Advances in Bayesian Machine Learning: From Uncertainty to Decision Making

Chao Ma

Computational and Biological Learning Lab
University of Cambridge

This dissertation is submitted for the degree of
Doctor of Philosophy

Homerton College

December 2021

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Chao Ma
December 2021

Advances in Bayesian Machine Learning: From Uncertainty to Decision Making

Chao Ma

Bayesian uncertainty quantification is the key element to many machine learning applications. To this end, approximate inference algorithms [176] are developed to perform inference at a relatively low cost. Despite the recent advancements of scaling approximate inference to "big model $\times$ big data" regimes, many open challenges remain. For instance, how to properly quantify the parameter uncertainties for complicated, non-identifiable models (such as neural networks)? How to properly handle the uncertainties caused by missing data, and perform learning/inference in a scalable way? Furthermore, how to optimally collect new information, so that missing data uncertainties can be further reduced, and better decisions can be made?

In this work, we propose new research directions and new technical contributions towards these research questions. This thesis is organized in two parts (theme A and theme B). In theme A, we consider quantifying model uncertainty under the supervised learning setting. To step aside some of the difficulties of parameter-space inference, we propose a new research direction called *function space* approximate inference. That is, by treating supervised probabilistic models as stochastic processes (measures over functions), we can now approximate the true posterior of the predictive functions by another class of (simpler) stochastic processes. We provide two different methodologies for function space inference and demonstrate that they return better uncertainty estimates, as well as improved empirical performances on complicated models.

In theme B, we consider the quantification of missing data uncertainty under the unsupervised learning setting. We propose a new approach for quantifying missing data uncertainty, based on deep generative models. It allows us to step aside from the computational burden of traditional methods, and perform accurate and scalable missing data imputation. Furthermore, by utilizing the uncertainty estimates returned by the generative models, we propose an information-theoretic framework for efficient, scalable, and personalized active information acquisition. This allows us to maximally reduce missing data uncertainty, and make improved decisions with new information.

This thesis is dedicated to my loving family and friends.

Acknowledgements

Without the help of many, this thesis would have been impossible to finish. I wish to express my deepest gratitude to my supervisor, Professor Jose Miguel Hernandez Lobato. In many ways, Miguel is the best supervisor that I can ever imagine. He is such a knowledgeable, humble, patient, and lovely person to work with, and I cannot thank him enough for recognizing my potential and enthusiasm and taking me as one of his first PhD students. He convinced me to explore original research ideas just for pure curiosity, which led to many unexpected successes. Thanks to Miguel, these four years become one of the most memorable experiences in my life, and being his student is such a great privilege.

I especially thank Yingzhen Li (Imperial College) and Cheng Zhang (Microsoft Research). Yingzhen's deep insights and enthusiasm inspired me to pursue PhD research in this field, despite all the difficulties that I had to overcome. She has been a fantastic mentor during my PhD, who always has time for research conversations, that are necessary for pushing forward my PhD. Cheng Zhang deserves special thanks for hosting me at Microsoft Research Cambridge, and providing me with the fantastic opportunity to conduct novel research in such a vibrant and exciting environment. She is a great person to work with, and constantly encourages me to step out of my comfort zone. It is my absolute pleasure for being able to participate in project Azua (now Causica) since its early days and witness how our theoretical research makes a difference in real-life applications.

My amazing collaborators and fellow PhD students have been a rich source of inspiration. In particular, I would like to thank my fellow colleagues from both CBL group at the University of Cambridge, and the Causica group at Microsoft Research Cambridge. It has been a great pleasure to be part of those intelligent research forces and learn from like-minded Bayesian cool kids in town. I am grateful to Prof. Carl Henrik Ek and Prof. Jes Frellsen, who kindly agreed to serve as my viva examiners and provided honest feedback on polishing this thesis.

Finally, the support from my family and friends has always been the pillar of my PhD. I will dedicate this thesis to my parents and my maternal grandparents, for everything that I have accomplished.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

“ *Fear comes from uncertainty; we can eliminate the fear within us when we know ourselves better.* ”

Bruce Lee,

UNCERTAINTIES are everywhere in the world. What is the weather like tomorrow? What are the chances of an earthquake in Japan? What is the valuation of a company? Is a bitcoin crash coming? Which movies will people watch? Which restaurant will people go to? All these questions involve processing uncertainties and making decisions given imperfect data or knowledge of the world.

In this chapter, we motivate the importance of performing inference under uncertainty in machine learning, which is the main focus of the thesis. Specifically, we first review the modern axiomatization systems of rational behaviors (i.e., von Neumann-Morgenstern [347] and Savage’s decision theory [296]) (Section 1.1), and show that (Bayesian) uncertainty is the only way that can mathematically induce rational decision-making behaviors [102]. Therefore, we argue that if rationality is the common principle behind both natural and machine intelligence systems, then the key to replicating human decision-making behavior relies on algorithms that can perform Bayesian inference efficiently and accurately (Section 1.2). Finally, we describe the challenges of Bayesian inference under model uncertainties and missing data uncertainties (Section 1.3), and summarize the contributions of this thesis towards addressing those challenges (1.4).

