove the context-specific components,  $\mathcal{Z}$ ,  $\mathcal{W}$  and  $\mathbf{C}$ , then PCRL collapses to the original Bayesian Poisson factorization (Cemgil, 2009; Gopalan et al., 2015) that would learn from user-item preferences alone.

For another, if we drop the collaborative filtering components, namely  $\mathbf{X}$  and  $\boldsymbol{\theta}$ , then we would recover an instance of Deep Exponential Families (DEFs) (Ranganath et al., 2015) for unsupervised feature learning. However it should be noted that our composition of Gamma distributed layers and Gaussian weights has not been investigated previously in (Ranganath et al., 2015). The PCRL’s representation learning component is also related to the Poisson Gamma Belief Network (PGBN) (Zhou et al., 2016). The key differences are: PGBN uses Dirichlet weights, it factorizes and chains the Gamma shape instead of the rate parameters.

If we further take the shape parameter  $\lambda_z^s$  to infinity, than PCRL is reduced to a Bayesian deep “decoder” neural network, with a stochastic Gamma top layer  $\boldsymbol{\beta}$ . Furthermore, starting from PCRL we can derive a Bayesian Gamma-Poisson variant of the variational auto-encoder

(Kingma and Welling, 2014). To our knowledge, such neural networks with Gamma stochastic layers have not been studied in prior literature.

## 4 INFERENCE & LEARNING

So far we describe PCRL as a generative model. In practice, we are given  $\mathbf{X}$  and  $\mathbf{C}$ , and we are interested in reversing the above generative process to infer the posterior distribution of the latent variables, i.e.,  $p(\boldsymbol{\theta}, \boldsymbol{\beta}, \mathcal{W} | \mathbf{X}, \mathbf{C})$  that would be the most likely to generate the observations. This allows us to explore data in different ways as well as predict unknown ratings for recommendations. Note that by taking  $\lambda_z^s$  to infinity the intermediate latent variables  $\mathcal{Z}$  become deterministic; this is why they are not considered in the above posterior.

As in many Bayesian models, the above posterior is intractable. We therefore resort to approximate inference. In particular, we rely on Variational Inference (VI) (Bishop, 2006; Blei et al., 2017), which is widely used in statistical learning to fit complex Bayesian models.

### 4.1 THE VARIATIONAL FAMILY

The key to variational inference is to introduce a tractable family of distributions  $q$ , governed by a set of *variational parameters*  $\nu$ . The objective is then to find the closest, typically in terms of the Kullback-Leibler (KL) divergence, member of this family to the true posterior.

We can ease inference in the collaborative part of PCRL by introducing an additional layer of auxiliary latent variables, leaving the original model intact when marginalized out. As in (Cemgil, 2009), we add  $K$  variables  $s_{uik} \sim \text{Poisson}(\theta_{uk}\beta_{ik})$  for each observed rating  $x_{ui}$ , such that  $x_{ui} = \sum_k s_{uik}$ . The marginal distribution of  $x_{ui}$  is preserved thanks to the additive property of Poisson random variables (Kingman, 1993). As the  $s_{uik}$ ’s are not random when  $x_{ui}$  is zero, we need to consider these variables for the non-zero elements in  $\mathbf{X}$  only.

One main source of intractability in our model is the coupling between the different latent variables. To overcome this difficulty, we adopt a *mean-field* variational family (Jordan et al., 1999),  $q(\cdot | \nu) = q(\boldsymbol{\theta}, \boldsymbol{\beta}, \mathbf{s}, \mathcal{W} | \nu)$ , which factorizes with respect to the latent variables:

$$q(\cdot | \nu, \mathbf{C}) = \prod_u q(\boldsymbol{\theta}_u | \tilde{\boldsymbol{\lambda}}_u^\theta) \prod_i q(\boldsymbol{\beta}_i | \tilde{\boldsymbol{\lambda}}_i^\beta) \prod_{u,i} q(s_{ui} | \tilde{\phi}_{ui}) \prod_{\ell=0}^L q(\mathbf{W}_\ell | \tilde{\boldsymbol{\xi}}_\ell) \quad (4)$$

where  $\nu = \{\tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\phi}}\}$ . Note that the variational distributions in the above equation are fully factorized, e.g.,  $q(\boldsymbol{\theta}_u | \tilde{\boldsymbol{\lambda}}_u^\theta) = \prod_k q(\theta_{uk} | \tilde{\lambda}_{uk}^\theta)$ . Each variational distribution is in the same family as the model distribution.

That is, the factors over the Gamma variables,  $\theta$  and  $\beta$ , are also Gamma distributions variational parameters  $\lambda$ , e.g.,  $\tilde{\lambda}_{uk}^\theta = (\tilde{\lambda}_{uk}^{\theta,s}, \tilde{\lambda}_{uk}^{\theta,r})$ . For the item attributes, we further amortize computations by using an inference network. More precisely, we let  $\tilde{\lambda}_i^\beta = (\tilde{\lambda}_i^{\beta,s}, \tilde{\lambda}_i^{\beta,r}) = \mathbf{f}_\omega(\mathbf{c}_i)$ , where  $\mathbf{f}_\omega(\mathbf{c}_i)$  is a deep ‘‘encoder’’ neural network (MLP), parameterized by  $\omega$ , whose input is  $\mathbf{c}_i$ ,  $\tilde{\lambda}_i^{\beta,s} = (\tilde{\lambda}_{ik}^{\beta,s}, \dots, \tilde{\lambda}_{iK}^{\beta,s})$  and  $\tilde{\lambda}_i^{\beta,r} = (\tilde{\lambda}_{ik}^{\beta,r}, \dots, \tilde{\lambda}_{iK}^{\beta,r})$ . Note that, the variational parameters over the item factors  $q(\beta)$  become  $\omega$ .

The factors over  $\mathbf{s}_{ui}$  are Multinomial distributions with free parameters  $\tilde{\phi}$ . This follows from the fact that the conditional distribution of a set of Poisson variables given their sum is a Multinomial (Cemgil, 2009).

The variational factor over  $\mathbf{W}_\ell$  takes this form:  $q(\mathbf{W}_\ell | \tilde{\xi}_\ell) = \prod_{k=1}^{K_\ell} q(\mathbf{w}_{\ell,k} | \tilde{\xi}_\ell^k)$ , where  $\tilde{\xi}_\ell^k = \{\tilde{\boldsymbol{\mu}}_\ell^k, (\tilde{\boldsymbol{\sigma}}_\ell^k)^2 \mathbf{I}_{K_{\ell+1}}\}$  indexes a multivariate Gaussian with a diagonal covariance structure.

Fitting the variational parameters  $\nu$  by minimizing the KL divergence between  $q$  and the true posterior is akin to maximizing the Evidence Lower Bound (ELBO), i.e.,

$$\mathcal{L} = \mathbb{E}_q[\log p(\mathbf{X}, \mathbf{C}, \mathcal{W}, \beta, \theta, \mathbf{s}) - \log(q(\cdot | \nu))] \quad (5)$$

Next we derive an algorithm to maximize (5).

## 4.2 COORDINATE ASCENT LEARNING

We now derive a variational algorithm to estimate PCRL from data. The principle is to alternate the update of each variational parameter while holding the others fixed.

**Updates for  $\tilde{\lambda}^\theta$  and  $\tilde{\phi}$ .** Thanks to the auxiliary variables  $\mathbf{s}$ ,  $\tilde{\lambda}^\theta$  and  $\tilde{\phi}$  have the following closed-form updates,

$$\tilde{\lambda}_{uk}^\theta = \left( \lambda_\theta^s + \sum_i x_{ui} \tilde{\phi}_{uik}, \lambda_\theta^r + \sum_i \frac{\tilde{\lambda}_{ik}^{\beta,s}}{\tilde{\lambda}_{ik}^{\beta,r}} \right), \quad (6)$$

$$\tilde{\phi}_{uik} \propto \exp \left( \psi(\tilde{\lambda}_{uk}^{\theta,s}) - \log \tilde{\lambda}_{uk}^{\theta,r} + \psi(\tilde{\lambda}_{ik}^{\beta,s}) - \log \tilde{\lambda}_{ik}^{\beta,r} \right) \quad (7)$$

where  $\psi(\cdot)$  denotes the digamma function. These updates are identical to those of Bayesian PF (Cemgil, 2009; Gopalan et al., 2015). For more details, please refer to the supplementary material (A.1).

**Parameter update for  $q(\beta)$  and  $q(\mathcal{W})$ .** The remaining variational parameters do not admit closed-form updates. We therefore rely on stochastic steepest gradient ascent to optimize the ELBO according to these parameters.

Keeping only terms which are function of  $\mathcal{W}$  or  $\beta$ , the ELBO can be rewritten, for each item  $i$ , as follows

$$\begin{aligned} \mathcal{L}_i &= \mathbb{E}_q[\log p(\mathbf{s} | \theta, \beta_i)] + \mathbb{E}_q[\log p(\mathbf{c}_i | \mathcal{W}, \beta_i)] \\ &\quad - \text{KL}(q(\beta_i) || p(\beta_i)) - \text{KL}(q(\mathcal{W}) || p(\mathcal{W})) + \text{const} \quad (8) \end{aligned}$$

with  $\mathcal{L} = \sum_i \mathcal{L}_i$ . While the first expectation and KL terms in (8) are available analytically, the second expectation over  $\log p(\mathbf{c}_i | \mathcal{W}, \beta_i)$  is intractable for general PCRL with respect to both  $\mathcal{W}$  and  $\beta_i$ . We cannot always push the expectations inward non-linear activation functions  $a_\ell$ . This makes the direct evaluation of the gradient of  $\mathcal{L}_i$  problematic. To overcome this difficulty we build a Monte Carlo estimator of the gradient of  $\mathbb{E}_q[\log p(\mathbf{c}_i | \mathcal{W}, \beta_i)]$ . To this end, we rely on the recent Rejection Sampling Variational Inference (RSVI) method (Naesseth et al., 2017), which generalizes the *reparameterization trick* (Kingma and Welling, 2014; Rezende et al., 2014).

RSVI requires continuous latent variables, and its applicability depends on whether we can sample from the variational distribution  $q(\beta; \omega)$  using the following reparameterization: (i) draw  $\epsilon \sim \pi(\epsilon; \omega)$ , (ii)  $\beta = \mathcal{G}(\epsilon, \omega)$ , where  $\mathcal{G}$  is a deterministic function (mapping) that must be differentiable with respect to  $\omega$ , and the distribution  $\pi(\epsilon; \omega)$ , defined by a rejection sampling algorithm, takes the following form,

$$\pi(\epsilon; \omega) = t(\epsilon) \frac{q(\mathcal{G}(\epsilon, \omega); \omega)}{r(\mathcal{G}(\epsilon, \omega); \omega)}, \quad (9)$$

where  $r$  and  $t$  are respectively the *proposal* and original distributions of  $\epsilon$  used in rejection sampling. In this procedure, some samples from  $t$  are not valid (and therefore rejected), here we are interested in the distribution of the accepted samples  $\pi(\epsilon; \omega)$ . For more details, please refer to the supplementary material (A.2.1) where we provide a brief review of the reparameterized acceptance-rejection algorithm in our notations.

Assuming that we have a reparameterized acceptance-rejection sampling procedure to simulate from  $q(\beta_{ik}; \omega)$ , the next step is to rewrite  $\mathbb{E}_{q(\beta_i; \omega)}[\log p(\mathbf{c}_i | \mathcal{W}, \beta_i)]$  as an expectation with respect to  $\pi(\epsilon_i; \omega)$  as follows

$$\begin{aligned} \mathbb{E}_{q(\beta_i; \omega)}[\log p(\mathbf{c}_i | \mathcal{W}, \beta_i)] \\ = \mathbb{E}_{\pi(\epsilon_i; \omega)}[\log p(\mathbf{c}_i | \mathcal{W}, \mathcal{G}(\epsilon_i, \omega))] \quad (10) \end{aligned}$$

where,  $\epsilon_i = \{\epsilon_{i1}, \dots, \epsilon_{iK}\}$ , and  $\pi$  fully factorizes over the components of  $\epsilon_i$ . The form of  $\mathcal{G}(\epsilon_i, \omega)$  will be given shortly. Based on (10) the gradient of  $\mathbb{E}_{q(\beta_i; \omega)}[\log p(\mathbf{c}_i | \mathcal{W}, \beta_i)]$  is

$$\begin{aligned} \nabla_\omega \mathbb{E}_{q(\beta_i; \omega)}[\log p(\mathbf{c}_i | \mathcal{W}, \beta_i)] \\ = \mathbb{E}_{\pi(\epsilon_i; \omega)}[\log p(\mathbf{c}_i | \mathcal{W}, \mathcal{G}(\epsilon_i, \omega)) \nabla_\omega \log \pi(\epsilon_i; \omega)] \\ + \mathbb{E}_{\pi(\epsilon_i; \omega)}[\nabla_\omega \log p(\mathbf{c}_i | \mathcal{W}, \mathcal{G}(\epsilon_i, \omega))] \quad (11) \end{aligned}$$

where we have pushed the gradient into the integral, used the log derivative-trick or REINFORCE (Glynn, 1990; Williams, 1992), and expressed integrals as expectations. All the derivations details of equations (11) and (10) are given in the supplementary material (A.2.2).

We can now form an unbiased Monte Carlo estimate of the above gradient as follows:

$$\begin{aligned} & \nabla_{\omega} \mathbb{E}_{q(\beta_i; \omega)} [\log p(\mathbf{c}_i | \mathcal{W}, \beta_i)] \\ & \simeq \frac{1}{M} \sum_{m=1}^M \log p(\mathbf{c}_i | \mathcal{W}, \beta_i^m) \nabla_{\omega} \log \frac{q(\mathcal{G}(\epsilon_i^m, \omega); \omega)}{r(\mathcal{G}(\epsilon_i^m, \omega); \omega)} \\ & + \frac{1}{M} \sum_{m=1}^M \nabla_{\omega} \log p(\mathbf{c}_i | \mathcal{W}, \beta_i^m) \end{aligned} \quad (12)$$

where  $\beta_i^m = \{\beta_{i1}^m, \dots, \beta_{iK}^m\}$ , and  $\beta_{ik}^m = \mathcal{G}(\epsilon_{ik}^m, \omega)$ , with  $\epsilon_{ik}^m \sim \pi(\epsilon_{ik}, \omega)$ . In practice we set  $M = 1$ .

Following Naesseth et al. (2017), for the Gamma random variables, we use the reparameterization proposed by Marsaglia and Tsang (2000). For a  $\text{Gamma}(\lambda_{\omega}^s, \lambda_{\omega}^r)$ , such that  $\lambda_{\omega}^s \geq 1$ , we use:

$$\mathcal{G}(\epsilon, \omega) = \frac{1}{\lambda_{\omega}^r} \left( \lambda_{\omega}^s - \frac{1}{3} \right) \left( 1 + \frac{\epsilon}{\sqrt{9\lambda_{\omega}^s - 3}} \right) \quad (13)$$

with  $\epsilon \sim t(\epsilon) = \mathcal{N}(0, 1)$ . When the shape parameter is less than 1,  $\lambda_{\omega}^s < 1$ , we use the shape augmentation technique (Marsaglia and Tsang, 2000). That is, if  $\beta \sim \text{Gamma}(\lambda^s + 1, \lambda^r)$ , and  $\beta = u^{\frac{1}{\lambda^s}} \tilde{\beta}$  with  $u \sim \mathcal{U}[0, 1]$ , then  $\beta \sim \text{Gamma}(\lambda^s, \lambda^r)$ .

Approximating the gradient of the ELBO with respect to  $\xi$  is simpler since the Gaussian satisfies the requirements of the original reparameterization trick (Kingma and Welling, 2014; Rezende et al., 2014). Roughly, the second expectation in (11) vanishes since the marginal distribution of the samples  $\epsilon$  is independent of the variational parameters  $\xi$ . Hence, the Monte Carlo estimator of  $\nabla_{\xi} \mathbb{E}_{q(\mathcal{W})} [\log p(\mathbf{c}_i | \mathcal{W}, \beta_i)]$  takes this form:

$$\begin{aligned} & \nabla_{\xi} \mathbb{E}_{q(\mathcal{W}; \xi)} [\log p(\mathbf{c}_i | \mathcal{W}, \beta_i)] \\ & \simeq \frac{1}{M} \sum_{m=1}^M \nabla_{\xi} \log p(\mathbf{c}_i | \mathcal{W}^m, \beta_i) \end{aligned} \quad (14)$$

where  $\mathcal{W}^m = \{\mathbf{W}_1^m, \dots, \mathbf{W}_L^m\}$ ,  $\mathbf{w}_{\ell, k}^m = \mathcal{T}(\boldsymbol{\eta}^m, \tilde{\xi}_{\ell}^k) = \tilde{\boldsymbol{\mu}}_{\ell}^k + \tilde{\boldsymbol{\sigma}}_{\ell}^k \odot \boldsymbol{\eta}^m$ , and  $\boldsymbol{\eta}^m \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ , the notation  $\odot$  refers to the Hadamard product.

Putting it all together, our Monte Carlo estimator for the gradient of the ELBO, is given by:

$$\begin{aligned} & \nabla_{\omega, \xi} \mathcal{L}_i \simeq I \nabla_{\omega} (\mathbb{E}_q [\log p(\mathbf{s} | \boldsymbol{\theta}, \beta_i)] - \text{KL}(q(\beta_i) || p(\beta_i))) \\ & + \frac{I}{M} \sum_{m=1}^M \nabla_{\omega, \xi} \log p(\mathbf{c}_i | \mathcal{W}^m, \beta_i^m) \\ & + \frac{I}{M} \sum_{m=1}^M \log p(\mathbf{c}_i | \mathcal{W}^m, \beta_i^m) \nabla_{\omega} \log \frac{q(\mathcal{G}(\epsilon_i^m, \omega); \omega)}{r(\mathcal{G}(\epsilon_i^m, \omega); \omega)} \\ & - \nabla_{\xi} \text{KL}(q(\mathcal{W}) || p(\mathcal{W})) \end{aligned} \quad (15)$$

With the estimator (15) in place, we perform stochastic gradient ascent over the parameters  $\omega$  and  $\xi$ . We use backpropagation to evaluate the gradients over the weights of the inference network  $\omega$ . In particular, we use RMSProp to scale the gradients before applying them. In practice, we take several stochastic gradient steps to nearly optimize the ELBO with respect to  $\omega$  and  $\xi$ , before to perform coordinate ascent step to update  $\tilde{\lambda}^{\theta}$  and  $\tilde{\phi}$ . More precisely, after each epoch of stochastic gradient ascent we update  $\tilde{\lambda}^{\theta}$  and  $\tilde{\phi}$ .

### 4.3 MISSING RATINGS ESTIMATION

Once PCRL is fit to the observations, we can estimate the unknown ratings for each user  $u$  and item  $i$  as follows

$$\hat{x}_{ui} = \mathbb{E}_q(\boldsymbol{\theta}_u^{\top} \beta_i) = \mathbb{E}_q(\boldsymbol{\theta}_u)^{\top} \mathbb{E}_q(\beta_i), \quad (16)$$

Note that this expectation is intractable with respect to the true posterior. These predicted values are then used to rank unrated items for each user so as to provide her with a recommendation list.

### 4.4 DESIRABLE PROPERTIES

The variational PCRL enjoys several desirable properties. In terms of efficiency, the operations involving user-item and item-context interactions need to be carried out only for the non-zero entries in  $\mathbf{X}$  and  $\mathbf{C}$ . It can be shown that the computational time complexity of the variational PCRL algorithm (its batch version) is linear in the number of non-zeros entries in  $\mathbf{X}$  and  $\mathbf{C}$ .

The main intuition behind PCRL is to learn item representations encoding various contextual regularities among items that are good at explaining the user behaviour. Interestingly, this intuition is reflected theoretically, as seen in the proposition below. Note that this result arises naturally from our formulation.

**Proposition 1** *Let  $q(\beta_i; \omega)$  be the variational distribution over the item factor in PCRL. Then, for fixed  $\tilde{\lambda}^{\theta}$ ,  $\tilde{\phi}$  and  $\tilde{\xi}$ , maximizing the ELBO (5) with respect to  $\omega$  is equivalent to maximizing the following criterion:*

$$\sum_i \mathbb{E}_q [\log p(\mathbf{c}_i | \mathcal{W}, \beta_i)] - \text{KL}(q(\beta_i; \omega) || \tilde{q}(\beta_i)). \quad (17)$$

where  $\tilde{q}(\beta_i)$  denotes the optimal mean-field variational distribution over the item attributes in Bayesian Poisson factorization. That is,  $\tilde{q}(\beta_i) = \prod_k \tilde{q}(\beta_{ik})$ , and  $\tilde{q}(\beta_{ik}) = \text{Gamma}(\lambda_{\beta}^s + \sum_u x_{ui} \tilde{\phi}_{uik}, \lambda_{\beta}^r + \sum_u \frac{\tilde{\lambda}_{uk}^s}{\lambda_{uk}^r})$ .

The proof is given below. The KL term in the above proposition can be viewed as a regularizer which encourages PCRL's variational factor over the items,  $q(\beta_i; \omega)$  to look like its optimal mean-field counterpart in Bayesian

Poisson factorization  $\tilde{q}(\beta_i)$ . Recall that  $\tilde{q}(\beta_i)$  is independent of the item context  $\mathbf{C}$ , and puts high probability on configurations of  $\beta_i$  that explain user preferences. This makes it clear how the collaborative PF component in PCRL guides or encourages the representation learning part to focus on extracting contextual features that might be useful for explaining user preferences. From this perspective, PCRL can be interpreted as regularizing a deep generative model with Bayesian Poisson Factorization.

**Proof.** If we fix all the variational parameters except  $\omega$ , then maximizing the ELBO with respect to the latter is equivalent to maximizing

$$\begin{aligned} \mathcal{L}_i = & \mathbb{E}_q[\log p(\mathbf{s}|\boldsymbol{\theta}, \beta_i) + \log p(\beta_i)] \\ & + \mathbb{E}_q[\log p(\mathbf{c}_i|\mathcal{W}, \beta_i) - \log q(\beta_i; \omega)] + \text{const.} \end{aligned} \quad (18)$$

In particular, we have

$$\log p(\beta_{ik}) \propto (\lambda_\beta^s - 1) \log(\beta_{ik}) - \lambda_\beta^r \beta_{ik}, \text{ and,}$$

$$\log p(s_{uik}|\theta_{uk}, \beta_{ik}) \propto s_{uik} \log(\beta_{ik}) - \theta_{uk} \beta_{ik}.$$

Therefore we get

$$\begin{aligned} \mathbb{E}_{q(\theta, s)}[\log p(\mathbf{s}|\boldsymbol{\theta}, \beta_i) + \log p(\beta_i)] = & -(\lambda_\beta^r + \sum_u \frac{\tilde{\lambda}_{uk}^s}{\tilde{\lambda}_{uk}^r}) \beta_{ik} \\ & + (\lambda_\beta^s + \sum_u x_{ui} \tilde{\phi}_{uik} - 1) \log \beta_{ik} + \text{const,} \end{aligned} \quad (19)$$

where we recognize the log (up to the normalizing constant) of the following Gamma( $\lambda_\beta^s + \sum_u x_{ui} \tilde{\phi}_{uik}$ ,  $\lambda_\beta^r + \sum_u \frac{\tilde{\lambda}_{uk}^s}{\tilde{\lambda}_{uk}^r}$ ) distribution. Adding the normalizing constant (which is independent of  $\omega$ ) and plugin (19) into (18), completes the proof. ■

## 5 EXPERIMENTS

In this section, we study the impact of item context, and our modeling assumptions, on personalized item recommendation as well as item representation learning.

**Datasets.** We use five datasets from `Amazon.com`<sup>3</sup>, provided by McAuley et al. (2015b,a). They include both user-item interactions and the ‘‘Also Viewed’’ lists that we treat as item contexts. We preprocess all datasets so that each user (resp. item) has at least ten (resp. two) ratings, and the sets of row and column items in  $\mathbf{C}$  are identical. Table 1 reports the resulting statistics.

**Comparative Models.** We benchmark PCRL<sup>4</sup> against strong comparable generative factorization models.

- MCF: Matrix Co-Factorization (Park et al., 2017) incorporates item-item relationships into Gaussian MF.

<sup>3</sup><http://jmcauley.ucsd.edu/data/amazon/>

<sup>4</sup>source code available at: <https://cornac.preferred.ai/>

Datasets	Characteristics					
	#Users	#Items	#Ratings	$nz_X$ (%)	# $nz_C$	$nz_C$ (%)
Office	3,703	6,523	53,282	0.22	108,466	0.25
Grocery	8,938	22,890	148,735	0.07	480,300	0.09
Automotive	7,280	15,635	63,477	0.05	365,634	0.15
Sports	19,049	24,095	211,582	0.04	531,148	0.09
Pet Supplies	16,462	20,049	164,017	0.05	631,102	0.16

Table 1: Statistics of the Datasets.

- PF: Bayesian Poisson Factorization (Gopalan et al., 2015) arises as a special case of our model without the context-specific components. Comparison to PF allows us to assess the impact of item contexts.
- CTPF: Collaborative Topic Poisson Factorization (Gopalan et al., 2014b) was developed for content-based recommendation, but can serve as baseline by substituting item-word matrix with item-context  $\mathbf{C}$ .
- CoCTPF: Content-only CTPF (Gopalan et al., 2014b) is a variant of CTPF without the document topic offsets; please refer to (Gopalan et al., 2014b) for details. Comparison to CoCTPF allows us to assess the impact of our modeling choice of multilayered representation learning, as opposed to PF, for item context.
- RL+PF: Representation Learning + PF is a two-stage pipelined approach, which models item context independently from user preferences. First, it infers  $q(\beta)$  from  $\mathbf{C}$  using PCRL’s representation learning-specific part. Second, it performs PF on  $\mathbf{X}$  to infer  $q(\theta)$  while holding the item factors fixed. Comparison to RL+PF allows us to assess the benefit of our unified modeling.

**Experimental Setup.** For each dataset, we randomly select 80% of the ratings as training data and the remaining 20% as test data. Random selection is carried out three times independently on each dataset. The reported result is the average performance over the three samples.

Following previous works (Gopalan et al., 2014a, 2015), we set the number of latent dimensions for user preferences  $\theta$  and item attributes  $\beta$  to 100. In all experiments, we use a two-layer PCRL ( $\mathbf{z}_1, \beta$ ) with dimensions (100, 300) in the item representation learning component. The activation functions at the layers ( $\mathbf{c}, \mathbf{z}_1$ ) are set to (sigmoid, relu). Similarly, we use a two-layer inference network (encoder) with dimensions (300, 100 + 100)—recall that this network outputs Gamma variational parameters, a total of 100 (shape) + 100 (rate) parameters—and activation functions (relu, sofplus). When necessary we add a small offset to ensure strict positivity, e.g., the rate of the Poisson, the shape and rate of the Gamma, all must be positive. To encourage sparse latent representations, we set Gamma prior parameters ( $\lambda^s, \lambda^r$ ) to (0.3, 0.3)—resulting in exponen-



Table 2: Average recommendation accuracy.

	Metric	MCF	PF	CTPF	CoCTPF	RL+PF	PCRL
Office Prod.	nDCG	0.1525	0.1663	0.1718	0.1806	0.1551	<b>0.1974</b>
	MRR	0.0239	0.0414	0.0467	0.0558	0.0237	<b>0.0708</b>
	Pre@20	0.0041	0.0096	0.0111	0.0129	0.0048	<b>0.0156</b>
	Rec@20	0.0293	0.0541	0.0615	0.0768	0.0325	<b>0.0873</b>
	Pre50	0.0033	0.0077	0.0075	0.0095	0.0039	<b>0.0116</b>
	Rec50	0.0569	0.0970	0.1021	0.1392	0.0654	<b>0.1627</b>
Grocery	nDCG	0.1286	0.1568	0.1553	0.1717	0.1295	<b>0.1801</b>
	MRR	0.0145	0.0452	0.0429	0.0529	0.0098	<b>0.0652</b>
	Pre20	0.0024	0.0095	0.0095	0.0116	0.0017	<b>0.0134</b>
	Rec20	0.0191	0.0571	0.0591	0.0739	0.0109	<b>0.0751</b>
	Pre50	0.0019	0.0070	0.0072	0.0086	0.0015	<b>0.0098</b>
	Rec50	0.0353	0.1021	0.1090	0.1213	0.0234	<b>0.1339</b>
Automotive	nDCG	0.1186	0.1123	0.1124	0.1417	0.1225	<b>0.1453</b>
	MRR	0.0121	0.0100	0.0103	0.0337	0.0111	<b>0.0350</b>
	Pre20	0.0022	0.0015	0.0016	0.0058	0.0017	<b>0.0063</b>
	Rec20	0.0228	0.0132	0.0143	<b>0.0566</b>	0.0147	0.0536
	Pre50	0.0016	0.0010	0.0012	0.0038	0.0016	<b>0.0043</b>
	Rec50	0.0393	0.0233	0.0262	<b>0.0920</b>	0.0325	0.0913
Sports	nDCG	0.1122	0.1179	0.1189	0.1398	0.1190	<b>0.1524</b>
	MRR	0.0071	0.0122	0.0119	0.0297	0.0073	<b>0.0375</b>
	Pre20	0.0011	0.0018	0.0022	0.0054	0.0014	<b>0.0070</b>
	Rec20	0.0096	0.0143	0.0170	0.0431	0.0113	<b>0.0507</b>
	Pre50	0.0009	0.0013	0.0017	0.0038	0.0013	<b>0.0051</b>
	Rec50	0.0192	0.0273	0.0318	0.0759	0.0298	<b>0.0942</b>
Pet Supplies	nDCG	0.1201	0.1288	0.1317	0.1585	0.1210	<b>0.1626</b>
	MRR	0.0136	0.0207	0.0237	0.0441	0.0094	<b>0.0461</b>
	Pre20	0.0022	0.0029	0.0034	0.0079	0.0019	<b>0.0088</b>
	Rec20	0.0237	0.0271	0.0314	0.0752	0.0167	<b>0.0776</b>
	Pre50	0.0016	0.0021	0.0028	0.0055	0.0016	<b>0.0063</b>
	Rec50	0.0397	0.0481	0.0561	0.1301	0.0359	<b>0.1455</b>

tially shaped Gamma distributions with mean equal to 1. For an illustration, please refer to Figure 2 in (Cemgil, 2009). We follow the same strategy, grid search, as in (Park et al., 2017) to set the different hyperparameters of MCF. In order for the comparisons to be fair, we use the same random parameters to initialize all PF-based models, where it is possible.

**Item Recommendation.** Here we look into the quality of item recommendation, and discuss item representation later. We assess the recommendation accuracy on a set of held-out items—the test set. We retain four widely used measures for top- $M$  recommendation, namely the Normalized Discount Cumulative Gain (nDCG), Mean Reciprocal Rank (MRR), Precision@ $M$  ( $P@M$ ) and Recall@ $M$  ( $R@M$ ), where  $M$  is the number of items in the recommendation list (Bobadilla et al., 2013). Intuitively, nDCG and MRR measures the ranking quality of a model, while Precision@ $M$  and Recall@ $M$  assess the quality of a user’s top- $M$  recommendation list. These measures vary from 0.0 to 1.0 (higher is better).

Table 2 depicts the average performances<sup>5</sup> of the competing models in terms of different metrics, over all datasets. For the sake of completeness we also report, in Table 3, the average log-likelihood values for the Poisson models, i.e.,  $\log p(\mathbf{X}|\boldsymbol{\theta}, \boldsymbol{\beta})$ , where we set  $\boldsymbol{\theta}$  and  $\boldsymbol{\beta}$  to their mean values under the corresponding variational distribution.

<sup>5</sup>Most of the standard deviation values are of order  $1e-3/1e-4$ , we do not report them to fit Table 2 into one column.

Table 3: Comparison of Poisson log-likelihood.

Models	Office Prod.	Grocery	Automotive	Sports	Pet Supplies
PF	-210522	-680546	-355671	-1187849	-838712
CTPF	-208633	-681832	-354239	-1180927	-838910
CoCTPF	-207840	-656676	-336319	-1138744	-786326
RL+PF	-227454	-761403	-341730	-1178502	-887624
PCRL	<b>-199066</b>	<b>-649054</b>	<b>-322889</b>	<b>-1061088</b>	<b>-760935</b>

The main points from these results are as follows.

*Item context is useful for personalized recommendation.*

The proposed PCRL substantially outperforms the other competing models in virtually all cases. In particular, the major difference between the original PF and our proposed PCRL as well as CTPF or CoCTPF is that the latter models incorporate item context. We can therefore attribute the performance improvements reached by those over PF to the modeling of the item context.

*Poisson Factorization performs better than its Gaussian counterpart.* Effectively CoCTPF is the closest Poisson alternative to the Gaussian MCF. The former outperforms the latter in all cases. Even when augmented with contextual item information, the Poisson remains a better alternative than the Gaussian for modeling user preferences, which is in line with the findings of previous work on PF.

*The hierarchical (multilayered) structure in PCRL is useful.* The model PCRL can be viewed as an alternative to CoCTPF, where a multilayered generative model is substituted for PF to model item contexts. From Tables 2 and 3, we note that PCRL substantially outperforms CoCTPF on almost all datasets and across all metrics, except in terms of recall on Automotive. Since the main difference between the two approaches lies in how they model item context, these results suggest that our multilayered architecture does a better job than PF in extracting latent features from item’s contexts.

*Joint modeling or learning is beneficial.* A key point to PCRL is to model user preferences and item contexts jointly. As Tables 2 and 3 show, PCRL outperforms the two-stage pipelined model RL+PF. Quite surprisingly, the latter performs even worse than PF on almost all datasets. This demonstrates the importance of joint modeling, and suggests that the PCRL’s collaborative component plays an important role in guiding item representation learning towards extracting contextual features that are relevant for explaining user preferences. Whereas modeling the item context independently yields item representations that capture other item aspects, which are not necessarily as good for predicting user preferences.

To gain further insight into the results, especially the latter two points above, we conduct another series of experiments where we compare the quality of the item representations produced by the different models.

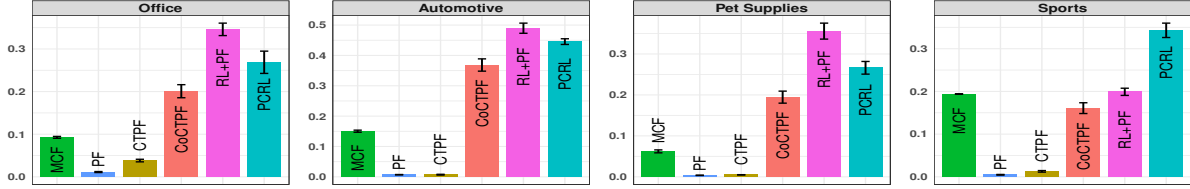


Figure 2: Average NMI over different datasets.

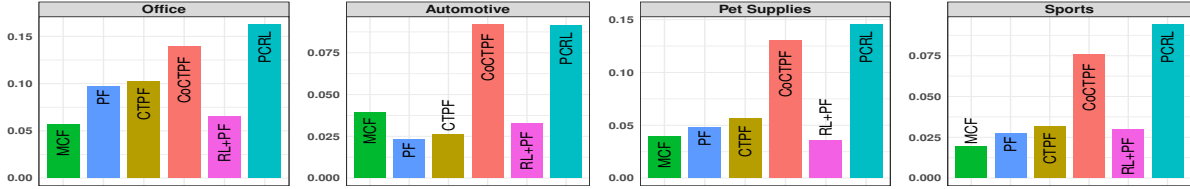


Figure 3: Average Recall@50 over different datasets.

**Item Representations.** Evaluating the quality of item representations is a challenging task. Here, we propose to make such an evaluation in terms of clustering. We seek to assess how well the representations produced by each model are good at organizing items into meaningful clusters. As evaluation measure, we use Normalized Mutual Information (NMI) (Strehl and Ghosh, 2002). Intuitively, NMI quantifies how much the estimated clustering is informative about the true clustering. As the “true” clustering, we retain the ten most frequent item categories (classes) in each dataset; these categories per dataset are listed in the supplementary material (B). We do not consider Grocery in this experiment, since its category labels are not available.

To form clusters based on learned item representations, we use the spherical k-means (*Skmeans*) (Dhillon and Modha, 2001). We perform fifty runs of (*Skmeans*), with different initial random points, and report the average NMI of the ten best runs—in terms of the *Skmeans*’ criterion—as the final results. The fifty random starting points used by *Skmeans* are the same across all models.

Figure 2 reports the clustering results. For reference, Figure 3 reproduces the Recall@50 (the results are consistent across all metrics) on the item recommendation task.

PF that relies solely on user-item interactions obtains the worst clustering results. Such sparse information is not rich enough to allow PF infer relationships among items. The other models that use contextual information perform better. In particular, we note that PCRL produces representations that are better suited to organize items into categories than the CoCTPF models. This provides additional empirical support for the importance of our hierarchical architecture to model items’ contexts.

Interestingly, RL+PF performs relatively well on clustering (Figure 2) even as it performs rather poorly on recommendation (Figure 3). One possible explanation of

this phenomenon is that RL+PF focuses on item similarity. While this is beneficial for clustering, this might not always be useful for recommendation. Hypothetically, two similar items may be alternatives. Instead of recommending alternatives to an item that a user has purchased, it may be useful to recommend complementary items (which may not belong to the same category).

## 6 DISCUSSION

PCRL composes Bayesian Poisson factorization with a multilayered latent variable model to join both sources of data: user preferences and item contexts. Empirical results provide strong support for the benefits of our modeling framework and reflect the underlying assumption in PCRL, namely: the collaborative component guides the item representation learning towards extracting contextual features that are useful for the recommendation task, whereas the representation learning encourages the collaborative part to rely on item’s contexts to explain recommendations, alleviating data sparsity.

While our focus here has been on item context, PCRL could potentially be extended to learn item representations from other modalities, e.g., text, images, etc. Another interesting direction of future work, is to investigate deeper variants of PCRL which would improve the feature learning.

We make PCRL’s implementation publicly available as part of the CORNAC<sup>6</sup> recommendation library.

## Acknowledgments

This research is supported by the National Research Foundation, Prime Minister’s Office, Singapore under its NRF Fellowship Programme (Award No. NRF-NRFF2016-07).

<sup>6</sup><https://cornac.preferred.ai/>

## References

- Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE TPAMI*, 35(8):1798–1828.
- Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.
- Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *JASA*.
- Bobadilla, J., Ortega, F., Hernando, A., and Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-based systems*, 46:109–132.
- Canny, J. (2004). Gap: a factor model for discrete data. In *SIGIR*, pages 122–129.
- Cemgil, A. T. (2009). Bayesian inference for nonnegative matrix factorisation models. *Computational Intelligence and Neuroscience*, 2009.
- Chaney, A. J., Blei, D. M., and Eliassi-Rad, T. (2015). A probabilistic model for using social networks in personalized item recommendation. In *RecSys*, pages 43–50.
- Charlin, L., Ranganath, R., McInerney, J., and Blei, D. M. (2015). Dynamic poisson factorization. In *RecSys*, pages 155–162.
- Dhillon, I. S. and Modha, D. S. (2001). Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1-2):143–175.
- Glynn, P. W. (1990). Likelihood ratio gradient estimation for stochastic systems. *CACM*, 33(10):75–84.
- Gopalan, P., Hofman, J. M., and Blei, D. M. (2015). Scalable recommendation with hierarchical poisson factorization. In *UAI*, pages 326–335.
- Gopalan, P., Ruiz, F. J., Ranganath, R., and Blei, D. (2014a). Bayesian nonparametric poisson factorization for recommendation systems. In *Artificial Intelligence and Statistics*, pages 275–283.
- Gopalan, P. K., Charlin, L., and Blei, D. (2014b). Content-based recommendations with poisson factorization. In *NIPS*, pages 3176–3184.
- Hu, Y., Koren, Y., and Volinsky, C. (2008). Collaborative filtering for implicit feedback datasets. In *ICDM*, pages 263–272.
- Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233.
- Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. *ICLR*.
- Kingman, J. F. C. (1993). *Poisson processes*. Wiley Online Library.
- Koenigstein, N., Dror, G., and Koren, Y. (2011). Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy. In *RecSys*, pages 165–172.
- Koren, Y., Bell, R., and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8).
- Li, X. and She, J. (2017). Collaborative variational autoencoder for recommender systems. In *KDD*, pages 305–314.
- Liang, D., Altosaar, J., Charlin, L., and Blei, D. M. (2016). Factorization meets the item embedding: Regularizing matrix factorization with item co-occurrence. In *RecSys*, pages 59–66.
- Ma, H., Yang, H., Lyu, M. R., and King, I. (2008). Sorec: social recommendation using probabilistic matrix factorization. In *CIKM*, pages 931–940.
- Marsaglia, G. and Tsang, W. W. (2000). A simple method for generating gamma variables. *TOMS*, 26(3):363–372.
- McAuley, J., Pandey, R., and Leskovec, J. (2015a). Inferring networks of substitutable and complementary products. In *KDD*, pages 785–794.
- McAuley, J., Targett, C., Shi, Q., and Van Den Hengel, A. (2015b). Image-based recommendations on styles and substitutes. In *SIGIR*, pages 43–52.
- Mnih, A. and Salakhutdinov, R. R. (2008). Probabilistic matrix factorization. In *NIPS*, pages 1257–1264.
- Naesseth, C., Ruiz, F., Linderman, S., and Blei, D. (2017). Reparameterization gradients through acceptance-rejection sampling algorithms. In *AISTATS*, pages 489–498.
- Park, C., Kim, D., Oh, J., and Yu, H. (2017). Do also-viewed products help user rating prediction? In *WWW*, pages 1113–1122.
- Park, Y.-J. and Tuzhilin, A. (2008). The long tail of recommender systems and how to leverage it. In *RecSys*, pages 11–18. ACM.
- Ranganath, R., Tang, L., Charlin, L., and Blei, D. (2015). Deep exponential families. In *AISTATS*, pages 762–771.
- Rao, N., Yu, H.-F., Ravikumar, P. K., and Dhillon, I. S. (2015). Collaborative filtering with graph information: Consistency and scalable methods. In *NIPS*, pages 2107–2115.
- Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, pages 1278–1286.

- Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *WWW*, pages 285–295. ACM.
- Shan, H. and Banerjee, A. (2010). Generalized probabilistic matrix factorizations for collaborative filtering. In *ICDM*, pages 1025–1030.
- Singh, A. P. and Gordon, G. J. (2008). Relational learning via collective matrix factorization. In *KDD*, pages 650–658.
- Strehl, A. and Ghosh, J. (2002). Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *JMLR*, 3:583–617.
- Wang, C. and Blei, D. M. (2011). Collaborative topic modeling for recommending scientific articles. In *KDD*, pages 448–456.
- Wang, H., Wang, N., and Yeung, D.-Y. (2015). Collaborative deep learning for recommender systems. In *KDD*, pages 1235–1244.
- Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Reinforcement Learning*, pages 5–32. Springer.
- Zhou, M., Cong, Y., and Chen, B. (2016). Augmentable gamma belief networks. *Journal of Machine Learning Research*, 17(163):1–44.
- Zhou, T., Shan, H., Banerjee, A., and Sapiro, G. (2012). Kernelized probabilistic matrix factorization: Exploiting graphs and side information. In *SDM*, pages 403–414.

---

# Reforming Generative Autoencoders via Goodness-of-Fit Hypothesis Testing

---

**Aaron Palmer**  
Computer Science Dept.  
University of Connecticut  
Storrs, CT 06269

**Dipak K. Dey**  
Statistics Dept.  
University of Connecticut  
Storrs, CT 06269

**Jinbo Bi**  
Computer Science Dept.  
University of Connecticut  
Storrs, CT 06269

## Abstract

Generative models, while not new, have taken the deep learning field by storm. However, the widely used training methods have not exploited the substantial statistical literature concerning parametric distributional testing. Having sound theoretical foundations, these goodness-of-fit tests enable parts of the black box to be stripped away. In this paper we use the Shapiro-Wilk and propose a new multivariate generalization of Shapiro-Wilk to respectively test for univariate and multivariate normality of the code layer of a generative autoencoder. By replacing the discriminator in traditional deep models with the hypothesis tests, we gain several advantages: objectively evaluate whether the encoder is actually embedding data onto a normal manifold, accurately define when convergence happens, explicitly balance between reconstruction and encoding training. Not only does our method produce competitive results, but it does so in a fraction of the time. We highlight the fact that the hypothesis tests used in our model asymptotically lead to the same solution of the  $L_2$ -Wasserstein distance metrics used by several generative models today.

## 1 INTRODUCTION

Recently a large variety of generative models have been proposed such as generative adversarial networks and generative autoencoders. A widely-used way to construct such a network requires training of a generator and a discriminator. There are great needs to understand the statistical foundation of these generative models. On the other hand, there exists substantial statistical literature

concerning parametric distributional hypothesis testing with a solid theoretical base. One particular group of deep generative models which, we show in this study, can benefit from hypothesis testing is the generative autoencoder (GAE). The objective of these models is to reconstruct the input as accurately as possible, while constraining the code layer to a specified distribution, usually normal. Often times once training has ended, this code layer distribution does not in fact match the required distribution. The spirit of these models is to embed data into a distribution that matches the prior to enable sampling, and thus it is of utmost importance we have ways to assess the quality of the fit. In other words, if the embedded distribution does not match the prior that is used to sample and generate instances, the method does not work in theory.

In this paper we propose to use goodness-of-fit hypothesis tests of normality on the code layer of an autoencoder as a new type of critic in both the univariate and multivariate case. Doing so leads to an adversary-free optimization problem. These hypothesis tests provide a more direct way to measure if the data representation, the latent code layer, matches a pre-specified distribution. More specifically, we test for normality using a composite test:

$$H_0 : \mathbf{X} \in \mathcal{G} \quad \text{vs} \quad H_1 : \mathbf{X} \notin \mathcal{G} \quad (1)$$

where  $\mathcal{G} = \{\pi : \pi = \mathcal{N}(\mu, \Sigma), -\infty < \mu < \infty, \Sigma \text{ is positive semi-definite (p.s.d)}\}$ . Many tests for comparing two distributions can be used in our model<sup>1</sup>. We specifically focus on the well studied univariate Shapiro-Wilk test (Shapiro and Wilk, 1965) and propose a new multivariate generalization of the Shapiro-Wilk test to demonstrate the effectiveness of the new method. We further highlight a link between these methods and those based on the Wasserstein distance by drawing attention to the fact that the Shapiro-Wilk test and the  $L_2$ -Wasserstein distance lead to the same asymptotic solution.

The remainder of the paper follows as such. Section 2 covers existing work that is most closely related to our method. In section 3 we present the basics of hypothe-

sis testing followed by a recap of the Shapiro-Wilk test and propose its multivariate generalization. Section 4 describes the new method in detail, followed by theoretical analysis in Section 5 where we explore the linkage between the hypothesis tests and several distance-based methods of training. Section 6 discusses the empirical results. Section 7 presents a discussion of the new method, followed by a conclusion section 8.

**Notation.** Boldface capital letters, e.g.,  $\mathbf{Y}$ , denote matrices while boldface lower case, e.g.,  $\mathbf{y}$ , denote vectors. Scalar values are denoted by lower case letters and no font change, such as  $y_i$ . Upper case without font change denote test statistics. Calligraphic capital letters, e.g.,  $\mathcal{Y}$ , denote sets. Probability density functions (PDF) are represented as  $p(\mathbf{z})$ , while cumulative distribution functions are the upper case version  $P(\mathbf{z})$ . Any modifications to this are explained in their respective context.