1.1 Uncertainty: the only consequence of rationality

One of the central tasks of machine learning research is to understand the theoretical and practical foundations of intelligent behaviors, such as learning, reasoning, and decision making. This allows us to replicate and automate the decision-making process of experts and thus helps us to understand massive datasets, even when we do not possess the governing theories that could have generated them.

Rationality However, it is often not clear what are the implications of "creating machines like humans". Over the decades, researchers have considered the connections/resemblances between natural and artificial intelligent systems from different perspectives, including neuroscience, cognitive science, psychology, engineering, etc. In this thesis, we instead begin with the mathematical characterizations of *rationality* (and rational decision making), as the common denominator among natural and artificial intelligent systems.

von Neumann-Morgenstern theory The modern axiomatization system for rational behaviors, known as the von Neumann-Morgenstern (VNM) expected utility theory, was formulated by von Neumann and Morgenstern back in 1944 [347]. VNM theory states that, when facing risky choices, a rational decision-maker (i.e., having VNM rational preferences over actions, see [83] e.g. for details), will act *as if* he/she is maximizing the expectation of *some* utility function under uncertainty (i.e., expected utility maximization, EUM). The beauty of VMN theory is that we do not need to know the psychological and biological details of how decision-makers feel and process uncertainties *per se*, to make predictions on who they would behave. The VMN theory implies that (rational) humans are nothing but animals of uncertainties. Conscious or not, when making decisions in a risky environment, we are taking account of the uncertainties of possible outcomes.

Savege's utility theory The VMN theory was later extended by *Savage's axiomatic system* [296] to the case where decision-makers do not know the ground truth probability measure of possible events. In this case, a rational decision-maker will behave as if he/she is maximizing the expected utility under his/her *subjective (Bayesian) assignment* of uncertainties. In other words, as argued by J. Harsanyi, "Bayesian criterion of EUM is the only decision criterion consistent with rationality" [102, 101]. In modern machine learning, such utility maximization problem is often formulated as Bayesian expected utility maximization (Bayesian EUM).

The theories of VNM and Savage (along with [349, 8], etc.) become the foundation of modern Bayesian decision theory [24]: if we agree that rationality shall be the common principle behind intelligent systems, then the results of axiomatizations of rationalities would strongly favor the Bayesian treatment of uncertainties and decision making in intelligent systems, i.e., the Bayesian EUM formulation. Next, we will describe Bayesian EUM, and how uncertainties play a role in Bayesian EUM in the context of machine learning.

1.2 Bayesian expected-ultility maximization

To understand how uncertainties and Bayesian statistics play a role in the context of machine learning, let us describe the Bayesian EUM problem in the notation of modern Bayesian decision theory. In this section, we follow the formulation of [235, 254]. The general idea of Bayesian EUM is, a decision maker will choose his/her actions among a set of available actions, which will cause certain consequences depending on the random state of the world. The decision maker will have certain utility over different consequences, which assigns preferences over different actions and forms the basis for decision making. Using a mathematical language, any Bayesian decision-theoretic problem can be specified by the following quantities:

- **The space of the states**, $\boldsymbol{\Theta}$, with $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ being the state of the world that is usually unobservable to the decision maker. We assume $\boldsymbol{\theta}$ follows some probability distribution (called prior) $p(\boldsymbol{\theta})$ over $(\boldsymbol{\Theta}, \mathcal{E})$. Here, $\mathcal{E}$ is a σ-algebra on $\boldsymbol{\Theta}$, usually referred as the space of events.
- **The space of observable variables**, $\mathcal{V}$, the elements of which are denoted by $\mathbf{v}$. These represent the information available for decision making. We assume $\mathbf{v}$ is generated according to the conditional probability function $p(\mathbf{v}|\boldsymbol{\theta})$ of $\mathbf{v}$ given the state $\boldsymbol{\theta}$.
- **The set of available actions** $\mathcal{A}$, each element of which $a(\cdot) \in \mathcal{A}$ is a *function* from $\boldsymbol{\Theta}$ to $\mathcal{C}$, where $\mathcal{C}$ is the space of consequences. That is, depending on the state of the world, an action will cause different consequences.
- **The loss function** $L(c)$, which assigns a real number for different consequences $c \in \mathcal{C}$. Since c is fully determined by both the state $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ and the action $a(\cdot) \in \mathcal{A}$, we can also view the loss L as a function of $\boldsymbol{\theta}$ and a. That is, we can rewrite the loss function as $L(\boldsymbol{\theta}, a(\theta))$ (since a is a function of θ), or simply $L(\boldsymbol{\theta}, a)$

Then, the rational decision making solution is obtained by the solving the following (conditional) Bayesian EUM problem:

$$a^{\star} = \arg\max_{a \in \mathcal{A}} \mathbb{E}_{p(\boldsymbol{\theta}|\mathbf{v})} L(\boldsymbol{\theta}, a(\boldsymbol{\theta})), \tag{1.1}$$

where the conditional distribution $p(\boldsymbol{\theta}|\mathbf{v})$ represents the decision maker's *belief* on the state of nature conditioned on observable information $\mathbf{v}$. $p(\boldsymbol{\theta}|v)$ is given by Bayes rule,

$$p(\boldsymbol{\theta}|\mathbf{v}) = \frac{p(\boldsymbol{\theta})p(\mathbf{v}|\boldsymbol{\theta})}{\int_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} p(\boldsymbol{\theta})p(\mathbf{v}|\boldsymbol{\theta})d\boldsymbol{\theta}}, \tag{1.2}$$

which quantifies the *uncertainty/risks* of the state $\boldsymbol{\theta}$, given the currently observed information. Common sources of uncertainty may include [74]:

- **Noisy data**: for example uncertainty from measurement noise; many real-world datasets are inevitably corrupted by noise due to imperfection of measurement. In this case, the observable information $\mathbf{v}$ in the Bayesian EUM problem corresponds to the noisy version of $\boldsymbol{\theta}$. That is, $\mathbf{v} = \tilde{\boldsymbol{\theta}}$, and the measurement noise is given by $p(\tilde{\boldsymbol{\theta}}|\boldsymbol{\theta})$, which is usually irreducible.

- **Model uncertainty**: given a certain set of observations, there may exist many models with different configurations of structures/parameters that give similar performance over training datasets, but behave differently in out-sample prediction. This typically corresponds to ill-posed tasks where the given dataset does not determine the uniqueness of the solution. This is also known as model non-identifiability in engineering problems.

- **Missing data**: uncertainty due to absence of certain entries in observable data records $\mathbf{v}$. This is common in data from responses to questionnaires, where participants may refuse to answer certain questions.

Remark (parameter uncertainty). A special case of model uncertainty is parameter uncertainty, which is common in *prediction problems*. In such roblems, we have a predictive model parameterized by $p_{\boldsymbol{\theta}}(y|x)$, where $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ is the parameter space, $x \in \mathcal{X}$ is the input location (features), and $y \in \mathcal{Y}$ is the predictive output (response). The parameter uncertainty is usually quantified by the posterior, $p(\boldsymbol{\theta}|\mathbf{v} = \mathcal{D})$, where $\mathcal{D} = \{(x_i, y_i)\}_{i=1:N}$ is the set of training data.

Remark (example of missing data uncertainty). Consider a medical diagnosis scenario. Given a patient experiencing a lip sore, we are uncertain of the cause of such a symptom. We might think of several possible diseases such as erythema multiforme, cold sores, syphilis, or even skin cancer, but certainly not a heart attack. However, at the beginning, most information regarding other symptoms and/or medical tests is missing and we may not be able to distinguish different diseases. This means that we have to (equally) consider multiple plausible explanations at the same time. To reduce such uncertainty, doctors will commonly ask to conduct more medical tests and/or ask more questions regarding symptoms.

aleatoric and epistemic uncertainty The noisy data example above is a case of *aleatoric uncertainty*, which accounts for variability in the outcome due to inherent randomness. The other two examples (model uncertainty and missing data) are cases of *epistemic uncertainty*, also known as *systematic uncertainty*, which accounts for uncertainties due to *ignorance*, i.e., lack of knowledge. In this thesis, we will focus on quantifying and reducing both epistemic and aleatoric uncertainty.

Remark (Generality of Bayesian EUM). Note that the setting of the Bayesian EUM problem (1.1) is quite general: common machine learning problems such as prediction, classification, statistical hypothesis testing, statistical parameter estimation, Bayesian inference, experimental design, etc. can all be described as special cases of Equation (1.1):

- **Point estimation.** In this case $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ would describe the unknown parameter of interest and $\mathcal{A} = \boldsymbol{\Theta}$. For convenience, $L(\boldsymbol{\theta}, a)$ can be the square loss $(\boldsymbol{\theta} - a)^2$, and the optimal action corresponds to the mean of the posterior, $p(\boldsymbol{\theta}|\mathbf{v})$. Other commonly used loss function include absolute error $|\boldsymbol{\theta} - a|$, which induces the median of the posterior.

- **Bayes interval.** In statistical inference, it is standard to obtain an estimation on $\boldsymbol{\theta}$, but instead of a point estimate, we would like to estimate an interval (a_1, a_2) in $\boldsymbol{\Theta}$. One example of framing interval estimation as a Bayesian EUM problem is illustrated by [254], where the loss is given by

$$L(\boldsymbol{\theta}, a_1, a_2) = L_1 \frac{a_2 - a_1}{2} + L_2 \left(\boldsymbol{\theta} - \frac{a_2 + a_1}{2}\right) \frac{2}{a_2 - a_1}.$$

  The optimal interval in this case is given by $\mathbb{E}[\boldsymbol{\theta}|\mathbf{v}] \pm \sqrt{\frac{L_1}{L_2}\mathbb{V}[\boldsymbol{\theta}|\mathbf{v}]}$.