## 2 RELATED WORK

The original generative adversarial network (GAN) (Goodfellow et al., 2014) sparked a surge in generative models, parameterized by neural networks, that has yet to abate. As this paper is focused on autoencoders, GAN can be thought of as just the decoder part of a regular autoencoder. This decoder,  $G(\cdot)$ , endeavors to learn a mapping from the sampled prior,  $p(\mathbf{z})$ , to the data distribution  $p(\mathbf{x})$ . An auxiliary network called the discriminator,  $D(\cdot)$ , serves to discern how close the generated data distribution,  $p_g(\mathbf{z})$ , is to the true data distribution  $p(\mathbf{x})$ . Training a GAN amounts to the widely known two-player minimax game  $\min_G \max_D V(D, G) = \mathbb{E}_{\mathbf{x} \sim p_d(\mathbf{x})} [\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))]$ .

Using an autoencoder styled network to train a generative model is not new; the main goal being to understand the code layer distribution given data,  $q(\mathbf{z}|\mathbf{x})$ , while minimizing a reconstruction error. These GAEs tend to fall into several classes dictated by their training and generating mechanisms, and include adversarial methods, variational methods, MCMC based procedures, and the most closely related to our work, statistical hypothesis tests.

The adversarial autoencoder (AAE) (Makhzani et al., 2015) is a modification of the original GAN in which an encoder network is included, and where the discriminator is shifted from the decoder network to the latent code space. The encoder creates the encoding distribution  $q(\mathbf{z}|\mathbf{x})$  which defines an aggregated posterior distribution  $q(\mathbf{z}) = \int_{\mathbf{x}} q(\mathbf{z}|\mathbf{x}) p_{data}(\mathbf{x}) d\mathbf{x}$  where  $p_{data}(\mathbf{x})$  is the input data distribution. As in GAN, training the AAE, in part, amounts to a minimax game between a generator network,  $G(\cdot)$ , and a discriminator network,  $D(\cdot)$  where the objective is to have  $q(\mathbf{z})$  match  $p(\mathbf{z})$ , the specified

prior distribution defined over the latent space  $\mathcal{Z}$ . The decoder maps back to data space  $\mathcal{X}$  giving  $p(\mathbf{x}|\mathbf{z})$ . A reconstruction loss is minimized as usual.

However, aside from possible mode collapse issues, there are questions regarding how the generator and discriminator should be balanced during training, the issue of when to stop still has not been satisfactorily addressed. Within the adversarially trained autoencoders it is possible to vary the loss function used in the discriminator. One possible change is to use a Wasserstein loss (Arjovsky et al., 2017) (WGAN) which alleviates the vanishing gradient problem. Improvements to the WGAN include the addition of a gradient penalty (Gulrajani et al., 2017). In (Tolstikhin et al., 2017) the authors modify the AAE to use a Wasserstein distance between the target distribution and the model distribution.

Variational autoencoder (VAE) methods include the work of (Kingma and Welling, 2013; Rezende et al., 2014; Mnih and Gregor, 2014). Despite being some of the most successful methods for generation, they have been found to produce unrealistic or blurry samples (Dosovitskiy and Brox, 2016). The VAE model makes use of a random decoder mapping  $p(\mathbf{x}|\mathbf{z})$ . Moreover, there is no auxiliary network needed for discrimination. A third line of thought comes from modification of the traditional autoencoder paradigm so as to recover the density using MCMC. These include (Rifai et al., 2012; Bengio et al., 2013b, 2014) and attempt to use contraction operators, or denoising criteria in order to generate a Markov chain by repeated perturbations during the encoding phase. However, it has been a challenge to ensure adequate mixing in that process (Bengio et al., 2013a).

To the best of our knowledge there is only one method, aside from our work, that falls into the class of statistical hypothesis tests for training generative networks. It is based on the maximum mean discrepancy (MMD) (Gretton et al., 2007, 2012). Two works utilizing the MMD for training came out simultaneously (Li et al., 2015) and (Dziugaite et al., 2015), each taking a different approach. Li et al. used the MMD on features learned from the autoencoder to shape the distribution of the output layer of the network to create a generative moment matching network (GMMN). On the other hand, Dziugaite et al. applied the MMD to directly compare the generated against true data. This latter method is the closest to our work. When using the MMD, the bandwidth parameter in the kernel plays a crucial role in determining the statistical efficiency of MMD, and it is still an open problem how to find its optimal value. Moreover, using kernels in MMD requires that the computation of the objective function scales quadratically with the amount of data. This is due to the requirement of a linear increase in sample size as

dimensionality increases, and is necessary to ensure the power, covered next, goes to 1 as  $n \rightarrow \infty$ . In (Li et al., 2017) the authors propose to mitigate the bandwidth problem by using adversarial kernel learning to replace the fixed Gaussian kernel in the GMMN, while in (Sutherland et al., 2016) the authors propose to maximize the power of the statistical test based on the MMD.

### 3 STATISTICAL HYPOTHESIS TESTS

Distinguishing between two distributions is often carried out in the form of a hypothesis test. Suppose  $\theta$  is a quantity of interest, the format of a hypothesis test between the null,  $H_0$ , and the alternative,  $H_1$ , is:  $H_0 : \theta \in \Theta_0$  vs  $H_1 : \theta \in \Theta_0^c$ . To understand a hypothesis test the concept of statistical power is required. Two types of errors associated with hypothesis testing exist: type I, and type II. A type I error occurs when the null is rejected when it is true; the rate of this is called  $\alpha$ . A type II error occurs when the null is not rejected when the alternative is true, its rate defined as  $\beta$ . Power is defined to be  $1 - \beta$ . It is not possible to control both type I and type II errors; therefore it is necessary to pre-specify the  $\alpha$  one is willing to tolerate. The more powerful the test, the better. By utilizing information about the null distribution, a test statistic can be computed such that, for a given  $\alpha$ , if its value is unlikely to be observed, then  $H_0$  is rejected in favor of the alternative hypothesis' conclusion. This discerning threshold is called the critical value, and comes from the null distribution of the test statistic. When this distribution is known, a p-value, which is the observed significance level of the test, can be calculated. The p-value is bounded between 0 and 1, and can be interpreted as the probability the test statistic being at least as extreme as the test statistic calculated on the sample under  $H_0$ . The p-value affords the ability for different users to judge whether to reject or fail to reject  $H_0$ .

Testing for goodness-of-fit takes up the task of testing whether the underlying data distribution belongs to some given family of distribution functions. One such example is determining if a sample  $x_1, \dots, x_n$  is normally distributed or not, for which the hypothesis test is denoted in Eq.(1). For a more thorough discussion of hypothesis testing and goodness-of-fit see (Casella and Berger, 2002; Lehmann and Romano, 2006).

Hypothesis testing techniques fall into several sub-categories. (Seier, 2002) describes these sub-categories as those tests belonging to: skewness and kurtosis tests, empirical distribution function tests, regression and correlation tests, and others. Hypothesis tests can also be split into parametric vs non-parametric. Parametric hypothesis tests make assumptions about the underlying distribution while non-parametric hypothesis tests (also

called distribution-free tests) do not. In this paper we focus on a test containing a parametric null hypothesis.

#### 3.1 UNIVARIATE: SHAPIRO-WILK TEST

A goodness-of-fit test for normality is the Shapiro-Wilk (SW) test. In a Monte Carlo simulation by (Razali et al., 2011) the authors compared the power of the SW test to several non-parametric tests on various alternative distributions concluding it was the most powerful. The SW test is a composite parametric test to determine if a univariate data sample comes from a normal distribution. The original SW test was limited to a sample size between 3 and 50, but (Royston, 1982) extended the approach to use up to 2000 samples. The SW original test statistic,  $W$ , is calculated as

$$W = \frac{(\sum_{i=1}^n a_i x_{(i)})^2}{\sum_{i=1}^n (x_i - \bar{x})^2}. \quad (2)$$

In the Eq.(2)  $x_{(i)}$  represents the  $i^{th}$  ordered statistic of the sample. The constants  $a_i$  are given by  $(a_1, a_2, \dots, a_n) = \frac{\mathbf{m}^T \mathbf{V}^{-1}}{(\mathbf{m}^T \mathbf{V}^{-1} \mathbf{V}^{-1} \mathbf{m})^{1/2}}$ , where  $\mathbf{m}$  is a vector consisting of the  $n$  expected values of the order statistics of independent and identically distributed random variables samples from the standard normal distribution.  $\mathbf{V}$  is the covariance matrix of those order statistics. The most extreme order statistics are weighted the largest, and decrease when approaching the median. This property of the SW test will be important in later sections. Calculation of the constants  $a_i$  can be computationally demanding, prompting Royston in (Royston, 1992) to approximate these coefficients. He found that for  $12 \leq n \leq 2000$  a two-parameter log-normal distribution fitted the upper half of the empirical distribution of  $1 - W$ . The associated p-value for  $W$  is referred to the upper tail of  $\mathcal{N}(0, 1)$ . Using the hypothesis test defined in Eq.(1), the SW test fails to reject  $H_0$  if  $W > W_\alpha$ , or the p-value is larger than  $\alpha$ , where  $W_\alpha$  is the critical value based on the chosen confidence level. Three analytical properties that will become useful in the later sections, originally presented as lemmas in (Shapiro and Wilk, 1965), are cited as follows:

**Lemma 3.1.**  $W$  is scale and origin invariant.

**Lemma 3.2.**  $W$  has a maximum value of 1

**Lemma 3.3.** The minimum value of  $W$  is  $\frac{na_1^2}{(n-1)}$

As our approach makes use of this test as a new loss function, one must be cognizant of its computational complexity. Enjoying the benefits of a strong test at the cost of long run time may not be appealing. However, the  $a_i$  are calculated a single time prior to training and are stored. The actual time complexity of the SW test during training is  $O(n \log(n))$ .

### 3.2 A NEW MULTIVARIATE GENERALIZATION OF SHAPIRO-WILK

**Definition 3.1.** (Multivariate Normal (Rao et al., 1973)) A  $d$ -dimensional random variable  $\mathbf{u}$ , that is, a random variable  $\mathbf{u}$  taking values in  $E_d$  (Euclidean space of  $d$ -dimensions) is said to have a  $d$ -variate normal distribution  $\mathcal{N}_d$  if and only if every linear function of  $\mathbf{u}$  has a univariate normal distribution.

In a review by (Mecklin and Mundfrom, 2005) the authors noted more than 50 methods for testing multivariate normality. However, finding a multivariate test that is both powerful, and has low time complexity proved challenging. This necessitated the creation of a new multivariate hypothesis test that was able to make use of the strengths of SW, and relies on a well-known characterization of the multivariate normal (MVN) distribution.

**Proposition 3.4.**  $\mathbf{x} \sim \mathcal{N}_d(\mu, \Sigma)$  if and only if  $\mathbf{z} = \Sigma^{-\frac{1}{2}}(\mathbf{x} - \mu) \sim \mathcal{N}_d(\mathbf{0}, \mathbf{I})$ .

Letting  $\bar{\mathbf{x}}$  and  $\mathbf{S}$  be respectively the sample mean and covariance matrix, define  $\mathbf{S}^{-\frac{1}{2}}$  as the symmetric positive definite square root of the inverse of  $\mathbf{S}$ . Therefore, when  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n \sim \mathcal{N}_d(\mu, \Sigma)$ , then  $\mathbf{z}_i = \mathbf{S}^{-\frac{1}{2}}(\mathbf{x}_i - \bar{\mathbf{x}}) \quad \forall i = 1, \dots, n$  should be approximately  $\mathcal{N}_d(\mathbf{0}, \mathbf{I})$ . Under the assumption that observations are independent, and writing  $\mathbf{z}_i = (z_{i1}, z_{i2}, \dots, z_{id})^T$ , then  $z_{ij} \sim \mathcal{N}(0, 1)$  approximately for each  $j = 1, \dots, d$  and  $i = 1, \dots, n$ .

To test the null hypothesis that the sample  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$  is from  $\mathcal{N}_d(\mu, \Sigma)$  where  $\mu$  and  $\Sigma$  are unknown we propose to vectorize the entire  $\mathbf{Z}$  matrix as  $\mathbf{z}_{vec} = \text{vec}(\mathbf{Z}) = (z_{11}, z_{12}, \dots, z_{nd})^T$ , and then use the SW test statistic of Eq.(2) on  $\mathbf{z}_{vec}$ . Under  $H_0$ ,  $W$  is expected to be close to 1. This multivariate generalization of the Shapiro-Wilk test (MSW) does not require any corrections for multiple testing, nor any simulation to calculate new critical values. Furthermore, it inherits the same good power properties while keeping the test complexity at  $O(nd \log(nd))$ .

### 3.3 SHAPIRO-WILK ASYMPTOTICS

For years after the original (Shapiro and Wilk, 1965) paper, the distribution of  $W$  remained unknown. While variations of the original  $W$  statistic were proposed, it was the modification by (De Wet et al., 1972) that produced the first correlation normality test with known asymptotic distribution. The de Wet-Venter statistic  $W^*$  is defined as

$$W^* = \sum_i \left( \frac{x_{(i)} - \bar{x}}{s_n} - \Phi^{-1} \left[ \frac{i}{(n+1)} \right] \right)^2, \quad (3)$$

where  $\Phi^{-1}(\cdot)$  is the inverse normal cumulative density function. In their paper it was shown that  $W^*$  converges

in distribution to

$$2n(1 - W^{*\frac{1}{2}}) - a_n \xrightarrow{\mathcal{D}} \xi, \quad (4)$$

where  $\xi = \sum_3^\infty \frac{y_i^2 - 1}{i}$ ,  $\{y_i, i \geq 1\}$  is a sequence of independent and identically distributed  $\mathcal{N}(0, 1)$  random variables, and with  $a_n = \frac{1}{n+1} \left\{ \sum_{i=1}^n \frac{j(1-j)}{(\phi\{\Phi^{-1}(j)\})^2} - \frac{3}{2} \right\}$ ,

where  $j = \frac{i}{n+1}$ , and  $\phi(\cdot)$  is the standard normal density. (Verrill and Johnson, 1983; Fotopolous et al., 1984) showed that the Shapiro-Francia (Shapiro and Francia, 1972) statistic  $W^\dagger$  and the de Wet-Venter statistic  $W^*$  were asymptotically equivalent via convergence in probability  $n(W^{*\frac{1}{2}} - W^{\dagger\frac{1}{2}}) \xrightarrow{\mathcal{P}} 0$ . (Leslie et al., 1986) produced the final result connecting Shapiro-Wilk to Shapiro-Francia showing that  $n(W^{\frac{1}{2}} - W^{\dagger\frac{1}{2}}) \xrightarrow{\mathcal{P}} 0$ .

## 4 PROPOSED GENERATIVE MODEL

We propose to replace the discriminator neural network with a goodness-of-fit hypothesis test; specifically the Shapiro-Wilk hypothesis test, and its multivariate generalization. As the main idea here, maximizing the associated test statistic forces the encoder to encode data to a distribution (from which the decoder learns to generate data) so that the null hypothesis is not rejected, hence allowing  $q(\mathbf{z})$  to be indistinguishable from the true distribution  $p(\mathbf{z})$ . Lemma 3.2 gives a target value for maximization, and from lemma 3.1 it can be seen that maximizing  $W$  to  $W \geq W_\alpha$  results in the encoding distribution  $q(\mathbf{z}|\mathbf{x})$  indistinguishable from the family  $\mathcal{G} = \{\pi : \pi = \mathcal{N}(\mu, \Sigma)\}$ .

Constraining the network to map  $q(\mathbf{z}|\mathbf{x})$  to a specific distribution in the Gaussian class, for instance  $\mathcal{N}(\mathbf{0}, \mathbf{I})$ , as is often done, is not necessary. It may present challenges when generating data if the decoder is not robust to deviations from the prior,  $p(\mathbf{z})$ . Our new model allows the network to find the right  $q(\mathbf{z}) = \pi^* \in \mathcal{G} = \{\pi : \pi = \mathcal{N}(\mu, \Sigma)\}$  that minimizes the reconstruction loss without requiring a specific prior as long as it is in the class. In the univariate case, SW is directly applied to the encoded data, and the decoder works off of this code layer. When training is complete,  $(\hat{\mu}, \hat{\Sigma})$  are estimated using all data and the decoder generates data from  $\mathcal{N}(\hat{\mu}, \hat{\Sigma})$ . In the multivariate case, however, we include a whitening step in the code layer, which is necessary in order to use the proposed multivariate SW. In other words, let  $\mathbf{y}$  be the encoded data of  $k$  samples, and  $(\bar{\mathbf{y}}, \mathbf{S})$  be the respective sample mean and covariance, we whiten the encoded data as  $\mathbf{S}^{-\frac{1}{2}}(\mathbf{y} - \bar{\mathbf{y}})$ . Then, Prop.3.4 allows the decoder to work off samples coming from  $\mathcal{N}(\mathbf{0}, \mathbf{I})$ . In our empirical evaluation, we found that this whitening step also helped when other hypothesis tests were used in the proposed approach (see the supplement).



Now the presence of normality for  $q(\mathbf{z})$  can be directly tested, i.e., cannot be rejected if  $W$  passes a critical value  $W_\alpha$ . The overall optimization problem that our neural network solves is formulated as

$$\min \|\mathbf{X} - F_\psi(G_\theta(\mathbf{X}))\|_2^2 \text{ s.t. } W(G_\theta(\mathbf{X})) > W_\alpha, \quad (5)$$

where  $G_\theta(\cdot)$  is the encoder, and  $F_\psi(\cdot)$  is the decoder, respectively parameterized by  $\theta$  and  $\psi$ . Using the mathematically equivalent multi-objective loss, we can also find the solution  $G_\theta^*, F_\psi^* = \arg \min_{G_\theta, F_\psi} \|\mathbf{X} - F_\psi(G_\theta(\mathbf{X}))\|_2^2 - \lambda W(G_\theta(\mathbf{X}))$  for some proper value of  $\lambda > 0$  although we propose an algorithm that directly solves Eq.(5).

#### 4.1 OPTIMIZATION OF W

Eq.(5) is commonly optimized using a flavor of gradient descent with mini-batches, e.g., Adam (Kingma and Ba, 2014) which is used in Alg.1. The following proposition characterizes how to compute the gradient of  $W$ .

**Proposition 4.1.** *Let  $k$  be the size of the mini-batch. For any layer  $\ell$ , denote  $\mathbf{Y}^\ell = \Theta^\ell \mathbf{Y}^{(\ell-1)}$ , where  $\mathbf{Y}^{(\ell-1)}$  is an  $(n \times k)$  matrix of arbitrary activation from layer  $(\ell - 1)$ ,  $\mathbf{Y}^\ell$  is an  $(m \times k)$  matrix of linear activation for layer  $\ell$ , and  $\Theta^\ell$  is the  $(m \times n)$  matrix of parameters connecting the layers. Let  $\mathbf{y}$  be the  $(mk \times 1)$  ascendingly sorted vectorization of  $\mathbf{Y}^\ell$ . Then,  $\mathbf{y}$  can be computed by  $\mathbf{A}\theta$  where  $\mathbf{A}$  and  $\theta$  are the re-organized  $\mathbf{Y}^{\ell-1}$  and the vectorization of  $\Theta^\ell$ . Specifically,  $\mathbf{A}$  is an  $(mk \times mn)$  matrix with each row containing the relevant  $\mathbf{Y}^{(\ell-1)}$  data for a particular node’s activation. The gradient of  $W$  can be computed by*

$$\nabla_\theta W = \frac{2\mathbf{a}^T \mathbf{A} \theta}{\theta^T \mathbf{Z} \theta} \mathbf{a}^T \mathbf{A} \left[ \mathbf{I} - \frac{\theta \theta^T \mathbf{Z}}{\theta^T \mathbf{Z} \theta} \right], \quad (6)$$

where  $\mathbf{Z} = \mathbf{A}(\mathbf{I} - \frac{\mathbf{J}}{mk})\mathbf{A}^T$ ,  $\mathbf{I}$  is  $mk$ -dimensional identity matrix, and  $\mathbf{J}$  is a  $(mk \times mk)$  matrix of ones.

Unlike the adversarial framework the hypothesis testing model is straightforward to train. As shown in the pseudocode for this method in Alg.1, only the encoder part of the GAE updates when  $H_0$  is rejected (by re-optimizing  $G_\theta$  to reach  $W > W_\alpha$  or a p-value if available). When whitening is used, and if  $W > W_\alpha$ , the decoder can generate new data by sampling with respect to  $\mathcal{N}(\mathbf{0}, \mathbf{I})$ . It must be reiterated that failure to reject does *not* imply normality, however in practice this procedure works well.

#### 4.2 INNER LOOP TERMINATION

Alg.1 follows a conditional alternating optimization procedure, or can also be referred to as a feasible direction method. The outer loop seeks to minimize the reconstruction loss, whereas the inner loop evaluates the hypothesis testing and identifies the updates of  $\theta$  that satisfy:

---

#### Algorithm 1 Hypothesis Testing Autoencoder

---

**Require:**  $\mathbf{X}$ =training data,  $N$ =number of iterations,  $W_\alpha$ =critical value,  $\lambda$ =regularization coefficient,  $m$ =size of mini-batch  $> d$  = dimension of latent code layer to be tested

- 1: **Initialize:**  $\theta, \psi$
  - 2: **for**  $i = 0$  to  $N$  **do**
  - 3:   Sample next mini-batch from training data  $\mathbf{X}_m$
  - 4:    $(\theta, \psi) \leftarrow \text{Adam}(\nabla_{(\theta, \psi)} \|\mathbf{X}_m - F_\psi(G_\theta(\mathbf{X}_m))\|_2^2)$
  - 5:   Compute  $W(G_\theta(\mathbf{X}_m))$
  - 6:   **while**  $W(G_\theta(\mathbf{X}_m)) \leq W_\alpha$  **do**
  - 7:     Sample next mini-batch from training data  $\mathbf{X}_m$
  - 8:      $\theta \leftarrow \text{Adam}(\nabla_{(\theta)} (-\lambda W(G_\theta(\mathbf{X}_m))))$
  - 9:     Compute  $W(G_\theta(\mathbf{X}_m))$
  - 10:   **end while**
  - 11: **end for**
  - 12: Estimate  $(\hat{\mu}_d, \hat{\Sigma}_d)$       (Not necessary if whitened)
- 

$W > W_\alpha$ , a constraint that implies failure to reject  $H_0$ . Note that the inner loop is activated only when the condition is *not* already met. A fundamental question is whether this “while” loop will terminate given we solve a highly non-convex optimization problem (ultimately, whether we can find a  $\theta$  such that the  $q(\mathbf{z})$  stays within the Gaussian class). In (Bottou, 1991a) results were given for a general non-convex setting and show that under specific conditions the computation will converge. The following theorem is cited from that paper for which the proof can be found in (Bottou, 1991b).

**Theorem 4.2.** *For any measure  $dP(\mathbf{z})$ , if the cost  $C(\theta) = \mathbb{E}(J(\mathbf{z}, \theta))$  is differentiable up to the third derivatives where  $J$  is an objective function to be optimized, with bounded second and third derivatives, and if the following assertions are true,*

- (i)  $\forall \theta, E(H(\theta)) = \nabla_\theta C(\theta)$
- (ii)  $\sum_{t=1}^\infty \epsilon_t = \infty, \quad \sum_{t=1}^\infty \epsilon_t^2 < \infty$
- (iii)  $\exists A, B, \quad \forall \theta, \quad \mathbb{E}(H(\theta)^2) < A + BC(\theta)$
- (iv)  $\exists C_{min}, \quad \forall \theta, \quad C_{min} < C(\theta)$

*then  $C(\theta_t)$  converges with probability 1 and  $\nabla_\theta C(\theta_t)$  converges to 0 with probability 1.*

In our case,  $W(\theta)$  is  $C$ , thus  $H(\theta) \equiv \nabla_\theta W(\theta)$ , and  $\epsilon_t$  is the learning rate. The inner loop terminates according to Thm.4.2 if its conditions are all satisfied (the detailed proof is given in a supplement). Using the same argument of (Bottou, 1991b) regarding the similarity of simulated annealing, denoting  $q_t(\theta)$  the density of probability that  $\theta_t$  follows, by Thm.4.2 the support of  $q_t(\theta)$  converges to the set of extrema of  $W(\theta)$ , i.e.,  $\theta_t \rightarrow \{\theta | W(\theta) = 1\}$

thus  $W(\theta_t) \rightarrow 1$ . In fact, it is not necessary to train until  $W(\theta) = 1$ , so the procedure exits once  $W > W_\alpha$ .

## 5 THEORETICAL EQUIVALENCY

There exists a link between a distance based method for comparing the goodness-of-fit of two distributions and hypothesis testing discovered in (del Barrio et al., 1999). The general class of Wasserstein distances is studied in (Villani, 2008). We recite the definition here.

**Definition 5.1.** (Wasserstein Distances) Let  $(\chi, d)$  be a Polish metric space, and let  $p \in [1, \infty)$ . For any two probability measures  $\mu, \nu$  on  $\chi$ , the Wasserstein distance of order  $p$  between  $\mu$  and  $\nu$  is defined by

$$W_p(\mu, \nu) = \left( \inf_{\pi \in \Pi(\mu, \nu)} \int_{\chi} d(x, y)^p d\pi(x, y) \right)^{\frac{1}{p}}$$

$$= \inf \left\{ \left[ \mathbb{E}d(X, Y)^p \right]^{\frac{1}{p}}, \text{ law}(X) = \mu, \text{ law}(Y) = \nu \right\},$$

where  $\Pi(\mu, \nu)$  is the set of all joint probability measures. When the Polish metric space under consideration is the one-dimensional Euclidean space,  $W(\mu, \nu) = W_2(\mu, \nu)$ .

Of primary interest is the  $L_2$ -Wasserstein distance. It is possible to consider the distance between distributions  $P_1$  and  $P_2$ , defined by  $\mathcal{W}(P_1, P_2) = \left[ \int_0^1 (F_1^{-1}(t) - F_2^{-1}(t))^2 dt \right]^{\frac{1}{2}}$ , where  $F_1^{-1}$  and  $F_2^{-1}$  are the quantile functions of  $P_1$  and  $P_2$  defined to be  $F_i^{-1}(t) = \inf\{s : F_i(s) \geq t\}$  for  $i = 1, 2$ . The distance between a distribution with cumulative distribution function (CDF)  $F$ , mean  $\mu_0$  and standard deviation  $\sigma_0$ , and the class of all normal distributions can be written as  $\mathcal{W}^2(F, \mathcal{G}') = \inf\{\mathcal{W}^2(F, \pi), \pi \in \mathcal{G}'\}$ , where  $\mathcal{G}' = \{\pi : \pi = \Phi\left(\frac{x-\mu}{\sigma}\right), -\infty < \mu < \infty, \sigma > 0\}$ .

By expressing a normal random variable with mean  $\mu$  and variance  $\sigma^2$  as  $F^{-1}(p) = \mu + \sigma\Phi^{-1}(p)$ , it can be shown that  $\frac{\mathcal{W}^2(F, \mathcal{G}')}{\sigma_0^2} = 1 - \frac{\left( \int_0^1 (F^{-1}(t))\Phi^{-1}(t) dt \right)^2}{\sigma_0^2}$ . With a random sample of data,  $x_1, x_2, \dots, x_n$ , with underlying CDF  $F$ , define  $\mathcal{R}_n = \frac{\mathcal{W}^2(F_n, \mathcal{G}')}{S_n^2} = 1 - \frac{\hat{\sigma}_n^2}{S_n^2}$ , where  $\hat{\sigma}_n = \int_0^1 F_n^{-1}(t)\Phi^{-1}(t) dt$  and  $S_n^2$  is the sample variance.  $\mathcal{R}_n$  can be utilized as a test statistic for testing the composite null hypothesis that the data are normally distributed, and it belongs to the class of minimum distance tests.

### 5.1 $L_2$ -WASSERSTEIN ASYMPTOTICS

To study the null asymptotics of  $\mathcal{R}_n$ , assuming normality, (del Barrio et al., 1999) used approximations of quantile processes by Brownian bridges,  $B(t)$ . Under normal-

ity del Barrio et al. show that there exist constants  $a_n$  such that  $n\mathcal{R}_n - a_n \xrightarrow{\mathcal{D}} \int_0^1 \hat{B}^2(t) - E\hat{B}^2(t) dt$  where  $\hat{B} = \frac{(B - \langle B, 1 \rangle)1 - \langle B, \Phi^{-1} \rangle \Phi^{-1}}{\phi(\Phi^{-1})}$ . By applying principle component decomposition, (del Barrio et al., 1999) obtains the final result and is repeated here for clarity.

**Theorem 5.1.** Let  $\{X_n\}_n$  be a sequence of i.i.d normal random variables. Then

$$\mathcal{R}_n - a_n \xrightarrow{\mathcal{D}} -\frac{3}{2} + \sum_{j=3}^{\infty} \frac{Y_j^2 - 1}{j},$$

where  $\{Y_n\}_n$  is a sequence of i.i.d  $\mathcal{N}(0, 1)$  random variables with  $a_n = \int \frac{\frac{t}{(n+1)}}{\left(\phi(\Phi^{-1}(t))\right)^2} dt$

Cross referencing the asymptotics in Section 3.3, we find the  $L_2$ -Wasserstein normality test to be equivalent to the Shapiro-Wilk test, thus attaining similar power properties.

## 6 EMPIRICAL EVALUATION

We evaluated our method, the hypothesis testing based autoencoder using univariate Shapiro-Wilk (HTAE-SW), and its multivariate generalization (HTAE-MSW) on the standard MNIST digits dataset (LeCun et al., 1998), comparing it against the models believed to be the most closely related: the adversarial autoencoder (Makhzani et al., 2015) (AAE), the adversarial autoencoder with Wasserstein discriminator loss (WAAE), and autoencoder with maximum mean discrepancy loss for the critic (MMDAE as the model is not adversarial). We note that MMD constitutes a true hypothesis test where an  $\alpha$ -level test may be performed by way of permutation or approximation tests. Using it as such would lead to the HTAE-MMD model, a variant of our model. However, we used it as others did by simply optimizing it for comparison against the HTAE. To further evaluate the proposed approach, four additional goodness-of-fit tests were used to replace the (M)SW test. A supplementary material provided more results and discussion on Royston's H (Royston, 1983) (HTAE-R), Mardia's Skewness (Mardia, 1970) (HTAE-M), Malkovich-Afifi (Malkovich and Afifi, 1973) (HTAE-MA), and Henze-Zirkler (Henze and Zirkler, 1990) (HTAE-HZ). All models used  $\|x - \hat{x}\|_2^2$  for the reconstruction loss.

The network architecture was a fully connected class conditional autoencoder with conditioning done at the code layer. Two hidden layers are used between the input and code layer, each consisting of 784 nodes. The decoder contained the same structure. For AAE and WAAE, discriminators contained 2 layers each of 784 nodes with their respective losses. Models requiring sampling from  $p(\mathbf{z})$  used Gaussian priors of appropriate dimension  $p(\mathbf{z}) = \mathcal{N}_d(0, I)$ . Dropout (Srivastava

et al., 2014) was used with a “keep” probability equal to 0.9. The Adam optimizer was used with  $\beta_1 = 0.9$ ,  $\beta_2 = 0.999$ ,  $\epsilon = 10^{-8}$ , and  $\eta = 0.001$ . The hypothesis test was computed at the mini-batch level consisting of 100 samples. The hypothesis tests require an  $\alpha$  to be set. While technically a hyperparameter, it is the level of the test with a clear meaning that training proceeds with respect to this level of confidence. A more stringent  $\alpha$ -level tends to increase inner loop iterations. We found the commonly used  $\alpha = 0.05$  to work well. Two experiments were conducted to gauge the efficacy of the hypothesis testing method: a univariate and multivariate test.

Several criteria were used to assess each model: reconstruction loss, generative quality, normality constraints, prior matching (where applicable), and run-time. We generated images of hand-written digits, and monitored the reconstruction loss during training. By considering the hypothesis test statistic as an objective measure for rejecting normality, the test statistic was monitored for each iteration, and its corresponding p-value was plotted, during training for all models. In particular, the null hypothesis was rejected when the p-value was less than 0.05; larger p-values were preferred. Q-Q plots were also provided. By plotting the theoretical quantiles of the normal distribution against the empirical quantiles of the data in a Q-Q plot, any departure from the straight line provides evidence against normality. This can be used as a diagnostic measure after training has completed. The run times are included in Table 1.

### 6.1 UNIVARIATE: SHAPIRO-WILK

There was only a single node in the latent code layer in the univariate case. Results from the univariate case can be seen in Fig.1. Along with the plots mentioned above, the univariate case presented an opportunity to monitor the trajectory of  $(\hat{\mu}, \hat{\sigma})$ . Initial  $(\hat{\mu}, \hat{\sigma})$  were calculated using the initialized network weights. By monitoring the p-values, it appeared that the  $q(\mathbf{z})$  distributions from many other methods were not in fact normal. Of the Q-Q plots, only HTAE-SW maintained close enough proximity to the straight line. Based on the final batch, WAAE and MMDAE were able to match the prior distribution  $(\mu, \sigma)$  parameters fairly closely, however neither maintained normality. AAE could neither match parameters nor maintain normality. As HTAE-SW was not restricted to a specific normal, it had the opportunity to explore the normal class for an optimal distribution for the given data.

### 6.2 MULTIVARIATE GENERALIZATION OF SHAPIRO-WILK

For the multivariate methods a latent code dimension of 8 was used. By employing the new multivariate gener-

alization of the SW test (MSW) it was possible to use Q-Q plots to lend visual support to the p-value outcome, however trajectory plots were no longer an option. Each model was run for 100,000 iterations with plots shown in Fig.2, and run time shown in Table 1 when the code layer had 8 nodes. The generated images seemed to get visually better the higher the simple moving average of the p-values was. Q-Q plots for the last mini-batch show improper tail behaviors, for normality, in all models but HTAE-MSW. As before, the p-value should be greater than  $\alpha = 0.05$  to fail to reject the null; again the higher the p-value the better. For models that failed to reject  $H_0$  but had poor generative quality, this suggested several possibilities: training time needed to be increased, more focus should be given to reconstruction, or the network size should be increased. As can be seen in Table 1 HTAE methods were substantially faster in all cases.

Table 1: Run time in seconds for  $10^5$  iterations in the 1-D and 8-D cases using an NVIDIA GTX 1080Ti GPU.

Method	1-D	8-D
AAE	765.39	982.61
WAAE	904.95	1092.19
MMDAE	597.67	756.69
HTAE-(M)SW	346.16	548.08

## 7 DISCUSSION

Our empirical results suggest that substituting a hypothesis test, notably  $W$  and its the new multivariate generalization, which do not require pre-specifying a mean and covariance, may be a competitive alternative to other members of the GAE class. Allowing  $q(\mathbf{z})$  to deviate in the feasible space,  $\mathcal{G}$ , during training means sampling is done with respect to  $q(\mathbf{z}) = \hat{\pi} = \mathcal{N}(\hat{\mu}, \hat{\Sigma}) \in \mathcal{G}$  where  $(\hat{\mu}, \hat{\Sigma})$  are estimated (or when whitening is used in the multivariate case, with respect to  $\mathcal{N}(\mathbf{0}, \mathbf{I})$ ). Consequently, there is less need to worry about discrepancy between the distribution we *want* to sample from, and the distribution we *are* sampling from, as we tacitly interpret failure to reject as within the class of normals. This need *not* be the case in the other models. However, as HTAE-MSW does make use of whitening, ensuring enough samples to adequately estimate  $\hat{\Sigma}^{-\frac{1}{2}}$  and  $\hat{\mu}$  is a necessity.

The AAE and WAAE both require training of a discriminator network. This network, with size on the order of the encoder or decoder, increases training time. Moreover, training of the discriminator needs to be scheduled in balance with that of the generator, and how exactly this should be done is still an open problem. On the other hand, using the hypothesis test abolishes this problem

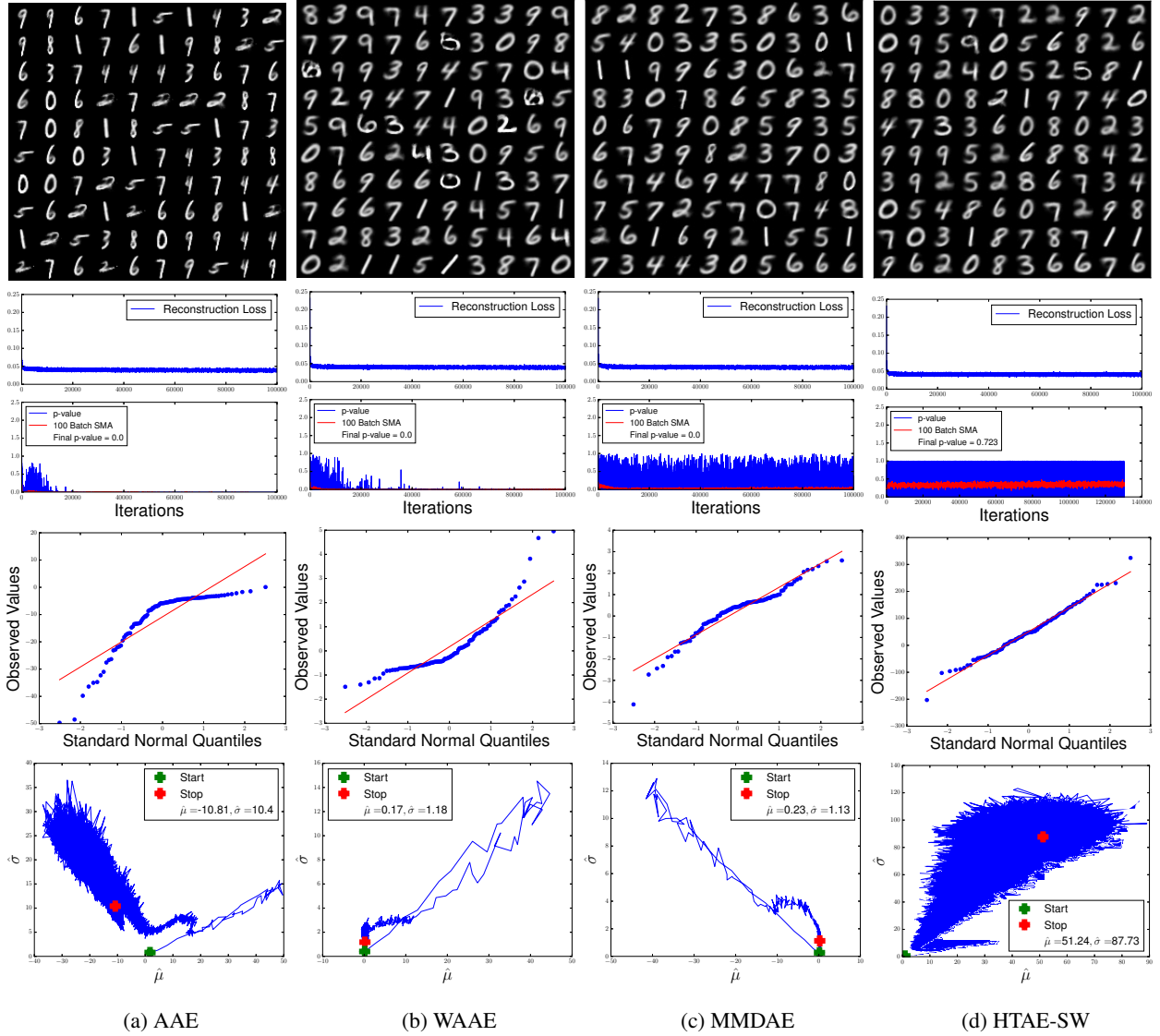


Figure 1: Row one illustrates sample digits generated by each model. Row two shows the reconstruction loss, and the simple moving average (SMA) of the p-value for a batch size of 100, along with the final mini-batch terminating p-value. Row three contains the Q-Q plots, and row four plots the trajectory of  $(\hat{\mu}, \hat{\sigma})$  over the course of training.

completely. By utilizing the critical values (or p-values) for the test statistic it is now possible to know *precisely* when to alternate between enforcing prior constraints, and minimizing the reconstruction loss. In our empirical evaluation, we also observed that using a parametric hypothesis test could improve gradient updates by utilizing information about the null distribution. A full exploration into this mechanism is left for future investigation.

A concern may be raised that Thm.4.2 merely guarantees the “while” loop will terminate, yet provides no indication of when it terminates. In all experiments, the speed never proved to be an issue. The inner loop was able to ensure  $q(\mathbf{z}) \in \mathcal{G}$  in a very small number of iterations as can be

seen in Fig.3 in supplementary materials. We attempt to understand why this is the case in the near future.

We experimented with four other goodness-of-fit tests (see the supplement please), but none provided the benefits that the Shapiro-Wilk and its generalization did. Issues included longer run times, and weaker power. The search for greater power motivates the following conjecture.

*Conjecture 1.* The more powerful the hypothesis test, the more precise the null distribution information contained within the test statistic that can be transmitted to the encoder to update  $\theta$  via the gradient.

A trade-off between the power of the test, time complex-

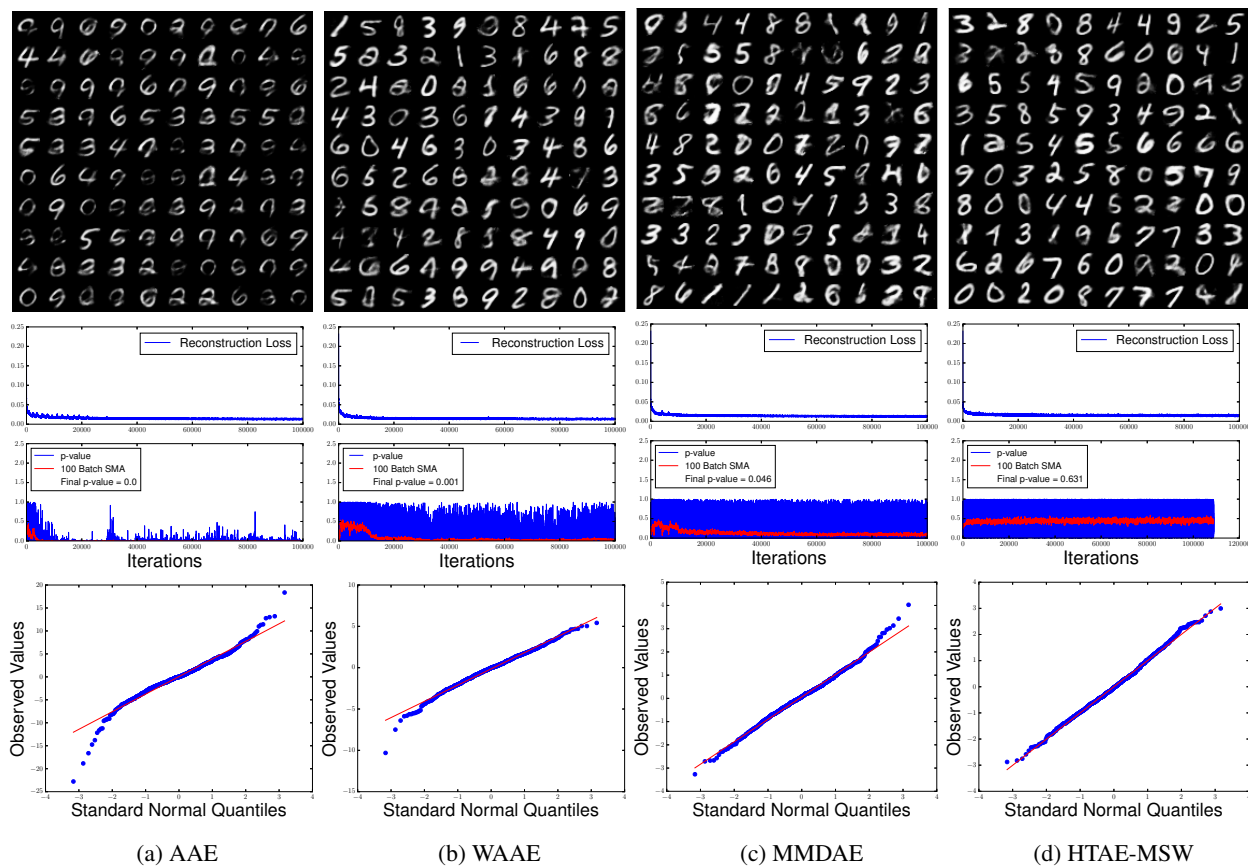


Figure 2: Similar to Fig.1, the top row plots random samples generated from each model. Row two contains the p-values with a 100 batch SMA. Row three are Q-Q plots associated with MSW.

ity and reconstruction quality likely exists. What is the cheapest computational complexity of a hypothesis test available for a given power? We expect that future research will reveal more on these questions.

## 8 CONCLUSION AND FUTURE WORK

In this paper we have proposed a new method for training generative autoencoders by explicitly testing the distribution of the code layer output via univariate and multivariate parametric hypothesis tests. We have shown a number of benefits to using such an approach including: objectively verifying if training has indeed pushed  $q(\mathbf{z}) \in \mathcal{G}$ , the ability to utilize the critical value of a hypothesis test as a threshold for determining when to switch between reconstruction and encoding iterations. Our method produces competitive results while showing computational efficiency. Moreover, we explored the link between the Shapiro-Wilk hypothesis test and the  $L_2$ -Wasserstein distance between two distributions.

Given the large numbers of univariate and multivariate

parametric hypothesis tests available, it remains to be seen how others compare when used in a generative autoencoder. In fact, any distribution for which a hypothesis test can be derived can be used for training a latent code layer. Furthermore, the proposed method of training can be applied to any of the models that takes a generative autoencoder style network. This initial work brings up additional interesting problems, so our future work will dive into the questions raised. It is also worth asking how other hypothesis testing methods can be included into neural network training.

## Acknowledgments

We thank Vladamir Pozdnyakov for insightful discussions, especially regarding the multivariate test. This work was supported by National Science Foundation (NSF) grants: IIS-1320586, CCF-1514357, and IIS-1718738. J. Bi was also supported by National Institutes of Health (NIH) grants: R01DA037349 and K02DA043063 as well as NSF grants: DBI-1356655 and IIS-1447711.

## References

- M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.
- Y. Bengio, G. Mesnil, Y. Dauphin, and S. Rifai. Better mixing via deep representations. In *International Conference on Machine Learning*, pages 552–560, 2013a.
- Y. Bengio, L. Yao, G. Alain, and P. Vincent. Generalized denoising auto-encoders as generative models. In *Advances in Neural Information Processing Systems*, pages 899–907, 2013b.
- Y. Bengio, E. Laufer, G. Alain, and J. Yosinski. Deep generative stochastic networks trainable by backprop. In *International Conference on Machine Learning*, pages 226–234, 2014.
- L. Bottou. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nimes*, 91(8):0, 1991a.
- L. Bottou. *Une Approche Theorique de L'Apprentissage Connexionniste et Applications A La Reconnaissance de la Parole*. PhD thesis, 1991b.
- G. Casella and R. L. Berger. *Statistical inference*, volume 2. Duxbury Pacific Grove, CA, 2002.
- J. De Wet et al. Asymptotic distributions of certain test criteria of normality. *South African Statistical Journal*, 6(2):135–149, 1972.
- E. del Barrio, J. A. Cuesta-Albertos, C. Matrán, and J. M. Rodríguez-Rodríguez. Tests of goodness of fit based on the l<sub>2</sub>-wasserstein distance. *Annals of Statistics*, pages 1230–1239, 1999.
- A. Dosovitskiy and T. Brox. Generating images with perceptual similarity metrics based on deep networks. In *Advances in Neural Information Processing Systems*, pages 658–666, 2016.
- G. K. Dziugaite, D. M. Roy, and Z. Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. *arXiv preprint arXiv:1505.03906*, 2015.
- S. Fotopolous, J. Leslie, and M. Stephens. Errors in approximations for expected normal order statistics with an application to goodness-of-fit. Technical report, Technical Report, 1984.
- I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.
- A. Gretton, K. M. Borgwardt, M. Rasch, B. Schölkopf, and A. J. Smola. A kernel method for the two-sample problem. In *Advances in neural information processing systems*, pages 513–520, 2007.
- A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.
- I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5769–5779, 2017.
- N. Henze and B. Zirkler. A class of invariant consistent tests for multivariate normality. *Communications in Statistics-Theory and Methods*, 19(10):3595–3617, 1990.
- D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- E. L. Lehmann and J. P. Romano. *Testing statistical hypotheses*. Springer Science & Business Media, 2006.
- J. Leslie, M. A. Stephens, and S. Fotopoulous. Asymptotic distribution of the shapiro-wilk w for testing for normality. *The Annals of Statistics*, pages 1497–1506, 1986.
- C.-L. Li, W.-C. Chang, Y. Cheng, Y. Yang, and B. Póczos. Mmd gan: Towards deeper understanding of moment matching network. In *Advances in Neural Information Processing Systems*, pages 2200–2210, 2017.
- Y. Li, K. Swersky, and R. Zemel. Generative moment matching networks. In *International Conference on Machine Learning*, pages 1718–1727, 2015.
- A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.
- J. F. Malkovich and A. Afifi. On tests for multivariate normality. *Journal of the american statistical association*, 68(341):176–179, 1973.
- K. V. Mardia. Measures of multivariate skewness and kurtosis with applications. *Biometrika*, 57(3):519–530, 1970.
- C. J. Mecklin and D. J. Mundfrom. A monte carlo comparison of the type i and type ii error rates of tests of multivariate normality. *Journal of Statistical Computation and Simulation*, 75(2):93–107, 2005.
- A. Mnih and K. Gregor. Neural variational inference and learning in belief networks. *arXiv preprint arXiv:1402.0030*, 2014.
- C. R. Rao, C. R. Rao, M. Statistiker, C. R. Rao, and C. R. Rao. *Linear statistical inference and its applications*, volume 2. Wiley New York, 1973.
- N. M. Razali, Y. B. Wah, et al. Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests. *Journal of statistical modeling and analytics*, 2(1):21–33, 2011.

- D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.
- S. Rifai, Y. Bengio, Y. Dauphin, and P. Vincent. A generative process for sampling contractive auto-encoders. *arXiv preprint arXiv:1206.6434*, 2012.
- J. Royston. An extension of shapiro and wilk's w test for normality to large samples. *Applied statistics*, pages 115–124, 1982.
- J. Royston. Some techniques for assessing multivariate normality based on the shapiro-wilk w. *Applied Statistics*, pages 121–133, 1983.
- P. Royston. Approximating the shapiro-wilk w-test for non-normality. *Statistics and Computing*, 2(3):117–119, 1992.
- E. Seier. Comparison of tests for univariate normality. *InterStat Statistical Journal*, 1:1–17, 2002.
- S. S. Shapiro and R. Francia. An approximate analysis of variance test for normality. *Journal of the American Statistical Association*, 67(337):215–216, 1972.
- S. S. Shapiro and M. B. Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4): 591–611, 1965.
- N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.
- D. J. Sutherland, H.-Y. Tung, H. Strathmann, S. De, A. Ramdas, A. Smola, and A. Gretton. Generative models and model criticism via optimized maximum mean discrepancy. *arXiv preprint arXiv:1611.04488*, 2016.
- I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schoelkopf. Wasserstein auto-encoders. *arXiv preprint arXiv:1711.01558*, 2017.
- S. P. Verrill and R. A. Johnson. *The asymptotic distributions of censored data versions of the Shapiro-Wilk test of normality statistic*. University of Wisconsin, Department of Statistics, 1983.
- C. Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

---

# Towards Flatter Loss Surface via Nonmonotonic Learning Rate Scheduling

---

Sihyeon Seong<sup>1</sup>, Yekang Lee<sup>1</sup>, Youngwook Kee<sup>1</sup>, Dongyoon Han<sup>1,2</sup>, Junmo Kim<sup>1</sup>

<sup>1</sup>School of Electrical Engineering, KAIST

<sup>2</sup>Clova AI Research, NAVER Corp.

{sihyun0826, askhow, youngwook.kee, dyhan, junmo.kim}@kaist.ac.kr

## Abstract

Whereas optimizing deep neural networks using stochastic gradient descent has shown great performances in practice, the rule for setting step size (i.e. learning rate) of gradient descent is not well studied. Although it appears that some intriguing learning rate rules such as ADAM (Kingma and Ba, 2014) have since been developed, they concentrated on improving convergence, not on improving generalization capabilities. Recently, the improved generalization property of the flat minima was revisited, and this research guides us towards promising solutions to many current optimization problems. In this paper, we analyze the flatness of loss surfaces through the lens of robustness to input perturbations and advocate that gradient descent should be guided to reach flatter region of loss surfaces to achieve generalization. Finally, we suggest a learning rate rule for escaping sharp regions of loss surfaces, and we demonstrate the capacity of our approach by performing numerous experiments.

## 1 INTRODUCTION

Overfitting is a core issue in the domain of machine learning. When it comes to deep learning, it becomes even more important, because of its high dimensionality. Owing to the huge number of parameters, deep learning models show some strange behaviors. For example, deep models easily fit random labeling of training data and furthermore training data replaced with random noise input (Zhang et al., 2016). A peculiar phenomenon called “fooling deep neural networks (DNNs)” was also reported in (Nguyen et al., 2015; Szegedy et al., 2013). These kinds of unexpected behavior might be a potential

risk when adopting DNNs for applications which require high precision. Classic DNN regularization methods include placing a Gaussian or Laplacian prior on parameters, called weight decays (Figueiredo, 2003; Bishop, 2006). A weight decay assumes that the desirable solutions of the parameters are placed near zero, and therefore does not consider solutions which may lie a bit far from zero but possibly show better test performance. Numerous interesting works are still being proposed, including Stochastic Gradient Langevin Dynamics (SGLD) (Raginsky et al., 2017), parametrization method for reducing overfitting for genomics (Romero et al., 2016), employing stochastic effects; randomly dropping features (Hinton et al., 2012) or using stochastic depth (Huang et al., 2016). Some interesting works analyzed these stochastic effects in the view of a  $L_2$ -regularization (Wager et al., 2013) or an ensemble method (Singh et al., 2016).

The concept of generalization via achieving flat minima was first proposed in (Hochreiter et al., 1995), and its importance has recently been revisited in the domain of deep learning optimization (Chaudhari et al., 2016), (Keskar et al., 2016). This another viewpoint of thinking generalization may become a promising direction for investigating the weight space property of DNNs; moreover, DNN loss surface property analysis has become a popular issue (Sagun et al., 2016; Swirszcz et al., 2016; Littwin and Wolf, 2016; Im et al., 2016). Recently, achieving generalization via flat minima is investigated in terms of optimizing PAC-Bayes bound (Dziugaite and Roy, 2017a,b; Neyshabur et al., 2017).

A stochastic gradient descent (SGD) walks around loss surfaces in DNNs, and its behavior can be controlled by learning rates. It leads us to an interesting question: “Is it possible to seek flatter valleys of loss surfaces by controlling learning rate schedules?”. We show the relation of learning rates and flatness of the loss surface, then show that a nonmonotonic scheduling of learning rates with an intermediate large learning rate stages are



beneficial to discover flat minima and therefore leads to improved generalization. On the scheduling of learning rates, recent studies achieve great convergence rates on training data, but they are prone to overfit, showing some degradation of test performance. Therefore, even state-of-the-art DNN models (Simonyan and Zisserman, 2014; Szegedy et al., 2014; He et al., 2016) have continued to use simple step or exponentially-decaying learning step sizes. Thus, our work has great implications on improving the theory of learning rates. To the best of our knowledge, this is the first work that pays substantial attention to learning rates with regard to generalization and overfitting of deep neural networks. We cite selected noteworthy work on learning rates; however, again note that none of the work concerns what we consider throughout the paper. Regarding fast convergence of given training data: RMSprop (Tieleman and Hinton, 2012), Adagrad (Duchi et al., 2011), Adadelata (Zeiler, 2012), Adam (Kingma and Ba, 2014), and an interesting work that searches for optimal adaptive learning rates without manual tuning (Schaul et al., 2012).

In this paper, we begin with robustness in the input space, which is a traditional way of explaining generalization. Then, we find the relation of robustness in the input perturbation and that in weight perturbation to show why flat minima work better (Section 2). Next, we explain how large learning rates can guide gradient descent to flatter losses (Section 3). Then, a simple nonmonotonic learning rate scheduling technique is introduced for adopting larger learning rates. Finally, our claims are demonstrated by performing numerous experiments.

## 2 THE RELATION OF ROBUSTNESS IN INPUT SPACE AND WEIGHT SPACE

In this section, we show that generalization can be achieved through robustness with respect to input perturbations, input perturbations can be equivalently transferred to weight perturbations, and therefore generalization can be achieved through robustness with respect to weight perturbations.

Generalization can be stated as the uniformity of the loss function with respect to input change (See Supplementary Material A). Denote  $\mathbf{x} \in \mathbb{R}^{n \times 1}$  as input,  $\mathbf{w} \in \mathbb{R}^d$  as the vectorized weight of the model and  $\mathcal{L}(\mathbf{w}; \mathbf{x})$  as the loss function of the neural network. Then, input perturbations  $\delta^{\mathbf{x}} \in \mathbb{R}^{n \times 1}$  should result in a small change in the loss function:

$$|\mathcal{L}(\mathbf{w}; \mathbf{x} + \delta^{\mathbf{x}}) - \mathcal{L}(\mathbf{w}; \mathbf{x})| < \epsilon,$$

A similar interpretation of generalization in DNNs can be found in (Rifai et al., 2011), where the authors attempted to reduce  $|f(\mathbf{w}; \mathbf{x} + \delta^{\mathbf{x}}) - f(\mathbf{w}; \mathbf{x})|$  for  $f$ , which is an auto-encoder.

Now, the generalization capability of the model is evaluated for the change of the loss function with respect to perturbations on the weight vector  $\delta^{\mathbf{w}}$ .

**Lemma 1.** *Let  $\delta^{\mathbf{x}}$  and  $\delta^{\mathbf{w}}$  be input and weight perturbations, respectively. For a single-layer neural network, the input perturbations  $\delta^{\mathbf{x}}$  can be transferred to the weight perturbations  $\delta^{\mathbf{w}}$ . More formally, suppose we have weight matrix  $\mathbf{W} \in \mathbb{R}^{m \times n}$  and weight perturbations  $\delta^{\mathbf{W}} \in \mathbb{R}^{m \times n}$ . Then for any  $\mathbf{W}$  and  $\mathbf{x}$  that satisfy  $\mathbf{x} \neq \mathbf{0}$  and  $\mathbf{W} \neq \mathbf{0}$ , there exists  $\delta^{\mathbf{W}}$  such that  $(\mathbf{W} + \delta^{\mathbf{W}})\mathbf{x} = \mathbf{W}(\mathbf{x} + \delta^{\mathbf{x}})$ . Consequently,  $|\mathcal{L}(\mathbf{W} + \delta^{\mathbf{W}}; \mathbf{x}) - \mathcal{L}(\mathbf{W}; \mathbf{x})| = |\mathcal{L}(\mathbf{W}; \mathbf{x} + \delta^{\mathbf{x}}) - \mathcal{L}(\mathbf{W}; \mathbf{x})|$ .*

*Proof.* The proof can be accomplished by finding  $\delta^{\mathbf{W}}$  which satisfies

$$\delta^{\mathbf{W}}\mathbf{x} = \mathbf{W}\delta^{\mathbf{x}} \quad (1)$$

The following choice of  $\delta^{\mathbf{W}}$  satisfies (1):

$$\delta^{\mathbf{W}} = \frac{\mathbf{W}\delta^{\mathbf{x}}}{\mathbf{x}^{\top}\mathbf{x}}\mathbf{x}^{\top} = \frac{\mathbf{W}\delta^{\mathbf{x}}}{\|\mathbf{x}\|^2}\mathbf{x}^{\top} \quad (2)$$

□

**Proposition 1.** *Suppose  $|\mathcal{L}(\mathbf{W} + \delta^{\mathbf{W}}; \mathbf{x}) - \mathcal{L}(\mathbf{W}; \mathbf{x})| < \epsilon$  holds for any  $\delta^{\mathbf{W}}$  such that  $\|\delta^{\mathbf{W}}\|_F < \delta$ ,  $\delta > 0$  and  $\epsilon > 0$ . Then  $|\mathcal{L}(\mathbf{W}; \mathbf{x} + \delta^{\mathbf{x}}) - \mathcal{L}(\mathbf{W}; \mathbf{x})| < \epsilon$  holds for any  $\delta^{\mathbf{x}}$  such that  $\frac{\|\delta^{\mathbf{x}}\|}{\|\mathbf{x}\|} < \frac{\delta}{\sigma_{max}(\mathbf{W})}$ , where  $\sigma_{max}(\mathbf{W})$  is the maximum singular value of  $\mathbf{W}$ .*

*Proof.* For any  $\delta^{\mathbf{x}}$  such that  $\frac{\|\delta^{\mathbf{x}}\|}{\|\mathbf{x}\|} < \frac{\delta}{\sigma_{max}(\mathbf{W})}$ , if we choose  $\delta^{\mathbf{W}}$  as in (2), then from the result of Lemma 1, we have

$$\begin{aligned} & |\mathcal{L}(\mathbf{W}; \mathbf{x} + \delta^{\mathbf{x}}) - \mathcal{L}(\mathbf{W}; \mathbf{x})| \\ &= |\mathcal{L}(\mathbf{W} + \delta^{\mathbf{W}}; \mathbf{x}) - \mathcal{L}(\mathbf{W}; \mathbf{x})| \quad (3) \end{aligned}$$

Then, corresponding bound on  $\|\delta^{\mathbf{W}}\|_F$  can be derived by

$$\|\delta^{\mathbf{W}}\|_F^2 = \frac{\|\mathbf{W}\delta^{\mathbf{x}}\|^2 \|\mathbf{x}^{\top}\|^2}{(\mathbf{x}^{\top}\mathbf{x})^2} \leq \frac{\sigma_{max}^2(\mathbf{W}) \|\delta^{\mathbf{x}}\|^2}{\|\mathbf{x}\|^2} < \delta^2, \quad (4)$$

where we used  $\|\mathbf{a}\mathbf{b}^{\top}\|_F^2 = \|\mathbf{a}\|^2 \|\mathbf{b}\|^2$  and  $\|\mathbf{W}\delta^{\mathbf{x}}\| \leq \sigma_{max}(\mathbf{W}) \|\delta^{\mathbf{x}}\|$ .

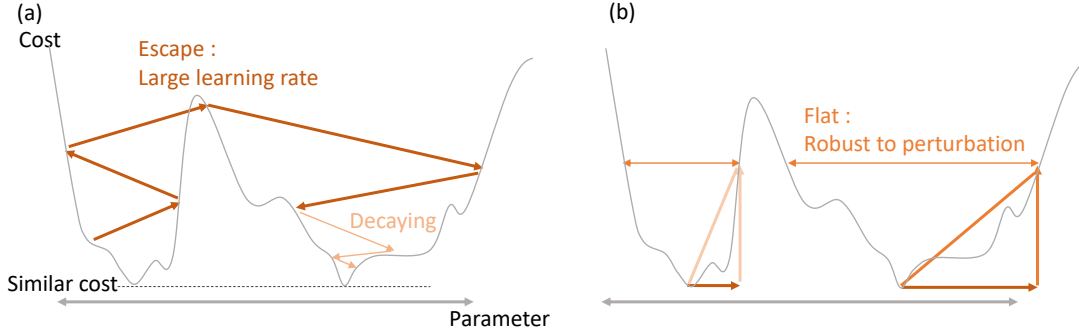


Figure 1: Conjectured illustration showing the relation of learning rate to loss surface. (a) A nonmonotonic scheduling of learning rates with an intermediate large learning rate stages lets the model escape from the steep valleys with high curvature. (b) Concept of what we call ‘wide valleys’. The scale of perturbation should be large enough so that the loss surface can effectively cope with proper perturbations.

Therefore, we have

$$\begin{aligned}
 & |\mathcal{L}(\mathbf{W}; \mathbf{x} + \delta^{\mathbf{x}}) - \mathcal{L}(\mathbf{W}; \mathbf{x})| \\
 &= |\mathcal{L}(\mathbf{W} + \delta^{\mathbf{W}}; \mathbf{x}) - \mathcal{L}(\mathbf{W}; \mathbf{x})| < \epsilon \quad (5)
 \end{aligned}$$

□

According to Lemma 1, for an arbitrary input perturbation, the first layer’s weight matrix can be perturbed so that  $\mathbf{W}(\mathbf{x} + \delta^{\mathbf{x}}) = (\mathbf{W} + \delta^{\mathbf{W}})\mathbf{x}$ , i.e. the first layer’s responses are the same for the two cases. Thus, the remaining layer’s responses are also the same throughout, and the results in Lemma 1 and Proposition 1 are applicable to general DNNs. In Supplementary Material B, we provide additional analysis that shows how to distribute weight perturbations to all DNN layers to match the input perturbations.

Now that we can transfer the perturbations of input to those of weight, generalization can be interpreted on the loss surface. Reduction of the effects on the loss function with respect to perturbations on the input, requires us to reduce the effects with respect to perturbations on the weight vector. Finally, we propose the relation of the loss surface and a measure for generalization as follows:

**Conjecture 1.** *Given the same cost value, if the loss surface is flatter, then it is more likely for neural networks to be more generalized.*

### 3 THE RELATION OF LEARNING RATES AND GENERALIZATION

High-dimensional loss surfaces are regarded as non-convex and extremely difficult to visualize. Let us consider the SGD algorithm, which corresponds well to current large-scale problems. Not only are there abundant pathways that the SGD can follow, but the pathways are

also highly dependent on learning rates. We can expect the loss surface of the model to have many locally convex areas and the learning rates to largely affect the outcome achieved by the algorithm. We justify this situation by evaluating the stationary characteristic of the stochastic gradients as the mean of the given values as follows:

$$\bar{\mathcal{L}}(\mathbf{w}; \mathbf{x}) := \mathbb{E}_{\mathbf{x}_i \sim \mathcal{D}}[\mathcal{L}(\mathbf{w}; \mathbf{x}_i)], \quad (6)$$

where the training data  $\mathbf{x}_i \in \mathbb{R}^{n \times 1}$  are drawn from a distribution  $\mathcal{D}$ . We can fit a quadratic approximation of the entire loss surface surrounding the local optimum  $\mathbf{w}^*$  using the positive definite Hessian matrix

$$\bar{\mathcal{L}}(\mathbf{w}; \mathbf{x}) \approx \bar{\mathcal{L}}(\mathbf{w}^*; \mathbf{x}) + \frac{1}{2}(\mathbf{w} - \mathbf{w}^*)^\top \mathbf{H}^*(\mathbf{w} - \mathbf{w}^*), \quad (7)$$

where  $\mathbf{H}^*$  is the mean (over  $\mathcal{D}$ ) of the Hessian of the loss function at  $\mathbf{w}^*$ . Let us denote  $\gamma_t$  as the learning rate at the iteration  $t$ . Then, the gradient descent can be calculated as follows:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \gamma_t \nabla \bar{\mathcal{L}}(\mathbf{w}; \mathbf{x}) \quad (8)$$

$$\approx \mathbf{w}_t - \gamma_t \mathbf{H}^*(\mathbf{w}_t - \mathbf{w}^*). \quad (9)$$

$$\mathbf{w}_{t+1} - \mathbf{w}^* \approx (I - \gamma_t \mathbf{H}^*)(\mathbf{w}_t - \mathbf{w}^*). \quad (10)$$

If we assume small and smooth changes of the learning rate during some epochs (equivalently,  $t_s \leq t \leq t_f$  where  $|\gamma_t - \gamma_{t_s}| < \epsilon_\gamma$ ), providing constant  $\gamma_{t_s}$ , the weight vector can be measured by both the optimal and the initial points:

$$\mathbf{w}_t \approx \mathbf{w}^* + (I - \gamma_{t_s} \mathbf{H}^*)^{t-t_s} (\mathbf{w}_{t_s} - \mathbf{w}^*) \quad (11)$$

Then, we can strictly obtain the range of the learning rate capable of enforcing the convergence. If  $\mathbf{H}^*$  is diagonal as (Schaul et al., 2012), and  $h_{max}$  is the maximum value

amongst the second-order gradients, the convergence criteria are expressed by the condition<sup>1</sup>

$$|1 - \gamma_{t_s} h_{max}| < 1, \quad (12)$$

which is equivalent to

$$0 < h_{max} < \frac{2}{\gamma_{t_s}}, \quad (13)$$

which provides the relationship of the learning rate and the curvature of the surface necessary to converge. A large learning rate has the critical requirement that the curvature of the surface should be sufficiently low to avoid diverging. Therefore, we anticipate that the SGD having a large learning rate allows for locating a smooth area.

Our remarks on the relationship between learning rates and generalization address “which local minimum should be chosen” as conceptually illustrated in Figure 1. Recent studies on loss surfaces, such as (Dauphin et al., 2014) and (Choromanska et al., 2015), both theoretically and empirically discovered that local minima are more likely to be located only where train losses are very close to those of the global minimum. Based on these results, we consider that the loss surface has basins of local minima that occur only at the bottom of the loss surface, i.e.,  $|\mathcal{L}(\mathbf{w}^*; \mathbf{x}) - \mathcal{L}_{gmin}| \approx \epsilon$  where  $\mathcal{L}_{gmin}$  is the global minimum. When such a basin has a high curvature  $h_{max}$  violating (13),  $w$  keeps drifting from  $w^*$  because of (11) until it reaches another basin with a smaller curvature, a flatter region satisfying (13).

Additionally, once it reaches the entrance of a basin of smaller curvature, it is more likely to converge to a flatter local minimum. This is discussed in Section 6.3 implying that the average slope of the loss surface depends on the width of the basin’s entrance. Finally, we propose the following chain of effects showing how generalization can be achieved via nonmonotonic learning rates scheduling:

**High learning rate  $\Rightarrow$  escape from high curvature valleys in weight space  $\Rightarrow$  smooth region in weight space  $\Rightarrow$  smooth region in input space  $\Rightarrow$  improved generalization.**

## 4 LEARNING RATES AND STOCHASTIC VARIANCE

The major claim of our work is that learning rates should be set large to escape sharp loss surface valleys. For setting large learning rates, curvature of loss surfaces is the

<sup>1</sup>(12) is still a valid condition for a non-diagonal  $\mathbf{H}^*$  with the maximum eigenvalue  $h_{max}$

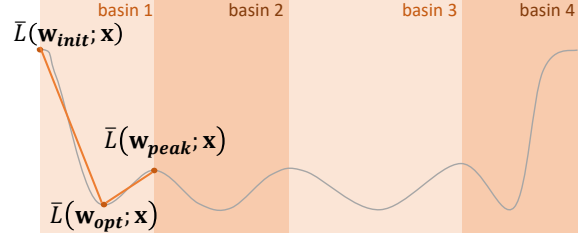


Figure 2: An illustration of the occurrence of maximum gradients. As the loss at random initial weights is significantly large, maximum gradients are most likely to occur near initial weights.

only constraint in plain gradient descent cases. However, stochastic variance (Reddi et al., 2016) further interferes with convergence in SGD cases. Stochastic variance is an inherent variance of gradients caused by minibatch selection in SGD. When stochastic variance is large, learning rates should be set smaller to prevent divergence. Therefore, the maximum learning rate can be achieved when the stochastic variance is small.

Stochastic variance is relatively large when the gradients are large (Johnson and Zhang, 2013). Therefore, learning rates should be set large after the occurrence of the maximum gradients. As in Figure 2, suppose the randomly initialized weights are  $\mathbf{w}_{init}$  for which  $\bar{\mathcal{L}}(\mathbf{w}_{init}; \mathbf{x})$  will be large (e.g. around  $\log(n_{class})$  in the case of cross-entropy loss where  $n_{class}$  is the number of classes).  $\mathbf{w}_{opt}$  is a local minimum that is closest to  $\mathbf{w}_{init}$  and  $\mathbf{w}_{peak}$  is a weights of local maximum which is the closest to  $\mathbf{w}_{opt}$  and under the condition  $\bar{\mathcal{L}}(\mathbf{w}_{init}; \mathbf{x}) \gg \bar{\mathcal{L}}(\mathbf{w}_{peak}; \mathbf{x})$ . We constrain our claims under

$$\frac{|\bar{\mathcal{L}}(\mathbf{w}_{init}; \mathbf{x}) - \bar{\mathcal{L}}(\mathbf{w}_{opt}; \mathbf{x})|}{|\bar{\mathcal{L}}(\mathbf{w}_{peak}; \mathbf{x}) - \bar{\mathcal{L}}(\mathbf{w}_{opt}; \mathbf{x})|} \gg \frac{\|\mathbf{w}_{init} - \mathbf{w}_{opt}\|}{\|\mathbf{w}_{peak} - \mathbf{w}_{opt}\|} \quad (14)$$

and consider other cases to be beyond the scope of this paper. In practice,  $\bar{\mathcal{L}}(\mathbf{w}_{init}; \mathbf{x}) \gg \bar{\mathcal{L}}(\mathbf{w}_{peak}; \mathbf{x})$ . Therefore (14) is a probable condition. If we divide the loss surfaces into subspaces of basins and assume Lipschitz continuity, then we get the lower bound of maximum gradients during the optimization  $g_{LB}$  as follows.

$$g_{LB} = \frac{|\bar{\mathcal{L}}(\mathbf{w}_{init}; \mathbf{x}) - \bar{\mathcal{L}}(\mathbf{w}_{opt}; \mathbf{x})|}{\|\mathbf{w}_{init} - \mathbf{w}_{opt}\|} \quad (15)$$

which makes maximum gradients mostly occur near the initial weights.

## 5 PEAK LEARNING STAGE : NONMONOTONIC LEARNING RATES SCHEDULING

To experimentally show that high learning rates are beneficial to discovering flatter loss surfaces, we propose non-monotonic learning rate schedules to utilize larger learning rates. Adopting larger learning rates presents the risk of learning to diverge because of (13) and Section 4. Rather than placing the maximum learning rate at the start of the learning, we place maximum learning rate in the middle, so the initial stage can stabilize learning (i.e. reduce stochastic variance) and prepare for the consecutive large learning rates. This learning rate schedule is motivated by the neurological phenomenon called *critical period* or *sensitive period*—a period in which learning plasticity reaches its peak (Wiesel et al., 1963; Ge et al., 2007). The plasticity gradually increases if biological systems are prepared to learn, after which plasticity is reduced over time. Considering the stabilization of learning, it is not surprising that plasticity *gradually* and *slowly* grows to a certain degree once it is switched on. We tried two nonmonotonic learning rate schedules,  $\gamma_t$  at normalized iteration  $t$ , as follows<sup>2</sup>:

- Gaussian Shape:  $\gamma_t = \gamma_{max} * \exp\left(\frac{-(t-0.5)^2}{\sigma^2}\right)$ .
- Laplacian Shape:  $\gamma_t = \gamma_{max} * \exp\left(-\frac{|t-0.5|}{\lambda}\right)$ .

Here,  $\gamma_{max}$  is the peak or maximum learning rate;  $\gamma_{min}$  is the final or minimum learning rate; and  $\gamma_{start}$  is the starting learning rate. Also,  $\sigma^2$  for the Gaussian-shaped schedule and  $\lambda$  for the Laplacian-shaped schedule is defined as  $-0.25 * \frac{1}{\ln(\gamma_{min}/\gamma_{max})}$  and  $-0.5 * \frac{1}{\ln(\gamma_{min}/\gamma_{max})}$ , respectively. Because the function starts with a value that is too small, we use a truncated function that is cut at the appropriate offset,  $t_{offset}$ , for faster convergence. The offset  $t_{offset}$  is  $0.5 - \sqrt{-\sigma^2 \ln(\gamma_{start}/\gamma_{max})}$  in the Gaussian schedules and  $0.5 + \lambda \ln(\gamma_{start}/\gamma_{max})$  in the Laplacian schedules.

The total number of iterations determines not only the total number of weight updates but also the sampling frequency from the continuous peak-shaped function. The sampling frequency controls the smoothness of the change in learning rate. By setting  $\gamma_{max}$  larger than the maximum learning rate of the classical learning rate schedule within convergence, the local optimum can be found in the flatter basin. We found both two peak shaped learning rates worked well, but the Gaussian-shaped learning rates worked slightly better than the

<sup>2</sup>For notation simplicity, the iteration  $t$  is normalized between 0 and 1.

Laplacian-shaped ones as shown in Figure 6. Therefore, we used Gaussian-shaped scheduling for most of the experiments. Note that a Gaussian-shaped curve is not the only way but just one way to implement the proposed peak learning stages.

**SLOW START** Because of (13) and stochastic variance of the SGD, learning rates that are too large cause the learning to diverge. If we start with a too large learning rate, the learning either diverges or finds a poor critical point. Therefore, considering the stability of optimization, learning should commence with a small learning rate. However, we verified experimentally that starting with a learning rate that is too small does not lead to success. Thus, adopting  $t_{offset}$  in the learning rate rules is required to eliminate the redundant initial phase.

**DECAYING LEARNING RATES** The peak learning stage induces some divergence to escape from the sharp minima. Thus, the learning rate must be decayed for the final convergence. Considering conventional methods, this stage is intuitively acceptable. It is noteworthy that most DNNs are optimized using SGD, which implies loss surfaces are substantially fluctuating for each minibatch. Therefore decaying stages are necessary for achieving stability of optimization results.

## 6 EVALUATING THE FLATNESS OF LOSS SURFACES

Evaluation of the effect of the learning rate inevitably requires its behavior on the loss surface to be analyzed. However, in neural networks, the loss surface is high-dimensional so that optimization trajectory on the loss function becomes difficult to visualize. First, we tracked the local property of high-dimensional loss surfaces by measuring the magnitude of the gradient. Throughout this paper, the magnitude of the gradient refers to the 2-norm of the gradient vector which represents the local steepness of loss surfaces.

However, because the gradient (or Hessian) of the loss is computed at a specified location in the weight space, its application is limited to a local area of the loss surface. Therefore, we visualize large-scale properties of loss surfaces by adopting the linear path experiments introduced by (Goodfellow and Vinyals, 2014). The linear path experiments measure the loss surface by sweeping the trajectory between two points in high-dimensional spaces. By visualizing a single cross section of the loss surface, we indirectly measure its large-scale flatness.

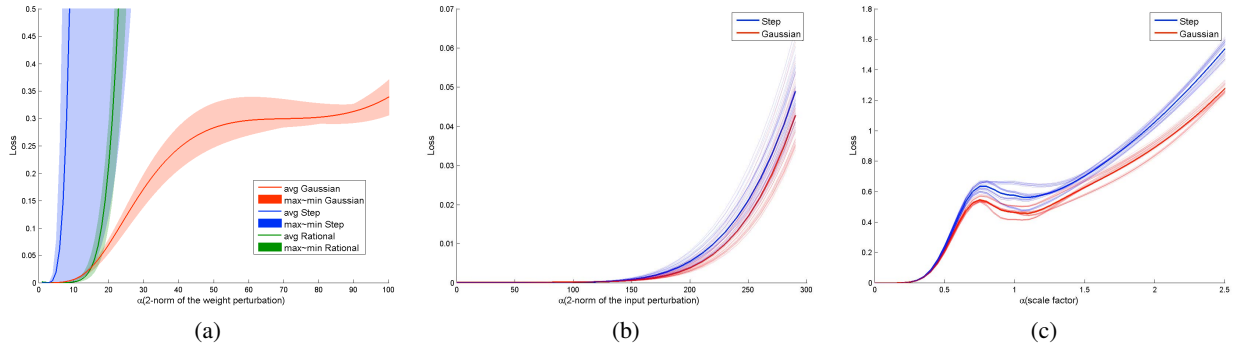


Figure 3: Linear path experiments in Section 6. (a) Loss versus weight perturbation along the line path, which follows the trajectory between the final state weight and peak stage weight. The model trained using the proposed learning rate clearly shows flatter loss surface. Each color represents different learning rates. Maximum and minimum losses are also presented from different models trained by random initialization. (b) Loss versus randomly generated input perturbation. Each transparent line represents different random noise directions, and the bold line indicates the ensemble average of these transparent lines. The model trained using the proposed learning rate shows smaller loss for the same perturbation, indicating that our method is more robust to input perturbations. (c) Loss versus input perturbations in the directions of training to validation data. Each transparent line represents different perturbation decided by the selection of nearby validation data. The bold line indicates the ensemble average of the transparent lines. This also shows that the loss surface determined by our method is flatter with respect to input perturbations.

## 6.1 LOSS WITH RESPECT TO WEIGHT PERTURBATIONS

Because our learning rate rule has a peak stage that is not considered in existing methods, we analyzed its effects by performing linear path experiments which follow the trajectory between the final state weight,  $\mathbf{w}_{\text{final}}$ , and peak stage weight,  $\mathbf{w}_{\text{peak}}$ . That is, we calculated  $\mathcal{L}(\mathbf{w}; \mathbf{x})$ , where  $\mathbf{w} = \mathbf{w}_{\text{final}} + \alpha \frac{\mathbf{w}_{\text{peak}} - \mathbf{w}_{\text{final}}}{\|\mathbf{w}_{\text{peak}} - \mathbf{w}_{\text{final}}\|}$  and  $\alpha > 0$ . We reported the flatness over a training set  $S$ . Thus, the average of losses,  $\frac{1}{|S|} \sum_{\mathbf{x} \in S} \mathcal{L}(\mathbf{w}; \mathbf{x})$ , was calculated. We also tested weight perturbations in random directions and fooling directions (Szegedy et al., 2013).

## 6.2 LOSS WITH RESPECT TO INPUT PERTURBATIONS

We also applied the linear path experiments to input spaces. Flatter loss surfaces can be easily expected to be robust, regarding random perturbations. Thus, we calculated  $\mathcal{L}(\mathbf{w}; \mathbf{x})$ , where  $\mathbf{x} = \mathbf{x}_{\text{train}} + \alpha \mathbf{x}_{\text{noise}}$ , given that  $\mathbf{x}_{\text{noise}}$  is randomly generated from the Normal distribution and then normalized to  $\|\mathbf{x}_{\text{noise}}\| = 1$ . Moreover, what the generalization tries to achieve is higher accuracy on the validation data. Therefore, we performed further experiments generating perturbations in the direction of validation data from training data. Thus, we calculated  $\mathcal{L}(\mathbf{w}; \mathbf{x})$ , where  $\mathbf{x} = \mathbf{x}_{\text{train}} + \alpha(\mathbf{x}_{\text{val}} - \mathbf{x}_{\text{train}})$ . In this case,  $\mathbf{x}_{\text{val}}$  was randomly selected from the ten closest validation candidates placed near  $\mathbf{x}_{\text{train}}$  for each trial. Finally, the average of losses over training set is

calculated.

## 6.3 THE FLATNESS OF LOSS SURFACES

The notion of flatness of a loss surface can be defined as follows. Let us consider a basin of a loss surface with local minimum  $\mathbf{w}_i^*$ , and define level set  $\mathbf{S}_{\mathcal{L}_i} = \{\mathbf{w} | \mathcal{L}(\mathbf{w}; \mathbf{x}) = c\}$  where  $c$  is the loss at the entrance of the basin. Choose any  $\mathbf{w}_0 \in \mathbf{S}_{\mathcal{L}_i}$ , and then the average slope of the surface along the line from  $\mathbf{w}_i^*$  to  $\mathbf{w}_0$  can be determined as  $\frac{c - \epsilon}{\|\mathbf{w}_i^* - \mathbf{w}_0\|}$ , where  $\epsilon$  is the loss at  $\mathbf{w}_i^*$ . If this slope is small, then we call the loss surface is flat.

## 7 EXPERIMENTS

Here, we verify the roles of these peak-shaped learning rates on baseline convolutional neural networks (Krizhevsky et al., 2012), which we call “small model”, and (Springenberg et al., 2014), which we call “large model” with the CIFAR-10 dataset. Hereinafter, all the experimental results are based on these settings, except for those in Section 7.2. We compared step decay, rational decay ( $\gamma_t = \gamma_0(1 + \lambda t)^{-1}$ ), and RMSprop (Tieleman and Hinton, 2012) as adaptive step size schedules and Gaussian-shaped schedules. For the “small model”, the best validation error was 18.79 % with the baseline step decaying learning rates. This result was reduced to 16.48% after adding our learning rate method. For the “large model”, the validation error, 9.53%, of the baseline method was reduced to 8.68%. We tried numerous random settings of hyperparameters (e.g., total number

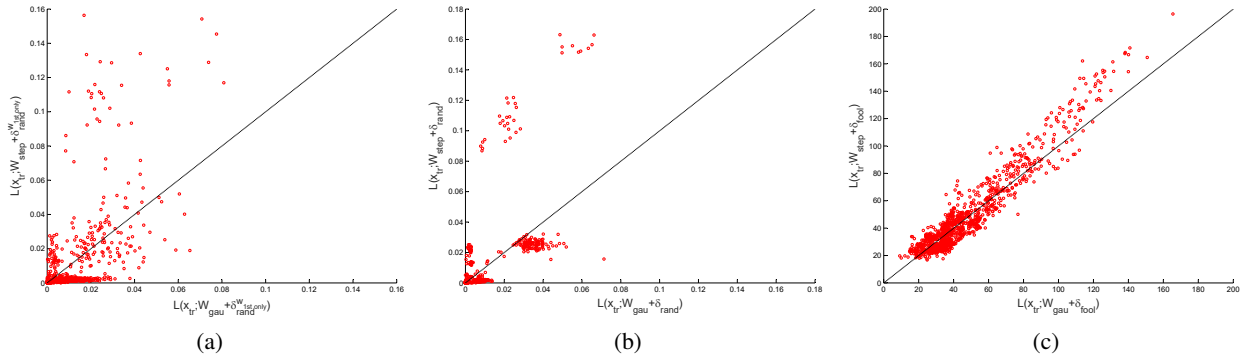


Figure 4: Measuring effects of various weight perturbations (at  $t = 1$ ) was introduced in Section 6. The x and y axes indicate the loss of the model trained by Gaussian and stepped learning rates, respectively. Because all graphs are tilted toward the y-axis, we can see that the nonmonotonic learning rate technique leads to flatter region of loss surfaces with less loss for weight perturbation. (a) Random weight perturbations are added to the first layer. (b) Random weight perturbations are added to the entire layers. (c) Weight perturbations along the fooling direction are added to the entire layers.

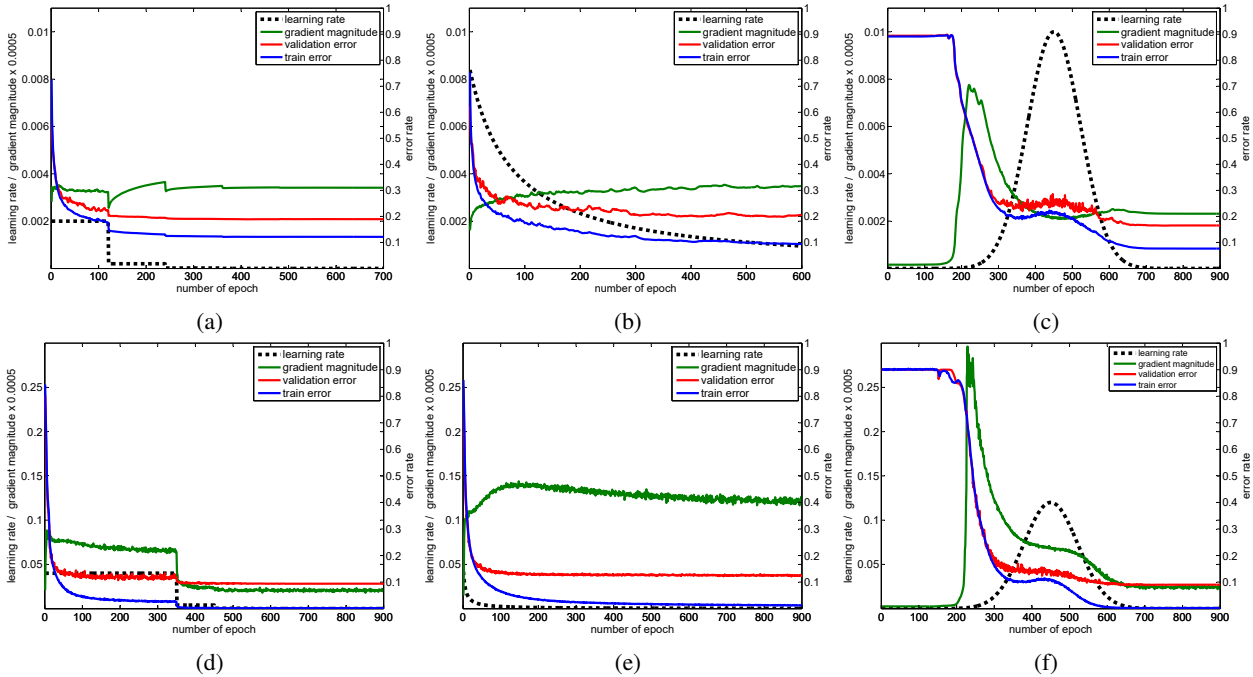


Figure 5: Examples of learning process of each learning rate schedule. (a), (b) and (c) :“small model” results, (d), (e) and (f) :“large model” results. (a), (b) and (c), or (d), (e) and (f) show learning rates, gradient magnitude, train and validation errors of each learning rate schedule (step, rational and Gaussian learning rates), respectively. All given values are calculated after each epoch. For the Gaussian learning case, the magnitude of the gradient peaks at the initial phase. Afterwards, it gradually decreases until the learning rate peaks. Other plots are provided for comparison.

of iterations, weight decay, momentum, and initial or peak learning rate) on the “small model” and confirmed that our methods are robust to selection of those settings (see Figure 7). However, we would like to clarify some experimental details on hyperparameters.

**STARTING OFFSET** This parameter can be ignored if the training time does not matter. It is safe to set  $t_{offset} = 0$ . However, going through all these extremely small learning rates is not necessary for achieving good performance.  $t_{offset}$  is controlled by  $\gamma_{start}$  and setting this value as maximum learning rate from the step decay-

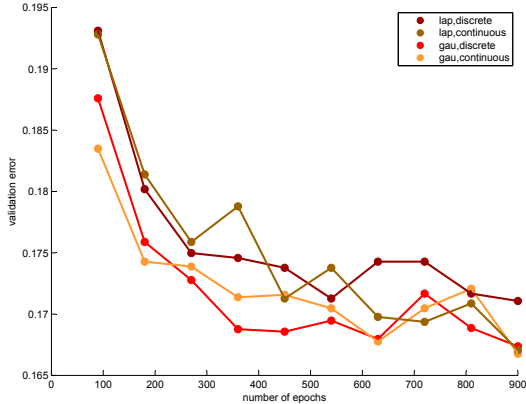


Figure 6: Comparisons of Gaussian-shaped and Laplacian-shaped learning rates, in terms of final validation error at  $s = 1$ . Horizontal axis is the total number of epochs. “Continuous” refers to learning rates that are updated at each iteration, whereas “Discrete” refers learning rates that are only updated at the beginning of each epoch.

ing scheduling is predominantly a good choice.

**MAXIMUM LEARNING RATE** Because we are claiming that a large learning rate increases the generalization capability of the model, it is necessary to increase the peak learning rate more than three times of the original maximum learning rate reported in the baseline model. Three times the conventional maximum learning rate is most often safe. However, one can further increase  $\gamma_{max}$  if enough epochs are taken.

**MINIMUM LEARNING RATE** Generally, using learning rates that are too small presents the risk of overfitting. However, because we regularize for the smooth loss surface in the peak learning stages, setting  $\gamma_{min}$  smaller than the conventional value does not harm the performance. It sometimes results in further performance improvement during trials. We experimentally found that setting  $\gamma_{min} = \gamma_{max}/10000$  works well.

**THE NUMBER OF EPOCHS** Our method spends times on peak learning rate, which does not reduce training loss. Therefore, we were required to take more epochs than the conventional schedule to get the best performance. However, even with the same number of epochs, our method can surpass other learning rate schedules, as reported in Section 7.2. Furthermore, we compared the performance of our method to the step-decaying learning rate using the model ensemble in Table 1. To reproduce the same performance as our method, the step decaying learning rate rule needs five model ensembles. Therefore, whereas Gaussian learning rate requires more epochs, our method is still practically efficient.

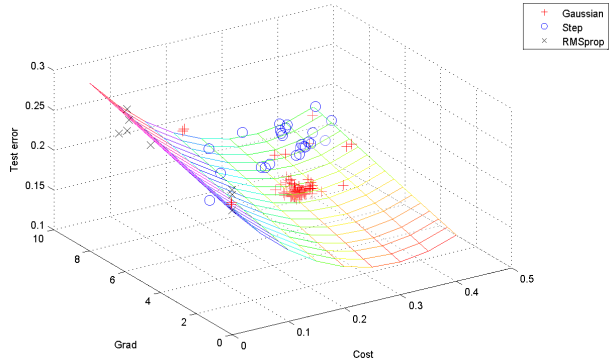


Figure 7: Prediction on test error with cost and  $\ell_2$ -norm of gradients.

## ROBUSTNESS TO HYPERPARAMETER TUNING

We reported the experimental results with several values for the peak learning rate,  $\gamma_{max}$ , and the total number of iterations in Figure 7. This clearly shows our method is robust to variations in the hyperparameter. For a wide range of  $\gamma_{max}$ , from 0.002 to 0.015, the worst validation error was 18.15%, which is still better than the baseline error (18.79%).

## 7.1 EVALUATING FLATNESS OF THE LOSS GUIDED BY PEAK LEARNING STAGES

Using the techniques presented in Section 6, we evaluated the flatness of the loss trained by peak learning stages. Peak learning stage showed robustness with respect to random input perturbations (Figure 3(b)) and flatness in the direction of train to validation data (Figure 3(c)). In terms of weight perturbations, our method showed flatness in the direction of  $\mathbf{w}^{final}$  to  $\mathbf{w}^{peak}$  (Figure 3(a)). It also showed flatness along random directions (Figure 4(a), (b)) and even with fooling directions (Figure 4(c)). Thus, our theory of flattening loss surface using peak learning stages is well supported by numerous experimental results.

Method	Error (%)	
	Step learning rate	Gaussian learning rate
1 CNN	18.79(360 epochs)	16.91(360 epochs)
2 CNNs	17.49	15.98
3 CNNs	17.13	15.37
4 CNNs	17.09	14.88
5 CNNs	16.75	14.89

Table 1: Model Ensemble.

## 7.2 PERFORMANCE EVALUATION

In what follows, all test error rates are evaluated at the last epoch, not at the best validation epoch. The per-

formance of our learning rate schedule(Gaussian-shaped learning rates) is compared to the state-of-the-art models (Zeiler and Fergus, 2013; Lin et al., 2013; Goodfellow et al., 2013; Lee et al., 2014; He et al., 2016).