- **Prediction problem.** Given a predictive model parameterized by $p_{\boldsymbol{\theta}}(y|x)$, our goal is to determine the best prediction y^* given a test input x^* and a training dataset, $\mathcal{D} = \{(x_i, y_i)\}_{i=1:N}$. The action space is $\mathcal{A} = \mathcal{Y}$, the action is defined by $a_y(\boldsymbol{\theta}) = y$ and the observable information is given by $\mathbf{v} = (\mathcal{D})$. At location x^*, the loss can be described by $L_{x^*}(\boldsymbol{\theta}, a_y) = p_{\boldsymbol{\theta}}(y|x^*)$. The Bayesian EUM problem is given by

$$y^\star = \arg\max_{a \in \mathcal{A}} \mathbb{E}_{p(\boldsymbol{\theta}|\mathcal{D})} p_{\boldsymbol{\theta}}(a|x^*),$$

which corresponds to the mode of the posterior predictive distribution $p(y|x^*, \mathcal{D})$.

- **Approximate inference.** In this case, the action $a(\cdot)$ is a distribution that assigns a probability density value to each $\boldsymbol{\theta}$ and represented by the approximate distribution $q(\boldsymbol{\theta})$. The loss can be defined as $L(\boldsymbol{\theta}, a) = \log q(\boldsymbol{\theta})$, and the Bayesian EUM problem can be given by

$$q^\star = \arg\max_{q} \mathbb{E}_{p(\boldsymbol{\theta}|\mathbf{v})} \log q(\boldsymbol{\theta}).$$

The optimal solution of which is the exact posterior $p(\boldsymbol{\theta}|\mathbf{v})$.

See [254] for more details and more examples of Bayesian EUM problem.

Takeaway So far, we have seen the following: 1), the theoretical results mentioned in Section 1.1 show that Bayesian EUM is the consequence of rationality, which could be the common principle behind natural and machine intelligent systems. 2), being able to compute $p(\boldsymbol{\theta}|\mathbf{v})$ is key in the Bayesian EUM problem, which includes many important tasks in machine learning and statistical inference as special cases. Therefore, this motivates us to develop effective techniques to: 1), quantify model uncertainty and missing data uncertainty (i.e., evaluating $p(\boldsymbol{\theta}|\mathbf{v})$) efficiently; and 2), use the information of uncertainty to correctly guide our decision-making process. My thesis will be focusing addressing these two questions in specific scenarios.

1.3 Challenges: model uncertainty and missing data uncertainty

From the unifying view of Bayesian EUM, uncertainty is captured in the decision maker's belief about the state of the system when conditioning on observed information. This is given by the posterior distribution $p(\boldsymbol{\theta}|\mathbf{v})$. The key limitation of this is that it often leads to computationally intractable solutions. This is due to the fact that calculating the posterior $p(\boldsymbol{\theta}|\mathbf{v}) = \frac{p(\boldsymbol{\theta})p(\mathbf{v}|\boldsymbol{\theta})}{\int_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} p(\boldsymbol{\theta})p(\mathbf{v}|\boldsymbol{\theta})d\boldsymbol{\theta}}$ requires solving the integration problem $\int_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} p(\boldsymbol{\theta})p(\mathbf{v}|\boldsymbol{\theta})d\boldsymbol{\theta}$

which is generally analytically and computationally intractable [176] except for a few special cases.

Approximate inference techniques [176] are often used to address this issue. This is done by finding another distribution, $q(\boldsymbol{\theta})$, that is easy to evaluate, while being able to *approximate* the ground truth posterior $p(\boldsymbol{\theta}|\mathbf{v})$ with low approximation error (subject to certain error criterion). The area of approximate inference has witnessed extraordinary advances in the past decade. However, highly challenging problems still remain, especially when dealing with *model uncertainties* and *missing data*.

1.3.1 Challenge I: accurate and scalable approximate inference for supervised learning models

With the rise of modern deep learning methods in the past ten years [22, 115, 288, 162, 162, 314, 62, 105, 58, 35], it becomes a daily routine to train and deploy huge deep neural networks with up to billions of parameters [35] and trained on billions of high-dimensional data points. While there has been tremendous recent advances of scaling Bayesian approximate inference to “big model $\times$ big data" regimes [75, 74, 111, 29, 178, 180, 177], inference in parameter space $p(\boldsymbol{\theta}|\mathcal{D})$ is still difficult [198, 330, 357, 127, 95] and often pathological [70]. Specifically, the following is a list with some challenging problems in this area:

- i) over-parameterized models like neural networks are often unidentifiable, meaning that there exists different parameter settings that leads to similar predictive functions. For example, there are symmetric modes in the posterior distribution of neural network weights that gives identical predictive distributions [206, 198]. These issues makes the distribution $p(\boldsymbol{\theta}|\mathcal{D})$ highly complicated and cannot be easily captured by simple approximations such as the mean-field approximation [357].
- ii) Some models involve the usage of so-called “*implicit distributions*” [179, 198], which are probability measures assigned implicitly by the specification of a process that generates samples from them. One of the most well-known implicit distributions is the generator in a generative adversarial net (GAN) [88], which transforms isotropic noise into high dimensional data and which can then be used to specify flexible distributions for $p(\boldsymbol{\theta})$. For such models, the evaluation of $p(\boldsymbol{\theta})$ is intractable, which adds additional difficulties when doing Bayesian inference.
- iii) In many models such as those defined by using neural networks as components, it is hard to interpret the implication of the choice of prior. That is, it is unclear

what effect the prior $p(\boldsymbol{\theta})$ over the weights $\boldsymbol{\theta}$ will have in the resulting predictive distribution.