**MNIST** For MNIST, we set the  $\gamma_{max} = 0.3$ ,  $\sigma^2 = 2*0.08^2$  (this value makes Gaussian-shape visually looks good) of the Gaussian shaped-learning schedule with total 167 epochs. We considered that the training time is not the issue in case of MNIST because training time is significantly short. Throughout our proposed learning rate policy, without any modification of loss functions and architectures, the accuracy already surpasses the previous works. Table 2 summarizes the result.

METHOD	ERROR (%)
CNN	0.53
STOCHASTIC POOLING	0.47
NIN	0.47
MAXOUT NETWORKS	0.45
DEEPLY SUPERVISED NET	0.39
<b>NIN + ours</b>	<b>0.34</b>

Table 2: Test error rates for the MNIST dataset (without data augmentation).

**CIFAR-10 and CIFAR-100** On the hyperparameter tuning, we followed same settings as in MNIST, except 720 total epochs for NIN (Lin et al., 2013), 270 epochs for ResNet (He et al., 2016). We preprocess the images similar to (Lin et al., 2013; Zeiler and Fergus, 2013; Goodfellow et al., 2013). Color jittering is also added in the ResNet experiment. For CIFAR-100, experiments on DenseNet (Huang et al., 2017) and Wide ResNet (Zagoruyko and Komodakis, 2016) are also performed. In case of Wide ResNet, we exceptionally set the peak learning rate twice as large as the start learning rate (this model adopts large dropout rate 0.3 and exceptionally small number of epochs, which interfere with convergence). Table 3 summarizes the result.

**ImageNet** Because ImageNet classification requires huge computational cost, we reduced shape of the Gaussian-shaped learning rate scheduling by adopting  $t_{offset}$ . We tested with the model of (Krizhevsky et al., 2012) and (Szegedy et al., 2014), which is well-reported and widely used as baseline model. First, we tried learning rate schedules with  $t_{offset} = 0$ ,  $\sigma^2 = 0.0128$  and total 270 epochs (Naive version). Then, to make training efficient, we adopt  $\gamma_{start} = 0.01$ ,  $\gamma_{max} = 0.03$  and  $\gamma_{min} = 0.000003$ , as suggested in the section 7. Finally, we report validation errors in Table 4.

METHOD	ERROR (%)	
	CIFAR-10	CIFAR-100
MAXOUT NETWORKS	9.38	-
DROPCONNECT	9.32	-
NIN	8.81	-
DEEPLY SUPERVISED NET	8.22	-
NIN + APL UNITS	7.51	-
RESNET(110-DEPTH)	6.41±0.21	27.815±0.15
DENSENET-BC (L=100, K=12)	-	22.47
DENSENET-BC (L=190, K=40)	-	17.18
WIDE RESNET (WRN-28-10-DROPOUT)	-	18.44
<b>NIN + ours</b>	<b>7.22</b>	-
<b>ResNet(110-depth) + ours</b>	<b>5.34±0.11</b>	<b>25.71±0.07</b>
<b>DenseNet-BC (L=100, k=12) + ours</b>	-	<b>22.17</b>
<b>DenseNet-BC (L=190, k=40) + ours</b>	-	<b>16.86</b>
<b>Wide ResNet (WRN-28-10-dropout) + ours</b>	-	<b>18.03</b>

Table 3: Test error rates for CIFAR-10 and CIFAR-100.

METHOD	ERROR (%)
ALEXNET(90 EPOCHS)	19.81
GOOGLENET(80 EPOCHS)	10.82
BATCHNORMALIZATION (80 EPOCHS)	9.05
<b>AlexNet + ours(90 epochs)</b>	<b>19.67</b>
<b>AlexNet + ours(180 epochs)</b>	<b>19.06</b>
<b>AlexNet + ours(Naive version, 270 epochs)</b>	<b>18.89</b>
<b>GoogLeNet + ours(80 epochs)</b>	<b>10.39</b>
<b>BatchNormalization + ours(80 epochs)</b>	<b>8.68</b>

Table 4: Validation error rates for the ImageNet dataset.

## 8 CONCLUSION

We showed why a flatter loss generalizes better in the view of robustness. Then we presented the relationship of flat losses and learning rates. Inspired by neuroscience, we further proposed peak learning stages for improving high-dimensional DNNs. We thoroughly analyzed how such learning rates affect conventional deep networks. To the best of our knowledge, the work presented in this paper is the first work in this line of research bridging the gap between the learning rate scheduling and the regularization theory of deep learning.

## ACKNOWLEDGEMENT

This work was supported by the National Research Council of Science & Technology (NST) grant by the Korea government (MSIP) (No. CRC-15-05-ETRI).



## References

- Bishop, C. M. (2006). Pattern recognition. *Machine Learning*.
- Chaudhari, P., Choromanska, A., Soatto, S., and LeCun, Y. (2016). Entropy-sgd: Biasing gradient descent into wide valleys. *arXiv preprint arXiv:1611.01838*.
- Choromanska, A., Henaff, M., Mathieu, M., Arous, G. B., and LeCun, Y. (2015). The loss surfaces of multilayer networks. In *AISTATS*.
- Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., and Bengio, Y. (2014). Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems*, pages 2933–2941.
- Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159.
- Dziugaite, G. K. and Roy, D. M. (2017a). Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*.
- Dziugaite, G. K. and Roy, D. M. (2017b). Entropy-sgd optimizes the prior of a pac-bayes bound: Data-dependent pac-bayes priors via differential privacy. *arXiv preprint arXiv:1712.09376*.
- Figueiredo, M. A. (2003). Adaptive sparseness for supervised learning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(9):1150–1159.
- Ge, S., Yang, C.-h., Hsu, K.-s., Ming, G.-l., and Song, H. (2007). A critical period for enhanced synaptic plasticity in newly generated neurons of the adult brain. *Neuron*, 54(4):559–566.
- Goodfellow, I. J. and Vinyals, O. (2014). Qualitatively characterizing neural network optimization problems. *arXiv preprint arXiv:1412.6544*.
- Goodfellow, I. J., Warde-Farley, D., Mirza, M., Courville, A., and Bengio, Y. (2013). Maxout networks. *arXiv preprint arXiv:1302.4389*.
- He, K., Zhang, X., Ren, S., and Sun, J. (2016). Identity mappings in deep residual networks. In *Computer Vision—ECCV 2016*.
- Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.
- Hochreiter, S., Schmidhuber, J., et al. (1995). Simplifying neural nets by discovering flat minima. *Advances in Neural Information Processing Systems*, pages 529–536.
- Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.
- Huang, G., Sun, Y., Liu, Z., Sedra, D., and Weinberger, K. (2016). Deep networks with stochastic depth. *arXiv preprint arXiv:1603.09382*.
- Im, D. J., Tao, M., and Branson, K. (2016). An empirical analysis of deep network loss surfaces. *arXiv preprint arXiv:1612.04010*.
- Johnson, R. and Zhang, T. (2013). Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pages 315–323.
- Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. (2016). On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*.
- Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.
- Lee, C.-Y., Xie, S., Gallagher, P., Zhang, Z., and Tu, Z. (2014). Deeply-supervised nets. *arXiv preprint arXiv:1409.5185*.
- Lin, M., Chen, Q., and Yan, S. (2013). Network in network. *arXiv preprint arXiv:1312.4400*.
- Littwin, E. and Wolf, L. (2016). The loss surface of residual networks: Ensembles and the role of batch normalization. *arXiv preprint arXiv:1611.02525*.
- Neyshabur, B., Bhojanapalli, S., McAllester, D., and Srebro, N. (2017). Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, pages 5949–5958.
- Nguyen, A., Yosinski, J., and Clune, J. (2015). Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 427–436.
- Raginsky, M., Rakhlin, A., and Telgarsky, M. (2017). Non-convex learning via stochastic gradient langevin dynamics: a nonasymptotic analysis. In *Conference on Learning Theory*, pages 1674–1703.
- Reddi, S. J., Hefny, A., Sra, S., Póczos, B., and Smola, A. (2016). Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pages 314–323.
- Rifai, S., Vincent, P., Muller, X., Glorot, X., and Bengio, Y. (2011). Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 833–840.
- Romero, A., Carrier, P. L., Erraqabi, A., Sylvain, T., Auvolat, A., Dejoie, E., Legault, M.-A., Dubé, M.-P., Hussin, J. G., and Bengio, Y. (2016). Diet networks: Thin parameters for fat genomic. *arXiv preprint arXiv:1611.09340*.
- Sagun, L., Bottou, L., and LeCun, Y. (2016). Singularity of the hessian in deep learning. *arXiv preprint arXiv:1611.07476*.
- Schaul, T., Zhang, S., and LeCun, Y. (2012). No more pesky learning rates. *arXiv preprint arXiv:1206.1106*.
- Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- Singh, S., Hoiem, D., and Forsyth, D. (2016). Swapout: Learning an ensemble of deep architectures. *arXiv preprint arXiv:1605.06465*.

- Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*.
- Swirszcz, G., Czarnecki, W. M., and Pascanu, R. (2016). Local minima in training of deep networks. *arXiv preprint arXiv:1611.06310*.
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2014). Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*.
- Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.
- Tieleman, T. and Hinton, G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 4.
- Wager, S., Wang, S., and Liang, P. S. (2013). Dropout training as adaptive regularization. In *Advances in neural information processing systems*, pages 351–359.
- Wiesel, T. N., Hubel, D. H., et al. (1963). Single-cell responses in striate cortex of kittens deprived of vision in one eye. *J Neurophysiol*, 26(6):1003–1017.
- Zagoruyko, S. and Komodakis, N. (2016). Wide residual networks. *arXiv preprint arXiv:1605.07146*.
- Zeiler, M. D. (2012). Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.
- Zeiler, M. D. and Fergus, R. (2013). Stochastic pooling for regularization of deep convolutional neural networks. *arXiv preprint arXiv:1301.3557*.
- Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2016). Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*.

---

# The Information Autoencoding Family: A Lagrangian Perspective on Latent Variable Generative Models

---

**Shengjia Zhao**

Computer Science Department  
Stanford University  
sjzhao@stanford.edu

**Jiaming Song**

Computer Science Department  
Stanford University  
tsong@stanford.edu

**Stefano Ermon**

Computer Science Department  
Stanford University  
ermon@stanford.edu

## Abstract

A large number of objectives have been proposed to train latent variable generative models. We show that many of them are Lagrangian dual functions of the same primal optimization problem. The primal problem optimizes the mutual information between latent and visible variables, subject to the constraints of accurately modeling the data distribution and performing correct amortized inference. By choosing to maximize or minimize mutual information, and choosing different Lagrange multipliers, we obtain different objectives including InfoGAN, ALI/BiGAN, ALICE, CycleGAN, beta-VAE, adversarial autoencoders, AVB, AS-VAE and InfoVAE. Based on this observation, we provide an exhaustive characterization of the statistical and computational trade-offs made by all the training objectives in this class of Lagrangian duals. Next, we propose a dual optimization method where we optimize model parameters as well as the Lagrange multipliers. This method achieves Pareto optimal solutions in terms of optimizing information and satisfying the constraints.

## 1 INTRODUCTION

Latent variable generative models are designed to accomplish a wide variety of tasks in computer vision (Radford et al., 2015; Kuleshov & Ermon, 2017), natural language processing (Yang et al., 2017), reinforcement learning (Li et al., 2017b), compressed sensing Dhar et al. (2018), etc. Prominent examples include Variational Autoencoders (VAE, Kingma & Welling (2013); Rezende et al. (2014)), with extensions such as  $\beta$ -VAE (Higgins et al., 2016), Adversarial Autoencoders (Makhzani et al., 2015), and InfoVAE (Zhao et al., 2017); Generative Adversarial Networks (Goodfellow et al., 2014), with extensions such as ALI/BiGAN (Dumoulin et al., 2016a; Don-

ahue et al., 2016), InfoGAN (Chen et al., 2016a) and ALICE (Li et al., 2017a); hybrid objectives such as CycleGAN (Zhu et al., 2017), DiscoGAN (Kim et al., 2017), AVB (Mescheder et al., 2017) and AS-VAE (Pu et al., 2017). All these models attempt to fit an empirical data distribution, but differ in multiple ways: how they measure the similarity between distributions; whether or not they allow for efficient (amortized) inference; whether the latent variables should retain or discard information about the data; and how the model is optimized, which can be likelihood-based or likelihood-free (Mohamed & Lakshminarayanan, 2016; Grover et al., 2018).

In this paper, we generalize existing training objectives for latent variable generative models. We show that all the above training objectives can be viewed as *Lagrangian dual functions* of a constrained optimization problem (primal problem). The primal problem optimizes over the parameters of a generative model and an (amortized) inference distribution. The optimization objective is to maximize or minimize mutual information between latent and observed variables; the constraints (which we term “*consistency constraints*”) are to accurately model the data distribution and to perform correct amortized inference. By considering the Lagrangian dual function and different settings of the Lagrange multipliers, we can obtain all the aforementioned generative modeling training objectives. Surprisingly, under mild assumptions, the aforementioned objectives can be linearly combined to produce every possible primal objective/multipliers in this model family.

In Lagrangian dual optimization, the dual function is maximized with respect to the Lagrange multipliers, and minimized with respect to the primal parameters. Under strong duality, the optimal parameters found by this procedure also solve the original primal problem. However, the aforementioned objectives use fixed (rather than maximized) multipliers. As a consequence, strong duality does not generally hold.

To overcome this problem, we propose a new learning approach where the Lagrange multipliers are also optimized. We show that strong duality holds in distribution space,

so this optimization procedure is guaranteed to optimize the primal objective while satisfying the consistency constraints. As an application of this approach, we propose *Lagrangian VAE*, a Lagrangian optimization algorithm for the InfoVAE (Zhao et al., 2017) objective. Lagrangian VAE can explicitly trade-off optimization of the primal objective and consistency constraint satisfaction. In addition, both theoretical properties (of Lagrangian optimization) and empirical experiments show that solutions obtained by Lagrangian VAE *Pareto dominate* solutions obtained with InfoVAE: Lagrangian VAE either obtains better mutual information or better constraint satisfaction, regardless of the hyper-parameters used by either method.

## 2 BACKGROUND

We consider two groups of variables: observed variables  $\mathbf{x} \in \mathcal{X}$  and latent variables  $\mathbf{z} \in \mathcal{Z}$ . Our algorithm receives input distributions  $q(\mathbf{x}), p(\mathbf{z})$  over  $\mathbf{x}$  and  $\mathbf{z}$  respectively. Each distribution is either specified *explicitly* through a tractable analytical expression such as  $\mathcal{N}(0, I)$ , or *implicitly* through a set of samples. For example, in latent variable generative modeling of images (Kingma & Welling, 2013; Goodfellow et al., 2014),  $\mathcal{X}$  is the space of images, and  $\mathcal{Z}$  is the space of latent features.  $q(\mathbf{x})$  is a dataset of sample images, and  $p(\mathbf{z})$  is a simple “prior” distribution, e.g., a Gaussian; in unsupervised image translation (Zhu et al., 2017),  $\mathcal{X}$  and  $\mathcal{Z}$  are both image spaces and  $q(\mathbf{x}), p(\mathbf{z})$  are sample images from two different domains (e.g., pictures of horses and zebras).

The underlying joint distribution on  $(\mathbf{x}, \mathbf{z})$  is not known, and we are not given any sample from it. Our goal is to nonetheless learn some model of the joint distribution  $r_{\text{model}}(\mathbf{x}, \mathbf{z})$  with the following desiderata:

**Desideratum 1. Matching Marginal** The marginals of  $r_{\text{model}}(\mathbf{x}, \mathbf{z})$  over  $\mathbf{x}, \mathbf{z}$  respectively match the provided distributions  $q(\mathbf{x}), p(\mathbf{z})$ .

**Desideratum 2. Meaningful Relationship**  $r_{\text{model}}(\mathbf{x}, \mathbf{z})$  captures a meaningful relationship between  $\mathbf{x}$  and  $\mathbf{z}$ . For example, in latent variable modeling of images, the latent variables  $\mathbf{z}$  should correspond to semantically meaningful features describing the image  $\mathbf{x}$ . In unsupervised image translation,  $r_{\text{model}}(\mathbf{x}, \mathbf{z})$  should capture the “correct” pairing between  $\mathbf{x}$  and  $\mathbf{z}$ .

We address desideratum 1 in this section, and desideratum 2 in section 3. The joint distribution  $r_{\text{model}}(\mathbf{x}, \mathbf{z})$  can be represented in factorized form by chain rule. To do so, we define conditional distribution families  $\{p_{\theta^p}(\mathbf{x}|\mathbf{z}), \theta^p \in \Theta^p\}$  and  $\{q_{\theta^q}(\mathbf{z}|\mathbf{x}), \theta^q \in \Theta^q\}$ . We require that for any  $\mathbf{z}$  we can both efficiently sample from  $p_{\theta^p}(\mathbf{x}|\mathbf{z})$  and compute  $\log p_{\theta^p}(\mathbf{x}|\mathbf{z})$ , and similarly for  $q_{\theta^q}(\mathbf{z}|\mathbf{x})$ . For compactness we use  $\theta = (\theta^p, \theta^q)$  to denote the parameters of both distributions  $p_{\theta}$  and  $q_{\theta}$ . We define the joint distribu-

tion  $r_{\text{model}}(\mathbf{x}, \mathbf{z})$  in two ways:

$$r_{\text{model}}(\mathbf{x}, \mathbf{z}) \stackrel{\text{def}}{=} p_{\theta}(\mathbf{x}, \mathbf{z}) \stackrel{\text{def}}{=} p(\mathbf{z})p_{\theta}(\mathbf{x}|\mathbf{z}) \quad (1)$$

and symmetrically

$$r_{\text{model}}(\mathbf{x}, \mathbf{z}) \stackrel{\text{def}}{=} q_{\theta}(\mathbf{x}, \mathbf{z}) \stackrel{\text{def}}{=} q(\mathbf{x})q_{\theta}(\mathbf{z}|\mathbf{x}) \quad (2)$$

Defining the model in two (redundant) ways seem unusual but has significant computational advantages: given  $\mathbf{x}$  we can tractably sample  $\mathbf{z}$ , and vice versa. For example, in latent variable models, given observed data  $\mathbf{x}$  we can sample latent features from  $\mathbf{z} \sim q_{\theta}(\mathbf{z}|\mathbf{x})$  (amortized inference), and given latent feature  $\mathbf{z}$  we can generate novel samples from  $\mathbf{x} \sim p_{\theta}(\mathbf{x}|\mathbf{z})$  (ancestral sampling).

If the two definitions (1), (2) are **consistent**, which we define as  $p_{\theta}(\mathbf{x}, \mathbf{z}) = q_{\theta}(\mathbf{x}, \mathbf{z})$ , we automatically satisfy desideratum 1:

$$\begin{aligned} r_{\text{model}}(\mathbf{x}) &= \int_{\mathbf{z}} r_{\text{model}}(\mathbf{x}, \mathbf{z}) d\mathbf{z} = \int_{\mathbf{z}} q_{\theta}(\mathbf{x}, \mathbf{z}) d\mathbf{z} = q(\mathbf{x}) \\ r_{\text{model}}(\mathbf{z}) &= \int_{\mathbf{x}} r_{\text{model}}(\mathbf{x}, \mathbf{z}) d\mathbf{x} = \int_{\mathbf{x}} p_{\theta}(\mathbf{x}, \mathbf{z}) d\mathbf{x} = p(\mathbf{z}) \end{aligned}$$

Based on this observation, we can design objectives that encourage consistency. Many latent variable generative models fit into this framework. For example, variational autoencoders (VAE, Kingma & Welling (2013)) enforce consistency by minimizing the KL divergence:

$$\min_{\theta} D_{\text{KL}}(q_{\theta}(\mathbf{x}, \mathbf{z}) \| p_{\theta}(\mathbf{x}, \mathbf{z}))$$

This minimization is equivalent to maximizing the evidence lower bound ( $\mathcal{L}_{\text{ELBO}}$ ) (Kingma & Welling, 2013):

$$\begin{aligned} &\max_{\theta} -D_{\text{KL}}(q_{\theta}(\mathbf{x}, \mathbf{z}) \| p_{\theta}(\mathbf{x}, \mathbf{z})) \quad (3) \\ &= -\mathbb{E}_{q_{\theta}(\mathbf{x}, \mathbf{z})} [\log(q_{\theta}(\mathbf{z}|\mathbf{x})q(\mathbf{x})) - \log(p_{\theta}(\mathbf{x}|\mathbf{z})p(\mathbf{z}))] \\ &= \mathbb{E}_{q_{\theta}(\mathbf{x}, \mathbf{z})} [\log p_{\theta}(\mathbf{x}|\mathbf{z})] + H_q(\mathbf{x}) \\ &\quad - \mathbb{E}_{q(\mathbf{x})} [D_{\text{KL}}(q_{\theta}(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z}))] \\ &\equiv \mathbb{E}_{q_{\theta}(\mathbf{x}, \mathbf{z})} [\log p_{\theta}(\mathbf{x}|\mathbf{z})] \\ &\quad - \mathbb{E}_{q(\mathbf{x})} [D_{\text{KL}}(q_{\theta}(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z}))] \quad \Big\} \mathcal{L}_{\text{ELBO}} \quad (4) \end{aligned}$$

where  $H_q(\mathbf{x})$  is the entropy of  $q(\mathbf{x})$  and is a constant that can be ignored for the purposes of optimization over model parameters  $\theta$  (denoted  $\equiv$ ).

As another example, BiGAN/ALI (Donahue et al., 2016; Dumoulin et al., 2016b) use an adversarial discriminator to approximately minimize the Jensen-Shannon divergence

$$\min_{\theta} D_{\text{JS}}(q_{\theta}(\mathbf{x}, \mathbf{z}) \| p_{\theta}(\mathbf{x}, \mathbf{z}))$$

Many other ways of enforcing consistency are possible. Most generally, we can enforce consistency with a vector of divergences  $\mathcal{D} = [D_1, \dots, D_m]$ , where each  $D_i$  takes two

probability measures as input, and outputs a non-negative value which is zero if and only if the two input measures are the same. Examples of possible divergences include Maximum Mean Discrepancy (MMD, Gretton et al. (2007)), denoted  $D_{\text{MMD}}$ ; Wasserstein distance (Arjovsky et al., 2017), denoted  $D_{\text{W}}$ ;  $f$ -divergences (Nowozin et al., 2016), denoted  $D_f$ ; and Jensen-Shannon divergence (Goodfellow et al., 2014), denoted  $D_{\text{JS}}$ .

Each  $D_i$  can be any divergence applied to a pair of probability measures. The pair of probability measures can be defined over either both variables  $(\mathbf{x}, \mathbf{z})$ , a single variable  $\mathbf{x}, \mathbf{z}$ , or conditional  $\mathbf{x}|\mathbf{z}, \mathbf{z}|\mathbf{x}$ . If the probability measure is defined over a conditional  $\mathbf{x}|\mathbf{z}, \mathbf{z}|\mathbf{x}$ , we also take expectation over the conditioning variable with respect to  $p_\theta$  or  $q_\theta$ . Some examples of  $D_i$  are:

$$\begin{aligned} & \mathbb{E}_{q_\theta(\mathbf{z})}[D_{\text{KL}}(q_\theta(\mathbf{x}|\mathbf{z})\|p_\theta(\mathbf{x}|\mathbf{z}))] \\ & D_{\text{MMD}}(q_\theta(\mathbf{z})\|p(\mathbf{z})) \\ & D_{\text{W}}(p_\theta(\mathbf{x}, \mathbf{z})\|q_\theta(\mathbf{x}, \mathbf{z})) \\ & \mathbb{E}_{q(\mathbf{x})}[D_f(q_\theta(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x}))] \\ & D_{\text{JS}}(q(\mathbf{x})\|p_\theta(\mathbf{x})) \end{aligned}$$

We only require that

$$D_i = 0, \forall i \in \{1, \dots, m\} \iff p_\theta(\mathbf{x}, \mathbf{z}) = q_\theta(\mathbf{x}, \mathbf{z})$$

so  $\mathcal{D} = \mathbf{0}$  implies consistency. Note that each  $D_i$  implicitly depends on the parameters  $\theta$  through  $p_\theta$  and  $q_\theta$ , but notationally we neglect this for simplicity.

Enforcing consistency  $p_\theta(\mathbf{x}, \mathbf{z}) = q_\theta(\mathbf{x}, \mathbf{z})$  by  $\mathcal{D} = \mathbf{0}$  satisfies desideratum 1 (matching marginal), but does not directly address desideratum 2 (meaningful relationship). A large number of joint distributions can have the same marginal distributions  $p(\mathbf{z})$  and  $q(\mathbf{x})$  (including ones where  $\mathbf{z}$  and  $\mathbf{x}$  are independent), and only a small fraction of them encode meaningful models.

### 3 GENERATIVE MODELING AS CONSTRAINT OPTIMIZATION

To address desideratum 2, we modify the training objective and specify additional preferences among consistent  $p_\theta(\mathbf{x}, \mathbf{z})$  and  $q_\theta(\mathbf{x}, \mathbf{z})$ . Formally we solve the following primal optimization problem

$$\min_{\theta} f(\theta) \quad \text{subject to } \mathcal{D} = \mathbf{0} \quad (5)$$

where  $f(\theta)$  encodes our preferences over consistent distributions, and depends on  $\theta$  through  $p_\theta(\mathbf{x}|\mathbf{z})$  and  $q_\theta(\mathbf{x}|\mathbf{z})$ .

An important preference is the mutual information between  $\mathbf{x}$  and  $\mathbf{z}$ . Depending on the downstream application, we may maximize mutual information (Chen et al., 2016b; Zhao et al., 2017; Li et al., 2017a; Chen et al., 2016a) so that the features (latent variables)  $\mathbf{z}$  can capture as much

information as possible about  $\mathbf{x}$ , or minimize mutual information (Zhao et al., 2017; Higgins et al., 2016; Tishby & Zaslavsky, 2015; Shamir et al., 2010) to achieve compression. To implement mutual information preference we consider the following objective

$$f_I(\theta; \alpha_1, \alpha_2) = \alpha_1 I_{q_\theta}(\mathbf{x}; \mathbf{z}) + \alpha_2 I_{p_\theta}(\mathbf{x}; \mathbf{z}) \quad (6)$$

where  $I_{p_\theta}(\mathbf{x}; \mathbf{z}) = \mathbb{E}_{p_\theta(\mathbf{x}, \mathbf{z})}[\log p_\theta(\mathbf{x}, \mathbf{z}) - \log p_\theta(\mathbf{x})p(\mathbf{z})]$  is the mutual information under  $p_\theta(\mathbf{x}, \mathbf{z})$ , and  $I_{q_\theta}(\mathbf{x}; \mathbf{z})$  is their mutual information under  $q_\theta(\mathbf{x}, \mathbf{z})$ .

The optimization problem in Eq.(5) with mutual information  $f(\theta)$  in Eq.(6) has the following Lagrangian dual function:

$$\alpha_1 I_{q_\theta}(\mathbf{x}; \mathbf{z}) + \alpha_2 I_{p_\theta}(\mathbf{x}; \mathbf{z}) + \boldsymbol{\lambda}^\top \mathcal{D} \quad (7)$$

where  $\boldsymbol{\lambda} = [\lambda_1, \dots, \lambda_m]$  is a vector of Lagrange multipliers, one for each of the  $m$  consistency constraints in  $\mathcal{D} = [D_1, \dots, D_m]$ .

In the next section, we will show that many existing training objectives for generative models minimize the Lagrangian dual in Equation 7 for some *fixed*  $\alpha_1, \alpha_2, \mathcal{D}$  and  $\boldsymbol{\lambda}$ . However, dual optimization requires maximization over the dual parameters  $\boldsymbol{\lambda}$ , which should *not* be kept fixed. We discuss dual optimization in Section 5.

## 4 GENERALIZING OBJECTIVES WITH FIXED MULTIPLIERS

Several existing objectives for latent variable generative models can be rewritten in the dual form of Equation 7 with *fixed* Lagrange multipliers. We provide several examples here and provide more in Appendix A.

**VAE (Kingma & Welling, 2013)** Per our discussion in Section 2, the VAE training objective commonly written as ELBO maximization in Eq.(4) is actually equivalent to Equation 3. This is a dual form where we set  $\mathcal{D} = [D_{\text{KL}}(q_\theta(\mathbf{x}, \mathbf{z})\|p_\theta(\mathbf{x}, \mathbf{z}))]$ ,  $\alpha_1 = \alpha_2 = 0$  and  $\boldsymbol{\lambda} = 1$ . Because  $\alpha_1 = \alpha_2 = 0$ , this objective has no information preference, confirming previous observations that the learned distribution can have high, low or zero mutual information between  $x$  and  $z$ . Chen et al. (2016b); Zhao et al. (2017).

**$\beta$ -VAE (Higgins et al., 2016)** The following objective  $\mathcal{L}_{\beta\text{-VAE}}$  is proposed to learn disentangled features  $\mathbf{z}$ :

$$-\mathbb{E}_{q_\theta(\mathbf{x}, \mathbf{z})}[\log p_\theta(\mathbf{x}|\mathbf{z})] + \beta \mathbb{E}_{q(\mathbf{x})}[D_{\text{KL}}(q_\theta(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))]$$

This is equivalent to the following dual form:

$$\begin{aligned} & \mathcal{L}_{\beta\text{-VAE}} \\ & \equiv \mathbb{E}_{q_\theta(\mathbf{x}, \mathbf{z})} \left[ \log \frac{q_\theta(\mathbf{x}|\mathbf{z})q(\mathbf{x})}{p_\theta(\mathbf{x}|\mathbf{z})q_\theta(\mathbf{x}|\mathbf{z})} + \beta \log \frac{q_\theta(\mathbf{z}|\mathbf{x})q_\theta(\mathbf{z})}{q_\theta(\mathbf{z})p(\mathbf{z})} \right] \\ & \equiv (\beta - 1) I_{q_\theta}(\mathbf{x}; \mathbf{z}) \quad (\text{primal}) \\ & \quad + \beta D_{\text{KL}}(q_\theta(\mathbf{z})\|p(\mathbf{z})) \quad (\text{consistency}) \\ & \quad + \mathbb{E}_{q_\theta(\mathbf{z})}[D_{\text{KL}}(q_\theta(\mathbf{x}|\mathbf{z})\|p_\theta(\mathbf{x}|\mathbf{z}))] \end{aligned}$$

$f(p, q)$	Likelihood Based	Unary Likelihood Free	Binary Likelihood Free
0	VAE (Kingma & Welling, 2013)	VAE-GAN (Makhzani et al., 2015)	ALI (Dumoulin et al., 2016b)
$\alpha_1 I_q$	$\beta$ -VAE (Higgins et al., 2016)	InfoVAE (Zhao et al., 2017)	ALICE (Li et al., 2017a)
$\alpha_2 I_p$	VMI (Barber & Agakov, 2003)	InfoGAN (Chen et al., 2016a)	-
$\alpha_1 I_q + \alpha_2 I_p$	-	CycleGAN (Zhu et al., 2017)	AS-VAE (Pu et al., 2017)

Table 1: For each choice of  $\alpha$  and computability class (Definition 2) we list the corresponding existing model. Several other objectives are also Lagrangian duals, but they are not listed because they are similar to models in the table. These objectives include DiscoGAN (Kim et al., 2017), BiGAN (Donahue et al., 2016), AAE (Makhzani et al., 2015), WAE (Tolstikhin et al., 2017).

where we use  $\equiv$  to denote “equal up to a value that does not depend on  $\theta$ ”. In this case,

$$\alpha_1 = \beta - 1, \quad \alpha_2 = 0$$

$$\lambda = [\beta, 1]$$

$$\mathcal{D} = [KL(q_\theta(\mathbf{z})\|p(\mathbf{z})), \mathbb{E}_{q_\theta(\mathbf{z})}[D_{\text{KL}}(q_\theta(\mathbf{x}|\mathbf{z})\|p_\theta(\mathbf{x}|\mathbf{z}))]]$$

When  $\alpha_1 > 0$  or equivalently  $\beta > 1$ , there is an incentive to minimize mutual information between  $\mathbf{x}$  and  $\mathbf{z}$ .

**InfoGAN (Chen et al., 2016a)** As another example, the InfoGAN objective <sup>1</sup> :

$$\mathcal{L}_{\text{InfoGAN}} = D_{\text{JS}}(q(\mathbf{x})\|p_\theta(\mathbf{x})) - \mathbb{E}_{p_\theta(\mathbf{x}, \mathbf{z})}[\log q_\theta(\mathbf{z}|\mathbf{x})]$$

is equivalent to the following dual form:

$$\begin{aligned} \mathcal{L}_{\text{InfoGAN}} &\equiv \mathbb{E}_{p_\theta(\mathbf{x}, \mathbf{z})}[-\log p_\theta(\mathbf{z}|\mathbf{x}) + \log p(\mathbf{z}) \\ &+ \log p_\theta(\mathbf{z}|\mathbf{x}) - \log q_\theta(\mathbf{z}|\mathbf{x})] + D_{\text{JS}}(q(\mathbf{x})\|p_\theta(\mathbf{x})) \\ &\equiv -I_{p_\theta}(\mathbf{x}; \mathbf{z}) \quad (\text{primal}) \\ &+ \mathbb{E}_{p_\theta(\mathbf{x})}[D_{\text{KL}}(p_\theta(\mathbf{z}|\mathbf{x})\|q_\theta(\mathbf{z}|\mathbf{x}))] \quad (\text{consistency}) \\ &+ D_{\text{JS}}(q(\mathbf{x})\|p_\theta(\mathbf{x})) \end{aligned}$$

In this case  $\alpha_1 = 0$ , and  $\alpha_2 = -1 < 0$ , the model maximizes mutual information between  $\mathbf{x}$  and  $\mathbf{z}$ .

In fact, all objectives in Table 1 belong to this class<sup>2</sup>. Derivations for additional models can be found in Appendix A.

#### 4.1 ENUMERATION OF ALL OBJECTIVES

The Lagrangian dual form of an objective reveals its mutual information preference ( $\alpha_1, \alpha_2$ ), type of consistency constraints ( $\mathcal{D}$ ), and weighting of the constraints ( $\lambda$ ). This suggests that the Lagrangian dual perspective may unify many existing training objectives. We wish to identify and categorize *all* objectives that have Lagrangian dual form as in Eq.7). However, this has two technical difficulties that we proceed to resolve.

<sup>1</sup>For conciseness we use  $\mathbf{z}$  to denote structured latent variables, which is represented as  $\mathbf{c}$  in (Chen et al., 2016a).

<sup>2</sup>Variational Mutual Information Maximization (VMI) is not truly a Lagrangian dual because it does not enforce consistency constraints ( $\lambda = 0$ ).

**1. Equivalence:** Many objectives appear different, but are actually identical for the purposes of optimization (as we have shown). To handle this we characterize “equivalent objectives” with a set of pre-specified transformations.

**Definition 1. Equivalence (Informal):** An objective  $\mathcal{L}$  is equivalent to  $\mathcal{L}'$  when there exists a constant  $C$ , so that for all parameters  $\theta$ ,  $\mathcal{L}(\theta) = \mathcal{L}'(\theta) + C$ . We denote this as  $\mathcal{L} \equiv \mathcal{L}'$ .

$\mathcal{L}$  and  $\mathcal{L}'$  are elementary equivalent if  $\mathcal{L}'$  can be obtained from  $\mathcal{L}$  by applying chain rule or Bayes rule to probabilities in  $\mathcal{L}$ , and addition/subtraction of constants  $\mathbb{E}_{q(\mathbf{x})}[\log q(\mathbf{x})]$  and  $\mathbb{E}_{p(\mathbf{z})}[\log p(\mathbf{z})]$ .

A more formal but verbose definition is in Appendix B, Definition 1.

Elementary equivalences define simple yet flexible transformations for deriving equivalent objectives. For example, all the transformations in Section 4 (VAE,  $\beta$ -VAE and InfoGAN) and Appendix A are elementary. This implies that all objectives in Table 1 are elementary equivalent to a Lagrangian dual function in Eq.(7). However, these transformations are not exhaustive. For example, transforming  $\mathbb{E}_{p_\theta}[g(\mathbf{x})]$  into  $\mathbb{E}_{q_\theta}[g(\mathbf{x})p_\theta(\mathbf{x})/q_\theta(\mathbf{x})]$  via importance sampling is not accounted for, hence the two objectives are not considered to be elementary equivalent.

**2. Optimization Difficulty:** Some objectives are easier to evaluate/optimize than others. For example, variational autoencoder training is robust and stable, adversarial training is less stable and requires careful hyper-parameter selection (Kodali et al., 2018), and direct optimization of the log-likelihood  $\log p_\theta(\mathbf{x})$  is very difficult for latent variable models and almost never used Grover et al. (2018).

To assign a “hardness score” to each objective, we first group the “terms” (an objective is a sum of terms) from easy to hard to optimize. An objective belongs to a “hardness class” if it cannot be transformed into an objective with easier terms. This is formalized below:

**Definition 2. Effective Optimization:** We define

1. Likelihood-based terms as the following set

$$T_1 = \{\mathbb{E}_{p_\theta(\mathbf{x}, \mathbf{z})}[\log p_\theta(\mathbf{x}|\mathbf{z})], \mathbb{E}_{p_\theta(\mathbf{x}, \mathbf{z})}[\log p_\theta(\mathbf{x}, \mathbf{z})], \\ \mathbb{E}_{p_\theta(\mathbf{z})}[\log p(\mathbf{z})], \mathbb{E}_{p_\theta(\mathbf{x}, \mathbf{z})}[\log q_\theta(\mathbf{z}|\mathbf{x})] \\ \mathbb{E}_{q_\theta(\mathbf{x}, \mathbf{z})}[\log p_\theta(\mathbf{x}|\mathbf{z})], \mathbb{E}_{q_\theta(\mathbf{x}, \mathbf{z})}[\log p_\theta(\mathbf{x}, \mathbf{z})], \\ \mathbb{E}_{q_\theta(\mathbf{z})}[\log p(\mathbf{z})], \mathbb{E}_{q_\theta(\mathbf{x}, \mathbf{z})}[\log q_\theta(\mathbf{z}|\mathbf{x})]\}$$

2. Unary likelihood-free terms as the following set

$$T_2 = \{D(q(\mathbf{x})\|p_\theta(\mathbf{x})), D(q_\theta(\mathbf{z})\|p(\mathbf{z}))\}$$

3. Binary likelihood-free terms as the following set

$$T_3 = \{D(q_\theta(\mathbf{x}, \mathbf{z})\|p_\theta(\mathbf{x}, \mathbf{z}))\}$$

where each  $D$  can be  $f$ -divergence, Jensen Shannon divergence, Wasserstein distance, or Maximum Mean Discrepancy. An objective  $\mathcal{L}$  is likelihood-based computable if  $\mathcal{L}$  is elementary equivalent to some  $\mathcal{L}'$  that is a linear combination of elements in  $T_1$ ; unary likelihood-free computable if  $\mathcal{L}'$  is a linear combination of elements in  $T_1 \cup T_2$ ; binary likelihood-free computable if  $\mathcal{L}'$  is a linear combination of elements in  $T_1 \cup T_2 \cup T_3$ .

The rationale of this categorization is that elements in  $T_1$  can be estimated by Monte-Carlo estimators and optimized by stochastic gradient descent effectively in practice (with low bias and variance) (Kingma & Welling, 2013; Rezende et al., 2014). In contrast, elements in  $T_2$  are optimized by likelihood-free approaches such as adversarial training (Goodfellow et al., 2014) or kernelized methods such as MMD (Gretton et al., 2007) or Stein variational gradient (Liu & Wang, 2016). These optimization procedures are known to suffer from stability problems (Arjovsky et al., 2017) or cannot handle complex distributions in high dimensions (Ramdas et al., 2015). Finally, elements in  $T_3$  are over both  $\mathbf{x}$  and  $\mathbf{z}$ , and they are empirically shown to be even more difficult to optimize (Li et al., 2017a). We do not include terms such as  $\mathbb{E}_{q(\mathbf{x})}[\log p_\theta(\mathbf{x})]$  because they are seldom feasible to compute or optimize for latent variable generative models.

Now we are able to fully characterize all Lagrangian dual objectives in Eq. (7) that are likelihood-based / unary likelihood free / binary likelihood free computable in Table 1.

In addition, Table 1 contains essentially *all* possible models for each optimization difficulty class in Definition 2. This is shown in the following theorem (informal, formal version and proof in Appendix B, Theorem 3,4,5)

**Theorem 1. Closure theorem (Informal):** Denote a Lagrangian objectives in the form of Equation 7 where all divergences are  $D_{\text{KL}}$  a KL Lagrangian objective. Under elementary equivalence defined in Definition 1,

1) Any KL Lagrangian objective that is elementary equivalent to a likelihood based computable objective is equivalent to a linear combination of VMI and  $\beta$ -VAE.

2) Any KL Lagrangian objective that is elementary equivalent to a unary likelihood computable objective is equivalent to a linear combination of InfoVAE and InfoGAN.

3) Any KL Lagrangian objective that is elementary equivalent to a binary likelihood computable objective is equivalent to a linear combination of ALICE, InfoVAE and InfoGAN.

We also argue in the Appendix (without formal proof) that this theorem holds for other divergences including  $D_{\text{MMD}}$ ,  $D_{\text{W}}$ ,  $D_f$  or  $D_{\text{JS}}$ .

Intuitively, this suggests a rather negative result: if a new latent variable model training objective contains mutual information preference and consistency constraints (defined through  $D_{\text{KL}}$ ,  $D_{\text{MMD}}$ ,  $D_{\text{W}}$ ,  $D_f$  or  $D_{\text{JS}}$ ), and this objective can be effectively optimized as in Definition 1 and Definition 2, then this objective is a linear combination of existing objectives. Our limitation is that we are restricted to elementary transformations and the set of terms defined in Definition 2. To derive new training objectives, we should consider new transformations, non-linear combinations and/or new terms.

## 5 DUAL OPTIMIZATION FOR LATENT VARIABLE GENERATIVE MODELS

While existing objectives for latent variable generative models have dual form in Equation 7, they are not solving the dual problem exactly because the Lagrange multipliers  $\lambda$  are predetermined instead of optimized. In particular, if we can show strong duality, the optimal solution to the dual is also an optimal solution to the primal (Boyd & Vandenberghe, 2004). However if the Lagrange multipliers are fixed, this property is lost, and the parameters  $\theta$  obtained via dual optimization may be suboptimal for  $\min_{\theta} f(\theta)$ , or violate the consistency conditions  $\mathcal{D} = \mathbf{0}$ .

### 5.1 RELAXATION OF CONSISTENCY CONSTRAINTS

This observation motivates us to directly solve the dual optimization problem where we also *optimize* the Lagrange multipliers.

$$\max_{\lambda \geq 0} \min_{\theta} f(\theta) + \lambda^T \mathcal{D}$$

Unfortunately, this is usually impractical because the consistency constraints are difficult to satisfy when the model has finite capacity, so in practice the primal optimization problem is actually infeasible and  $\lambda$  will be optimized to  $+\infty$ .

One approach to this problem is to use relaxed consistency constraints, where compared to Eq.(5) we require consis-

tency up to some error  $\epsilon > 0$ :

$$\min_{\theta} f(\theta) \quad \text{subject to} \quad \mathcal{D} \leq \epsilon \quad (8)$$

For a sufficiently large  $\epsilon$ , the problem is feasible. This has the corresponding dual problem:

$$\max_{\lambda \geq 0} \min_{\theta} f(\theta) + \lambda^{\top} (\mathcal{D} - \epsilon) \quad (9)$$

When  $\lambda$  is constant (instead of maximized), Equation 9 still reduces to existing latent variable generative modeling objectives since  $\lambda^{\top} \epsilon$  is a constant, so the objective simply becomes

$$\min_{\theta} f(\theta) + \lambda^{\top} \mathcal{D} + \text{constant}$$

In contrast, we propose to find  $\lambda^*$ ,  $\theta^*$  that optimize the Lagrangian dual in Eq.(9). If we additionally have strong duality,  $\theta^*$  is also the optimal solution to the primal problem in Eq.(8).

## 5.2 STRONG DUALITY WITH MUTUAL INFORMATION OBJECTIVES

This section aims to show that strong duality for Eq.(8) holds in distribution space if we replace mutual informations in  $f$  with upper and lower bounds. We prove this via Slater's condition (Boyd & Vandenberghe, 2004), which has three requirements: 1.  $\forall D \in \mathcal{D}$ ,  $D$  is convex in  $\theta$ ; 2.  $f(\theta)$  is convex for  $\theta \in \Theta$ ; 3. the problem is strictly feasible:  $\exists \theta$  s.t.  $\mathcal{D} < \epsilon$ . We propose weak conditions to satisfy all three in distribution space, so strong duality is guaranteed.

For simplicity we focus on discrete  $\mathcal{X}$  and  $\mathcal{Z}$ . We parameterize  $q_{\theta}(z|\mathbf{x})$  with a parameter matrix  $\theta^q \in \mathbb{R}^{|\mathcal{X}||\mathcal{Z}|}$  (we add the superscript  $q$  to distinguish parameters of  $q_{\theta}$  from that of  $p_{\theta}$ ) where

$$q_{\theta}(z = j|\mathbf{x} = i) = \theta_{ij}^q, \forall i \in \mathcal{X}, j \in \mathcal{Z} \quad (10)$$

The only restriction is that  $\theta^q$  must correspond to valid conditional distributions. More formally, we require that  $\theta^q \in \Theta^q$ , where

$$\Theta^q = \left\{ \theta^q \in \mathbb{R}^{|\mathcal{X}||\mathcal{Z}|} \text{ s.t. } 0 \leq \theta_{ij}^q \leq 1, \sum_j \theta_{ij}^q = 1 \right\} \quad (11)$$

Similarly we can define  $\theta^p \in \Theta^p$  for  $p_{\theta}$ . We still use

$$\theta = [\theta^q, \theta^p], \quad \Theta = \Theta^q \times \Theta^p \quad (12)$$

to denote both sets of parameters.

**1) Constraints  $D \in \mathcal{D}$  are convex:** We show that some divergences used in existing models are convex in distribution space.

**Lemma 1** (Convex Constraints (Informal)).  $D_{\text{KL}}$ ,  $D_{\text{MMD}}$ , or  $D_f$  over any marginal distributions on  $x$  or  $z$  or joint distributions on  $(x, z)$  are convex with respect to  $\theta \in \Theta$  as defined in Eq.(12).

Therefore if one only uses these convex divergences, the first requirement for Slater's condition is satisfied.

**2) Convex Bounds for  $f(\theta)$ :**  $f(\theta) = \alpha_1 I_{q_{\theta}}(x; z) + \alpha_2 \bar{I}_{p_{\theta}}(x; z)$  is not itself guaranteed to be convex in general. However we observe that mutual information has a convex upper bound, and a concave lower bound, which we denote as  $\bar{I}_{q_{\theta}}$  and  $\underline{I}_{q_{\theta}}$  respectively:

$$\begin{aligned} I_{q_{\theta}}(\mathbf{x}; \mathbf{z}) & \quad (13) \\ &= \mathbb{E}_{q(\mathbf{x})}[D_{\text{KL}}(q_{\theta}(z|\mathbf{x})||p(z))] \quad \text{convex upper bound } \bar{I}_{q_{\theta}} \\ &\quad - D_{\text{KL}}(q_{\theta}(z)||p(z)) \quad \text{bound gap } \bar{I}_{q_{\theta}} - I_{q_{\theta}} \\ &= \mathbb{E}_{q_{\theta}(\mathbf{x}, \mathbf{z})}[\log p_{\theta}(\mathbf{x}|\mathbf{z})] + H_q(\mathbf{x}) \quad \text{concave lower bound } \underline{I}_{q_{\theta}} \\ &\quad + \mathbb{E}_{p(\mathbf{z})} D_{\text{KL}}(q(\mathbf{x}|\mathbf{z})||p_{\theta}(\mathbf{x}|\mathbf{z})) \quad \text{bound gap } I_{q_{\theta}} - \underline{I}_{q_{\theta}} \end{aligned}$$

The convexity/concavity of these bounds is shown by the following lemma, which we prove in the appendix

**Lemma 2** (Convex/Concave Bounds).  $\bar{I}_{q_{\theta}}$  is convex with respect to  $\theta \in \Theta$  as defined in Eq.(12), and  $\underline{I}_{q_{\theta}}$  is concave with respect to  $\theta \in \Theta$ .

A desirable property of these bounds is that if we look at the bound gaps (difference between bound and true value) in Eq.(13), they are 0 if the consistency constraint is satisfied (i.e.,  $p_{\theta}(\mathbf{x}, \mathbf{z}) = q_{\theta}(\mathbf{x}, \mathbf{z})$ ). They will be tight (bound gaps are small) when consistency constraints are approximately satisfied (i.e.,  $p_{\theta}(\mathbf{x}, \mathbf{z}) \approx q_{\theta}(\mathbf{x}, \mathbf{z})$ ). In addition we also denote identical bounds for  $I_{p_{\theta}}$  as  $\bar{I}_{p_{\theta}}$  and  $\underline{I}_{p_{\theta}}$ . Similar bounds for mutual information have been discussed in (Alemi et al., 2017).

**3) Strict Feasibility:** the optimization problem has non empty feasible set, which we show in the following lemma:

**Lemma 3** (Strict Feasibility). For discrete  $\mathcal{X}$  and  $\mathcal{Z}$ , and  $\epsilon > 0$ ,  $\exists \theta \in \Theta$  such that  $\mathcal{D} < \epsilon$ .

Therefore we have shown that for convex/concave upper and lower bounds on  $f$ , all three of Slater's conditions are satisfied, so strong duality holds. We summarize this in the following theorem.

**Theorem 2** (Strong Duality). If  $\mathcal{D}$  contains only divergences in Lemma 1, then for all  $\epsilon > 0$ :

If  $\alpha_1, \alpha_2 \geq 0$  strong duality holds for the following problems:

$$\min_{\theta \in \Theta} \alpha_1 \bar{I}_{q_{\theta}} + \alpha_2 \bar{I}_{p_{\theta}} \quad \text{subject to} \quad \mathcal{D} \leq \epsilon \quad (14)$$

If  $\alpha_1, \alpha_2 \leq 0$ , strong duality holds for the following problem

$$\min_{\theta \in \Theta} \alpha_1 \underline{I}_{q_{\theta}} + \alpha_2 \underline{I}_{p_{\theta}} \quad \text{subject to} \quad \mathcal{D} \leq \epsilon \quad (15)$$



---

**Algorithm 1** Dual Optimization for Latent Variable Generative Models

---

**Input:** Analytical form for  $p(\mathbf{z})$  and samples from  $q(\mathbf{x})$ ; constraints  $\mathcal{D}$ ;  $\alpha_1, \alpha_2$  that specify maximization / minimization of mutual information;  $\epsilon > 0$  which specifies the strength of constraints; step size  $\rho_\theta, \rho_\lambda$  for  $\theta$  and  $\lambda$ .

**Output:**  $\theta$  (parameters for  $p_\theta(\mathbf{x}|\mathbf{z})$  and  $q_\theta(\mathbf{z}|\mathbf{x})$ ).

---

Initialize  $\theta$  randomly

Initialize the Lagrange multipliers  $\lambda := 1$

**if**  $\alpha_1, \alpha_2 > 0$  **then**

$$f(\theta) \leftarrow \alpha_1 \bar{I}_{q_\theta} + \alpha_2 \bar{I}_{p_\theta}$$

**else**

$$f(\theta) \leftarrow \alpha_1 \underline{I}_{q_\theta} + \alpha_2 \underline{I}_{p_\theta}$$

**end if**

**for**  $t = 0, 1, 2, \dots$  **do**

$$\theta \leftarrow \theta - \rho_\theta (\nabla_\theta f(\theta) + \lambda^\top \nabla_\theta \mathcal{D})$$

$$\lambda \leftarrow \lambda + \rho_\lambda (\mathcal{D} - \epsilon)$$

**end for**

---

### 5.3 DUAL OPTIMIZATION

Because the problem is convex in distribution space and satisfies Slater’s condition, the  $\theta^*$  that achieves the saddle point

$$\lambda^*, \theta^* = \arg \max_{\lambda \geq 0} \arg \min_\theta f(\theta) + \lambda^\top (\mathcal{D} - \epsilon) \quad (16)$$

is also a solution to the original optimization problem Eq.(8) (Boyd & Vandenberghe, 2004)(Chapter 5.4). In addition the max-min problem Eq.(16) is convex with respect to  $\theta$  and concave (linear) with respect to  $\lambda$ , so one can apply iterative gradient descent/ascent over  $\theta$  (minimize) and  $\lambda$  (maximize) and achieve stable convergence to saddle point (Holding & Lestas, 2014). We describe the iterative algorithm in Algorithm 1.

In practice, we do not optimize over distribution space and  $\{p_\theta(\mathbf{x}|\mathbf{z})\}, \{q_\theta(\mathbf{z}|\mathbf{x})\}$  are some highly complex and non-convex families of functions. We show in the experimental section that this scheme is stable and effective despite non-convexity.

## 6 LAGRANGIAN VAE

In this section we consider a particular instantiation of the general dual problem proposed in the previous section. Consider the following primal problem, with  $\alpha_1 \in \mathbb{R}$ :

$$\begin{aligned} \min_{\theta} \quad & \alpha_1 I_{q_\theta}(\mathbf{x}; \mathbf{z}) & (17) \\ \text{subject to} \quad & D_{\text{KL}}(q_\theta(\mathbf{x}, \mathbf{z}) \| p_\theta(\mathbf{x}, \mathbf{z})) \leq \epsilon_1 \\ & D_{\text{MMD}}(q_\theta(\mathbf{z}) \| p(\mathbf{z})) \leq \epsilon_2 \end{aligned}$$

For mutual information minimization / maximization, we respectively replace the (possibly non-convex) mutual information by upper bound  $\bar{I}_{q_\theta}$  if  $\alpha_1 \geq 0$  and lower bound

$\underline{I}_{q_\theta}$  if  $\alpha_1 < 0$ . The corresponding dual optimization problem can be written as:

$$\max_{\lambda \geq 0} \min_{\theta} \begin{cases} \alpha_1 \bar{I}_{q_\theta} + \lambda^\top (\mathcal{D}_{\text{InfoVAE}} - \epsilon), & \alpha_1 \geq 0 \\ \alpha_1 \underline{I}_{q_\theta} + \lambda^\top (\mathcal{D}_{\text{InfoVAE}} - \epsilon), & \alpha_1 < 0 \end{cases} \quad (18)$$

where  $\epsilon = [\epsilon_1, \epsilon_2]$ ,  $\lambda = [\lambda_1, \lambda_2]$  and

$$\begin{aligned} \mathcal{D}_{\text{InfoVAE}} &= [D_{\text{KL}}(q_\theta(\mathbf{x}, \mathbf{z}) \| p_\theta(\mathbf{x}, \mathbf{z})), \\ &D_{\text{MMD}}(q_\theta(\mathbf{z}) \| p(\mathbf{z}))] \end{aligned}$$

We call the objective in 18 Lagrangian (Info)VAE (LagVAE). Note that setting a constant  $\lambda$  for the dual function recovers the InfoVAE objective (Zhao et al., 2017). By Theorem 2 strong duality holds for this problem and finding the max-min saddle point of LagVAE in Eq.(18) is identical to finding the optimal solution to original problem of Eq.(17).

The final issue is choosing the  $\epsilon$  hyper-parameters so that the constraints are feasible. This is non-trivial since selecting  $\epsilon$  that describe feasible constraints depends on the task and model structure. We introduce a general strategy that is effective in all of our experiments. First we learn a parameter  $\theta^*$  that satisfies the consistency constraints “as well as possible” without considering mutual information maximization/minimization. Formally this is achieved by the following optimization (for any choice of  $\lambda > 0$ ),

$$\theta^* = \arg \min_{\theta} \lambda^\top \mathcal{D}_{\text{InfoVAE}} \quad (19)$$

This is the original training objective for InfoVAE with  $\alpha_1 = 0$  and can be optimized by

$$\begin{aligned} \min_{\theta} \lambda^\top \mathcal{D}_{\text{InfoVAE}} &= \lambda_1 D_{\text{KL}}(q_\theta(\mathbf{x}, \mathbf{z}) \| p_\theta(\mathbf{x}, \mathbf{z})) + \lambda_2 D_{\text{MMD}}(q_\theta(\mathbf{z}) \| p(\mathbf{z})) \\ &\equiv \lambda_1 \mathcal{L}_{\text{ELBO}}(\theta) + \lambda_2 D_{\text{MMD}}(q_\theta(\mathbf{z}) \| p(\mathbf{z})) \end{aligned} \quad (20)$$

where  $\mathcal{L}_{\text{ELBO}}(\theta)$  is the evidence lower bound defined in Eq.(4). Because we only need a rough estimate of how well consistency constraints can be satisfied, the selection of weighing  $\lambda_1$  and  $\lambda_2$  is unimportant. The recommendation in (Zhao et al., 2017) works well in all our experiments ( $\lambda_1 = 1, \lambda_2 = 100$ ).

Now we introduce a “slack” to specify how much we are willing to tolerate consistency error to achieve higher/lower mutual information. Formally, we define  $\hat{\epsilon}$  as the divergences  $\mathcal{D}_{\text{InfoVAE}}$  evaluated at the above  $\theta^*$ . Under this  $\hat{\epsilon}$  the following constraint must be feasible (because  $\theta^*$  is a solution):

$$\mathcal{D}_{\text{InfoVAE}} \leq \hat{\epsilon}$$

Now we can safely set  $\epsilon = \gamma + \hat{\epsilon}$ , where  $\gamma > 0$ , and the constraint

$$\mathcal{D}_{\text{InfoVAE}} \leq \epsilon$$

must still be feasible (and strictly feasible).  $\gamma$  has a very nice intuitive interpretation: it is the “slack” that we are willing to accept. Compared to tuning  $\alpha_1$  and  $\lambda$  for InfoVAE, tuning  $\gamma$  is much more interpretable: we can anticipate the final consistency error before training.

Another practical consideration is that the one of the constraints  $D_{\text{KL}}(q_\theta(\mathbf{x}, \mathbf{z}) \| p_\theta(\mathbf{x}, \mathbf{z}))$  is difficult to estimate. However, we have

$$D_{\text{KL}}(q_\theta(\mathbf{x}, \mathbf{z}) \| p_\theta(\mathbf{x}, \mathbf{z})) = -\mathcal{L}_{\text{ELBO}} - H_q(\mathbf{x})$$

where  $\mathcal{L}_{\text{ELBO}}$  is again, the evidence lower bound in Eq.(4) of Section 2, and  $H_q(\mathbf{x})$  is the entropy of the true distribution  $q(\mathbf{x})$ .  $\mathcal{L}_{\text{ELBO}}$  is empirically easy to estimate, and  $H_q(\mathbf{x})$  is a constant irrelevant to the optimization problem. The optimization problem is identical if we replacing the more difficult constraint  $D_{\text{KL}}(q_\theta(\mathbf{x}, \mathbf{z}) \| p_\theta(\mathbf{x}, \mathbf{z})) \leq \epsilon_1$  with the easier-to-optimize/estimate constraint  $-\mathcal{L}_{\text{ELBO}} \leq \epsilon'_1$  (where  $\epsilon'_1 = \epsilon_1 + H_q(\mathbf{x})$ ). In addition,  $\epsilon'_1$  can be selected by the technique in the previous paragraph.

## 7 EXPERIMENTS

We compare the performance of **LagVAE**, where we learn  $\lambda$  automatically, and **InfoVAE**, where we set  $\lambda$  in advance (as hyperparameters). Our primal problem is to find solutions that maximize / minimize mutual information under the consistency constraints. Therefore, we consider two performance metrics:

- $I_q(\mathbf{x}, \mathbf{z})$  the mutual information between  $\mathbf{x}$  and  $\mathbf{z}$ . We can estimate the mutual information via the identity:

$$I_q(\mathbf{x}; \mathbf{z}) = \mathbb{E}_{q_\theta(\mathbf{x}, \mathbf{z})} [\log q_\theta(\mathbf{z} | \mathbf{x}) - \log q_\theta(\mathbf{z})] \quad (21)$$

where we approximate  $q_\theta(\mathbf{z})$  with a kernel density estimator.

- the consistency divergences  $D_{\text{KL}}(q_\theta(\mathbf{x}, \mathbf{z}) \| p_\theta(\mathbf{x}, \mathbf{z}))$  and  $D_{\text{MMD}}(q_\theta(\mathbf{z}) \| p(\mathbf{z}))$ . As stated in Section 6, we replace  $D_{\text{KL}}(q_\theta(\mathbf{x}, \mathbf{z}) \| p_\theta(\mathbf{x}, \mathbf{z}))$  with the evidence lower bound  $\mathcal{L}_{\text{ELBO}}$ .

In the remainder of this section we demonstrate the following empirical observations:

- LagVAE reliably maximizes/minimizes mutual information without violating the consistency constraints. InfoVAE, on the other hand, makes unpredictable and task-specific trade-offs between mutual information and consistency.
- LagVAE is Pareto optimal, as no InfoVAE hyperparameter choice is able to achieve both better mutual information and better consistency (measured by  $D_{\text{MMD}}$  and  $\mathcal{L}_{\text{ELBO}}$ ) than LagVAE.

### 7.1 VERIFICATION OF DUAL OPTIMIZATION

We first verify that LagVAE reliably maximizes/minimizes mutual information subject to consistency constraints. We train LagVAE on MNIST according to Algorithm 1.  $\epsilon$  is selected according to Section 6, where we first compute  $\hat{\epsilon} = (\hat{\epsilon}_1, \hat{\epsilon}_2)$  without information maximization/minimization by Eq.(20). Next we choose slack variables  $\gamma = (\gamma_1, \gamma_2)$ , and set  $\epsilon = \hat{\epsilon} + \gamma$ . For  $\gamma_1$  we explore values from 0.1 to 4.0, and for  $\gamma_2$  we use the fixed value  $0.5\hat{\epsilon}_2$ .

The results are shown in Figure 1, where mutual information is estimated according to Eq.(21). For any given slack  $\gamma$ , setting  $\alpha_1$  to positive values and negative values respectively minimizes or maximizes the mutual information within the feasible set  $\mathcal{D} \leq \epsilon$ . In particular, the absolute value of  $\alpha_1$  does not affect the outcome, and only the sign matters. This is consistent with the expected behavior (Figure 1 Left) where the model finds the maximum/minimum mutual information solution within the feasible set.

### 7.2 VERIFICATION OF PARETO IMPROVEMENTS

In this section we verify Pareto optimality of LagVAE. We evaluate LagVAE and InfoVAE on the MNIST dataset with a wide variety of hyper-parameters. For LagVAE, we set  $\epsilon_1$  for  $\mathcal{L}_{\text{ELBO}}$  to be  $\{83, 84, \dots, 95\}$  and  $\epsilon_2$  for  $D_{\text{MMD}}$  to be 0.0005. For InfoVAE, we set  $\alpha \in \{1, -1\}$ ,  $\lambda_1 \in \{1, 2, 5, 10\}$  and  $\lambda_2 \in \{1000, 2000, 5000, 10000\}$ <sup>3</sup>.

Figure 2 plots the mutual information and  $\mathcal{L}_{\text{ELBO}}$  achieved by both methods. Each point is the outcome of one hyperparameter choice of LagVAE / InfoVAE. Regardless of the hyperparameter choice of both models, no InfoVAE hyperparameter lead to better performance on both mutual information and  $\mathcal{L}_{\text{ELBO}}$  on the training set. This is expected because LagVAE always finds the maximum/minimum mutual information solution out of all solutions with given consistency value. The same trend is true even on the test set, indicating that it is not an outcome of over-fitting.

## 8 CONCLUSION

Many existing objectives for latent variable generative modeling are Lagrangian dual functions of the same type of constrained optimization problem with fixed Lagrangian multipliers. This allows us to explore their statistical and computational trade-offs, and characterize all models in this class. Moreover, we propose a practical dual optimization method that optimizes both the Lagrange multipliers and the model parameters, allowing us to specify interpretable constraints and achieve Pareto-optimality empirically.

<sup>3</sup>Code for this set of experiments is available at <https://github.com/ermongroup/lagvae>

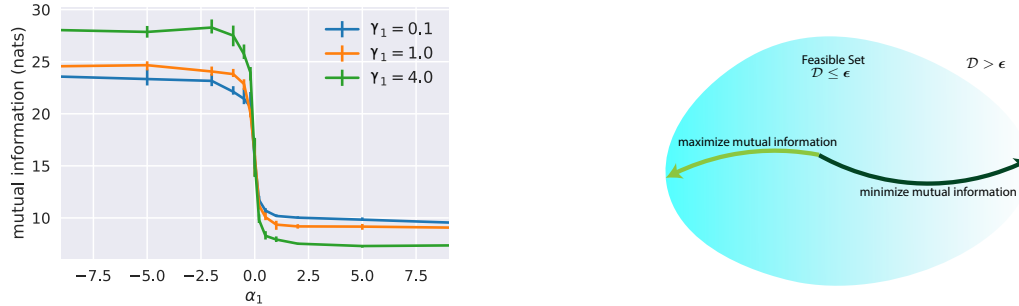


Figure 1: **Left:** Effect of  $\alpha_1$  and  $\gamma_1$  on the primal objective (mutual information). When  $\alpha_1$  is positive we minimize mutual information within the feasible set, and when  $\alpha_1$  is negative we maximize mutual information. When  $\alpha_1$  is zero the preference is undetermined, and mutual information varies depending on initialization. Note that mutual information does not depend on the absolute value of  $\alpha_1$  but only on its sign. **Right:** An illustration of this effect. Lagrangian dual optimization finds the maximum/minimum mutual information solution in the feasible set  $\mathcal{D} \leq \epsilon$ .

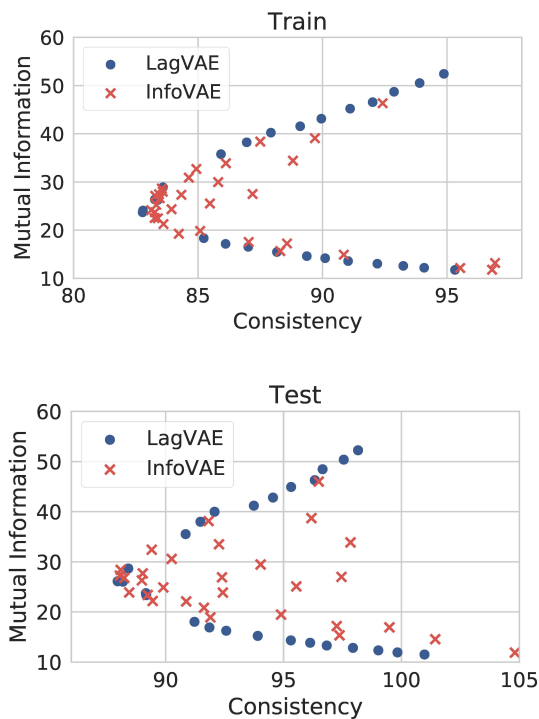


Figure 2: LagVAE Pareto dominates InfoVAE with respect to Mutual information and consistency ( $\mathcal{L}_{ELBO}$  values) on train (top) and test (bottom) set. Each point is the outcome of one hyper-parameter choice for LagVAE / InfoVAE. When we maximize mutual information ( $\alpha_1 < 0$ ), for any given  $\mathcal{L}_{ELBO}$  value, LagVAE always achieve similar or larger mutual information; when we minimize mutual information ( $\alpha_1 > 0$ ), for any given ELBO value, LagVAE always achieve similar or smaller mutual information.

In this work, we only considered Lagrangian (Info)VAE, but the method is generally applicable to other Lagrangian dual objectives. In addition we only considered mutual information preference. Exploring different preferences is a promising future directions.

### Acknowledgements

This research was supported by Intel Corporation, TRI, NSF (#1651565, #1522054, #1733686) and FLI (#2017-158687).

### References

Alemi, Alexander A., Poole, Ben, Fischer, Ian, Dillon, Joshua V., Saurous, Rif A., and Murphy, Kevin. An information-theoretic analysis of deep latent-variable models. *CoRR*, abs/1711.00464, 2017. URL <http://arxiv.org/abs/1711.00464>.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein GAN. *ArXiv e-prints*, January 2017.

Barber, David and Agakov, Felix. The im algorithm: a variational approach to information maximization. In *Proceedings of the 16th International Conference on Neural Information Processing Systems*, pp. 201–208. MIT Press, 2003.

Boyd, Stephen and Vandenberghe, Lieven. *Convex optimization*. Cambridge university press, 2004.

Chen, Xi, Duan, Yan, Houthoofd, Rein, Schulman, John, Sutskever, Ilya, and Abbeel, Pieter. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *arXiv preprint arXiv:1606.03657*, 2016a.

Chen, Xi, Kingma, Diederik P, Salimans, Tim, Duan, Yan, Dhariwal, Prafulla, Schulman, John, Sutskever, Ilya, and Abbeel, Pieter. Variational lossy autoencoder. *arXiv preprint arXiv:1611.02731*, 2016b.

- Dhar, Manik, Grover, Aditya, and Ermon, Stefano. Sparsegen: Modeling sparse deviations for compressed sensing using generative models. *International Conference on Machine Learning*, 2018.
- Donahue, Jeff, Krähenbühl, Philipp, and Darrell, Trevor. Adversarial feature learning. *CoRR*, abs/1605.09782, 2016. URL <http://arxiv.org/abs/1605.09782>.
- Dumoulin, Vincent, Belghazi, Ishmael, Poole, Ben, Lamb, Alex, Arjovsky, Martin, Mastropietro, Olivier, and Courville, Aaron. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016a.
- Dumoulin, Vincent, Belghazi, Ishmael, Poole, Ben, Lamb, Alex, Arjovsky, Martin, Mastropietro, Olivier, and Courville, Aaron. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016b.
- Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and Bengio, Yoshua. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.
- Gretton, Arthur, Borgwardt, Karsten M, Rasch, Malte, Schölkopf, Bernhard, and Smola, Alex J. A kernel method for the two-sample-problem. In *Advances in neural information processing systems*, pp. 513–520, 2007.
- Grover, Aditya, Dhar, Manik, and Ermon, Stefano. FlowGAN: Combining maximum likelihood and adversarial learning in generative models. In *AAAI Conference on Artificial Intelligence*, 2018.
- Higgins, Irina, Matthey, Loic, Pal, Arka, Burgess, Christopher, Glorot, Xavier, Botvinick, Matthew, Mohamed, Shakir, and Lerchner, Alexander. beta-vae: Learning basic visual concepts with a constrained variational framework. 2016.
- Holding, Thomas and Lestas, Ioannis. On the convergence to saddle points of concave-convex functions, the gradient method and emergence of oscillations. In *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, pp. 1143–1148. IEEE, 2014.
- Kim, Taeksoo, Cha, Moon-su, Kim, Hyunsoo, Lee, Jung Kwon, and Kim, Jiwon. Learning to discover cross-domain relations with generative adversarial networks. *CoRR*, abs/1703.05192, 2017. URL <http://arxiv.org/abs/1703.05192>.
- Kingma, D. P and Welling, M. Auto-Encoding Variational Bayes. *ArXiv e-prints*, December 2013.
- Kodali, Naveen, Hays, James, Abernethy, Jacob, and Kira, Zsolt. On convergence and stability of gans. 2018.
- Kuleshov, Volodymyr and Ermon, Stefano. Deep hybrid models: Bridging discriminative and generative approaches. In *Proceedings of the Conference on Uncertainty in AI (UAI)*, 2017.
- Li, Chunyuan, Liu, Hao, Chen, Changyou, Pu, Yunchen, Chen, Liqun, Henao, Ricardo, and Carin, Lawrence. Towards understanding adversarial learning for joint distribution matching. 2017a.
- Li, Yunzhu, Song, Jiaming, and Ermon, Stefano. Inferring the latent structure of human decision-making from raw visual inputs. 2017b.
- Liu, Qiang and Wang, Dilin. Stein variational gradient descent: A general purpose bayesian inference algorithm. *arXiv preprint arXiv:1608.04471*, 2016.
- Makhzani, Alireza, Shlens, Jonathon, Jaitly, Navdeep, and Goodfellow, Ian. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.
- Mescheder, Lars, Nowozin, Sebastian, and Geiger, Andreas. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. *arXiv preprint arXiv:1701.04722*, 2017.
- Mohamed, Shakir and Lakshminarayanan, Balaji. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*, 2016.
- Nowozin, Sebastian, Cseke, Botond, and Tomioka, Ryota. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems*, pp. 271–279, 2016.
- Pu, Yuchen, Wang, Weiyao, Henao, Ricardo, Chen, Liqun, Gan, Zhe, Li, Chunyuan, and Carin, Lawrence. Adversarial symmetric variational autoencoder. In *Advances in Neural Information Processing Systems*, pp. 4333–4342, 2017.
- Radford, Alec, Metz, Luke, and Chintala, Soumith. Un-supervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.
- Ramdas, Aaditya, Reddi, Sashank Jakkam, Póczos, Barnabás, Singh, Aarti, and Wasserman, Larry A. On the decreasing power of kernel and distance based non-parametric hypothesis tests in high dimensions. In *AAAI*, pp. 3571–3577, 2015.
- Rezende, D., Mohamed, S., and Wierstra, D. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. *ArXiv e-prints*, January 2014.
- Shamir, Ohad, Sabato, Sivan, and Tishby, Naftali. Learning and generalization with the information bottleneck. *Theoretical Computer Science*, 411(29-30):2696–2711, 2010.
- Tishby, Naftali and Zaslavsky, Noga. Deep learning and the information bottleneck principle. *CoRR*, abs/1503.02406, 2015. URL <http://arxiv.org/abs/1503.02406>.

- Tolstikhin, Ilya, Bousquet, Olivier, Gelly, Sylvain, and Schoelkopf, Bernhard. Wasserstein auto-encoders. *arXiv preprint arXiv:1711.01558*, 2017.
- Yang, Zichao, Hu, Zhiting, Salakhutdinov, Ruslan, and Berg-Kirkpatrick, Taylor. Improved variational autoencoders for text modeling using dilated convolutions. *CoRR*, abs/1702.08139, 2017. URL <http://arxiv.org/abs/1702.08139>.
- Zhao, Shengjia, Song, Jiaming, and Ermon, Stefano. Infovae: Information maximizing variational autoencoders. *CoRR*, abs/1706.02262, 2017. URL <http://arxiv.org/abs/1706.02262>.
- Zhu, Jun-Yan, Park, Taesung, Isola, Phillip, and Efros, Alexei A. Unpaired image-to-image translation using cycle-consistent adversarial networks. *CoRR*, abs/1703.10593, 2017. URL <http://arxiv.org/abs/1703.10593>.

---

# Bayesian optimization and attribute adjustment

---

**Stephan Eismann**  
CS Department  
Stanford University  
Stanford, CA 94305

**Daniel Levy**  
CS Department  
Stanford University  
Stanford, CA 94305

**Rui Shu**  
CS Department  
Stanford University  
Stanford, CA 94305

**Stefan Bartzsch**  
iRT Department  
Helmholtz-Zentrum Munich  
Munich, Germany

**Stefano Ermon**  
CS Department  
Stanford University  
Stanford, CA 94305

## Abstract

Automatic design via Bayesian optimization holds great promise given the constant increase of available data across domains. However, it faces difficulties from high-dimensional, potentially discrete, search spaces. We propose to probabilistically embed inputs into a lower dimensional, continuous latent space, where we perform gradient-based optimization guided by a Gaussian process. Building on variational autoencoders, we use both labeled and unlabeled data to guide the encoding and increase its accuracy. In addition, we propose an adversarial extension to render the latent representation invariant with respect to specific design attributes, which allows us to transfer these attributes across structures. We apply the framework both to a functional-protein dataset and to perform optimization of drag coefficients directly over high-dimensional shapes without incorporating domain knowledge or handcrafted features.

## 1 INTRODUCTION

Developing enhanced designs is an overarching goal across engineering disciplines ranging from the optimization of planes in aeronautics (Simpson et al., 2001) and batteries for electric vehicles (Grover et al., 2018) to the development of proteins in bioengineering (Damborsky and Brezovsky, 2014). The different optimization efforts often face the same challenges in form of search-space complexity and costly design evaluations which render naive exhaustive search infeasible and make human-expert knowledge a key success factor. The ever increasing amounts of experimental data have to be considered, however, pose new challenges to manual analysis.

Increasing amounts of data open new opportunities for statistical design approaches. In this context Bayesian optimization has emerged as a data-driven tool for automated design optimization (Shahriari et al., 2016). Bayesian optimization is a model-based approach with a prescribed prior belief on the functional score of designs. Given data, we sequentially update this belief and optimize a surrogate function to our true objective. The choice of this surrogate function thereby trades exploration vs. exploitation in the design space.

By leveraging the problem structure, Bayesian optimization can be much more sample efficient than random search (Snoek et al., 2012b), but it is not immune to the curse of dimensionality. Signal is often sparse in high-dimensional input spaces of many real-world design problems. In addition, desirable target applications like drug or material design involve optimization over *discrete* structures, where even optimizing the model-based surrogate is difficult.

We target the input-space challenges of high-dimensionality and discrete designs by combining Bayesian optimization with deep generative modeling. Specifically, we built on the architecture by Gómez-Bombarelli et al. (2016a; 2016b) combining a variational autoencoder (VAE) and Gaussian process (GP) regression. VAEs (Kingma and Welling, 2013) are probabilistic models which map high-dimensional, possibly discrete inputs to a lower dimensional continuous space. The encoder consists of a neural network whose feature-construction ability is leveraged for dimensionality reduction. We learn a GP on top of the latent space as its predictions enjoy uncertainty estimates such that we can explore the design space based on a confidence measure. At the same time, the continuous space now allows us to use gradient-based methods for optimization.

In this work we make the following contributions: (i) We propose to use parametric label-guidance for the autoencoder and demonstrate how this results in increased

label-prediction accuracy for the datasets we consider. (ii) We present a corresponding graphical model and derive a variational lower bound on the marginal log-likelihood. The variational bound provides us with a principled way of incorporating unlabeled data in the joint training procedure. We show that incorporating unlabeled data results in enhanced reconstruction accuracy for our datasets. (iii) We perform Bayesian design optimization on two different domains: proteins and shapes in laminar flow. Having access to a physics simulator, we show the validity of the approach in the hydrodynamics setting. (iv) We propose an adversarial model extension to render the latent representation invariant with respect to specific, real-valued design attributes. To compensate for the loss of information, we provide these attributes as additional arguments to the decoder. This allows us to transfer attributes across designs when we generate a design from its latent representation.

## 2 PROBLEM SETUP

We consider the setting of a high-dimensional, possibly discrete input space of designs  $\mathcal{X}$ . Each design  $x$  is associated with a real-valued score  $y \in \mathbb{R}$  drawn from a conditional distribution  $p^*(y|x)$ .

Our goal is to find a design  $x \in \mathcal{X}$  that maximizes the expectation  $\mathbb{E}[y|x]$ . We are given access to an oracle providing a sample of  $y \sim p^*(\cdot|x)$ , however, we assume that obtaining a sample (evaluating  $y$  for a given design  $x$ ) is expensive. For example, it might require an expensive simulation or conducting a lab experiment.

Furthermore, we assume to have access to samples  $D_u = \{x_1, \dots, x_{N_u}\}$  from  $\mathcal{X}$ , and a (small) number of labeled examples  $D_\ell = \{(x_1, y_1), \dots, (x_{N_\ell}, y_{N_\ell})\}$ , where each pair  $(x, y)$  corresponds to a design and a measurement of its score. We assume that labeled and unlabeled examples are sampled from the same marginal distribution.

### 2.1 BAYESIAN OPTIMIZATION

Bayesian optimization is a data-driven tool to optimize expensive black-box functions. In this model-based approach we start with a prior belief on the functional relationship between inputs and outputs, and update it sequentially as new data is acquired. As actual function evaluations are expensive, we aim to optimize a surrogate or acquisition function instead. A popular choice of acquisition function is expected improvement (EI) (Jones et al., 1998) which strikes to balance exploration vs. exploitation in the search space. To calculate EI we require a prediction with uncertainty for the black-box function values. Gaussian processes (GPs) provide uncertainty quantification and as such they are a standard model in Bayesian

optimization. In the framework of GPs we assume that given a finite number of  $n$  inputs  $x_{1:n}$ , the function values  $f(x_{1:n})$  are jointly Gaussian and the observations  $y_{1:n}$  are normally distributed given  $f$  (Rasmussen and Williams, 2006). Because of the GP guidance, it can be more sample efficient than random search, however, Bayesian optimization still faces the challenges of data sparsity when operating in high dimensions. In addition, gradient-based optimization of the (EI) surrogate is not a priori applicable for discrete inputs. Finally, the benefit of uncertainty quantification comes at a price, as learning in the nonparametric GP model is cubic in the number of inputs. To circumvent this bottleneck, different sparse approximations have been developed. One approach to reduce the computational complexity is to calculate the covariance matrix with respect to  $m$  inducing points instead of  $n$  data points and typically  $m \ll n$  with complexity  $\mathcal{O}(m^2n)$  (Titsias, 2009; McIntire et al., 2016).

### 2.2 VARIATIONAL AUTOENCODERS

A variational autoencoder is a generative model defining a joint probability distribution between a latent variable  $z$  and inputs  $x$ . We commonly assume a simple Gaussian prior distribution  $p(z)$  and model the input data distribution as a more complex conditional distribution  $p_\Psi(x|z)$  where  $\Psi$  are the parameters of a neural network. Directly optimizing the marginal likelihood is intractable as it requires integration over the latent space. Kingma and Welling (2013) circumvent this obstacle by proposing an auxiliary inference distribution  $q_\Phi(z|x)$  and derive a variational lower bound on the log likelihood

$$\begin{aligned} \mathcal{L}_{\text{ELBO}} &= \mathbb{E}_{q_\Phi(z|x)} [\log p_\Psi(x|z)] - \text{D}_{\text{KL}}(q_\Phi(z|x) || p(z)) \\ &\leq \log p(x) \end{aligned} \tag{1}$$

Maximizing this objective can be naturally interpreted as minimizing the reconstruction loss of a probabilistic autoencoder and regularizing the posterior distribution towards the prior. Kingma et al. (2014) extend this work to the semi-supervised setting considering both labeled and unlabeled data.

## 3 BAYESIAN OPTIMIZATION AND A SHAPED LATENT SPACE

We address the optimization challenges of high dimensions and discrete spaces in Bayesian Optimization by combining Gaussian process regression with variational autoencoding. In addition, we further shape the latent space through adversarial training. By learning an invariant latent representation regarding input-specific attributes we are able to transfer these attributes across inputs.

We consider the directed graphical model of inputs  $x$ , corresponding labels  $y$  and latent variables  $z$  shown in Figure 1. Given  $z$ , we assume  $x$  and  $y$  to be independent:

$$p(x, y|z) = p_{\Psi}(x|z) p_{\Theta}(y|z). \quad (2)$$

The data distribution  $p_{\Psi}(x|z)$  is modeled as either multinomial (protein dataset) or multivariate normal with fixed covariance (shape dataset). The discriminative  $p_{\Theta}(y|z)$  is modeled as standard normal  $\mathcal{N}(\mu_{\theta}(z), 1)$ . Both distributions are parametrized through neural networks with parameters  $\Psi$  and  $\Theta$ .

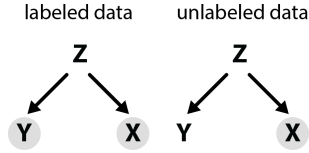


Figure 1: Graphical model connecting input space  $x$  with latent variable  $z$  and label  $y$ . The model assumes conditional independence of  $x$  and  $y$  given  $z$ . The gray shading marks observed quantities.

### 3.1 SEMI-SUPERVISED LEARNING

Given labeled and unlabeled data  $D_{\ell}$  and  $D_u$ , respectively, we aim to optimize the likelihoods  $p(x, y)$  and  $p(x)$ . As in the case of the VAE this is intractable to compute due to integration over  $z$  and we instead resort to variational lower bounds.

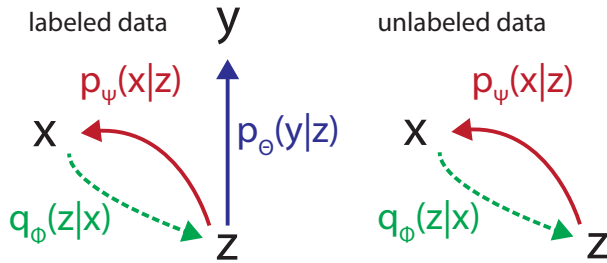


Figure 2: Illustration of the parametrized distributions involved in the derivation of the variational lower bounds.

Introducing the auxiliary model  $q_{\Phi}(z|x)$  and using Jensen’s inequality, we derive a variational lower bound on the log-likelihood of labeled data  $(x, y)$ . The independence of the proposal distribution regarding  $y$  reflects the view that  $x$  contains all information on  $y$ . An illustration

of the parametrized distributions is shown in Figure 2.

$$\begin{aligned} \log p(x, y) &= \log \int p(x, y|z) p(z) dz \\ &= \log \int \frac{p(x, y|z)}{q_{\Phi}(z|x)} q_{\Phi}(z|x) p(z) dz \\ &= \log \mathbb{E}_{z \sim q_{\Phi}(z|x)} \left[ p(x, y|z) \frac{p(z)}{q_{\Phi}(z|x)} \right] \\ &\geq \mathbb{E}_{z \sim q_{\Phi}(z|x)} [\log p(x, y|z)] \\ &\quad - D_{\text{KL}}(q_{\Phi}(z|x) || p(z)) \\ &= \mathbb{E}_{z \sim q_{\Phi}(z|x)} [\log p_{\Psi}(x|z) + \log p_{\Theta}(y|z)] \\ &\quad - D_{\text{KL}}(q_{\Phi}(z|x) || p(z)) \\ &\equiv \mathcal{L}_{\ell} \end{aligned} \quad (3)$$

where  $D_{\text{KL}}$  indicates Kullback-Leibler divergence.

In contrast, we find in the case of unlabeled data

$$\begin{aligned} \log p(x) &= \log \int \int p(x, y|z) p(z) dz dy \\ &= \log \int \int q_{\Theta}(y|z) \frac{p(x, y|z)}{q_{\Theta}(y|z)} \\ &\quad q_{\Phi}(z|x) \frac{p(z)}{q_{\Phi}(z|x)} dz dy \\ &\geq \mathbb{E}_{z \sim q_{\Phi}(z|x)} [\log p_{\Psi}(x|z)] \\ &\quad - D_{\text{KL}}(q_{\Phi}(z|x) || p(z)) \\ &\equiv \mathcal{L}_u \end{aligned} \quad (4)$$

where we recover the ELBO objective of the VAE. Modeling the proposal distribution  $q_{\Phi}(z|x)$  as multivariate Gaussian and assuming a Gaussian prior  $p(z) = \mathcal{N}(0, I)$  results in an analytic expression for the divergence term. Together  $\mathcal{L}_u$  and  $\mathcal{L}_{\ell}$  form a joint lower bound in the semi-supervised setting. During training we optimize a weighted sum of the two bounds. Specifically, labeled and unlabeled data share the same encoder  $q_{\Phi}(z|x)$  and decoder  $p_{\Psi}(x|z)$ .

### 3.2 DESIGN OPTIMIZATION ON THE LATENT SPACE

After the designs are embedded in a lower dimensional continuous latent space we can now use a GP to perform Bayesian optimization. The algorithm for the joined procedure of latent embedding and following optimization is outlined in pseudocode in Algorithm 1.

For each  $(x_i, y_i)$  pair from the labeled set, we can compute the mean value of  $q_{\Phi}(z|x_i)$  that we denote  $z_i$ . This effectively embeds the labeled inputs from  $D_{\ell}$  in the lower-dimensional latent space. We can then fit a GP on



the set  $D'_\ell := \{(z_i, y_i)\}_{i \leq N_\ell}$ . Depending on the dataset size this is either a full GP or a sparse approximation with inducing points (Titsias, 2009). Subsequently we perform iterative optimization by sampling points from the latent prior  $p(z)$  and maximizing expected improvement via gradient ascent. Next, we generate new designs corresponding to the latent points with largest EI value using the decoder, leveraging the generative capability of a VAE. Finally we evaluate the black-box function for these designs. The new data is appended to our dataset and we continue ad libitum. For simplicity, we do not retrain the VAE with each dataset expansion.

---

**Algorithm 1** VAE-guided Bayesian Optimization

---

**Input:** Unlabeled data  $D_u$ , labeled data  $D_\ell = \{(x_i, y_i)\}_{i \leq N_\ell}$ , fitness function  $f$ , parameters  $\alpha, \beta, \kappa$ .

$y_{\max} \leftarrow \max_{i \leq N_\ell} y_i$

**TrainVAE:**

minimize $_{\Theta, \Phi, \Psi} \alpha \mathcal{L}_u(D_u) + \beta \mathcal{L}_\ell(D_\ell)$   
 $D'_\ell \leftarrow \{(z_i, y_i)\}_{i \leq N_\ell}$  with  $z_i$  mean of  $q_\Phi(\cdot|x_i)$

**loop:**

GP  $\leftarrow$  **FitGP** ( $D'_\ell; \kappa$ )

**parallel loop:**

sample  $z_i^0 \sim \mathcal{N}(0, I)$

$(z_i^*, f_i^*) \leftarrow \max_z \text{EI}(z, z_0 = z_i^0, y_{\max}, GP)$

$\hat{z} \leftarrow z_b$  with  $b \leftarrow \arg \max_i f_i^*$

$\hat{x} \leftarrow \text{decoder}_\Phi(\hat{z})$  ▷ Create design

$\hat{y} \leftarrow f(\hat{x})$  ▷ Evaluate design  $\hat{x}$

Add  $(\hat{x}, \hat{y})$  to  $D_\ell$  and  $(\hat{z}, \hat{y})$  to  $D'_\ell$

$y_{\max} \leftarrow \max\{y_{\max}, \hat{y}\}$

**return**  $D_\ell$

---

### 3.3 ADJUSTING ATTRIBUTES AT TIME OF DECODING

We consider a setting in which we have successfully used our Bayesian optimization framework to find an enhanced design  $x$  which we generated by decoding its corresponding latent representation  $z$ . In addition, let  $a$  be a real-valued attribute intrinsic to a given input design  $x$  which is uncorrelated to the functional score  $y$  of the design. Taking the case of car designs as an example,  $y$  could be a measure of the car’s aerodynamic properties and  $a$  reflect the car’s color.

The joint probability distribution  $p(a, x, y, z)$  factorizes as

$$p(a, x, y, z) = p_\Psi(x|z, a) p_\gamma(a|z) p_\Theta(y|z) p(z) \quad (5)$$

with the additional inference network  $p_\gamma(a|z)$  (Figure 3). We assume that the input design  $x$  and the attribute  $a$  are observed variables, and that the label  $y$  is sometimes observed (i.e. the semi-supervised setting).

The question we consider is whether we can transfer an attribute across designs, i.e., can we decode our optimized latent representation to designs which share the optimal score but differ with respect to the value of attribute  $a$ . Referring again to our example of car designs, the attribute adjustment would consist in changing a car’s color after finding an aerodynamically optimal design.

Our strategy to enable attribute adjustment is to enforce a latent representation which is invariant to  $a$ . If the latent space contains no information on the attribute, the decoder  $p_\Psi(x|z, a)$  is forced to learn how to impose  $a$  on  $z$  in order to achieve proper design reconstruction. We can then adjust attributes by decoding optimized latent points with an attribute value of our choice.

To enforce this invariance, we add adversarial training to the training objective. We formalize this in the following maxmin expression:

$$\max_{\Phi} \min_{\gamma} \mathbb{E}_{x \sim q(x)} \mathbb{E}_{z \sim q_\Phi(z|x)} \left[ (a(x) - \hat{a}_\gamma(z))^2 \right] \quad (6)$$

Here  $a$  is the attribute we want to be invariant to, and  $\hat{a}_\gamma$  is an estimator of  $a$  given the latent set  $z$ . The notation  $a(x)$  emphasizes the fact that every design  $x$  has an intrinsic attribute value  $a$ . We model  $p_\gamma(a|z)$  as Gaussian with fixed covariance and predict the mean using a neural network with parameters  $\gamma$ . If we assume that  $p_\gamma$  is expressive enough and the network trained such that it can take advantage of all information  $z$  has on  $a$ , then the objective minimizes the mutual information  $I(a; z)$  and  $p_\gamma$  is forced to settle on mean prediction. Note that this adversarial training objective does not depend on the observation of the label  $y$  and can thus leverage both labeled and unlabeled data.

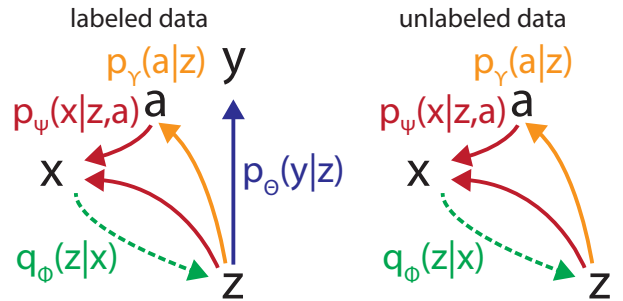


Figure 3: The augmented architecture with adversarial network  $p_\gamma(a|z)$  mapping from latent space  $z$  to attribute  $a$ . Providing  $a$  as additional input to the decoder encourages an  $a$ -invariant latent representation.

## 4 DATASETS

We empirically evaluate the performance of our method on two datasets.

### 4.1 PROTEIN-FITNESS LANDSCAPE

Proteins are of paramount importance for biological systems and in industrial applications such as food processing or biomass conversion. As such the ability to design enhanced proteins is desirable. Proteins form both a high-dimensional as well as discrete design space as a protein is defined by an amino-acid sequence with alphabet size 20.

Protein optimization is especially challenging as (i) the number of target amino-acid sequences grows exponentially with the number of considered amino-acid mutations and (ii) only a very small fraction of all amino-acid sequences results in a functional protein (Keefe and Szostak, 2001).

We base our protein-optimization approach on a large fitness-landscape exploration study of the green fluorescent protein from *Aequorea victoria* (avGFP) (Sarkisyan et al., 2016). GFP is a widely used label-protein in fluorescence microscopy with a sequence of 237 amino acids. Our specific dataset consists of 51,715 different protein sequences  $D_\ell$  generated by random mutagenesis from the avGFP sequence and associated fluorescence values  $y$  as measured by fluorescence-activated cell sorting. On average each protein sequence contains 3.7 mutations compared to avGFP.

Amino acid sequences of the avGFP variants are encoded in a one-hot-style manner through a matrix of size  $20 \times 237$  – accounting for the 20 essential amino acids and the sequence length of avGFP. All entries of the columns are 0 except for one 1 encoding the amino acid at the specific sequence position.