All those challenging problem listed above present unignorable difficulties for performing inference in the parameter space. This motivates us to propose new methodologies to address these issues. We will provide a more in-depth analysis of those challenges in Chapter 3.

1.3.2 Challenge II: Unsupervised learning and inference under the presence of missing data

Missing data is an obstacle in many data analysis problems, which may seriously compromise the performance of machine learning models. Being able to successfully handle missing data is the key to understanding the structure of real-world data. Suppose we have a system comprising a set of observable variables, $\mathbf{x} = (x_1, x_2, ..., x_D)$ (also referred as *complete data*). Due to the presence of missing mechanisms, for each data point, we may only be able to observe a *subset* of variables (denoted by $\mathbf{x}_O \subset \mathbf{x}$) each time. The rest of the variables (denote by $\mathbf{x}_U = \mathbf{x} \setminus \mathbf{x}_O$) remain unknown. This induces uncertainty due to missing data, represented by the posterior distribution $p(\mathbf{x}_U|\mathbf{x}_O)$, which presents two challenging questions:

- i) Unsupervised learning under missing data. That is, how to specify/learn the distribution of the complete data $p(\mathbf{x})$, given only partially observed data points with missing values? Furthermore, can this be done in the large data/large model regime? This problem is often referred to as *unsupervised learning* under missing values.

- ii) Efficient missing data imputation. There are many possible partitions of the complete data into missing/observed subsets, U, O. For D observable variables, there exists 2^D different combinations for the observed subset O. Therefore, there are 2^D different posterior distributions of the form $p(\mathbf{x}_U|\mathbf{x}_O)$ that may need to be computed, with each of them requiring exact/approximate Bayesian inference. How can we efficiently address this significant computational challenge?

We will provide a more detailed analysis of challenges for missing data uncertainty quantification in Chapter 6.

1.3.3 Challenge III: Replicating human expert's ability to collect high-value information

Human experts are not only good at evaluating the level of uncertainty in certain decision-making problems [157, 158] but also at actively collecting the additional information *that is most useful* for reducing those uncertainties [245, 246]. Consider again the medical diagnosis example that we described in Section 1.2:

Remark (example of missing data uncertainty, cont.) When a patient experiencing a lip sore shows up at the hospital, a doctor usually knows little about the patient's current status. Therefore, he/she is uncertain of the cause of such symptoms. However, an experienced doctor might actively *acquire information* to reduce such uncertainty. He/she would first evaluate the current situation, investigate what are the possible scenarios, and then he/she will ask questions accordingly. He/she might ask whether the patient has a mouth ulcer and whether he/she is experiencing a high temperature. Based on the answers, the doctor can evaluate again the possibilities of each possible outcome. For example, if the answers are positive and negative, respectively, then the doctor might conclude that the patient is most likely having a cold sore. Meanwhile, it is also equally important to acknowledge other high-risk possibilities such as skin cancers are not excluded, at least not without further medical tests.

The above scenario highlights one of the most important applications of Bayesian approaches, i.e., how to answer *what does our model know*? or equivalently, *how can we know if the model does not know*? Bayesian approaches provide a principled answer: when there is not enough information to make predictions/decisions, the estimated uncertainty level (either model uncertainty $p(\boldsymbol{\theta}|\mathbf{v})$ and/or missing data uncertainty $p(\mathbf{x}_U|\mathbf{x}_O)$) should be quite high. This will indicate that the model does not quite know what it is doing. Then, we must either *refuse* to make decisions (hand over to human experts), or proceed to *collect more information* (acquire and add more variables to $\mathbf{x}_O$), until we feel significantly more certain. This inspires us to use Bayesian approaches for automating the human expert's ability to collect high-value information.

1.4 Outline of this thesis

The thesis begins with a introductory Chapter (Chapter2) that provides basic foundations for Bayesian machine learning and approximate inference. The rest of the thesis focuses on two themes: theme A addresses Challenge I and theme B addresses Challenge II and III, which explores topics in supervised learning and unsupervised learning, respectively.

1. **Theme A (supervised learning): proposing a new research direction—variational inference in *function space*.** In this theme, we try to address Challenge I, i.e., obtaining efficient and accurate *model uncertainty* in supervised learning problems. We follow a recently fresh idea in approximate inference: performing Bayesian inference in the space of functions as opposed to the space of parameters. We treat supervised probabilistic models as stochastic processes (i.e., measures over functions); then, we can apply the idea of variational inference, by approximating the true posterior stochastic processes by another class of (simpler) stochastic processes. This theme begins with Chapter 3, a short chapter that discusses the limitations of parameter-space inference, and motivates the necessity of function-space inference. Then, the main technical contributions of Theme A are presented in the following chapters:

   - **Chapter 4: Variational implicit processes.**

     Contributions This chapter is one of the first works that demonstrates the idea of performing approximate inference for modern probabilistic models in function space. Our key contribution is twofold. First, we introduced a flexible class of stochastic process priors, namely the variational implicit processes, for the sake of Bayesian modelling. Secondly, we proposed a new inference method for this type of priors, based on Gaussian process (GP) approximations in function space.

     Outline In this chapter, we first review the basics of Gaussian processes (GPs) in Bayesian machine learning. We argue that one of the key ideas of GPs is that they directly specify prior and posterior distributions over functions, which could be useful to address some of the aforementioned challenges in model uncertainty estimation. We address the key question of how to extend such function space inference ideas to Bayesian *parametric* models such as Bayesian neural networks. Then, we show how to perform efficient function-space inference using approximate Bayesian inference techniques. We develop the variational implicit process (VIP) as a solution. Similar to Gaussian processes (GPs), in implicit processes (IPs) an implicit multivariate prior is placed over any finite collections of random variables. Based on Gaussian process approximations, a novel and efficient approximate inference algorithm for IPs is then derived based on a generalized version of the wake-sleep algorithm. We finally perform experiments to demonstrate the effectiveness of VIPs on a number of tasks on which VIPs return better uncertainty estimates and superior performance than other existing inference methods.

- **Chapter 5: Functional variational inference.**

  Contributions The method proposed in Chapter 4 is limited in the sense that it only performs Gaussian approximations in function space, and it uses wake-sleep updates which does not correspond to a coherent optimization objective. In this chapter, we further generalize the idea proposed in Chapter 4, to a more general method that performs non-Gaussian approximations under the framework of variational inference. This Chapter has three main contributions: first, we propose a new functional divergence measure between stochastic processes, which can be used for variational inference in function space. We also show that it is able to avoid some of the pathologies found in the original KL divergence. Second, we propose a new class of flexible variational family for posterior approximation in function space, namely the stochastic process generators (SPGs). SPGs are non-Gaussian generalizations of the GP approximations used in Chapter 4, and can serve as a very flexible variational family in VI. Finally, we propose that our proposed functional divergence can be estimated efficiently using SPG approximations, which achieves a significant speed-up against the gradient estimators commonly used in variational inference.

  Outline In this chapter, we propose a new functional-space inference method, called Functional Variational Inference (FVI). We first present the general framework of performing variational inference in function space [330], which is solely based on the idea of functional Kullback-Leibler divergence. We also review the recent work [39] that analyzed the pathologies of Kullback-Leibler divergence in function space. To partially address this issue, we propose to minimize a new divergence measure between the variational distribution and the posterior process, namely the gird-functional divergence, which will be more well-behaved than the functional KL-divergence typically used in the literature. We further derive the evidence lower bound (ELBO) in function space, based on the proposed grid-functional divergence. Based on this framework, we proposed to parameterize the variational approximation by stochastic process generators (SPGS), which is a class of flexible, non-Gaussian stochastic process based on variational autoencoders (VAEs). We further show how the proposed divergence measure can be estimated using analytic solutions and mini-batch sampling. Finally, we provide empirical examples to showcase the effectiveness of our approach.

2. **Theme B (unsupervised learning): potential solutions to Challenges II and III, i.e., learning, inference, and high-value information acquisition under the presence of missing data.** Our contributions regarding this topic are presented in chapters 7 and 8 of this thesis. The theme begins with Chapter 6, which introduces basic concepts regarding missing data in machine learning, and proposes to apply deep generative models to quantify missing data uncertainty, perform missing value imputation, and actively acquire information that maximally reduces missing data uncertainty. The main contributions of Theme B are presented in the following chapters:

   - **Chapter 7: Efficient dynamic discovery of high-value information with partial VAE.**

     Contributions In this Chapter, we propose a principled framework, named EDDI (Efficient Dynamic Discovery of high-value Information), for learning, inference, and high-value information acquisition with missing data. We have two major contributions: i), we develop a new probabilistic model, namely, the partial variational autoencoder (Partial VAE), to capture missing data uncertainty [260, 374]. The Partial VAE, as a probabilistic framework in the presence of missing data, is highly scalable, and serves as the base for the EDDI framework; and ii), we propose an information-theoretic framework for efficient, scalable, and personalized active information acquisition. By harnessing the missing data uncertainties modeled by the partial VAEs, we come up with an acquisition function that is implemented in practice using novel efficient approximations. This allows us to actively select the unobserved variable which contributes most to a prediction task. This function is particularly useful in customer surveys and health assessments.