### 4.2 DRAG IN LAMINAR FLUID FLOW

The second dataset consists of 5100 two-dimensional shapes  $x$  and scalar drag coefficients  $y$  associated with the resistance these shapes experience in a constant fluid flow. We consider the case of laminar flow around an object in two dimensions as it allows us to generate training and test data, and perform Bayesian optimization at relatively low computational cost.

We generate the dataset by numerically solving the Navier-Stokes equations (Lifshitz and Landau, 1959) which provide a theoretical description of fluid flow around objects. Figure 4 shows example simulations from the dataset generation. Generated shapes are resized to  $42 \times 56$  pixels

to reduce memory requirements. Further details on the hydrodynamics simulation can be found in the appendix.

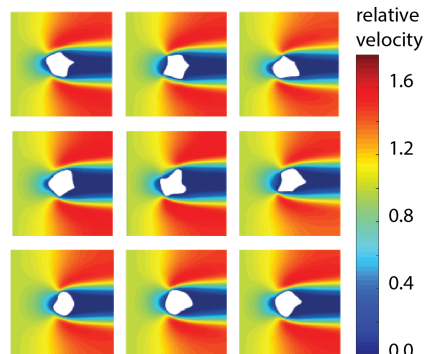


Figure 4: Finite-element simulations of fluid flow around random shapes in two dimensions. The left wall defines the fluid inlet.

## 5 EXPERIMENTS

In a first set of experiments we investigate the effect label-guided encoding and training with additional unlabeled data have on inference and autoencoder reconstruction error. The second part is concerned with design optimization of proteins and shapes. Finally, we demonstrate how we can use invariant learning to adjust shape attributes, namely area, with little effect on the drag coefficient.

### 5.1 INFERENCE AND RECONSTRUCTION ERROR

We consider the effect of label-guided encoding and adding unlabeled data to the training on 1) inference of  $y$  and 2) autoencoder reconstruction error as defined through the first term in the ELBO objective for the test set.

Tables 1 and 2 summarize the results for the protein and shape dataset, respectively. Details on the model architectures can be found in the appendix.

In general, we notice the positive effect label-guided encoding has on test set prediction. Column ‘NN’ shows the relative prediction error when using  $p_\Theta(y|z)$  for inference. In absence of unlabeled data ( $N_U = 0$ ) we consider two settings whose corresponding values are separated by a backslash: i) training  $p_\Theta(y|z)$  and the autoencoder jointly (left value) and ii) training autoencoder and discriminative network sequentially (right value). We observe that the discriminative network guides the dimensionality reduction such that label-relevant information

Table 1: Protein dataset: Normalized test set prediction errors for different allocations of labeled ( $N_L$ ) and unlabeled ( $N_U$ ) training data. NN, GP-NN and GP-LAT describe the neural network parametrized by  $\Theta$  and GPs trained on the neural-network features and the latent space, respectively. REC describes relative reconstruction error for test set designs.

$N_L / N_U$	NN	GP-NN	GP-LAT	REC
30k/10k	1.04	2.43	1.82	<b>1.00</b>
30k/0	<b>1.00</b> /1.48	2.45	1.79	1.03
15k/15k	1.22	2.49	1.93	1.14
15k/0	1.31/1.57	2.58	2.18	1.64
5k/5k	1.18	2.62	1.89	1.90
5k/0	1.44/1.70	2.58	1.97	3.11

Table 2: Hydrodynamic dataset: Normalized test set prediction errors for different allocations of labeled ( $N_L$ ) and unlabeled ( $N_U$ ) training data. NN, GP-NN and GP-LAT describe the neural network parametrized by  $\Theta$  and GPs trained on the neural-network features and the latent space, respectively. REC describes relative reconstruction error for test set designs.

$N_L / N_U$	NN	GP-NN	GP-LAT	REC
2500/2000	<b>1.00</b>	1.41	1.02	<b>1.00</b>
2500/0	1.21/1.38	1.44	1.22	1.04
1500/2000	1.10	1.31	1.24	1.07
1500/0	1.19/1.41	1.19	1.07	1.17
500/2000	1.91	1.64	1.79	1.31
500/0	2.05/1.98	1.61	2.02	1.82

in the amino-acid sequence or shape is encoded with enhanced accuracy.

In addition, we also observe a positive effect of adding unlabeled data with respect to the reconstruction error (REC). The effect is more pronounced when less labeled data is available. Incorporating unlabeled data is also beneficial for NN prediction error in almost all cases.

In order to obtain uncertainty measures for the predictions we train and compare two GP models with squared-exponential kernel. Model 1 is trained on the latent-space embedding of the training data (GP-LAT). Model 2 is trained on the features learned by the discriminative neural network  $p_{\Theta}(y|z)$  (GP-NN). Both models consider the situation in which  $p_{\Theta}(y|z)$  and the autoencoder have previously been trained jointly.

We account for dimension-specific length scales in the covariance function such that the Gaussian process can filter irrelevant dimensions. For the protein dataset we use a sparse approximation with 500 inducing points such

that the prediction performance of both GP models is impaired compared to the neural network (NN columns in the Tables). Comparing the GPs among each other we note the in general much better performance of the GP trained on the latent-space coordinates.

## 5.2 OPTIMIZATION OF PROTEIN AND SHAPE DESIGNS

We follow the algorithm outlined in Algorithm 1 to optimize shape and protein designs.

### 5.2.1 Design of New Protein Variants

A schematic of the optimization framework is shown in Figure 5A. To find the best local EI maxima we independently sample 20,000 start points from the prior  $p(z) \sim \mathcal{N}(0, I)$  and perform gradient ascent for each point. Figure 5B visualizes amino-acid mutation sites apparent in the highest-ranked protein variants on the structure of avGFP.

While only experimental verification can provide a precise assessment of the model performance, comparison with the data from literature on development of GFP variants shows that genotypes predicted by our model are free from known deleterious mutations such as mutations in the chromophore-forming amino acids and catalytically active E222 residue (Chudakov et al., 2010). Most of the mutated amino acid side chains in predicted genotypes are oriented towards the solvent in the protein beta barrel structure, in accordance with experimental observations (Sarkisyan et al., 2016) and with the evolutionary conservation of internally oriented residues (Chudakov et al., 2010). Moreover, some predicted genotypes carry combinations of substitutions known to increase brightness of avGFP, such as the F99S/M153T pair of substitutions that in combination with mutation V163A was reported to result in avGFP being 42 times brighter when expressed *in vivo* (Battistutta et al., 2000).

### 5.2.2 Design of Drag-reduced Shapes

The time and resources required for protein synthesis and functional testing render it expensive to use this application for several rounds of Bayesian optimization given the purpose of this paper and to explore technical model aspects in more detail. For this reason we use a set of two-dimensional shapes and the drag these shapes face under laminar flow conditions. We can calculate the drag forces based on the Navier-Stokes equations in a physics simulation. As such we can perform closed-loop optimization with the goal of finding drag-reduced shapes.

Figure 6 shows a schematic of the optimization procedure which is analogous to the protein case. We generate

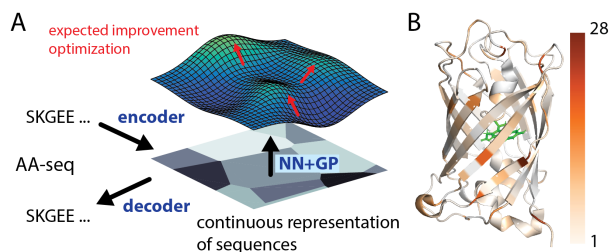


Figure 5: (A) Schematic illustrating the optimization of amino-acid sequences to generate variants of the green fluorescent protein (GFP) with enhanced brightness. (B) The structure of GFP annotated with the distribution of mutations among 100 sequences suggested by the design algorithm. The main chromophore complex is shown in green.

new object shapes by optimizing EI with respect to the smallest drag coefficient in our training set. Promising latent points are decoded to shape images and their drag coefficients evaluated in our hydrodynamics simulator.

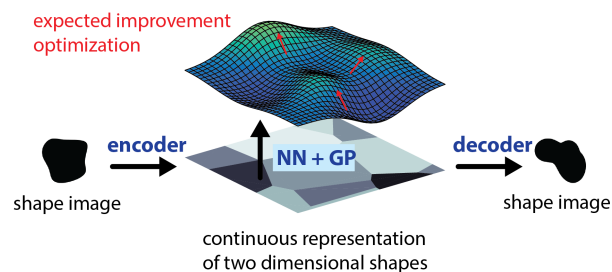


Figure 6: Schematic of the Bayesian optimization procedure to design drag-reduced shapes.

To illustrate the benefits of the joint autoencoder-GP framework we compare three different strategies for Bayesian optimization. In the first setting, we sample random points from the Gaussian prior distribution  $p(z)$  and for each point optimize EI via gradient ascent. We contrast this with optimization working directly at the level of shapes. Inputs are random shapes generated in the same way as our training dataset. We consider training the GP directly on the input space ( $GP(x)$ ) as well as including the intermediary mapping of the encoder ( $GP(z(x))$ ). To be able to optimize EI via gradient ascent we relax the assumption of binary pixel values to continuous values in  $[0, 1]$ .

Figure 7 shows the best drag coefficient generated as a function of shape evaluations for the three strategies. In

all cases we sample 600 starting points for gradient ascent and evaluate the resulting 100 shapes corresponding to the largest EI values in our simulator. All models share the same GP kernel function (squared-exponential), optimization parameters and stopping criteria.

Optimizing the acquisition function over the latent space consistently yields the largest reduction in drag coefficient for all rounds of Bayesian optimization. Optimizing the acquisition function over shapes does not improve upon directly simulating the drag coefficient for the random shapes which are chosen as gradient-ascent start points. The GP kernel is unable to extract drag-relevant features from the pixel input. As a consequence of the high-dimensionality of the pixel space, mean predictions for unknown shapes are close to the Gaussian prior and the EI gradient vanishes. We can recover part of the performance by using the ‘deep kernel’ of the encoder while still optimizing directly on shape images. We reason that the remaining performance gap is due to the relaxation of continuous pixel values.

Another advantage of the latent-space optimization consists in the fact that we can generate gradient-ascent starting points by simply sampling from  $z \sim \mathcal{N}(0, I)$ . In contrast, optimization on the input space requires us to have access to new valid structures, i.e. shapes, as naive sampling in the  $[0, 1]^{42 \times 56}$  pixel space breaks the optimization routine as expected.

Figure 8 shows examples of drag-reduced shapes over the course of 500 calls to the hydrodynamics simulator during optimization.

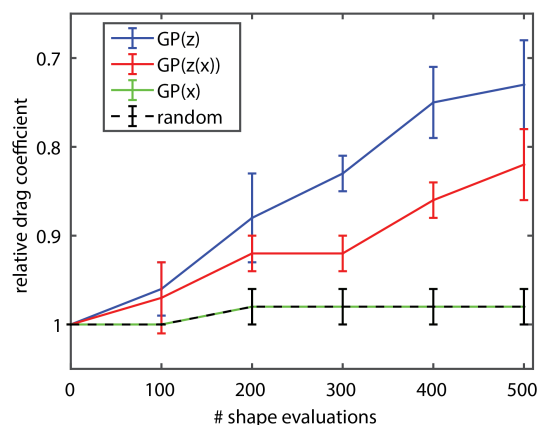


Figure 7: Best relative drag coefficient as a function of number of shape evaluations. The plot compares optimization based on random shapes (x) versus latent points (z). Error bars indicate standard deviation from three independent experiments.

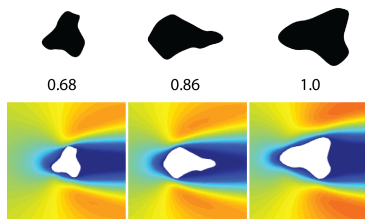


Figure 8: The Bayesian optimization routine produces shapes of reduced drag ( $< 1$ ) compared to the smallest drag coefficient in the training set ( $= 1$ ). Shape inputs to the hydrodynamic simulator (top) and corresponding flow fields (bottom).

### 5.3 ATTRIBUTE ADJUSTMENT ACROSS SHAPES

Given a set of shape designs with a real-valued attribute  $a$  our goal is for the decoder to learn how to generate a design with a given attribute value based on a latent coordinate  $z$ .

We consider two scalar attributes for our shape dataset: color and surface area. To introduce ‘color’ we create a separate channel for each shape which contains the binary mask multiplied by a random number from  $[0, 1]$ . During training we provide the true attribute value for each shape to the adversarial network and decoder. To make the learning more stable we slowly fade in the adversarial loss over half the number of maximal training epochs. Early stopping is only considered after this point.

At test time we take a given latent point and feed it along with a desired attribute value from the aforementioned range to the decoder. Figure 9 shows five example shapes decoded each with four different attribute values  $[0.2, 0.4, 0.6, 0.8]$  resulting in a specific black-white intensity.

Adjusting color is a relatively easy task as the additionally introduced color channel is entirely orthogonal to the drag coefficient - the quantity our discriminative model  $p_{\Theta}(y|z)$  promotes to be accurately encoded in the latent space. For this reason we consider the area of a shape as another more challenging attribute value to control for. Invariance to shape size requires the architecture to apply a non-trivial geometric transformation or to learn and store the shape information in the latent space in a more abstract way, e.g. through scale-invariant Fourier descriptors.

Figure 10 shows a selection of five latent points and their associated drag coefficients decoded with four different area values in analogy to the color-adjustment example.

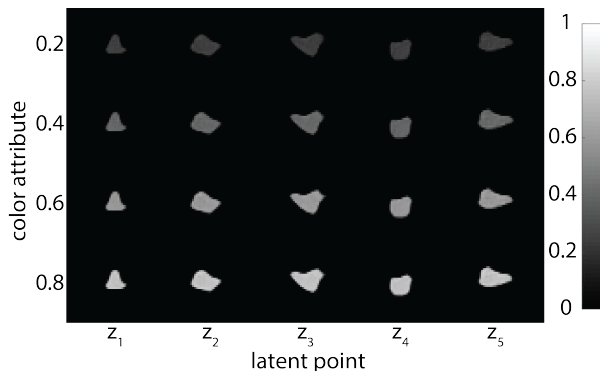


Figure 9: Shapes generated from five exemplary latent points  $z_1 \dots z_5$  and four different color-attribute values each.

The shapes are not cherry-picked. It is difficult to name an exact drag coefficient for the very small shapes due to the necessary shape rescaling, smoothing and boundary point extraction before any finite-element simulation can be performed. The simulations indicate that the drag coefficient of the scaled shapes stay within 25% of their original values with the largest deviations occurring at the smallest scale. At the same time area values change consistently for all shapes by about 300% (compare Table 3). The consistency of the scaling is remarkable as less than 4.5% of training shapes have an area smaller or larger than 67 and 175 pixels, respectively.

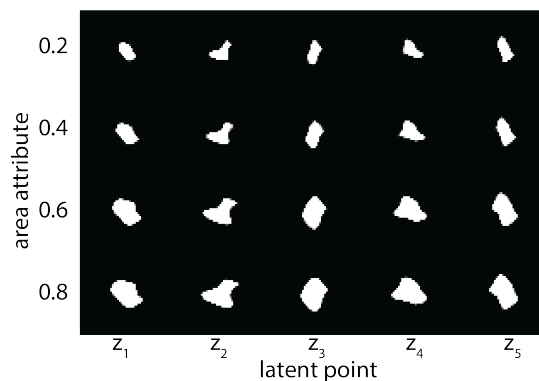


Figure 10: Shapes generated from five exemplary latent points  $z_1 \dots z_5$  and four different area-attribute values each.

## 6 RELATED WORK

Our approach draws on a broad basis of prior work and is generally related to conditional VAEs (Kingma et al., 2014) and hybrid models bridging generative and discrim-

Table 3: Pixel areas corresponding to shapes generated from latent points  $z_1 \dots z_5$  and four different area-attribute values each (compare Figure 10)

$z_1$	$z_2$	$z_3$	$z_4$	$z_5$
60	67	64	64	64
91	91	90	89	89
149	143	149	152	152
174	175	176	177	177

inative architectures (Maaløe et al., 2016; Shu et al., 2016; Kuleshov and Ermon, 2017). In contrast to the conditional VAE, we choose a label-independent encoder distribution stressing the perspective on the latent space as continuous embedding of inputs  $\mathcal{X}$ .

Using neural-network learned features as input to a GP model in the context of autoencoders dates back to Bengio et al. (2007). More recently, Snoek et al. (2012a) proposed non-parametric autoencoder guidance. Wilson et al. (2016a) present deep-kernel learning in which a stacked architecture of neural network and GP is trained jointly. Wilson et al. (2016b) and Huang et al. (2015) target the scalability of these hybrid GP models. Successful optimization within our framework depends on the ability to encode high-dimensional designs in a continuous latent space of lower dimensionality. This assumption is similar to the notion of low effective dimensionality in Wang et al. (2013) and related to Garnett et al. (2014). We considered joint training of GP and autoencoder in the case of the hydrodynamic dataset (5.2.2) but did not find this to outperform the sequential setting in which we train the GP on the latent space after parametric label guidance.

The idea of attribute adjustment draws on learning of fair representations, notably the fair VAE (Louizos et al., 2015). Purushotham et al. (2016) and recently Lample et al. (2017) explore enforcing invariance for adaptation in time-series data and natural images. We propose an adversarial objective based on mean-squared error maximization and demonstrate that attribute adjustment is feasible concurrent with Bayesian optimization.

GPs have been used for protein design and shape optimization - the two domains considered in our datasets. Romero et al. (2013) demonstrated the application of GPs to navigate the fitness landscape of proteins given limited data. The GP model is directly trained on amino acid sequences through a kernel function which relates sequence similarity to the spatial distance of amino acid positions in the folded protein structure. Previous works demonstrating the successful application of GP regression for the optimization of parametric designs in aerodynamic applications include (Simpson et al., 2001; Martin and

Simpson, 2005; Jeong et al., 2005; Jouhaud et al., 2007). The aforementioned publications leverage domain knowledge for optimization. In contrast, the approach presented here is solely data-driven and based on deep-generative models.

## 7 DISCUSSION

Bayesian optimization is a data-efficient approach to optimize complex black-box functions without the need to supply gradients. Nevertheless discrete, high-dimensional input spaces pose a challenge to successful design optimization.

In this work we explore a framework combining variational autoencoding and Gaussian process regression to approach this challenge. We present a variational bound for the underlying graphical model and show how label-guidance enhances the predictive performance and incorporating unlabeled data leads to an increase in reconstruction accuracy.

We apply the optimization framework to the design of enhanced functional proteins. One round of Bayesian optimization proposes meaningful new protein variants which are free of known deleterious mutations. We further introduce a physics-based dataset of two-dimensional shapes and associated drag coefficients in laminar flow. Having access to a simulator allows us to perform closed-loop Bayesian optimization, such that after five rounds we improve by about 30% on the best value in the training set. We demonstrate that optimization based in the latent space outperforms optimization in the design space.

Finally, we consider an adversarial extension to our model. By forcing the latent space to be invariant w.r.t. an attribute value of choice, we are able to select this attribute value when decoding a latent point and impose it on our design. The combination of label-guidance and attribute-adversarial training shapes the information encoded in the latent space. We envision that the adversarial-model extension might be a fruitful approach to transfer functional groups or domains across molecules. In addition, the performance of discriminative models trained on the latent space could benefit when uninformative factors of variation are removed from the latent code based on domain knowledge.

### Acknowledgements

The authors thank Jonathan Kuck, Neal Jean and Aditya Grover for fruitful discussions. This research was supported by TRI, NSF (#1651565, #1522054, #1733686) and a Hellman Faculty Fellowship.

## References

- R. Battistutta, A. Negro, and G. Zanotti. Crystal structure and refolding properties of the mutant F99s/M153t/V163a of the green fluorescent protein. *Proteins: Structure, Function, and Bioinformatics*, 41(4):429–437, 2000.
- Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In *Advances in neural information processing systems*, pages 153–160, 2007.
- D. M. Chudakov, M. V. Matz, S. Lukyanov, and K. A. Lukyanov. Fluorescent proteins and their applications in imaging living cells and tissues. *Physiological reviews*, 90(3):1103–1163, 2010.
- J. Damborsky and J. Brezovsky. Computational tools for designing and engineering enzymes. *Current opinion in chemical biology*, 19:8–16, 2014.
- R. Garnett, M. A. Osborne, and P. Hennig. Active learning of linear embeddings for Gaussian processes. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pages 230–239. AUAI Press, 2014. ISBN 0-9749039-1-4.
- R. Gómez-Bombarelli, J. Aguilera-Iparraguirre, T. D. Hirzel, D. Duvenaud, D. Maclaurin, M. A. Blood-Forsythe, H. S. Chae, M. Einzinger, D.-G. Ha, and T. Wu. Design of efficient molecular organic light-emitting diodes by a high-throughput virtual screening and experimental approach. *Nature materials*, 15(10):1120, 2016a.
- R. Gómez-Bombarelli, D. Duvenaud, J. M. Hernández-Lobato, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *arXiv preprint arXiv:1610.02415*, 2016b.
- A. Grover, T. Markov, P. Attia, N. Jin, N. Perkins, B. Cheong, M. Chen, Z. Yang, S. Harris, W. Chueh, and S. Ermon. Best arm identification in multi-armed bandits with delayed feedback. In A. Storkey and F. Perez-Cruz, editors, *Proceedings of Machine Learning Research*, volume 84, pages 833–842, Proceedings of Machine Learning Research, 2018. PMLR.
- W. B. Huang, D. Zhao, F. Sun, H. Liu, and E. Y. Chang. Scalable Gaussian process regression using deep neural networks. In *IJCAI*, pages 3576–3582, 2015.
- S. Jeong, M. Murayama, and K. Yamamoto. Efficient optimization design method using kriging model. *Journal of aircraft*, 42(2):413–420, 2005.
- D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.
- J.-C. Jouhaud, P. Sagaut, M. Montagnac, and J. Laurenceau. A surrogate-model based multidisciplinary shape optimization method with application to a 2d subsonic airfoil. *Computers & Fluids*, 36(3):520–529, 2007.
- A. D. Keefe and J. W. Szostak. Functional proteins from a random-sequence library. *Nature*, 410(6829):715, 2001.
- D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014.
- V. Kuleshov and S. Ermon. Deep hybrid models: Bridging discriminative and generative approaches. UAI, 2017.
- G. Lample, N. Zeghidour, N. Usunier, A. Bordes, and L. Denoyer. Fader networks: Manipulating images by sliding attributes. In *Advances in Neural Information Processing Systems*, pages 5967–5976, 2017.
- E. M. Lifshitz and L. D. Landau. *Course of theoretical physics, volume 6, fluid mechanics*. Pergamon Press, Oxford UK, 1959.
- C. Louizos, K. Swersky, Y. Li, M. Welling, and R. Zemel. The variational fair autoencoder. *arXiv preprint arXiv:1511.00830*, 2015.
- L. Maaløe, C. K. Sønderby, S. K. Sønderby, and O. Winther. Auxiliary deep generative models. *arXiv preprint arXiv:1602.05473*, 2016.
- J. D. Martin and T. W. Simpson. Use of kriging models to approximate deterministic computer models. *AIAA journal*, 43(4):853–863, 2005.
- D. G. Matthews, G. Alexander, M. Van Der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. León-Villagrà, Z. Ghahramani, and J. Hensman. GPflow: A Gaussian process library using TensorFlow. *The Journal of Machine Learning Research*, 18(1):1299–1304, 2017.
- M. McIntire, D. Ratner, and S. Ermon. Sparse Gaussian processes for Bayesian optimization. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pages 517–526, Jersey City, New Jersey, USA, 2016. AUAI Press. ISBN 978-0-9966431-1-5.
- S. Purushotham, W. Carvalho, T. Nilanon, and Y. Liu. Variational recurrent adversarial deep domain adaptation. *openreview.net*, 2016.
- C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, Cambridge, Mass, 2006.

- P. A. Romero, A. Krause, and F. H. Arnold. Navigating the protein fitness landscape with Gaussian processes. *Proceedings of the National Academy of Sciences*, 110(3):E193–E201, 2013.
- K. S. Sarkisyan, D. A. Bolotin, M. V. Meer, D. R. Usmanova, A. S. Mishin, G. V. Sharonov, D. N. Ivankov, N. G. Bozhanova, M. S. Baranov, and O. Soylemez. Local fitness landscape of the green fluorescent protein. *Nature*, 533(7603):397, 2016.
- B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.
- R. Shu, H. H. Bui, and M. Ghavamzadeh. Bottleneck conditional density estimation. *arXiv preprint arXiv:1611.08568*, 2016.
- T. W. Simpson, T. M. Mauery, J. J. Korte, and F. Mistree. Kriging models for global approximation in simulation-based multidisciplinary design optimization. *AIAA journal*, 39(12):2233–2241, 2001.
- J. Snoek, R. P. Adams, and H. Larochelle. Nonparametric guidance of autoencoder representations using label information. *Journal of Machine Learning Research*, 13(Sep):2567–2588, 2012a.
- J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012b.
- M. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial Intelligence and Statistics*, pages 567–574, 2009.
- Z. Wang, M. Zoghi, F. Hutter, D. Matheson, and N. De Freitas. Bayesian optimization in high dimensions via random embeddings. In *IJCAI*, pages 1778–1784, 2013.
- A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing. Deep kernel learning. In *Artificial Intelligence and Statistics*, pages 370–378, 2016a.
- A. G. Wilson, Z. Hu, R. R. Salakhutdinov, and E. P. Xing. Stochastic variational deep kernel learning. In *Advances in Neural Information Processing Systems*, pages 2586–2594, 2016b.



---

# Join Graph Decomposition Bounds for Influence Diagrams

---

Junkyul Lee    Alexander Ihler    Rina Dechter

Donald Bren School of Information and Computer Sciences

University of California, Irvine

{junkyul, ihler, dechter} @ics.uci.edu

## Abstract

We introduce a new decomposition method for bounding the maximum expected utility of influence diagrams. While most current schemes use reductions to the Marginal Map task over a Bayesian Network, our approach is direct, aiming to avoid the large explosion in the model size that often results by such reductions. In this paper, we extend to influence diagrams the principles of decomposition methods that were applied earlier to probabilistic inference, utilizing an algebraic framework called valuation algebra which effectively captures both multiplicative and additive local structures present in influence diagrams. Empirical evaluation on four benchmarks demonstrates the effectiveness of our approach compared to reduction-based approaches and illustrates significant improvements in the upper bounds on maximum expected utility.

## 1 INTRODUCTION

An influence diagram (ID) [Howard and Matheson, 2005] is a graphical model for sequential decision-making under uncertainty that compactly captures the local structure of the conditional independence of probability functions and the additivity of utility functions. Its structure is captured by a directed acyclic graph (DAG) over nodes representing the variables (decision and chance variables). The standard query on an ID is finding the maximum expected utility (MEU) and an optimal policy, at each decision subject to the history of observations and decisions. Our focus is on computing an upper bound on the MEU in a single agent sequential decision-making scenario when we assume perfect recall. The computation of upper bounds is desirable not only because exact computation is exponential in the number of variables appearing

in the history, but also because it can be used as a building block of algorithmic frameworks such as heuristic search and sampling.

**Earlier work.** One early work that yields bounds on many inference tasks in an anytime manner is the minbucket elimination (MBE) scheme that provides upper and lower bounds of graphical model queries by enforcing problem decomposition during the variable elimination process [Dechter and Rish, 2003]. In particular, Dechter [2000] presented an MBE algorithm for influence diagrams. A different principle for generating bounds on the MEU is to relax the constraints imposed on the information available at each stage and for each decision (thus making more observations visible to each decision). This *information relaxation scheme* relaxes the constraints imposed on the information available at each stage and it also permits variable reordering during processing [Nilsson and Hohle, 2001]. In particular, Yuan et al. [2010] presented an AND/OR depth-first branch and bound search algorithm guided by upper bounds generated by such flexible variable orderings.

An alternative set of schemes exploit translations between the maximum a posteriori inference (MMAP) in a Bayesian network (BN) and the MEU inference in an ID [Mauá, 2016]. The idea is to compute the upper bound of the MMAP of the BN translated from an input ID. However, the number of auxiliary variables introduced by the translation is exponential in the size of the history under the perfect recall assumption. If all utility functions were multiplicative, an ID could be treated as an unnormalized distribution and MMAP inference would be applied directly. Liu and Ihler [2012] presented a variational formulation for computing the MEU and message passing algorithms for such IDs where the additive utilities are converted into multiplicative utilities by introducing a latent selector variable. However, such a translation can make it difficult to exploit decompositions present in the additive utility functions.

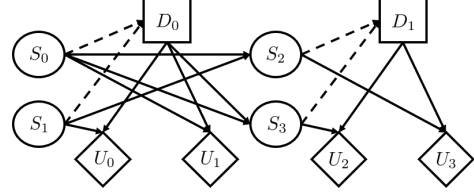
**Contributions.** We develop a decomposition scheme applied directly to IDs. It extends the decomposition scheme for MMAP [Ping et al., 2015] to IDs to accommodate multiplicative and additive terms. In particular, the upper bound is generated by relaxing the sequential structure of an ID to locally coupled decision subproblems. Subsequently, we present a message passing algorithm derived from the optimization framework that tightens the upper bound over various parameters including the reparameterization of both probability and utility functions. In summary:

- We present a new graphical model decomposition bound specialized for IDs called join graph decomposition bounds for IDs (JGDID), and a message passing algorithm that optimizes the bound.
- The proposed algorithm is compared empirically with current schemes on four benchmarks, showing a significant improvement in the quality of the upper bounds.

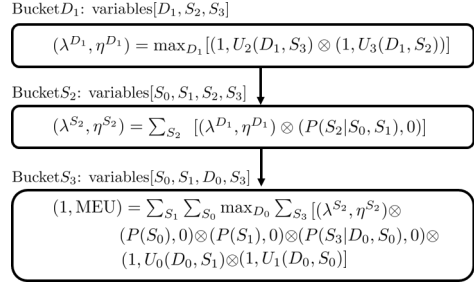
## 2 PRELIMINARIES

### 2.1 INFLUENCE DIAGRAMS

An ID is a tuple  $\mathcal{M} := \langle \mathbf{C}, \mathbf{D}, \mathbf{P}, \mathbf{U}, \mathcal{O} \rangle$  of a set of discrete random variables  $\mathbf{C} = \{C_i | i \in \mathcal{I}_{\mathbf{C}}\}$ , a set of discrete decision variables  $\mathbf{D} = \{D_i | i \in \mathcal{I}_{\mathbf{D}}\}$ , a set of conditional probability functions  $\mathbf{P} = \{P_i | P_i \in \mathcal{I}_{\mathbf{P}}\}$ , and a set of real-valued additive utility functions  $\mathbf{U} = \{U_i | U_i \in \mathcal{I}_{\mathbf{U}}\}$ . We use  $\mathcal{I}_{\mathbf{S}} = \{0, 1, \dots, |\mathbf{S}| - 1\}$  to denote the set of indices of each element in a set  $\mathbf{S}$ , where  $|\mathbf{S}|$  is the cardinality of  $\mathbf{S}$ . As illustrated in Figure 1(a), an ID can be associated with a DAG of three types of nodes: the chance nodes  $\mathbf{C}$  drawn as circles, the decision nodes  $\mathbf{D}$  drawn as squares, and the value nodes  $\mathbf{U}$  drawn as diamonds. There are also three types of directed edges: edges directed into a chance node  $C_i$  from its parents  $pa(C_i) \subset \mathbf{C} \cup \mathbf{D}$  representing the conditional probability function  $P_i(C_i | pa(C_i))$ , edges directed into a value node  $U_i$  denoting the utility function  $U_i(\mathbf{X}_i)$  from its scope  $\mathbf{X}_i \subset \mathbf{C} \cup \mathbf{D}$ , and informational arcs (dashed arrows in Figure 1(a)) directed from chance nodes to a decision node. The set of parent nodes associated with a decision node  $D_i$  is called the information set  $I_i$ , and denotes chance nodes that are assumed to be observed immediately before making decision  $D_i$ . The constrained variable ordering  $\mathcal{O}$  obeys a partial ordering which alternates between information sets and decision variables  $\{\mathbf{I}_0 < D_0 < \dots < \mathbf{I}_{|\mathbf{D}|-1} < D_{|\mathbf{D}|-1} < \mathbf{I}_{|\mathbf{D}|}\}$ . The partial elimination ordering should ensure the regularity of the ID (a decision can only be preceded by at most one decision), and dictates the available information at each decision  $D_i$  so that the *non-forgetting agent* makes decisions in multi-staged manner based on the history available at each stage



(a) Factored MDP as Influence Diagram



(b) Computing Maximum Expected Utility

Figure 1: Influence Diagram Example. A 2-step factored MDP is represented by an influence diagram, and the lower figure shows a schematic trace of the variable elimination with the valuation algebra.

$i$ ,  $H(D_i) := \cup_{k=0}^i \{D_k\} \cup \cup_{k=0}^i \mathbf{I}_k$ . The standard task of solving Influence Diagrams is computing the maximum expected utility  $E[\sum_{U_i \in \mathbf{U}} U_i | \Delta]$  and finding a set of optimal policies  $\Delta = \{\Delta_i | \Delta_i : R(D_i) \mapsto D_i, \forall D_i \in \mathbf{D}\}$ , where  $\Delta_i$  is a deterministic decision rule for  $D_i$  and  $R(D_i) \subseteq H(D_i)$  is the subset of history called  *requisite information* to  $D_i$ , namely, the only relevant history for making a decision [Nielsen and Jensen, 1999].

### 2.2 COMPUTING EXPECTED UTILITY

Unlike probabilistic graphical models, Influence Diagrams hold two type of functions combined by different operators: a product of probability functions, and a summation of utility functions. Jensen et al. [1994] presented a variable elimination algorithm that avoids the complication of dealing with two types of functions by generalizing the combination and marginalization operators such that operators act on a pair of probability and utility functions called a potential. The *valuation algebra* is an algebraic framework for computing the expected utility values, or values for short, based on the combination and marginalization on potentials [Mauá et al., 2012]. Here, we briefly summarize the essence of valuation algebra since it is what we use for developing the decomposition scheme. Let a valuation  $\Psi(\mathbf{X})$  be a pair of probability and value functions  $(P(\mathbf{X}), V(\mathbf{X}))$  over a set of variables  $\mathbf{X}$  called its scope. Occasionally, we will abuse the notation by dropping the scope from a function, e.g., writing  $P_1(\mathbf{X}_1)$  as  $P_1$ . The combination and marginalization operators

are defined as follows.

**Definition 1. (combination of two valuations)**

Given two valuations  $\Psi_1(\mathbf{X}_1) := (P_1(\mathbf{X}_1), V_1(\mathbf{X}_1))$  and  $\Psi_2(\mathbf{X}_2) := (P_2(\mathbf{X}_2), V_2(\mathbf{X}_2))$ , the combination of the two valuations over  $\mathbf{X}_1 \cup \mathbf{X}_2$  is defined by

$$\Psi_1(\mathbf{X}_1) \otimes \Psi_2(\mathbf{X}_2) := (P_1 P_2, P_1 V_2 + P_2 V_1).$$

Following Definition 1, the identity is  $(1, 0)$  and the inverse of  $(P(\mathbf{X}), V(\mathbf{X}))$  is  $(1/P(\mathbf{X}), -V(\mathbf{X})/P^2(\mathbf{X}))$ .

**Definition 2. (marginalization of a valuation)** Given a valuation  $\Psi(\mathbf{X}) := (P(\mathbf{X}), V(\mathbf{X}))$ , marginalizing over  $\mathbf{Y} \subseteq \mathbf{X}$  by summation or maximization are defined by

$$\sum_{\mathbf{Y}} \Psi(\mathbf{X}) := \left( \sum_{\mathbf{Y}} P(\mathbf{X}), \sum_{\mathbf{Y}} V(\mathbf{X}) \right),$$

$$\max_{\mathbf{Y}} \Psi(\mathbf{X}) := \left( \max_{\mathbf{Y}} P(\mathbf{X}), \max_{\mathbf{Y}} V(\mathbf{X}) \right).$$

An ID  $\mathcal{M}$  can be compactly represented by the valuation algebra as  $\mathcal{M}' := \langle \mathbf{X}, \Psi, \mathcal{O} \rangle$ , where  $\mathbf{X} = \mathbf{C} \cup \mathbf{D}$  and  $\Psi = \{(P_i, 0) | P_i \in \mathbf{P}\} \cup \{(1, U_i) | U_i \in \mathbf{U}\}$ . The MEU can be written as

$$\sum_{\mathbf{I}_0} \max_{D_0} \cdots \sum_{\mathbf{I}_{|\mathbf{D}|-1}} \max_{D_{|\mathbf{D}|-1}} \sum_{\mathbf{I}_{|\mathbf{D}|}} \bigotimes_{\alpha \in \mathcal{I}_\Psi} \Psi_\alpha(\mathbf{X}_\alpha), \quad (1)$$

where  $\mathbf{X}_\alpha$  denotes the scope of  $\Psi_\alpha$ .

The following example illustrates the variable elimination algorithm with the valuation algebra.

**Example 1.** Figure 1(b) shows a schematic trace of the variable elimination algorithm [Dechter, 1999] using the valuation algebra. We use  $\mathcal{O} : \{D_1, S_2, S_3, D_0, S_0, S_1\}$  as a legal elimination ordering. The first eliminated variable is  $D_1$ , so the variable elimination algorithm collects all valuations whose scope includes  $D_1$  in Bucket  $D_1$ . Then it generates the outgoing message  $(\lambda^{D_1}, \eta^{D_1})$  and sends it to Bucket  $S_2$ . Bucket  $S_2$  combines the preallocated valuations and the incoming message, and generates the outgoing message  $(\lambda^{S_2}, \eta^{S_2})$ . This elimination process continues until we obtain the MEU. We refer to Mauá et al. [2012] for more details.

### 2.3 JOIN GRAPH DECOMPOSITION

A probabilistic graphical model  $\mathcal{G} := \langle \mathbf{X}, \mathbf{F} \rangle$  is a tuple of a set of discrete variables  $\mathbf{X}$  and a set of non-negative real valued functions  $\mathbf{F} = \{F_\alpha(\mathbf{X}_\alpha) | \alpha \in \mathcal{I}_\mathbf{F}\}$ , where  $\mathbf{X}_\alpha \subset \mathbf{X}$  is the scope of  $F_\alpha$ . Graphical model inference tasks can be viewed as eliminating a set of variables from the joint function by either summation or maximization. The MMAP task computes the mode of the marginal of the joint function by

$$\max_{\mathbf{X}_M} \sum_{\mathbf{X}_S} \prod_{\alpha \in \mathcal{I}_\mathbf{F}} F_\alpha(\mathbf{X}_\alpha), \quad (2)$$

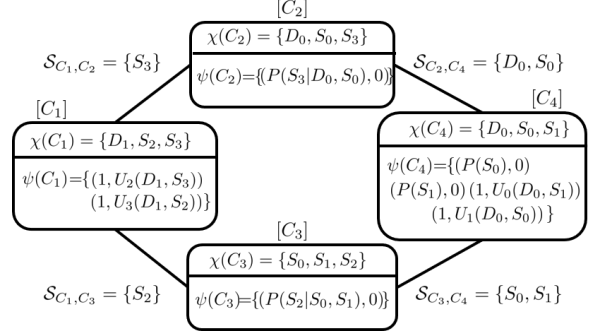


Figure 2: Join Graph Decomposition Example. Figure shows a join graph of the influence diagram in Figure 1(a). The join graph is generated by limiting the maximum cluster size from 4 to 3. The scope from a node labeling function  $\chi(C_i)$  is shown inside each node  $C_i$  and functions (valuations) are also allocated by  $\psi(C_i)$ . Separators  $S_{C_i, C_j}$  are shown next to edge  $(C_i, C_j)$ .

where  $\mathbf{X}_M$  denotes the maximization variables and  $\mathbf{X}_S$  denotes the summation variables. The relevance relation between variables is captured by an undirected graph  $\mathcal{G}_p = (V, E)$  called primal graph, where the set of nodes  $V$  represents variables, and an edge  $e \in E$  connects two nodes if both variables associated with those nodes appear in the scope of some function. A tree decomposition of  $\mathcal{G}_p$  produces a tree of clusters that captures the underlying structure of non-serial dynamic programming subject to the sequence of variable elimination operations. Namely, each cluster collects a set of functions that should be processed together. The worst-case space and time complexity of an inference query is exponential in the maximum cluster size called induced width  $w$  of  $\mathcal{G}_p$ .

Join graph decomposition [Mateescu et al., 2010] is an approximation scheme that further decomposes clusters in a join tree into a possibly loopy graph of finer grained clusters and, hence, it can control the complexity by limiting the maximum number of variables that are allowed to appear in cluster nodes.

**Definition 3. (join graph decomposition)** A join graph decomposition of a graphical model  $\mathcal{G}$  is a tuple  $\mathcal{D} := \langle \mathcal{G}_J, \chi, \psi \rangle$ , where  $\mathcal{G}_J = (\mathcal{C}, \mathcal{S})$  is a graph with nodes  $\mathcal{C}$  and edges  $\mathcal{S}$ , and  $\chi$  and  $\psi$  are labeling function that  $\chi$  maps a node  $C \in \mathcal{C}$  to a set of variables  $\chi(C_i) = \mathbf{X}_C$ , and  $\psi$  allocates each function  $F_\alpha$  exclusively to a node  $C \in \mathcal{C}$  such that  $\mathbf{X}_\alpha \subseteq \mathbf{X}_C$ . An edge  $(C_i, C_j) \in \mathcal{S}$  is associated with a subset of variables shared between the two clusters  $\chi(C_i) \cap \chi(C_j)$ , called separator  $S_{C_i, C_j}$ . The labeling function should ensure the running intersection property; for each variable  $X_i \in \mathbf{X}$ , the set  $\{C \in \mathcal{C} | X_i \in \psi(C)\}$  induces a connect subgraph.

A valid join graph can be systematically structured from a mini bucket tree produced by the MBE algorithm with the bounding parameter  $i$ -bound that controls the maximum

cluster size [Dechter and Rish, 2003]. The following example illustrates a join graph decomposition of the ID shown in Figure 1(a).

**Example 2.** Figure 2 shows a join graph decomposition  $\mathcal{D} : \langle \mathcal{G}_J, \chi, \psi \rangle$  of the ID in Figure 1(a). The primal  $\mathcal{G}_p$  of an ID can be obtained by removing all informational arcs before moralization and then removing the value nodes. From the  $\mathcal{G}_p$  and a legal variable elimination ordering compatible with the MEU query, a join graph  $\mathcal{G}_J$  can be generated by standard methods. The labeling functions  $\chi$  and  $\psi$  are displayed inside nodes and separators  $\mathcal{S}_{C_i, C_j}$  are displayed next to edges. Compared with the join tree shown in Figure 1(b), we see that the additional cluster node  $C_2$  contains a function  $P(S_3|D_0, S_0)$  that was included in Bucket  $S_3$  in the join tree.

## 2.4 DECOMPOSITION BOUNDS

Decomposition methods for bounding graphical model inference queries are based on two techniques. Namely, the graphical model decomposition with some auxiliary parameters and the optimization procedure that optimizes the parameters to improve the bound. For example, dual decomposition for MAP optimizes the dual variables, corresponding to Lagrange multipliers enforcing a set of local consistency constraints defined on the factor graph [Sontag et al., 2011]. Various decomposition bounds are available in the literature depending on decomposition schemes, methods of parameterization, and optimization frameworks. The common characteristic of such decomposition bounds is that they decompose the original graphical model to a relaxed lower complexity model, compute the global bound from decomposed local bounds and optimize the bound by additional local computations.

We review the generalized dual decomposition (GDD) bound for MMAP [Ping et al., 2015] that our bounding scheme is built upon. First, define a powered-sum elimination operator  $\sum_X^w$  by

$$\sum_X^w f(X) = [\sum_X |f(X)|^{1/w}]^w, \quad (3)$$

which generalizes maximization and summation with a weight  $0 \leq w \leq 1$  for the variable  $X$ . Note that the powered-sum elimination operator reduces to maximization and summation when  $w \rightarrow 0$  and  $w=1$ , respectively. Given a graphical model  $\mathcal{G} : \langle \mathbf{X}, \mathbf{F} \rangle$ , the MMAP task in Eq. (2) can be expressed by the powered-sum elimination operator as,

$$\sum_{\mathbf{X}}^w \prod_{\alpha \in \mathcal{I}_{\mathbf{F}}} F_{\alpha}(\mathbf{X}_{\alpha}), \quad (4)$$

where each weight  $w_i \in \mathbf{w}$  of a variable  $X_i \in \mathbf{X}$  is assigned 0 for the maximization variables and 1 for the summation variables. The bounding scheme of GDD is based on the generalization of the Hölder's inequality,

$$\sum_{\mathbf{X}}^w \prod_{\alpha \in \mathcal{I}_{\mathbf{F}}} F_{\alpha}(\mathbf{X}_{\alpha}) \leq \prod_{\alpha \in \mathcal{I}_{\mathbf{F}}} \sum_{\mathbf{X}_{\alpha}}^{w_{\alpha}} F_{\alpha}(\mathbf{X}_{\alpha}), \quad (5)$$

where  $\mathcal{I}_{\mathbf{F}}$  is the index set of functions  $\mathbf{F}$ ,  $\mathbf{w}$  is the set of weights that are either 0 or 1,  $w^{\alpha}$  is the set of non-negative weights distributed to  $\mathbf{X}_{\alpha}$  such that  $w_i = \sum_{\alpha \in \mathcal{I}_{\mathbf{F}}} w_i^{\alpha}$ . Note that the upper bound on the right-hand side of Eq. (5) combines local MMAP values only computed from a subset of variables  $\mathbf{X}_{\alpha}$ . The upper bounds of the MMAP in Eq. (5) can be formulated as an optimization problem by introducing cost-shifting parameters defined over a join graph decomposition  $\langle \mathcal{G}_J(\mathcal{C}, \mathcal{S}), \chi, \psi \rangle$  by the following equation,

$$\prod_{C_i \in \mathcal{C}} \sum_{\mathbf{X}_{C_i, \alpha \in \psi(C_i)}^{w^{C_i}}} [\prod_{F_{\alpha}(\mathbf{X}_{C_i})}] [\prod_{(\mathcal{S}_{C_i, C_j}) \in \mathcal{S}} \delta_{C_i, C_j}(\mathcal{S}_{C_i, C_j})], \quad (6)$$

where each cluster node  $C_i$  in the join graph  $\mathcal{G}_J$  collects a set of functions mapped by  $\chi(C_i)$ , and the cost-shifting parameters  $\delta_{C_j, C_i}(\mathcal{S}_{C_i, C_j})$  and  $\delta_{C_i, C_j}(\mathcal{S}_{C_i, C_j})$  are introduced on the separators  $\mathcal{S}_{C_i, C_j} \in \mathcal{S}$  such that the costs on the both sides cancel each other. The complexity of computing the upper bound is bounded by the maximum number of variables appearing in the cluster nodes. The optimization framework takes Eq. (6) as an objective function with weights  $w^{C_i}$  and cost-shifting functions  $\delta_{C_i, C_j}(\mathcal{S}_{C_i, C_j})$  as optimization parameters. Since the objective function is convex after taking log on Eq. (6), efficient optimization procedures are available for tightening the upper bound.

## 3 DECOMPOSITION BOUNDS FOR INFLUENCE DIAGRAMS

### 3.1 DECOMPOSING INFLUENCE DIAGRAMS

In this section, we develop a decomposition bound for IDs based on the valuation algebra. First, we generalize the powered-sum elimination operator in Eq. (3) to the valuation algebra.

**Definition 4. (powered-sum elimination for a valuation)** The powered-sum elimination operator for a valuation  $\Psi(X) := (P(X), V(X))$  is defined by

$$\sum_X^{(w, A)} \Psi(X) := \left( \sum_X^w P(X), \sum_X^w h(P(X), V(X), A) \otimes (1, -A) \right) \quad (7)$$

with the activation function  $h$  that adds an arbitrary utility constant  $A$  to the normalized expected utility value  $\frac{V(X)}{P(X)}$  and clips negative expected utility value as

$$h(P(X), V(X), A) = \begin{cases} P(X) \left( \frac{V(X)}{P(X)} + A \right) & \text{if } \frac{V(X)}{P(X)} + A > 0 \\ 0, & \text{otherwise.} \end{cases} \quad (8)$$

Since the powered-sum elimination applies to the absolute values of a function, we introduce the activation function  $h$  so that the powered-sum elimination

on the value component converges to the usual sum-elimination with a constant shift by  $A$  when weights  $\mathbf{w}$  are close to 1 and the value  $V(\mathbf{X})$  is negative. Namely,  $[\sum_X^w h(P(X), V(X))] \rightarrow \sum_X V(x) + A$  when  $w \rightarrow 1$  and  $A + \min(\frac{V(X)}{P(X)}) \geq 0$ .

Next, we define the comparison operator for the valuation algebra as a partial order as follows.

**Definition 5. (comparison of two valuations)** Given two valuations  $\Psi_1 := (P_1, V_1)$  and  $\Psi_2 := (P_2, V_2)$ , we define inequality  $\Psi_1 \leq \Psi_2$  iff.  $P_1 \leq P_2$  and  $V_1 \leq V_2$ .

Equipped with the powered-sum elimination and comparison operator for the valuation algebra, we state the decomposition bounds for IDs as follows.

**Theorem 1. (decomposition bounds for IDs)** Given an ID  $\mathcal{M}' := \langle \mathbf{X}, \Psi, \mathcal{O} \rangle$ , the MEU can be bounded by

$$\sum_{\mathcal{O}}^w \otimes_{\alpha \in \mathcal{I}_{\Psi}} \Psi_{\alpha}(\mathbf{X}_{\alpha}) \leq \otimes_{\alpha \in \mathcal{I}_{\Psi}} \sum_{\mathcal{O}}^{(\mathbf{w}^{\alpha}, \mathbf{A})} \Psi_{\alpha}(\mathbf{X}_{\alpha}). \quad (9)$$

The left-hand side is the MEU in Eq. (1), by rewriting the sequence of elimination operators as the powered-sum elimination operators  $\sum_{\mathcal{I}_0}^{\mathbf{w}^{\mathcal{I}_0}} \sum_{D_0}^{\mathbf{w}^{\mathcal{I}_0}} \dots \sum_{\mathcal{I}_{|\mathcal{D}|}}^{\mathbf{w}^{\mathcal{I}_{|\mathcal{D}|}}}$  following the partial ordering  $\mathcal{O}$ , where the weights  $\mathbf{w}^{\mathcal{I}_k}$  are 1 for the summation variables, and  $\mathbf{w}^{\mathcal{I}_k}$  are 0 for the maximization variables. The right-hand side switches the order of the elimination and combination operators, hence it combines fully eliminated local valuations to the global valuation with weights  $\mathbf{w}^{\alpha}$  that are distributed from  $\mathbf{w}$  such that  $w_i = \sum_{\alpha \in \mathcal{I}_{\Psi}} w_i^{\alpha}$ , where  $w_i$  is the weight of  $X_i \in \mathbf{X}$  and  $w_i^{\alpha}$  is the weight of  $X_i$  at  $\Psi_{\alpha}$ .

*Proof.* The decomposition bound can be obtained by applying Minkowski's inequality in Eq. (10) and Hölder's inequality in Eq. (11) to the probability and value functions of the valuations.

$$\sum_X^w f(X) + g(X) \leq \sum_X^w f(x) + \sum_X^w g(X) \quad (10)$$

$$\sum_X^w f(X)g(X) \leq \sum_X^{w_1} f(x) \sum_X^{w^{-w_1}} g(X) \quad (11)$$

The probability component in the left-hand side of Eq. (9) can be bounded by applying Hölder's inequality as shown in Eq. (12).

$$\sum_{\mathcal{O}}^w \prod_{i \in \mathcal{I}_{\Psi}} P_i \leq \prod_{i \in \mathcal{I}_{\Psi}} \sum_{\mathcal{O}}^{w_i} P_i. \quad (12)$$

The value component can be bounded by the following steps. We rewrite the MEU by introducing constant utilities  $A_i$  as shown in Eq. (13), and bound the MEU by the activation function  $h_i$  defined in Eq. (8) as shown in Eq. (14). The non-constant term in Eq. (14) can be further bounded by applying Minowski's inequality and Hölder's

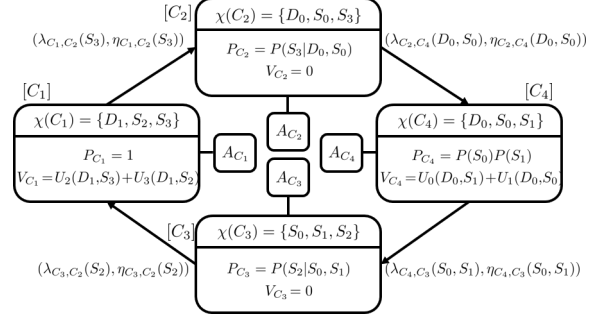


Figure 3: Optimization Parameters for Join Graph Decomposition Bounds. Figure shows the optimization parameters introduced in Proposition 1. The cost-shifting valuations are displayed next to the separators  $\mathcal{S}_{C_i, C_j}$  as  $(\lambda_{C_i, C_j}, \eta_{C_i, C_j})$ , and utility constants  $A_{C_i}$  are attached next to each cluster node  $C_i$ .

inequality as shown in Eq. (15) and (16), respectively.

$$\sum_{\mathcal{O}}^w \sum_{i \in \mathcal{I}_{\Psi}} [V_i + P_i(A_i - A_i)] [\prod_{j \neq i} P_j] \quad (13)$$

$$\leq \sum_{\mathcal{O}}^w \sum_{i \in \mathcal{I}_{\Psi}} h_i(P_i, V_i, A_i) [\prod_{j \neq i} P_j] - \sum_{i \in \mathcal{I}_{\Psi}} A_i \quad (14)$$

$$\leq \sum_{i \in \mathcal{I}_{\Psi}} [\sum_{\mathcal{O}}^w h_i(P_i, V_i, A_i)] [\prod_{j \neq i} P_j] - \sum_{i \in \mathcal{I}_{\Psi}} A_i \quad (15)$$

$$\leq \sum_{i \in \mathcal{I}_{\Psi}} [\sum_{\mathcal{O}}^{w_i} h_i(P_i, V_i, A_i)] [\prod_{j \neq i} \sum_{\mathcal{O}}^{w_j} P_j] - \sum_{i \in \mathcal{I}_{\Psi}} A_i \quad (16)$$

Note that the weights  $\mathbf{w}^i$  assigned to each valuation  $\Psi_i$  in Eq. (12) and Eq. (16) are the same. The final result can be obtained by combining the upper bound on the probability component in Eq. (12) and the upper bound on the value component in Eq. (16) as a valuation with the powered-sum elimination operator for a valuation.  $\square$

### 3.2 OPTIMIZING THE UPPER BOUNDS

The following Proposition 1 presents the parameterized decomposition bounds based on the Theorem 1 by introducing optimization parameters relative to a join graph decomposition. Then, the partial derivatives of the parameterized decomposition bounds are derived to be used in the first order optimization framework.

**Proposition 1. (parameterized decomposition bounds for IDs)** Given an ID  $\mathcal{M}' := \langle \mathbf{X}, \Psi, \mathcal{O} \rangle$  and its join graph decomposition  $\mathcal{D} := \langle \mathcal{G}_J(\mathcal{C}, \mathcal{S}), \chi, \psi \rangle$ , the decomposition bounds in Theorem 1 can be parameterized relative to a join graph decomposition  $\mathcal{G}_J$  as a pair of upper bounds on the probability component and the value component  $(L_{MMAP}, L_{MEU})$  as follows.

$$L_{MMAP} = \prod_{C_i \in \mathcal{C}} \sum_{\mathcal{O}}^{w_{C_i}} P'_{C_i}, \quad (17)$$

$$L_{MEU} = \sum_{C_i \in \mathcal{C}} [\sum_{\mathcal{O}}^{w_{C_i}} h_{C_i}(P'_{C_i}, V'_{C_i})] [\prod_{C_j \neq C_i} \sum_{\mathcal{O}}^{w_{C_j}} P'_{C_j}] - A_{C_i}. \quad (18)$$

In Proposition 1, the  $P'_{C_i}$  and  $V'_{C_i}$  are probability and value functions after incorporating cost-shifting parameters that can be written as

$$P'_{C_i} = P_{C_i} \prod_{(C_i, C_j) \in \mathcal{S}} \lambda_{C_i, C_j}, \quad (19)$$

$$V'_{C_i} = P'_{C_i} \left[ \frac{V_{C_i}}{P_{C_i}} + \sum_{(C_i, C_j) \in \mathcal{S}} \frac{\eta_{C_i, C_j}}{\lambda_{C_i, C_j}} \right]. \quad (20)$$

Each node  $C_i \in \mathcal{C}$  of the  $\mathcal{G}_J$  collects the probability and value functions by the labeling function  $\psi$ , and each edge  $(C_i, C_j) \in \mathcal{S}$  introduces a cost-shifting parameters  $\delta_{C_i, C_j} = (\lambda_{C_i, C_j}, \eta_{C_i, C_j})$  over the variables  $\mathcal{S}_{C_i, C_j}$  such that  $\delta_{C_i, C_j} \otimes \delta_{C_j, C_i} = (1, 0)$ . The utility constant parameters  $A_{C_i}$  is introduced through the activation function  $h_{C_i}$ , and the weight parameters  $\mathbf{w}^{C_i}$  are distributed from  $\mathbf{w}$  such that  $w_X = \sum_{C_i \in \mathcal{C}} w_X^{C_i}$  for all  $X \in \chi(C_i)$ . Note that the reparameterized decomposition bound for IDs subsumes the GDD bound for MMAP in Eq. (6) at the probability component,  $L_{\text{MMAP}}$ , and the new parameterized upper bound for the MEU at the value component,  $L_{\text{MEU}}$ . Note that the  $L_{\text{MEU}}$  in Eq. (20) is non-convex.

The following example illustrates the optimization parameters for the ID shown in Figure 1(a).

**Example 3.** Figure 3 illustrates the optimization parameters introduced by the join graph decomposition  $\mathcal{D} := \langle \mathcal{G}_J(\mathcal{C}, \mathcal{S}), \chi, \psi \rangle$  of Example 2. The valuations at each node  $C_i \in \mathcal{C}$  is displayed inside each node and the utility constant  $A_{C_i}$  is attached next to the node. The cost-shifting valuation  $(\lambda_{C_i, C_j}, \eta_{C_i, C_j})$  is shown next to the directed edges from  $C_i$  to  $C_j$  implying that the cost is moving from  $C_i$  to  $C_j$ , and hence  $\delta_{C_i, C_j}$  and  $\delta_{C_j, C_i}$  cancel each other.

Next, we present the first-order optimization procedures for tightening the parameterized decomposition bounds,  $L_{\text{MEU}}$  in Eq. (18). For efficiency and simplicity, we apply a block coordinate method that cycles through a subset of optimization parameters associated with the nodes and edges of the join graph  $\mathcal{G}_J$ , which we call the inner optimization problems. To optimize cost-shifting parameters  $\{\delta_{C_i, C_j} | (C_i, C_j) \in \mathcal{S}\}$  and utility constants  $\{A_{C_i} | C_i \in \mathcal{C}\}$ , we use gradient descent with line search [Wright and Nocedal, 1999] by

$$\mathbf{x}^{t+1} = \mathbf{x}^t - s \cdot [\nabla f(\mathbf{x}^t)], \quad (21)$$

where  $f$  is the objective function,  $s$  is the step size,  $\mathbf{x}^t$  is the optimization parameter at the  $t$ -th iteration. The weights  $\{\mathbf{w}^{C_i} | C_i \in \mathcal{C}\}$  are updated by exponentiated gradient descent [Kivinen and Warmuth, 1997] by

$$\mathbf{w}^{C_i, t+1} = \frac{\mathbf{w}^{C_i, t} \exp[-s \cdot [\nabla_{\mathbf{w}^{C_i}} L_{\text{MEU}}(\mathbf{w}^{C_i, t})]]}{\sum_{C_i \in \mathcal{C}} \mathbf{w}^{C_i, t} \exp[-s \cdot [\nabla_{\mathbf{w}^{C_i}} L_{\text{MEU}}(\mathbf{w}^{C_i, t})]]}. \quad (22)$$

Now, we define pseudo marginals and some useful expressions before deriving the gradients of each subset of the optimization parameters.

**Definition 6. (pseudo marginals)** Given a non-negative real-valued function  $Z_0(\mathbf{X}_{1:n})$  over a set of variables  $\mathbf{X}_{1:n} = \{X_1, \dots, X_n\}$ , and the weights  $\mathbf{w} = \{w_1, \dots, w_n\}$  associated with  $\mathbf{X}_{1:n}$ , we define a partial powered-sum elimination of variables  $\mathbf{X}_{1:i}$  from  $Z_0(\mathbf{X}_{1:n})$  by

$$Z_i(\mathbf{X}_{i+1:n}) = \sum_{\mathbf{x}_i}^{w_i} Z_{i-1}(\mathbf{X}_{i:n}).$$

The pseudo marginal of  $Z_0(\mathbf{X}_{1:n})$  is defined by

$$\Lambda(Z_0(\mathbf{X}_{1:n})) = \left[ \frac{Z_{n-1}(X_n)}{Z_n} \right]^{1/w_n} \dots \left[ \frac{Z_0(\mathbf{X}_{1:n})}{Z_1(\mathbf{X}_{2:n})} \right]^{1/w_1}.$$

Note that  $\Lambda(Z_0(\mathbf{X}_{1:n}))$  is a normalized distribution over  $\mathbf{X}_{1:n}$ , and each  $\left[ \frac{Z_{i-1}(\mathbf{X}_{i:n})}{Z_i(\mathbf{X}_{i+1:n})} \right]^{1/w_i}$  is a conditional distribution over  $X_i$  given  $\mathbf{X}_{i+1:n}$ .

Let's define a selector  $F_{C_j | C_i}$  that selects a probability or a value component depending on the indices  $i$  and  $j$  by

$$F_{C_j | C_i} = \begin{cases} h_{C_j}(P^{C_j}, V^{C_j}), & \text{if } j = i \\ P^{C_j}, & \text{otherwise.} \end{cases} \quad (23)$$

By using Eq. (23), the upper bound of the expected utility value at  $C_i$  can be expressed by

$$\Theta_{C_i} = \prod_{C_j \in \mathcal{C}} \sum_{\mathcal{O}}^{w^{C_j}} F_{C_j | C_i}. \quad (24)$$

In the following, we summarize the gradients of  $L_{\text{MEU}}$ ,  $\nabla L_{\text{MEU}}$  with respect to subsets of parameters.

$$\nabla L_{\text{MEU}}(w_k^{C_i}) = \sum_{C_j \in \mathcal{C}} \rho_{C_j} H(X_k | \mathbf{X}_{i+1:n} | \mathcal{O}; F_{C_j | C_i}) \quad (25)$$

$$\nabla L_{\text{MEU}}(A_{C_i}) = \Theta_{C_i} \sum_{\mathbf{x}_{C_i} \in \mathcal{S}_{C_i, C_j}} \frac{P^{C_i} \Lambda(h_{C_i}(P^{C_i}, V^{C_i}))}{h_{C_i}(P^{C_i}, V^{C_i})} - 1 \quad (26)$$

$$\nabla L_{\text{MEU}}(\lambda_{C_i, C_j}) = \sum_{C_k \in \mathcal{C}} \Theta_{C_k} \left[ \sum_{\mathbf{x}_{C_j} \in \mathcal{S}_{C_i, C_j}} \Lambda(F_{C_i | C_k}) - \sum_{\mathbf{x}_{C_i} \in \mathcal{S}_{C_i, C_j}} \Lambda(F_{C_j | C_k}) \right] \quad (27)$$

$$\nabla L_{\text{MEU}}(\eta_{C_i, C_j}) = \Theta_{C_j} \sum_{\mathbf{x}_{C_j} \in \mathcal{S}_{C_i, C_j}} \frac{P^{C_j} \Lambda(h_{C_j}(P^{C_j}, V^{C_j}))}{h_{C_j}(P^{C_j}, V^{C_j})} - \Theta_{C_i} \sum_{\mathbf{x}_{C_i} \in \mathcal{S}_{C_i, C_j}} \frac{P^{C_i} \Lambda(h_{C_i}(P^{C_i}, V^{C_i}))}{h_{C_i}(V^{C_i})} \quad (28)$$

The term  $H(X_k | \mathbf{X}_{i+1:n} | \mathcal{O}; F_{C_j | C_i})$  in Eq. (25) is the conditional entropy  $H(X_k | \mathbf{X}_{i+1:n} | \mathcal{O})$  of the pseudo marginal of the function selected by  $F_{C_j | C_i}$ .

### 3.3 MESSAGE PASSING ALGORITHM

Applying the parameterized decomposition bounds for IDs and the first order optimization procedures described in Section 3.2, we develop an iterative message passing algorithm that updates the optimization parameters defined relative to a join graph decomposition.

---

**Algorithm 1** Join Graph Decomposition for IDs (JGDID)

**Require:** Influence diagram  $\mathcal{M}' = \langle \mathbf{X}, \Psi, \mathcal{O} \rangle$ , initial weights  $w_i$  associated with a variable  $X_i \in \mathbf{X}$ ,  $i$ -bound, total iteration limit  $M_1$ , iteration limit  $M_2$  for updating weights and costs before updating utility constants.

**Ensure:** an upper bound of the MEU,  $L_{\text{MEU}}$ ,

```

1: generate a join graph decomposition  $\mathcal{D} = \langle \mathcal{G}_J(\mathcal{C}, \mathcal{S}), \chi, \psi \rangle$ 
   by MBE with  $i$ -bound and assign valuations to nodes by
   labeling function  $\psi$ .
2: execute single pass cost-shifting by messages generated by
   MBE algorithm based on the valuation algebra (MBE-VA)
3: initialize weights  $\mathbf{w}^{C_i}, \forall C_i \in \mathcal{C}$  by uniform weights.
4:  $iter=0, L_{\text{MEU}} = \inf$ 
5: while  $iter < M_1$  or  $L_{\text{MEU}}$  is not converged do
6:   for each variable  $X_i \in \mathbf{X}$  do
7:      $L_{\text{MEU}} \leftarrow \min(L_{\text{MEU}}, \text{UPDATE-WEIGHTS}(\mathcal{G}_J, X_i))$ 
8:   end for
9:   for each edge  $(C_i, C_j) \in \mathcal{S}$  do
10:     $L_{\text{MEU}} \leftarrow \min(L_{\text{MEU}}, \text{UPDATE-COSTS}(\mathcal{G}_J, \{C_i, C_j\}))$ 
11:   end for
12:   if  $iter > M_2$  then
13:     for each node  $C_i \in \mathcal{C}$  do
14:        $L_{\text{MEU}} \leftarrow \min(L_{\text{MEU}}, \text{UPDATE-UTIL-CONST}(\mathcal{G}_J, C_i))$ 
15:     end for
16:   end if
17:    $iter = iter + 1$ 
18: end while

```

---

Algorithm 1 outlines the procedure for updating the parameters, the Join Graph Decomposition Bound for IDs (JGDID). Given an input ID  $\mathcal{M}' := \langle \mathbf{X}, \Psi, \mathcal{O} \rangle$ , we first generate a join graph decomposition  $\mathcal{D} = \langle \mathcal{G}_J(\mathcal{C}, \mathcal{S}), \chi, \psi \rangle$  by executing the MBE algorithm with an input  $i$ -bound. For details on how to structure a join graph decomposition, see Mateescu et al. [2010]. Then, we assign valuations to the nodes in  $\mathcal{C}$  by labeling function  $\psi$  and run a single pass cost-shifting over the join graph using the messages generated by the MBE algorithm [Dechter and Rish, 2003] based on valuation algebra (MBE-VA). In our empirical evaluation, this preliminary step significantly improves the speed of convergence and the quality of the upper bound. The initial weights  $\mathbf{w}^{C_i}$  at each node  $C_i \in \mathcal{C}$  for the summation variables are uniform, and a small constant  $\epsilon \approx 10^{-6}$  initializes the weights for the maximization variables.

The block coordinate method updates the subset of the parameters by solving inner optimization problems following the structure of  $\mathcal{G}_J$ . The UPDATE-WEIGHTS routine in line 7 updates the weights  $w_i^{C_j}$  for a variable  $X_i$  over  $\forall C_j \in \mathcal{C}$  by Eq. (22) with the gradient in Eq. (25), the UPDATE-COSTS routine in line 10 updates cost-shifting valuations  $\delta_{C_i, C_j}$  over each edge  $(C_i, C_j) \in \mathcal{S}$  by the gradient descent with the gradients in Eq. (27) and in Eq. (28), and the UPDATE-UTIL-CONST routine in line 14 updates the utility constants  $A_{C_i}$  for each node  $C_i \in \mathcal{C}$  by the gradient descent with the gradient in Eq. (26). Since

Table 1: Benchmark statistics. Table shows the minimum, median, and maximum instance statistics from 10 instances. n is the number of chance and decision variables, f is the number of probability and utility functions, k is the maximum domain size, s is the maximum scope size, and w is the induced width.

Domain	n	f	k	s	w
FH-MDP	25, 110, 170	30, 143, 170	2, 3, 5	4, 7, 9	5, 25, 43
FH-POMDP	15, 51, 96	18, 61, 140	2, 2, 3	3, 5, 9	10, 27, 47
RAND	22, 57, 91	22, 57, 91	2, 2, 2	3, 3, 3	6, 18, 41
BN	54, 100, 115	54, 100, 115	2, 2, 2	6, 8, 10	18, 28, 45

the optimization objective function  $L_{\text{MEU}}$  is non-convex, the block coordinate method with our first-order optimization procedures is not guaranteed to provide a globally minimum bound, yet often performs well in practice.

In our empirical evaluation, we set the hyperparameters for the number of gradient updates for the inner optimization to 10, and the number of updates  $M_2$  for the weights and costs before updating utility constants to 20 and 50, which yielded a good convergence behavior.

## 4 EXPERIMENTS

We empirically compare the performance of our proposed algorithm JGDID with earlier approaches for bounding the MEU on four problem domains.

**Benchmarks.** The benchmarks are generated as follows. (1) Factored FH-MDP instances are generated from two stage factored MDP templates by controlling the number of state and action variables, the scope of functions, and the time horizon. (2) Factored FH-POMDP instances are generated similarly to FH-MDP instances, but we incorporate partial observability. (3) Random influence diagram (RAND) instances are generated by randomly generating influence diagram topology given the number of chance, decision, and value nodes. (4) BN instances are converted to ID from existing Bayesian networks used in the UAI-2006 probabilistic inference competitions by adding utility functions and randomly selecting decision variables. We generated 10 instances for each benchmark with increasing difficulty; Table 1 summarizes the instance statistics of the 4 benchmarks.

**Algorithms.** The first approach we compare against is the upper bounding algorithms for MMAP using the reduction from ID to MMAP. The reduction of Mauá [2016] generates standard MMAP instances, while Liu and Ihler [2012] generates MMAP instances with interleaved max and sum operators, which we call mixed MMAP. For the standard MMAP instances, we applied weighted mini-bucket with moment matching (WMBMM) [Liu and Ihler, 2011; Marinescu et al., 2014], and for the mixed MMAP instances, we applied GDD [Ping et al., 2015].

The second set of algorithms are applied directly to the In-

Table 2: Instance statistics of MMAP translation. Table shows the minimum, median, and maximum instance statistics of the standard MMAP reduction (MM) and the mixed MMAP reduction (MI).

Domain	Trans	n	k	w
FH-MDP	ID	25, 110, 170	2, 3, 5	5, 25, 43
	MI	26, 111, 171	10, 27, 80	15, 86, 160
FH-POMDP	ID	15, 51, 96	2, 3, 2	10, 27, 47
	MM	28, 188, 5277	6, 16, 48	14, 141, 5192
	MI	16, 52, 97	6, 16, 48	10, 28, 48
RAND	ID	22, 57, 91	2, 2, 2	6, 18, 41
	MM	29, 79, 142	2, 8, 21	8, 25, 58
	MI	23, 58, 92	2, 8, 21	6, 20, 42
BN	ID	54, 100, 115	2, 2, 2	18, 28, 45
	MM	69, 126, 202	3, 6, 12	20, 40, 92
	MI	55, 101, 116	3, 6, 12	19, 29, 46

fluence Diagram. One scheme is based on the mini-bucket idea which bounds the induced width by the  $i$ -bound, and then applies variable elimination using valuation algebra, yielding algorithm MBE-VA. The second is an information relaxation scheme, which relaxes the constrained variable ordering, denoted IR-SIS. [Nilsson and Hohle, 2001; Yuan et al., 2010]. The information relaxation is orthogonal to approximate elimination, and so both can be applied together; when we apply MBE-VA and JGDD together with the relaxed ordering of IR-SIS we call the hybrid algorithms MBE-VA+IR-SIS and JGDID-IR-SIS.

In summary, we evaluated 6 algorithms. We have JGDID and MBE-VA applied directly to the input IDs assuming only constrained ordering. We have JGDID+IR-SIS and MBE-VA+IR-SIS applied to the IDs but allowing relaxed ordering by IR-SIS, and finally we have WMBMM and GDD that are applied to MMAP reductions from IDs.

**Performance measure.** We report the quality of upper bounds for individual instances, and the average quality of the upper bounds by the mean of the ratio between the best upper bound found by all configurations (6 algorithms with  $i$ -bounds 1 and 15) and the upper bound of each under comparison; the closer the value to 1.0, the better the quality.

#### 4.1 COMPARING AGAINST MMAP TRANSLATIONS

Next, we compare our JGDID approach with WMBMM and GDD bounds based on MMAP translations.

**Impact of MMAP translation.** Table 2 summarizes the changes in the number of variables, the maximum domain size, and the induced width  $w$  due to the translation. When computing the induced width, we used the randomized min-fill algorithm. The reduction from ID to MMAP for the FH-MDP benchmark is not shown in the table because the translation was not feasible for most of the instances. Note that the standard MMAP reduction (MM)

Table 3: Average Quality of Upper Bounds.

Algorithm	FH-MDP	FH-POMDP	RAND	BN
JGDID+IR-SIS( $i=1$ )	NA	0.88	0.87	0.99
JGDID+IR-SIS( $i=15$ )	NA	0.76	0.85	0.64
JGDID( $i=1$ )	0.88	0.38	0.86	0.89
JGDID( $i=15$ )	0.49	0.38	0.85	0.64
MBE-VA+IR-SIS( $i=1$ )	NA	0.03	0.01	0.00
MBE-VA+IR-SIS( $i=15$ )	NA	0.54	0.46	0.15
MBE-VA( $i=1$ )	0.00	0.00	0.00	0.00
MBE-VA( $i=15$ )	0.40	0.29	0.46	0.17
GDD( $i=1$ )	0.87	0.03	0.11	0.24
GDD( $i=15$ )	0.22	0.11	0.15	0.05
WMBMM( $i=1$ )	0.00	0.00	0.01	0.01
WMBMM( $i=15$ )	0.01	0.23	0.35	0.24

inflates all input statistics. The number of variables is exponential in the size of the largest information set, the maximum domain size is increased to the total number of utility functions, and the induced width also increased significantly higher than input IDs. The mixed MMAP translation (MI) increases the number of variables by 1 which has domain size equal to the total number of utility functions. The induced width is increased by 1 except for the FH-MDP domain.

**Upper bounds from individual instances.** Figure 4 illustrates the quality of the obtained upper bounds for instances having the largest induced width in each benchmark. We ran JGDID and GDD algorithms up to 2000 iterations or until convergence. We can see from Figure 4 that JGDID dominates GDD and WMBMM on all instances except pomdp8. Comparing the upper bounds across  $i$ -bounds, JGDID and GDD do not show notable improvement on the speed of convergence with higher  $i$ -bounds due to the large overhead of a single iteration. We see that JGDID shows the step-wise improvement of upper bounds when it optimizes the utility constants.

**Average quality of upper bounds.** Table 3 shows the average quality of upper bounds. We see that the average quality of JGDID dominates both GDD and WMBMM in all benchmarks. Comparing JGDID and GDD, both generated upper bounds with similar quality on average in the FH-MDP benchmark. However, bounds from GDD are significantly worse than JGDID in other benchmarks. The upper bounds from WMBMM( $i=1$ ) is so large that the average ratios for all instances are close to 0.0.

#### 4.2 COMPARING AGAINST DIRECT ALGORITHMS

In this section we compare JGDID, MBE-VA, JGDID+IR-SIS, and MBE-VA+IR-SIS that are applied directly to IDs. They are all obtained by relaxing the input IDs either by decomposing graphical model or information relaxation.

**Impact of IR-SIS.** The IR-SIS relaxation often produces



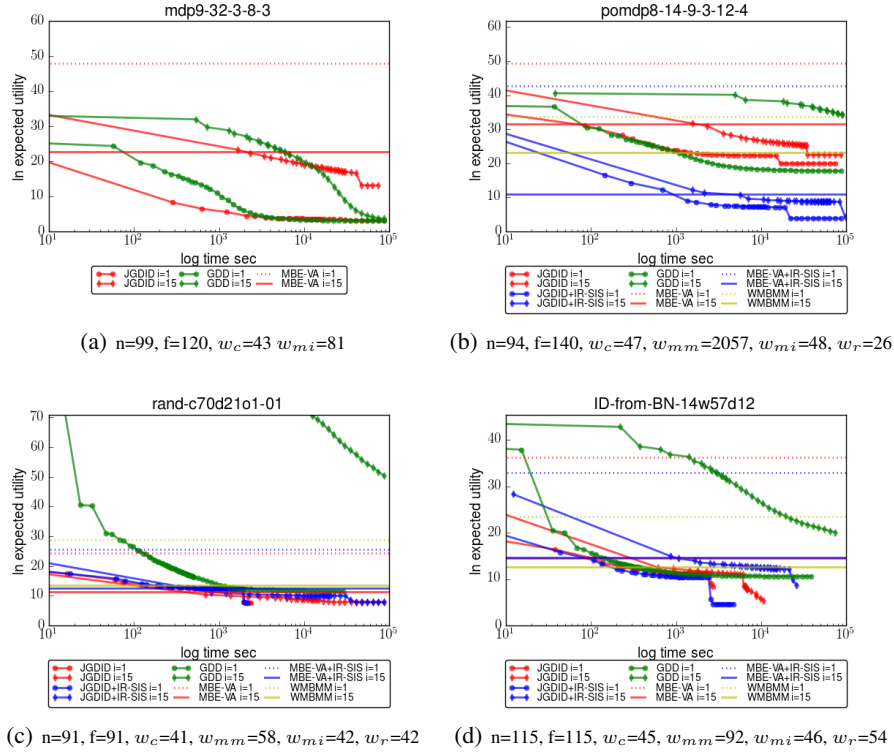


Figure 4: Upper Bounds of the MEU. Each plot shows the convergence of upper bounds over time. We draw horizontal lines for the bounds computed by non-iterative algorithms. The  $w_c$ ,  $w_{mm}$ ,  $w_{mi}$ , and  $w_r$  are the induced width of the input ID, standard MMAP reduction, mixed MMAP reduction, and relaxed ID with IR-SIS, respectively.

IDs with a lower induced width by reordering the variables in the information sets. It is shown to be very effective for FH-POMDP instances because IR-SIS transforms POMDP instances to MDP. The minimum, the median and the maximum induced width of FH-POMDP instances decreased from 10 to 3, 27 to 14, and 47 to 35, respectively via this relaxation. In other benchmarks, the improvement was negligible and often IR-SIS increased induced width.

**Upper bounds from individual instances.** Figure 4 illustrates also upper bounds obtained by direct algorithms. Comparing JGDID with MBE-VA, we see in the figure that JGDID improved the quality of the upper bound significantly in all instances. IR-SIS often significantly improves the quality of the bounds when the upper bounds from decomposition methods is still weak. In case of `pomdp8`, we see that the JGDID+IR-SIS improved the upper bound significantly compared to JGDID.

**Average quality of upper bounds.** Table 3 also shows the average quality of the upper bounds for the 4 direct algorithms. We see that the average quality of JGDID dominates MBE-VA. The average quality of the upper bounds obtained from MBE-VA( $i=1$ ) are so weak that their value is closed to 0.0 in all benchmarks. Both JGDID and MBE-VA improve the average quality of the upper bounds when they are combined with IR-SIS, and

JGDID+IR-SIS presents the best average quality overall.

## 5 CONCLUSIONS

We present a new algorithm, Join Graph Decomposition for solving Influence Diagrams, called JGDID, using the valuation algebra. Our scheme subsumes the decomposition bounds for marginal MAP, and also provide a bound for the MEU. Our experiments show the effectiveness of the translation free approach and the significant improvement in the quality of upper bounds compared with earlier state-of-the-art approaches. We also demonstrate that a join graph decomposition scheme can be combined with information relaxation scheme to yield superior bounds. The principle of decomposing a sequence of decision problems to a collection of weakly coupled subproblems can be further developed by incorporating advanced optimization frameworks, and the resulting, effective upper bounding schemes can be applied to probabilistic planning and stochastic programming.

## ACKNOWLEDGEMENTS

This work was supported in part by NSF grants IIS-1526842 and IIS-1254071, the US Air Force (Contract FA9453-16-C-0508) and DARPA (Contract W911NF-18-C-0015).

## References

- Dechter, R. (1999). Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113(1):41–85.
- Dechter, R. (2000). An anytime approximation for optimizing policies under uncertainty. In *AIPS-2000 Workshop on Decision Theoretic Planning*.
- Dechter, R. and Rish, I. (2003). Mini-buckets: A general scheme for bounded inference. *Journal of the ACM (JACM)*, 50(2):107–153.
- Howard, R. A. and Matheson, J. E. (2005). Influence diagrams. *Decision Analysis*, 2(3):127–143.
- Jensen, F., Jensen, F. V., and Dittmer, S. L. (1994). From influence diagrams to junction trees. In *Proceedings of the 10th international conference on Uncertainty in artificial intelligence*, pages 367–373.
- Kivinen, J. and Warmuth, M. K. (1997). Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–63.
- Liu, Q. and Ihler, A. (2011). Bounding the partition function using Hölder’s inequality. In *Proceedings of the 28th International Conference on Machine Learning, ICML ’11*, pages 849–856.
- Liu, Q. and Ihler, A. (2012). Belief propagation for structured decision making. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*, pages 523–532.
- Marinescu, R., Dechter, R., and Ihler, A. (2014). AND/OR search for marginal MAP. In *Uncertainty in Artificial Intelligence (UAI)*, pages 563–572, Quebec City, Canada.
- Mateescu, R., Kask, K., Gogate, V., and Dechter, R. (2010). Join-graph propagation algorithms. *Journal of Artificial Intelligence Research*, 37:279–328.
- Mauá, D. D. (2016). Equivalences between maximum a posteriori inference in bayesian networks and maximum expected utility computation in influence diagrams. *Int. J. Approx. Reasoning*, 68(C):211–229.
- Mauá, D. D., de Campos, C. P., and Zaffalon, M. (2012). Solving limited memory influence diagrams. *Journal of Artificial Intelligence Research*, 44:97–140.
- Nielsen, T. D. and Jensen, F. V. (1999). Welldefined decision scenarios. In *Proceedings of The 15th Conference on Uncertainty in Artificial Intelligence*, pages 502–511.
- Nilsson, D. and Hohle, M. (2001). Computing bounds on expected utilities for optimal policies based on limited information. *Technical Report*, (94).
- Ping, W., Liu, Q., and Ihler, A. T. (2015). Decomposition bounds for marginal MAP. In *Proceedings of Advances in Neural Information Processing Systems 28*, pages 3267–3275.
- Sontag, D., Globerson, A., and Jaakkola, T. (2011). Introduction to dual decomposition for inference. *Optimization for Machine Learning*, 1(219-254):1.
- Wright, S. and Nocedal, J. (1999). Numerical optimization. *Springer Science*, 35(67-68):7.
- Yuan, C., Wu, X., and Hansen, E. A. (2010). Solving multistage influence diagrams using branch-and-bound search. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, pages 691–700.

---

# Causal Discovery with Linear Non-Gaussian Models under Measurement Error: Structural Identifiability Results

---

Kun Zhang<sup>†</sup>, Mingming Gong<sup>\*†</sup>, Joseph Ramsey<sup>†</sup>, Kayhan Batmanghelich<sup>\*</sup>,  
Peter Spirtes<sup>†</sup>, Clark Glymour<sup>†</sup>

<sup>†</sup>Department of philosophy, Carnegie Mellon University

<sup>\*</sup>Department of Biomedical Informatics, University of Pittsburgh

## Abstract

Causal discovery methods aim to recover the causal process that generated purely observational data. Despite its successes on a number of real problems, the presence of measurement error in the observed data can produce serious mistakes in the output of various causal discovery methods. Given the ubiquity of measurement error caused by instruments or proxies used in the measuring process, this problem is one of the main obstacles to reliable causal discovery. It is still unknown to what extent the causal structure of relevant variables can be identified in principle. This study aims to take a step towards filling that void. We assume that the underlining process or the measurement-error free variables follows a linear, non-Gaussian causal model, and show that the so-called ordered group decomposition of the causal model, which contains major causal information, is identifiable. The causal structure identifiability is further improved with different types of sparsity constraints on the causal structure. Finally, we give rather mild conditions under which the whole causal structure is fully identifiable.

## 1 INTRODUCTION

Understanding and using causal relations among variables of interest has been a fundamental problem in various fields, including biology, neuroscience, and social sciences. Since interventions or controlled randomized experiments are usually expensive or even impossible to conduct, discovering causal information from observational data, known as causal discovery (Spirtes et al., 2001; Pearl, 2000), has been an important task and received much attention in computer science, statistics, and philosophy. Roughly speaking, methods for causal

discovery are categorized into constraint-based ones, such as the PC algorithm (Spirtes et al., 2001), and score-based ones, such as Greedy Equivalence Search (GES) (Chickering, 2002).

Almost all current causal discovery methods assume that the recorded values are realizations of the variables of interest. Typically, however, the measured values are not identical to the values of the variables that they are intended to measure. The measuring process may involve nonlinear distortion, as already address by the post-nonlinear causal model (Zhang & Hyvärinen, 2009; Zhang & Chan, 2006), and may introduce a lot of error. For instance, in neuroscience the measured brain signals obtained by functional magnetic resonance (fMRI) usually contain error introduced by instruments. In this paper, we consider the so-called random measurement error model, as defined by Scheines & Ramsey (2017), in which observed variables  $X_i$ ,  $i = 1, \dots, n$ , are generated from the underlying measurement-error-free variables  $\tilde{X}_i$  with additive measurement errors  $E_i$ :

$$X_i = \tilde{X}_i + E_i. \quad (1)$$

We further assume that the errors  $E_i$  are mutually independent and independent from  $\tilde{X}_i$ . Putting the causal model for  $\tilde{X}_i$  and the random measurement error model together, we have the whole process that generates the measured data. We call this process the CAusal Model with Measurement Error (CAMME).

Generally speaking, because of the presence of measurement errors, the d-separation patterns among  $X_i$  are different from those among the underlying variables  $\tilde{X}_i$ . This generating process has been called the random measurement error model in (Scheines & Ramsey, 2017). According to the causal Markov condition (Spirtes et al., 2001; Pearl, 2000), observed variables  $X_i$  and the underlying variables  $\tilde{X}_i$  may have different conditional independence/dependence relations and, as a consequence, the output of approaches to causal discovery that exploit conditional independence and dependence relations are unreliable in the

presence of such errors, as demonstrated in (Scheines & Ramsey, 2017). In Section 2 we will give an example to show how conditional independence/dependence between the variables is changed by measurement error, and discuss its implication in applications of causal discovery to real problems. Furthermore, because of the measurement error, the structural equation models according to which the measurement-error-free variables  $\tilde{X}_i$  are generated usually do not hold for the observed variables  $X_i$ . (In fact,  $X_i$  follow error-in-variables models, for which the identifiability of the underlying causal relation is not clear.) Hence, approaches based on structural equation models, such as the linear, non-Gaussian, acyclic model (LiNGAM (Shimizu et al., 2006)), will generally fail to find the correct causal direction.

In this paper, we aim to estimate the causal model underlying the measurement-error-free variables  $\tilde{X}_i$  from their observed values  $X_i$  contaminated by random measurement error. We assume linearity of the causal model and causal sufficiency relative to  $\{\tilde{X}_i\}_{i=1}^n$ . We particularly focus on the case where the causal structure for  $\tilde{X}_i$  is represented by a Directed Acyclic Graph (DAG), although this condition can be weakened. In order to develop principled causal discovery methods to recover the causal model for  $\{\tilde{X}_i\}_{i=1}^n$  from observed values of  $\{X_i\}_{i=1}^n$ , we have to address theoretical issues include 1) whether the causal model of interest is completely or partially identifiable from the contaminated observations and 2) what are the precise identifiability conditions.

There exist causal discovery methods, such as the Fast Causal Inference (FCI) algorithm (Spirtes et al., 2001), to deal with confounders, i.e., hidden direct common causes. However, they cannot estimate the causal relations among the "latent" variables, which is what we aim to recover in this paper. Silva et al. (2006) and Kummerfeld et al. (2014) have provided algorithms for recovering latent variables and their causal relations when each latent variable has multiple measured effects; Shimizu et al. (2011a) further applied LiNGAM to the recovered latent variables to improve the estimated causal relations between them. Their problem is different from the measurement error setting we consider, where clustering for latent common causes is not required and each measured variable is the direct effect of a single "true" variable. As discussed in Section 3, their models can be seen as special cases of our setting.

## 2 EFFECT OF MEASUREMENT ERROR

Suppose we observe variables  $X_1$ ,  $X_2$ , and  $X_3$ , which are generated from measurement-error-free variables  $\tilde{X}_i$  according to the structure given in Figure 1. By

the Markov condition and Faithfulness assumption, all three of the  $\tilde{X}_i$  variables are dependent on one another, while  $\tilde{X}_1$  and  $\tilde{X}_3$  are conditionally independent given  $\tilde{X}_2$ . That conditional independence does not hold for  $X_i$ , the variables actually observable. The measurement error  $E_2$  produces the trouble. We will treat the distributions as Gaussian purely for illustration; again, the point is general.

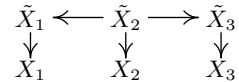


Figure 1: A linear CAMME to demonstrate the effect of measurement error on conditional independence and dependence relationships. For simplicity, we consider the special case where there is measurement error only in  $X_2$ , i.e.,  $X_2 = \tilde{X}_2 + E_2$ , but  $X_1 = \tilde{X}_1$  and  $X_3 = \tilde{X}_3$ .

Let  $\tilde{\rho}_{12}$  be the correlation coefficient between  $\tilde{X}_1$  and  $\tilde{X}_2$  and  $\tilde{\rho}_{13,2}$  be the partial correlation coefficient between  $\tilde{X}_1$  and  $\tilde{X}_3$  given  $\tilde{X}_2$ , which is zero. Let  $\rho_{12}$  and  $\rho_{13,2}$  be the corresponding correlation coefficient and partial correlation coefficient in the presence of measurement error. We let  $\tilde{\rho}_{12} = \tilde{\rho}_{23} = \tilde{\rho}$  to make the argument simpler, but the point is quite general. So we have  $\rho_{13} = \tilde{\rho}_{13} = \tilde{\rho}_{12}\tilde{\rho}_{23} = \tilde{\rho}^2$ . Let  $\gamma = \frac{\text{Std}(E_2)}{\text{Std}(\tilde{X}_2)}$ . For the data with measurement error, we have

$$\begin{aligned} \rho_{12} &= \frac{\text{Cov}(X_1, X_2)}{\text{Var}^{1/2}(X_1)\text{Var}^{1/2}(X_2)} \\ &= \frac{\text{Cov}(\tilde{X}_1, \tilde{X}_2)}{\text{Var}^{1/2}(\tilde{X}_1)(\text{Var}(\tilde{X}_2) + \text{Var}(E_2))^{1/2}} \\ &= \frac{\tilde{\rho}}{(1 + \gamma^2)^{1/2}}; \\ \rho_{13,2} &= \frac{\rho_{13} - \rho_{12}\rho_{23}}{(1 - \rho_{12}^2)^{1/2}(1 - \rho_{23}^2)^{1/2}} \\ &= \frac{\tilde{\rho}_{13} - \frac{\tilde{\rho}_{12}\tilde{\rho}_{23}}{1 + \gamma^2}}{(1 - \frac{\tilde{\rho}^2}{(1 + \gamma^2)})^{1/2}(1 - \frac{\tilde{\rho}^2}{(1 + \gamma^2)})^{1/2}} \\ &= \frac{\tilde{\rho}^2}{1 + \gamma^2 - \tilde{\rho}^2}. \end{aligned}$$

As the variance of the measurement error in  $X_2$  increases,  $\gamma$  become larger, and  $\rho_{12}$  decreases and finally goes to zero; in contrast,  $\rho_{13,2}$ , which is zero for the measurement-error-free variables, is increasing and finally converges to  $\tilde{\rho}^2$ . See Figure 2 for an illustration. In other words, in this example as the variance of the measurement error in  $X_2$  increases,  $X_1$  and  $X_2$  become more and more independent, while  $X_1$  and  $X_3$  are conditionally more and more dependent given  $X_2$ . However, for the measurement-error-free variables,  $\tilde{X}_1$  and  $\tilde{X}_2$  are dependent and  $\tilde{X}_1$  and  $\tilde{X}_3$  are conditionally independent given  $\tilde{X}_2$ . The PC algorithm and other methods that explicitly or implicitly exploit con-

ditional independence and dependence relations will find an edge between  $X_1$  and  $X_3$  that does not exist between  $X_1$  and  $X_3$ . Multiple regression of  $X_3$  on  $X_1$  and  $X_2$ , or  $X_1$  on  $X_3$  and  $X_2$ , will make the same error.

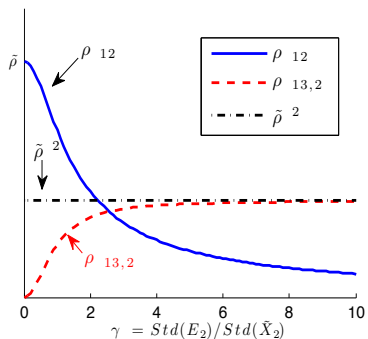


Figure 2: The correlation coefficient  $\rho_{12}$  between  $X_1$  and  $X_2$  and partial correlation coefficient  $\rho_{13,2}$  between  $X_1$  and  $X_3$  given  $X_2$  as functions of  $\gamma$ , the ratio of the standard deviation of measurement error to the that of  $\tilde{X}_2$ . We have assumed that the correlation coefficient between  $\tilde{X}_1$  and  $\tilde{X}_2$  and that between  $\tilde{X}_2$  and  $\tilde{X}_3$  are the same (denoted by  $\tilde{\rho}$ ), and that there is measurement error only in  $X_2$ .