     Outline In this chapter, we begin by reviewing the necessity of proposing a framework for learning, inference, and active information acquisition under missing data uncertainties. We argue that this task boils down to two components:

an unsupervised learning model that captures missing data uncertainties, and a key mechanism to make decisions of what information to collect next. To this end, we review related works in the literature, and propose to create a modern solution based on recent advances in deep learning and Bayesian approximate inference. We first introduce a new partial amortized inference method for generative models under partially observed data. This allows us to extend the variational autoencoder (VAE) [150, 272] to account for partial observations. The resulting method, which we call the Partial VAE, is inspired by the set formulation of the observed data [260, 374], which can be used to perform learning and inference efficiently under missing data. Then, we proceed to describe an information-theoretic acquisition function which yields a novel variable-wise active information collection method. Based on the Partial VAE, we actively select the unobserved variable which contributes the most to the prediction task. This acquisition function does not have an analytical solution. Therefore, we derive a novel estimator based on Monte Carlo sampling and latent-space approximations. We demonstrate the performance of our framework in various settings and apply it in real-life healthcare scenarios.

- **Chapter 8: Identifiable generative models under missing not at random data**.

  Contributions The framework proposed in Chapter 7 implicitly assumes that the missing data follows a missing at random (MAR) assumption. That is, the probability of a variable being missing only depends on the observed data, which is quite restrictive. In this Chapter, we further extend the work in Chapter 7 to general missing not at random (MNAR) assumptions (the causes of a variable being missing is unobserved). Unfortunately, under MNAR, most commonly used deep generative models lack model identifiability guarantees, which will introduce biases when performing missing data imputation. Our main contribution of this chapter is filling in this gap by systematically analyzing the identifiability of generative models under MNAR. Furthermore, we propose a practical deep generative model which can provide identifiability guarantees under mild assumptions for a wide range of MNAR mechanisms.

  Outline In this chapter, we begin by reviewing three basic types of missing data mechanism: missing completely at random (MCAR), missing at random (MAR), and missing not at random (MNAR). We argue that model identifiability is a main obstacle for unsupervised learning under MNAR, and is overlooked by many modern deep generative models. To this end, we provide a theoretical analysis of identifiability for generative models under different MNAR assumptions. More specifically, we propose a set of sufficient conditions, under which the ground truth parameters can be uniquely identified via optimizing the partial ELBO proposed in Chapter 7. We also demonstrate how the assumptions can be slightly relaxed under model mis-specification. Based on our theoretical result, we propose a practical algorithm model based on identifiable variational auto-encoders, which enables us to apply flexible deep generative models in a principled way, even in the presence of MNAR data. This Chapter ends with an empirical evaluation of the proposed model across different tasks.

To summarize, the outline of the thesis is as follows:

- Chapter 2: introduces the basic ideas of Bayesian machine learning, reviews prior work in approximate inference, and provides introductions to Bayesian neural networks, Gaussian processes, and generative models.
- Chapter 3: discusses the limitations of parameter space inference, and motivates function space inference from the perspective of model identifiability and posterior consistency.
- Chapter 4: presents the variational implicit process (VIP), a function-space inference method based on GP approximations and the wake-sleep method.
- Chapter 5: presents functional variational inference (FVI), an extension to the VIP method that uses non-GP approximations together with a newly proposed functions-space inference objective.
- Chapter 6: discusses the backgrounds of missing data uncertainty and the challenges of performing unsupervised learning under missing data using generative models.
- Chapter 7: presents EDDI (Efficient Dynamic Discovery of high-value Information), which performs learning, inference, and high-value information acquisition under MAR (missing at random) missing values.
- Chapter 8: extends the work in Chapter 7 to more general missing not at random (MNAR) assumptions and studies the model identifiability of deep generative models under MNAR.

- Chapter 9: summarizes the thesis and suggests directions for future directions.

1.5 List of publications

Part of the thesis has been published in major conferences in the area. Below I list each of the papers published/submitted during my PhD, which are classified into two categories. In the category of direct publications, I list all the papers (in the order of their appearances in the thesis) that directly correspond to the research themes described before. In the category of indirect publications, I list the publications that are related to the thesis but are not described in full detail.

Directly used publications

- **Chao Ma**, Yingzhen Li, and José Miguel Hernández-Lobato, *"Variational implicit processes"*, in International Conference on Machine Learning (ICML), 2019.
- **Chao Ma**, and José Miguel Hernández-Lobato, *"Functional Variational Inference based on Stochastic Process Generators"*, in Neural Information Processing Systems (NeurIPS), 2021.
- **Chao Ma**, Sebastian Tschiatschek, Konstantina Palla, José Miguel Hernández-Lobato, Sebastian Nowozin, and Cheng Zhang, *"EDDI: Efficient Dynamic Discovery of High-Value Information with Partial VAE"*, in International Conference on Machine Learning (ICML), 2019
- **Chao Ma**, and Cheng Zhang, *"Identifiable Generative Models for Missing Not at Random Data Imputation"*, in Neural Information Processing Systems (NeurIPS), 2021.