Roughly speaking, originally conditionally independent (or dependent) variables will become less independent (or dependent), due to the effect of measurement error. In order to correctly detect conditional independence relations between measurement-error-free variables from the observed noisy values, one may use a very small significance level (or type I error level,  $\alpha$ ) when performing conditional independence tests—the smaller the significance level, the less often the independence null hypothesis is rejected, and more pairs of variables are likely to be considered as conditionally independent. This, inevitably, risks high type II errors (i.e., conditionally dependent variable pairs are likely to be considered as independent), especially when the sample size is relatively small. Therefore it is desirable to develop principled causal discovery methods to deal with measurement error.

One might apply other types of methods instead of the constraint-based ones for causal discovery from data with measurement error. In fact, as the measurement-error-free variables are not observable,  $\tilde{X}_2$  in Figure 1 is actually a confounder for observed variables. As a consequence, generally speaking, due to the effect of the confounders, the independence noise assumption underlying functional causal model-based approaches, such as the method based on the linear, non-Gaussian, acyclic model (Shimizu et al., 2006), will not hold for the observed variables any more. Figure 3 gives an

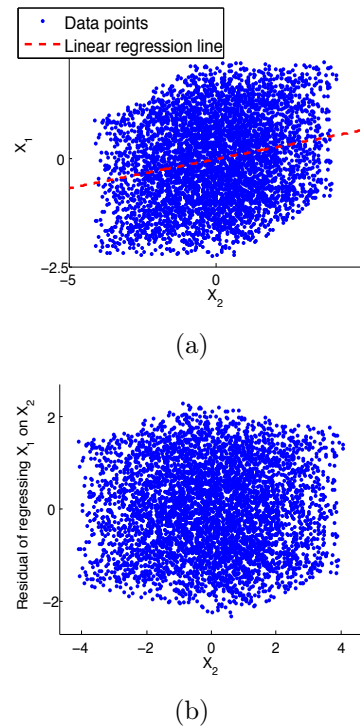


Figure 3: Illustration on how measurement error leads to dependence between regression residual and contaminated cause. (a) Scatter plot of  $X_2$  and  $X_1$  with measurement error in  $X_2$  together with the regression line. (b) Scatter plot of the regression residual and  $X_2$ . Note that if we regress  $\tilde{X}_1$  on  $\tilde{X}_2$ , the residual is independent from  $\tilde{X}_2$ .

illustration on this. Figure 3(a) shows the scatter plot of  $X_1$  vs.  $X_2$  and the regression line from  $X_2$  to  $X_1$ , where  $\tilde{X}_2$ , the noise in  $\tilde{X}_1$ , and the measurement error  $E_2$ , are all uniformly distributed ( $\rho = 0.4$ , and  $\gamma = 1.4$ ). As seen from Figure 3(b), the residual of regressing  $X_1$  on  $X_2$  is not independent from  $X_2$ , although the residual of regressing  $\tilde{X}_1$  on  $\tilde{X}_2$  is independent from  $\tilde{X}_2$ . As a result, the functional causal model-based approaches to causal discovery may also fail to find the causal structure of the measurement-error-free variables from their contaminated observations. The effect of measurement error on causal direction identification in the two-variable case was also studied by Wiedermann et al. (2018) under some further assumptions.

### 3 MODEL CANONICAL REPRESENTATION

Let  $\tilde{G}$  be the acyclic causal model over  $\tilde{X}_i$ . Here we call it *measurement-error-free causal model*. Let  $\mathbf{B}$  be the corresponding causal adjacency matrix for  $\tilde{X}_i$ , in which  $B_{ij}$  is the coefficient of the direct causal influence

from  $\tilde{X}_j$  to  $\tilde{X}_i$  and  $B_{ii} = 0$ . We have,

$$\tilde{\mathbf{X}} = \mathbf{B}\tilde{\mathbf{X}} + \tilde{\mathbf{E}}, \quad (2)$$

where the components of  $\tilde{\mathbf{E}}$ ,  $\tilde{E}_i$ , have non-zero, finite variances. Then  $\tilde{\mathbf{X}}$  is actually a linear transformation of the error terms in  $\tilde{\mathbf{E}}$  because (2) implies

$$\tilde{\mathbf{X}} = \underbrace{(\mathbf{I} - \mathbf{B})^{-1}}_{\triangleq \mathbf{A}} \tilde{\mathbf{E}}. \quad (3)$$

Now let us consider two types of nodes of  $\tilde{G}$ , namely, leaf nodes (i.e., those that do not influence any other node) and non-leaf nodes. Accordingly, the noise term in their structural equation models also has distinct behaviors: If  $\tilde{X}_i$  is a leaf node, then  $\tilde{E}_i$  influences only  $\tilde{X}_i$ , not any other; otherwise  $\tilde{E}_i$  influences  $\tilde{X}_i$  and at least one other variable,  $\tilde{X}_j$ ,  $j \neq i$ . Consequently, we can decompose the noise vector into two groups:  $\tilde{\mathbf{E}}^L$  consists of the  $l$  noise terms that influence only leaf nodes, and  $\tilde{\mathbf{E}}^{\text{NL}}$  contains the remaining noise terms. Equation (3) can be rewritten as

$$\tilde{\mathbf{X}} = \mathbf{A}^{\text{NL}}\tilde{\mathbf{E}}^{\text{NL}} + \mathbf{A}^L\tilde{\mathbf{E}}^L = \tilde{\mathbf{X}}^* + \mathbf{A}^L\tilde{\mathbf{E}}^L, \quad (4)$$

where  $\tilde{\mathbf{X}}^* \triangleq \mathbf{A}^{\text{NL}}\tilde{\mathbf{E}}^{\text{NL}}$ ,  $\mathbf{A}^{\text{NL}}$  and  $\mathbf{A}^L$  are  $n \times (n-l)$  and  $n \times l$  matrices, respectively. Here both  $\mathbf{A}^L$  and  $\mathbf{A}^{\text{NL}}$  have specific structures. All entries of  $\mathbf{A}^L$  are 0 or 1; for each column of  $\mathbf{A}^L$ , there is only one non-zero entry. In contrast, each column of  $\mathbf{A}^{\text{NL}}$  has at least two non-zero entries, representing the influences from the corresponding non-leaf noise term.

We give a more formal way to derive the above result and make it clear how  $\mathbf{A}^{\text{NL}}$  and  $\mathbf{A}^L$  depend on  $\mathbf{B}$ . For any graph  $\tilde{G}$  there always exists a suitable permutation matrix, denoted by  $\Omega$ , such that the last  $l$  elements of the permuted variables  $\Omega\tilde{\mathbf{X}}$  are all leaf nodes. Hence,  $\Omega\tilde{\mathbf{E}} = \begin{bmatrix} \tilde{\mathbf{E}}^{\text{NL}} \\ \tilde{\mathbf{E}}^L \end{bmatrix}$ . Accordingly, (2) implies that

$$\Omega\tilde{\mathbf{X}} = \mathbf{B}_\Omega \cdot \Omega\tilde{\mathbf{X}} + \Omega\tilde{\mathbf{E}}, \quad (5)$$

where  $\mathbf{B}_\Omega = \Omega\mathbf{B}\Omega^\top$ . Since the last  $l$  variables in  $\Omega\tilde{\mathbf{X}}$  are leaf nodes, the last  $l$  columns of  $\mathbf{B}_\Omega$  are zero. Let  $\mathbf{B}_\Omega^{\text{NL}}$  be the causal influence matrix for the non-leaf nodes and  $\mathbf{B}_\Omega^L$  denote the causal influence from non-leaf nodes to leaf nodes. We have  $\mathbf{B}_\Omega = \begin{bmatrix} \mathbf{B}_\Omega^{\text{NL}} & \mathbf{0} \\ \mathbf{B}_\Omega^L & \mathbf{0} \end{bmatrix}$ . Consequently,

$$(\mathbf{I} - \mathbf{B}_\Omega)^{-1} = \begin{bmatrix} (\mathbf{I} - \mathbf{B}_\Omega^{\text{NL}})^{-1} & \mathbf{0} \\ \mathbf{B}_\Omega^L(\mathbf{I} - \mathbf{B}_\Omega^{\text{NL}})^{-1} & \mathbf{I} \end{bmatrix}. \quad (6)$$

Combining (5) and (6) gives

$$\begin{aligned} \tilde{\mathbf{X}} &= \Omega^\top(\mathbf{I} - \mathbf{B}_\Omega)^{-1}\Omega\tilde{\mathbf{E}} \\ &= \underbrace{\Omega^\top \cdot \begin{bmatrix} \mathbf{I} \\ \mathbf{B}_\Omega^L \end{bmatrix}}_{\mathbf{A}^{\text{NL}}} \cdot (\mathbf{I} - \mathbf{B}_\Omega^{\text{NL}})^{-1} \tilde{\mathbf{E}}^{\text{NL}} + \underbrace{\Omega^\top \begin{bmatrix} \mathbf{0} \\ \tilde{\mathbf{E}}^L \end{bmatrix}}_{\mathbf{A}^L\tilde{\mathbf{E}}^L}. \end{aligned}$$

Further consider the generating process of observed variables  $X_i$ . Combining (1) and (4) gives

$$\begin{aligned} \mathbf{X} &= \tilde{\mathbf{X}}^* + \mathbf{A}^L\tilde{\mathbf{E}}^L + \mathbf{E} = \mathbf{A}^{\text{NL}}\tilde{\mathbf{E}}^{\text{NL}} + (\mathbf{A}^L\tilde{\mathbf{E}}^L + \mathbf{E}) \\ &= \mathbf{A}^{\text{NL}}\tilde{\mathbf{E}}^{\text{NL}} + \mathbf{E}^* \\ &= \begin{bmatrix} \mathbf{A}^{\text{NL}} & \mathbf{I} \end{bmatrix} \cdot \begin{bmatrix} \tilde{\mathbf{E}}^{\text{NL}} \\ \mathbf{E}^* \end{bmatrix}, \end{aligned} \quad (7)$$

where  $\mathbf{E}^* = \mathbf{A}^L\tilde{\mathbf{E}}^L + \mathbf{E}$  and  $\mathbf{I}$  denotes the identity matrix. To make it more explicit, we give how  $\tilde{X}_i^*$  and  $E_i^*$  are related to the original CAMME process:

$$\tilde{X}_i^* = \begin{cases} \tilde{X}_i, & \text{if } \tilde{X}_i \text{ is not a leaf node in } \tilde{G}; \\ \tilde{X}_i - \tilde{E}_i, & \text{otherwise;} \end{cases}, \text{ and} \quad (9)$$

$$E_i^* = \begin{cases} E_i, & \text{if } \tilde{X}_i \text{ is not a leaf node in } \tilde{G}; \\ E_i + \tilde{E}_i, & \text{otherwise.} \end{cases}$$

Clearly  $E_i^*$ s are independent across  $i$ , and as we shall see in Section 4, the information shared by difference  $X_i$  is still captured by  $\tilde{\mathbf{X}}^*$ . For each CAMME specified by (2) and (1), there always exists an observationally equivalent representation in the form of (7). We call the representation (7) the canonical representation of the CAMME (CR-CAMME).

**Example Set 1** Consider the following example with three observed variables  $X_i$ ,  $i = 1, 2, 3$ , for which  $\tilde{X}_1 \rightarrow \tilde{X}_2 \leftarrow \tilde{X}_3$ , with causal relations  $\tilde{X}_2 = a\tilde{X}_1 + b\tilde{X}_3 + \tilde{E}_2$ . That is,

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 \\ a & 0 & b \\ 0 & 0 & 0 \end{bmatrix}, \text{ and } \mathbf{A} = \begin{bmatrix} 1 & 0 & 0 \\ a & 1 & b \\ 0 & 0 & 1 \end{bmatrix}.$$

Therefore,

$$\begin{aligned} \mathbf{X} &= \tilde{\mathbf{X}} + \mathbf{E} = \tilde{\mathbf{X}}^* + \mathbf{E}^* \\ &= \begin{bmatrix} 1 & 0 \\ a & b \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \tilde{E}_1 \\ \tilde{E}_3 \end{bmatrix} + \begin{bmatrix} E_1 \\ \tilde{E}_2 + E_2 \\ E_3 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 & | & 1 & 0 & 0 \\ a & b & | & 0 & 1 & 0 \\ 0 & 1 & | & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \tilde{E}_1 \\ \tilde{E}_3 \\ E_1 \\ \tilde{E}_2 + E_2 \\ E_3 \end{bmatrix}. \end{aligned}$$

In causal discovery from observations in the presence of measurement error, we aim to recover information of the measurement-error-free causal model  $\tilde{G}$ . Let us define a new graphical model,  $\tilde{G}^*$ . It is obtained by replacing variables  $\tilde{X}_i$  in  $\tilde{G}$  with variables  $\tilde{X}_i^*$ . In other words, it has the same causal structure and causal parameters (given by the  $\mathbf{B}$  matrix) as  $\tilde{G}$ , but with

variables  $\tilde{X}_i^*$  as its nodes. If we manage to estimate the structure of and the involved causal parameters in  $\tilde{G}^*$ , then the causal model of interest,  $\tilde{G}$ , is recovered. We defined the graphical model  $\tilde{G}^*$  because we cannot fully estimate the distribution of measurement-error-free variables  $\tilde{\mathbf{X}}$ , but might be able to estimate that of  $\tilde{\mathbf{X}}^*$  under proper assumptions, as shown in Section 4.

Compared to  $\tilde{G}$ ,  $\tilde{G}^*$  involves some deterministic causal relations because each leaf node is a deterministic function of its parents (the noise in leaf nodes has been removed; see (9)). For instance, suppose in  $\tilde{G}^*$ ,  $\text{PA}(\tilde{X}_3^*) = \{\tilde{X}_1^*, \tilde{X}_2^*\}$ , where  $\text{PA}(\tilde{X}_3^*)$  denotes the set of parents of  $\tilde{X}_3^*$  in  $\tilde{G}^*$ , and that  $\tilde{X}_3$  is a leaf node. Then each of  $\tilde{X}_1$ ,  $\tilde{X}_2$ , and  $\tilde{X}_3$  is a deterministic function of the remaining two. More generally, let  $\tilde{X}_l^*$  be a leaf node in the causal graph  $\tilde{G}^*$ ; then each of the variables in  $\{\tilde{X}_l^*\} \cup \text{PA}(\tilde{X}_l^*)$ , denoted by  $\tilde{X}_k^*$ , is a deterministic function of the remaining variables.

To make it possible to identify the structure of  $\tilde{G}$  from the distribution of  $\mathbf{X}$ , in what follows we assume the distribution of  $\tilde{\mathbf{X}}^*$  satisfies the following assumption.

A0. The causal Markov condition holds for  $\tilde{G}$  and the distribution of  $\tilde{X}_i$  is faithful w.r.t.  $\tilde{G}$ . Furthermore, the distribution of  $\tilde{X}_i^*$  is *non-deterministically faithful* w.r.t.  $\tilde{G}^*$ , in the sense that if there exists  $\mathbf{S}$ , a subset of  $\{\tilde{X}_k^* : k \neq i, k \neq j\}$ , such that neither of  $\tilde{X}_i^*$  and  $\tilde{X}_j^*$  is a deterministic function of  $\mathbf{S}$  and  $\tilde{X}_i^* \perp\!\!\!\perp \tilde{X}_j^* \mid \mathbf{S}$  holds, then  $\tilde{X}_i^*$  and  $\tilde{X}_j^*$  (or  $\tilde{X}_i$  and  $\tilde{X}_j$ ) are d-separated by  $\mathbf{S}$  in  $\tilde{G}^*$ .

This non-deterministically faithfulness assumption excludes a particular type of parameter coupling in the causal model for  $\tilde{X}_i$ . In Figure 4 we give a causal model in which the causal coefficients are carefully chosen so that this assumption is violated: because  $\tilde{X}_3^* = a\tilde{X}_1^* + b\tilde{X}_2^*$  and  $\tilde{X}_4^* = 2a\tilde{X}_1^* + 2b\tilde{X}_2^* + E_4^*$ , we have  $\tilde{X}_4^* = 2\tilde{X}_3^* + E_4^*$ , implying  $\tilde{X}_4^* \perp\!\!\!\perp \tilde{X}_1^* \mid \tilde{X}_3^*$  and  $\tilde{X}_4^* \perp\!\!\!\perp \tilde{X}_2^* \mid \tilde{X}_3^*$ , which are not given by the causal Markov condition on  $\tilde{G}$ . We note that this non-deterministic faithfulness is defined for the distribution of the constructed variables  $\tilde{X}_i^*$ , not the measurement-error-free variables  $\tilde{X}_i$ . (Bear in mind their relationship given in (9).) This assumption is generally stronger than the faithfulness assumption for the distribution of  $\tilde{X}_i$ . In particular, in the causal model given in Figure 4, the distribution of  $\tilde{X}_i$  is still faithful w.r.t.  $\tilde{G}$ . Below we call the conditional independence relationship between  $\tilde{X}_i^*$  and  $\tilde{X}_j^*$  given  $\mathbf{S}$  where neither of  $\tilde{X}_i^*$  and  $\tilde{X}_j^*$  is a deterministic function of  $\mathbf{S}$  *non-deterministic conditional independence*.

Now we have two concerns. One is whether essential information of the CR-CAMME is identifiable from

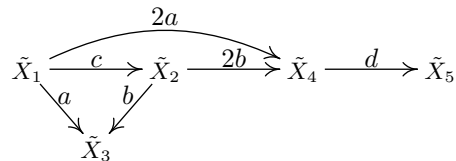


Figure 4: A specification of the causal model  $\tilde{G}$  in which  $\tilde{X}_i^*$  are not *non-deterministically faithful* w.r.t.  $\tilde{G}$  because of parameter coupling.

observed values of  $\mathbf{X}$ . The other is what information of the original CAMME, in particular, the causal model over  $\tilde{X}_i$ , can be estimated from the above identifiable information of the CR-CAMME. Although the transformation from the original CAMME to a CR-CAMME is straightforward, without further knowledge there does not necessarily exist a unique CAMME corresponding to a given CR-CAMME: first, the CR-CAMME does not tell us which nodes  $\tilde{X}_i$  are leaf nodes in  $\tilde{G}$ ; second, even if  $\tilde{X}_i$  is known to be a leaf node, it is impossible to separate the measurement error  $E_i$  from the noise  $\tilde{E}_i$  in  $E_i^*$ . Fortunately, we are not interested in everything of the original CAMME, but only the causal graph  $\tilde{G}$  and the corresponding causal influences  $\mathbf{B}$ . Accordingly, in the next section we will explore what information of the CR-CAMME is identifiable from the observations of  $\mathbf{X}$  and how to further reconstruct necessary information of the original CAMME.

In the measurement error model (1) we assumed that each observed variable  $X_i$  is generated from its own latent variable  $\tilde{X}_i$ . We note that in case multiple observed variables are generated from a single latent variable or a single observed variable is generated by multiple latent variables (see, e.g., Silva et al. (2006)), we can still use the CR-CAMME to represent the process. In the former case, certain rows of  $\mathbf{A}^{\text{NL}}$  are identical. For instance, if  $X_1$  and  $X_2$  are generated as noisy observations of the same latent variable, then in (7) the first two rows of  $\mathbf{A}_{\text{NL}}$  are identical. (More generally, if one allows different coefficients to generate them from the latent variable, the two rows are proportional to each other.) Let us then consider an example in the latter case. Suppose  $X_3$  is generated by latent variables  $\tilde{X}_1$  and  $\tilde{X}_2$ , for each of which there is also an observable counterpart. Write the causal model as  $X_3 = f(\tilde{X}_1, \tilde{X}_2) + E_3$  and introduce the latent variable  $\tilde{X}_3 = f(\tilde{X}_1, \tilde{X}_2)$ , and then we have  $X_3 = \tilde{X}_3 + E_3$ . The CR-CAMME formulation then follows.

#### 4 IDENTIFIABILITY IN THE LINEAR, NON-GAUSSIAN CASE

The CR-CAMME (7) has a form of the factor analysis model (FA) (Everitt, 1984), which has been a funda-

mental tool in data analysis. Accordingly, one can study the identifiability for CAMME by making use of the identifiability of FA, as reported by Zhang et al. (2017). The identifiability of FA, however, relies heavily on the assumption that there are a relatively large number of leaf variables in the causal graph  $\tilde{G}$  (Bekker & ten Berge, 1997), which seems rather strong. Moreover, it has been shown that second-order statistics usually is not informative enough to recover a unique causal model (Spirtes et al., 2001). Interestingly, we show that the identifiability results can greatly benefit from the non-Gaussianity assumption on the data. In this paper we make the following assumption on the distribution of  $\tilde{E}_i$ :

A1. All  $\tilde{E}_i$  are non-Gaussian.

We note that under the above assumption,  $\mathbf{A}^{\text{NL}}$  in (8) can be estimated up to the permutation and scaling indeterminacies (including the sign indeterminacy) of the columns, as given in the following lemma. This can be achieved by using overcomplete Independent Component Analysis (ICA) (Hyvärinen et al., 2001).

**Lemma 1.** *Suppose assumption A1 holds. Given  $\mathbf{X}$  which is generated according to (8),  $\mathbf{A}^{\text{NL}}$  is identifiable up to permutation and scaling of columns as the sample size  $N \rightarrow \infty$ .*

*Proof.* This lemma is implied by Theorem 10.3.1 in (Kagan et al., 1973) or Theorem 1 in (Eriksson & Koivunen, 2004).  $\square$

What information of the causal structure  $\tilde{G}$  can we recover? Can we apply existing methods for causal discovery based on LiNGAM, such as ICA-LiNGAM (Shimizu et al., 2006) and Direct-LiNGAM (Shimizu et al., 2011b), to recover it? LiNGAM assumes that the system is non-deterministic: each variable is generated as a linear combination of its direct causes plus a non-degenerate noise term. As a consequence, the linear transformation from the vector of observed variables to the vector of independent noise terms is a square matrix; ICA-LiNGAM applies certain operations to this matrix to find the causal model, and Direct-LiNGAM estimates the causal ordering by enforcing the property that the residual of regressing the effect on the root cause is always independent from the root cause.

In our case,  $\mathbf{A}^{\text{NL}}$ , the essential part of the mixing matrix in (8), is  $n \times r$ , where  $r < n$ . In other words, for some of the variables  $\tilde{X}_i^*$ , the causal relations are deterministic. (In fact, if  $\tilde{X}_k$  is a leaf node in  $\tilde{G}$ ,  $\tilde{X}_k^*$  is a deterministic function of  $\tilde{X}_k$ 's direct causes.) As a consequence, unfortunately, the above causal analysis methods based on LiNGAM, including ICA-LiNGAM

and Direct-LiNGAM, do not apply. We will see how to recover information of  $\tilde{G}$  by analyzing the estimated  $\mathbf{A}^{\text{NL}}$ .

We will show that some group structure and the group-wise causal ordering in  $\tilde{G}$  can always be recovered. Before presenting the results, let us define the following ordered group decomposition according to causal structure  $\tilde{G}$ .

**Definition 2 (ordered group decomposition).** *Consider the causal model  $\tilde{G}^*$ . Decompose all involved nodes into disjoint groups in the following way. First put all leaf nodes which share the same direct-and-only-direct cause in the same group; further incorporate the corresponding direct-and-only-direct cause in the same group. Here we say a node  $\tilde{X}_i^*$  is the "direct-and-only-direct" cause of  $\tilde{X}_j^*$  if and only if  $\tilde{X}_i^*$  is a direct cause of  $\tilde{X}_j^*$  and there is no other directed path from  $\tilde{X}_i^*$  to  $\tilde{X}_j^*$ . After forming all groups each of which involves at least one leaf node, each of the remaining nodes forms a separate group. **Each node is guaranteed to be in one and only one group.** We call the set of all such groups ordered according to the causal ordering of the non-leaf nodes in DAG  $\tilde{G}^*$  an ordered group decomposition of  $\tilde{G}^*$ , denoted by  $\mathcal{G}_{\tilde{G}^*}$ .*

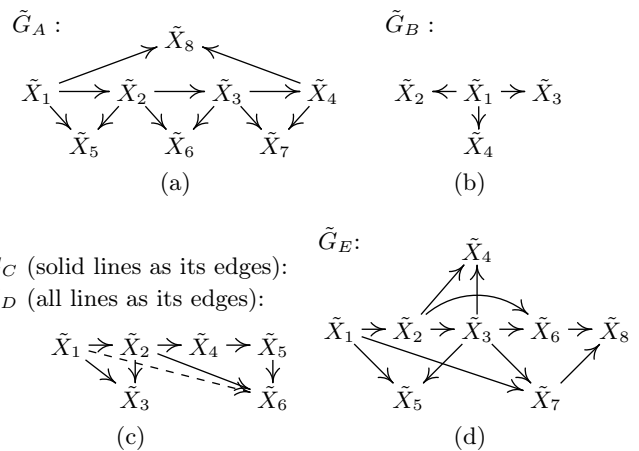


Figure 5: A set of causal DAGs  $\tilde{G}$  as illustrative examples. (a) DAG  $\tilde{G}_A$ . (b)  $\tilde{G}_B$ . (c) Two DAGs  $\tilde{G}_C$  and  $\tilde{G}_D$ . (d)  $\tilde{G}_E$ .

**Example Set 2** As seen from the process of ordered group decomposition, each non-leaf node is in one and only one ordered group, and it is possible for multiple leaf nodes to be in the same group. Therefore, in total there are  $(n - l)$  ordered groups. For example, for  $\tilde{G}_A$  given in Figure 5(a), a corresponding group structure for the corresponding  $\tilde{G}^*$  is  $\mathcal{G}_{\tilde{G}_A^*} = (\{\tilde{X}_1^*\} \rightarrow \{\tilde{X}_2^*, \tilde{X}_5^*\} \rightarrow \{\tilde{X}_3^*, \tilde{X}_6^*\} \rightarrow \{\tilde{X}_4^*, \tilde{X}_7^*, \tilde{X}_8^*\})$ , and for  $\tilde{G}_B$  in Figure 5(b), there is only one group:



$\mathcal{G}_{\tilde{G}_B^*} = (\{\tilde{X}_1^*, \tilde{X}_2^*, \tilde{X}_3^*, \tilde{X}_4^*\})$ . For both  $\tilde{G}_C$  and  $\tilde{G}_D$ , given in Figure 5(c), an ordered group decomposition is  $(\{\tilde{X}_1^*\} \rightarrow \{\tilde{X}_2^*, \tilde{X}_3^*\} \rightarrow \{\tilde{X}_4^*\} \rightarrow \{\tilde{X}_5^*, \tilde{X}_6^*\})$ .

Note that the causal ordering and the ordered group decomposition of given variables according to the graphical model  $\tilde{G}^*$  may not be unique (this will actually give rise to the possibility of distinguishing between the non-leaf and leaf node in the group, as shown next). For instance, if  $\tilde{G}^*$  has only two variables  $\tilde{X}_1^*$  and  $\tilde{X}_2^*$  which are not adjacent, both decompositions  $(\{\tilde{X}_1^*\} \rightarrow \{\tilde{X}_2^*\})$  and  $(\{\tilde{X}_2^*\} \rightarrow \{\tilde{X}_1^*\})$  are correct. Consider  $\tilde{G}^*$  over three variables,  $\tilde{X}_1^*, \tilde{X}_2^*, \tilde{X}_3^*$ , where  $\tilde{X}_1^*$  and  $\tilde{X}_2^*$  are not adjacent and are both causes of  $\tilde{X}_3^*$ ; then both  $(\{\tilde{X}_1^*\} \rightarrow \{\tilde{X}_2^*, \tilde{X}_3^*\})$  and  $(\{\tilde{X}_2^*\} \rightarrow \{\tilde{X}_1^*, \tilde{X}_3^*\})$  are valid ordered group decompositions.

We first present a procedure to construct the ordered group decomposition and the causal ordering among the groups from the estimated  $\mathbf{A}^{\text{NL}}$ . We will further show that the recovered ordered group decomposition is always asymptotically correct under assumption A1.

#### 4.1 Construction and Identifiability of ordered Group Decomposition

First of all, Lemma 1 tells us that  $\hat{\mathbf{A}}^{\text{NL}}$  in (8) is identifiable up to permutation and scaling columns. Let us start with the asymptotic case, where the columns of the estimated  $\mathbf{A}^{\text{NL}}$  from values of  $X_i$  are a permuted and rescaled version of the columns of  $\mathbf{A}^{\text{NL}}$ . In what follows the permutation and rescaling of the columns of  $\mathbf{A}^{\text{NL}}$  does not change the result, so below we just work with the true  $\mathbf{A}^{\text{NL}}$ , instead of its estimate.

$\tilde{X}_i^*$  and  $\tilde{X}_i$  follow the same causal DAG,  $\tilde{G}$ , and  $\tilde{X}_i^*$  are causally sufficient, although some variables among them (corresponding to leaf nodes in  $\tilde{G}^*$ ) are determined by their direct causes. Let us find the causal ordering of  $\tilde{X}_i^*$ . If there are no deterministic relations and the values of  $\tilde{X}_i^*$  are given, the causal ordering can be estimated by recursively performing regression and checking independence between the regression residual and the predictor (Shimizu et al., 2011b). Specifically, if one regresses all the remaining variables on the root cause, the residuals are always independent from the predictor (the root cause). After detecting a root cause, the residuals of regressing all the other variables on the discovered root cause are still causally sufficient and follow a DAG. One can repeat the above procedure to find a new root cause over such regression residuals, until no variable is left.

However, in our case we have access to  $\mathbf{A}^{\text{NL}}$  but not the values of  $\tilde{X}_i^*$ . Fortunately, the independence between regression residuals and the predictor can still be checked by analyzing  $\mathbf{A}^{\text{NL}}$ . Recall that  $\tilde{\mathbf{X}}^* = \mathbf{A}^{\text{NL}} \tilde{\mathbf{E}}^{\text{NL}}$ ,

where the components of  $\tilde{\mathbf{E}}^{\text{NL}}$  are independent. Without loss of generality, here we assume that all components of  $\tilde{\mathbf{E}}^{\text{NL}}$  are standardized, i.e., they have a zero mean and unit variance. Denote by  $\mathbf{A}_{i \cdot}^{\text{NL}}$  the  $i$ th row of  $\mathbf{A}^{\text{NL}}$ . We have  $\mathbb{E}[\tilde{X}_j^* \tilde{X}_i^*] = \mathbf{A}_{j \cdot}^{\text{NL}} \mathbf{A}_{i \cdot}^{\text{NL}\top}$  and  $\mathbb{E}[\tilde{X}_i^{*2}] = \mathbf{A}_{i \cdot}^{\text{NL}} \mathbf{A}_{i \cdot}^{\text{NL}\top} = \|\mathbf{A}_{i \cdot}^{\text{NL}}\|^2$ . The regression model for  $\tilde{X}_j^*$  on  $\tilde{X}_i^*$  is

$$\tilde{X}_j^* = \frac{\mathbb{E}[\tilde{X}_j^* \tilde{X}_i^*]}{\mathbb{E}[\tilde{X}_i^{*2}]} \tilde{X}_i^* + R_{j \leftarrow i} = \frac{\mathbf{A}_{j \cdot}^{\text{NL}} \mathbf{A}_{i \cdot}^{\text{NL}\top}}{\|\mathbf{A}_{i \cdot}^{\text{NL}}\|^2} \tilde{X}_i^* + R_{j \leftarrow i}.$$

Here the residual can be written as

$$\begin{aligned} R_{j \leftarrow i} &= \tilde{X}_j^* - \frac{\mathbf{A}_{j \cdot}^{\text{NL}} \mathbf{A}_{i \cdot}^{\text{NL}\top}}{\|\mathbf{A}_{i \cdot}^{\text{NL}}\|^2} \tilde{X}_i^* \\ &= \underbrace{\left( \mathbf{A}_{j \cdot}^{\text{NL}} - \frac{\mathbf{A}_{j \cdot}^{\text{NL}} \mathbf{A}_{i \cdot}^{\text{NL}\top} \mathbf{A}_{i \cdot}^{\text{NL}}}{\|\mathbf{A}_{i \cdot}^{\text{NL}}\|^2} \right)}_{\triangleq \alpha_{j \leftarrow i}} \tilde{\mathbf{E}}^{\text{NL}}. \end{aligned} \quad (10)$$

If for all  $j$ ,  $R_{j \leftarrow i}$  is either zero or independent from  $\tilde{X}_i^*$ , we consider  $\tilde{X}_i^*$  as the current root cause and put it and all the other variables which are deterministically related to it in the first group, which is a *root cause group*. Now the problem is whether we can check for independence between nonzero residuals  $R_{j \leftarrow i}$  and the predictor  $\tilde{X}_i^*$ . Interestingly, the answer is yes, as stated in the following proposition.

**Proposition 3.** *Suppose assumption A1 holds. For variables  $\tilde{\mathbf{X}}^*$  generated by (7), regression residual  $R_{j \leftarrow i}$  given in (10) is independent from variable  $\tilde{X}_i^*$  if and only if*

$$\left\| \alpha_{j \leftarrow i} \circ \mathbf{A}_{i \cdot}^{\text{NL}} \right\|_2 = 0, \quad (11)$$

where  $\circ$  denotes entrywise product.

So we can check for independence between the predictor and regression residual as if the values of  $\tilde{\mathbf{X}}^*$  were given. Consequently, we can find the root cause group.

We then consider the residuals of regressing all the remaining variables  $\tilde{X}_k^*$  on the discovered root cause as a new set of variables. Note that like the variables  $\tilde{X}_j^*$ , these variables are again linear mixtures of  $\tilde{E}_i$ . Repeating the above procedure on this new set of variables will give the second root cause and its ordered group. Applying this procedure repeatedly until no variable is left finally discovers all ordered groups following the causal ordering. The constructed ordered group decomposition is asymptotically correct, as stated in the following proposition. We denote by **OICA+Reg** the above two-stage procedure: we first apply overcomplete ICA to find an estimate of  $\mathbf{A}^{\text{NL}}$ , and then do regression and check for independence between the residuals and the current candidate root cause by analyzing  $\mathbf{A}^{\text{NL}}$ .

**Proposition 4. (Identifiable ordered group decomposition)** Let  $X_i$  be generated by the CAMME with the corresponding measurement-error-free variables generated by the causal DAG  $\tilde{G}$  and suppose assumptions A0 and A1 hold. The ordered group decomposition constructed by the above procedure is asymptotically correct, in the sense that as the sample size  $N \rightarrow \infty$ , if non-leaf node  $\tilde{X}_i$  is a cause of non-leaf node  $\tilde{X}_j$ , then the ordered group which  $\tilde{X}_i$  is in precedes the group which  $\tilde{X}_j$  belongs to. However, the causal ordering among the nodes within the same ordered group may not be identifiable.

The result of Proposition 4 applies to any DAG structure  $\tilde{G}$ . Clearly, the identifiability can be naturally improved if additional assumptions on the causal structure  $\tilde{G}$  hold. In particular, to recover information of  $\tilde{G}$ , it is essential to answer the following questions.

- Can we determine which nodes in an ordered group are leaf nodes?
- Can we find the causal edges into a particular node?

Below we will show that under rather mild assumptions, the answers to both questions are yes.

## 4.2 Identifying Leaf Nodes and Individual Causal Edges

If for each ordered group we can determine which variable is the non-leaf node, the causal ordering among the variables  $\tilde{X}_i^*$  is then fully known. The causal structure in  $\tilde{G}^*$  as well as the causal model can then be readily estimated by regression: for a leaf node, its direct causes are those non-leaf nodes that determine it; for a non-leaf node, we can regress it on all non-leaf nodes that precede it according to the causal ordering, and those predictors with non-zero linear coefficients are its parents. This way the structure can be estimated uniquely under Assumption A0, although whether the causal parameters in the causal model are uniquely identifiable is another issue for investigation.

Now the goal is to see whether it is possible to find out which variables in a given ordered group are leaf nodes; if all leaf nodes are found, then the remaining one is the (only) non-leaf node in the considered ordered group. Below we will show that it is possible to find leaf nodes by “looking backward” or “looking forward”; the former makes use of the parents of the variables in the considered group, and the latter exploits the fact leaf nodes do not have any child.

**Proposition 5. (Leaf node determination by “looking backward”)** Suppose the observed data were

generated by the CAMME where Assumptions A0 and A1 hold.<sup>1</sup> Let the sample size  $N \rightarrow \infty$ . Then if assumption A2 holds, leaf node  $O$  is correctly identified from observations of  $\mathbf{X}$  (more specifically, from the estimated  $\mathbf{A}^{\text{NL}}$  or the distribution of  $\tilde{\mathbf{X}}^*$ ).

A2. According to  $\tilde{G}^*$ , for leaf node  $O$  in the considered ordered group  $g^{(k)}$ , at least one of its parents is not a parent of the non-leaf node in  $g^{(k)}$  or some other leaf node in  $g^{(k)}$ .

**Example Set 3** Suppose Assumptions A0 and A1 hold.

- For  $\tilde{G}_A$  in Figure 5(a), assumption A2 holds for  $\tilde{X}_7^*$  and  $\tilde{X}_8^*$  in the ordered group  $\{\tilde{X}_4^*, \tilde{X}_7^*, \tilde{X}_8^*\}$ : each of them has a parent which is not a parent of the other; so both of them are identified to be leaf nodes from the estimated  $\mathbf{A}^{\text{NL}}$  or the distribution of  $\tilde{\mathbf{X}}^*$ , and  $\tilde{X}_4^*$  can then be determined as a non-leaf node.
- For  $\tilde{G}_B$ , we cannot detect which node is a leaf node or a non-leaf node.
- For both  $\tilde{G}_C$  and  $\tilde{G}_D$  in Figure 5(c),  $\tilde{X}_6^*$ , in the ordered group  $\{\tilde{X}_5^*, \tilde{X}_6^*\}$ , follows assumption A2 and can be found to be a leaf node from the matrix  $\mathbf{A}^{\text{NL}}$ ; accordingly,  $\tilde{X}_5^*$  has to be a non-leaf node.
- For  $\tilde{G}_E$  in Figure 5(d), assumption A2 holds for all leaf nodes,  $\tilde{X}_4^*$ ,  $\tilde{X}_5^*$ , and  $\tilde{X}_8^*$ , which can then be found to be leaf nodes.

We can also determine leaf nodes by looking at the relationships between the considered variables and the variables causally following them, as stated in the following proposition.

**Proposition 6. (Leaf node determination by “looking forward”)** Suppose the observed data were generated by the CAMME where Assumptions A0 and A1 hold. Then as the sample size  $N \rightarrow \infty$ , we can correctly identify the leaf node  $U$  in the considered ordered group  $g^{(k)}$  from values of  $\mathbf{X}$  if assumption A3 holds for it:

A3. For leaf node  $U$  in  $g^{(k)}$ , there exists at least one node causally following  $g^{(k)}$  that 1) is  $d$ -separated from  $U$  by a subset of variables in  $g^{(1)} \cup g^{(2)} \dots \cup g^{(k)} \setminus \{U\}$  which does not include all parents of  $U$  and 2) is a child of the non-leaf node in  $g^{(k)}$ .

<sup>1</sup>In this non-Gaussian case (implied by assumption A1), the result reported in this proposition may still hold if one avoids the non-deterministic faithfulness assumption and assumes a weaker condition; however, for simplicity of the proof we currently still assume non-deterministic faithfulness.

**Example Set 4** Let Assumptions A0 and A1 hold.

- For data generated by  $\tilde{G}_A$  in Figure 5(a), we already found  $\tilde{X}_4^*$  in ordered group  $\{\tilde{X}_4^*, \tilde{X}_7^*, \tilde{X}_8^*\}$  to be a non-leaf node because of Proposition 5. Proposition 6 further indicates that  $\tilde{X}_2^*$  (in group  $\{\tilde{X}_2^*, \tilde{X}_5^*\}$ ) and  $\tilde{X}_3^*$  (in group  $\{\tilde{X}_3^*, \tilde{X}_6^*\}$ ) are non-leaf nodes, and all leaf nodes are identified.
- For  $\tilde{G}_B$  in Figure 5(b), there is only one ordered group, and it does not provide further information by looking "backward" or "forward", and it is impossible to find the non-leaf node with Proposition 5 or 6.
- For both  $\tilde{G}_C$  and  $\tilde{G}_D$  in Figure 5(c),  $\tilde{X}_6^*$  was found to be a leaf node due to Proposition 5; thanks to Proposition 6, the other leaf node,  $\tilde{X}_3^*$ , was also detected. In particular, in  $\tilde{G}_C$ , for leaf node  $\tilde{X}_3^*$  both  $\tilde{X}_4^*$  and  $\tilde{X}_6^*$  satisfy the two conditions in Assumption A3; however, in  $\tilde{G}_D$ , for leaf node  $\tilde{X}_3^*$  only  $\tilde{X}_4^*$  satisfies them. All leaf nodes were successfully found.
- For  $\tilde{G}_E$  in Figure 5(d), Proposition 5 already allows us to identify all leaf nodes,  $\tilde{X}_4^*$ ,  $\tilde{X}_5^*$ , and  $\tilde{X}_8^*$ . The assumptions in Propositions 5 and 6 are not exclusive: Assumption A3 also holds for  $\tilde{X}_4^*$  (for it  $\tilde{X}_7^*$  satisfies the two conditions), we can alternatively identify this leaf node by making use of Proposition 6.

For contaminated data generated by any of  $\tilde{G}_A$ ,  $\tilde{G}_C$ ,  $\tilde{G}_D$ , and  $\tilde{G}_E$ , now we can find all leaf nodes in the measurement-error-free causal model. One can then immediately estimate the whole structure of the measurement-error-free model.

The above two propositions are about the identifiability of leaf nodes in the measurement-error-free causal model. By applying them to all leaf nodes, we have (sufficient) conditions under which the causal graph of  $\tilde{G}$  is fully identifiable.

**Proposition 7. (Full identifiability)** *Suppose the observed data were generated by the CAMME where Assumptions A0 and A1 hold. Assume that for each leaf node in  $\tilde{G}^*$ , at least one of the two assumptions, A2 and A3, holds. Then as the sample size  $N \rightarrow \infty$ , the causal structure  $\tilde{G}$  is fully identifiable from the observations with random measurement error.*

In the general case, the causal structure  $\tilde{G}$  might not be fully identifiable, and the above propositions may allow partial identifiability of the underlying causal structure. Roughly speaking, the ordered group decomposition is identifiable in the non-Gaussian case; with Propositions 5 and 6 one can further identify some leaf nodes as well as their parents.

## 5 CONCLUSION AND DISCUSSIONS

The measured values of variables of interest in various fields, including the social sciences, neuroscience, and biology, are often contaminated by measurement error. Unfortunately, the output of existing causal discovery methods is sensitive to the existence of measurement error, and it is desirable to develop causal discovery methods that can estimate the causal model for the measurement-error-free variables without using much prior knowledge about the measurement error. To this end, this paper investigates identifiability conditions for the underlying measurement-error-free causal structure given contaminated observations. We have shown that under appropriate conditions, the causal structure of interest is partially or even fully identifiable.

We formulated four assumptions. Assumption A0 is about the Markov condition and non-deterministic faithfulness assumption for causal model  $\tilde{G}^*$ . Assumption A1 is about the distribution of the underlying noise terms in the causal process. The remaining two are about particular types of "sparsity" of the underlying causal graph. We note that in principle, all assumptions except A0 are testable from the observed data. This suggests that it is possible to develop practical causal discovery methods to deal with measurement error that are able to produce reliable information at least in the asymptotic case. In addition, it is worth noting that some involved assumptions may be weakened. For instance, faithfulness is not required to find the correct ordered group decomposition, but just needed for detecting leaf nodes in the ordered groups. Suppose Assumptions A0 and A1 hold; we conjecture that the necessary and sufficient condition for the non-leaf node to be identifiable is that at least one of the two assumptions, A2 and A3, holds. To falsify or prove this conjecture is part of our future work.

It is worth noting that various kinds of background knowledge of the causal model may further help improve the identifiability of the measurement-error-free causal model. For instance, if one knows that all causal coefficients are smaller than one in absolute value, then the measurement-error-free causal model in Figure 5(b) is immediately identifiable from contaminated data. Our future research further includes 1) establishing identifiability conditions that allow cycles in the measurement-error-free causal model in light of ubiquity of cycles in causal models, 2) developing computationally efficient algorithms for causal discovery under measurement error based on the established theory, and 3) proposing efficient methods for particular cases where each measurement-error-free variable has multiple measured effects or multiplied measurement-error-free variables generate a single measured effect.

## References

- Bekker, P. A. and ten Berge, J. M. F. Generic global identification in factor analysis. *Linear Algebra and its Applications*, 264:255–263, 1997.
- Chickering, D. M. Optimal structure identification with greedy search. *Journal of machine learning research*, 3(Nov):507–554, 2002.
- Eriksson, J. and Koivunen, V. Identifiability, separability, and uniqueness of linear ICA models. *IEEE Signal Processing Letters*, 11(7):601–604, 2004.
- Everitt, B. S. *An introduction to latent variable models*. London: Chapman and Hall, 1984.
- Hyvärinen, A., Karhunen, J., and Oja, E. *Independent Component Analysis*. John Wiley & Sons, Inc, 2001.
- Kagan, A. M., Linnik, Y. V., and Rao, C. R. *Characterization Problems in Mathematical Statistics*. Wiley, New York, 1973.
- Kummerfeld, E., Ramsey, J., Yang, R., Spirtes, P., and Scheines, R. Causal clustering for 2-factor measurement models. In Calders, T., Esposito, F., Hüllermeier, R., and Meo, R. (eds.), *Proc. ECML PKDD*, pp. 34–49, 2014.
- Pearl, J. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge, 2000.
- Scheines, R. and Ramsey, J. Measurement error and causal discovery. In *Proc. CEUR Workshop 2016*, pp. 1–7, 2017.
- Shimizu, S., Hoyer, P.O., Hyvärinen, A., and Kerminen, A.J. A linear non-Gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7: 2003–2030, 2006.
- Shimizu, S., Hoyer, P. O., and Hyvärinen, A. Estimation of linear non-gaussian acyclic models for latent factors. *Neurocomputing*, 72:2024–2027, 2011a.
- Shimizu, S., Inazumi, T., Sogawa, Y., Hyvärinen, A., Kawahara, Y., Washio, T., Hoyer, P. O., and Bollen, K. Directlingam: A direct method for learning a linear non-gaussian structural equation model. *Journal of Machine Learning Research*, pp. 1225–1248, 2011b.
- Silva, R., Scheines, R., Glymour, C., and Spirtes, P. Learning the structure of linear latent variable models. *Journal of Machine Learning Research*, 7:191–246, 2006.
- Spirtes, P., Glymour, C., and Scheines, R. *Causation, Prediction, and Search*. MIT Press, Cambridge, MA, 2nd edition, 2001.
- Wiedermann, W., Merkle, E. C., and von Eye, A. Direction of dependence in measurement error models. *British Journal of Mathematical and Statistical Psychology*, 71:117–145, 2018.
- Zhang, K. and Chan, L. Extensions of ICA for causality discovery in the hong kong stock market. In *Proc. 13th International Conference on Neural Information Processing (ICONIP 2006)*, 2006.
- Zhang, K. and Hyvärinen, A. On the identifiability of the post-nonlinear causal model. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, Montreal, Canada, 2009.
- Zhang, K., Gong, M., Ramsey, J., Batmanghelich, K., Spirtes, P., and Glymour, C. Causal discovery in the presence of measurement error: Identifiability conditions. In *UAI 2017 Workshop on Causality: Learning, Inference, and Decision-Making*, 2017.