Indirect publications

- **Chao Ma**, Sebastian Tschiatschek, José Miguel Hernández-Lobato, Richard Turner, and Cheng Zhang, *"VAEM: a Deep Generative Model for Heterogeneous Mixed Type Data"*, in Neural Information Processing Systems (NeurIPS), 2020.
- Ignacio Peis, **Chao Ma**, and José Miguel Hernández-Lobato. *"Missing Data Imputation and Acquisition with Deep Hierarchical Models and Hamiltonian Monte Carlo."*, in Neural Information Processing Systems (NeurIPS), 2022.

- Weijie He, Xiaohao Mao, **Chao Ma**, Yu Hang, José Miguel Hernández-Lobato, and Ting Cheng, *"BSODA: A Bipartite Scalable Framework for Online Disease Diagnosis."*, Proceedings of the ACM Web Conference, 2022.
- **Chao Ma**,Wenbo Gong, José Miguel Hernández-Lobato, Noam Koenigstein, Sebastian Nowozin, and Cheng Zhang, *"Partial VAE for hybrid recommender system"*, in NeurIPS Workshop on Bayesian Deep Learning, 2018.
- **Chao Ma**, Sebastian Tschiatschek, Yingzhen Li, Richard Turner, José Miguel Hernández-Lobato, and Cheng Zhang, *"HM-VAEs: a deep generative model for real-valued data with heterogeneous marginals"*, in Symposium on Advances in Approximate Bayesian Inference, 2020.

In all of the paper described above, the author (Chao Ma) is responsible for research question identification, major research contributions, theoretical derivations and proofs, model designs and experiments, under the supervision of other co-authors, with the following exceptions:

- The paper titled *"EDDI: Efficient Dynamic Discovery of High-Value Information with Partial VAE"*, in which the research question is originally identified by Sebastian Nowozin and Cheng Zhang, and the main technical details, derivations, designs and implementations are done by this author (Chao Ma) during the internship at Microsoft Research.
- The papers titled *"BSODA: A Bipartite Scalable Framework for Online Disease Diagnosis."* and *"Missing Data Imputation and Acquisition with Deep Hierarchical Models and Hamiltonian Monte Carlo"*, I was only partly involved in the initial design of the algorithm, the discussions of the experimental results and a small fraction of paper writing. Most of the implementation, experiments and paper writing are conducted by Weijie He and Ignacio Peis.

Chapter 2

Inference and Models in Bayesian Machine Learning

“ *The most important maxim for data analysis to heed, and one which many statisticians seem to have shunned is this: ‘Far better an approximate answer to the right question, which is often vague, than an exact answer to the wrong question, which can always be made precise.’ Data analysis must progress by approximate answers, at best, since its knowledge of what the problem is will at best be approximate.* ”

John W Tukey, "The Future of Data Analysis", 1962,

IN Chapter 1, we have motivated the importance of uncertainty in machine learning. In particular, we argued that Bayesian uncertainty quantification (and more generally, Bayesian decision theory) is crucial for building intelligent systems: it is an inevitable consequence of the axiomatization of human rationality. To this end, Bayesian probability theory offers a foundation for the quantitative formulation of uncertainty. We might want to ask: how is it going to help us in practical machine learning scenarios? In this chapter, we review the basic conceptual and computational ideas for Bayesian modeling in the context of machine learning. We will first introduce the key concept of Bayesian machine learning, and how to perform approximate inference in the large data regime. We will cover variational inference, expectation propagation, black-box variational inference, alpha-divergence minimization, and implicit variational inference. Then, we will introduce specific models where Bayesian inference will be performed, including Bayesian neural networks, Gaussian processes, variational autoencoders, and Helmholtz machines.

Remark (Relevant mathematical backgrounds). Certain mathematical concepts and tools such as real analysis, measure-theoretic probability theory, stochastic processes, and functional analysis will be useful for certain concepts and theoretical results of the thesis. The introduction of these materials is out of the scope of the thesis. Nevertheless, the thesis has been written aiming to be as self-contained as possible. For example, throughout the thesis, we will avoid the use of measure-theoretic presentation of probability theory, and will only mildly discuss those related concepts in the **Remark** boxes when necessary (e.g., when introducing the general definition of KL divergence). For more concrete introductions to those mathematical backgrounds, readers may refer to resources such as [283, 129, 92].

2.1 Bayesian machine learning principles

Probability theory, or more specifically Bayesian probability theory, is the scientific language of uncertainty quantification. It tells us how to describe uncertainty in mathematical languages, and how to *infer* the uncertainty of unknown variables. To begin with, we work with a set of assumptions (or, in the language of machine learning, a model) that explicitly describes how observations are generated from the underlying system.

Using the notation from Chapter 1, this is described by the joint probability distribution $p(\mathbf{v}, \boldsymbol{\theta})$ of some observable vector $\mathbf{v}$, and the (latent, unknown) “state of nature" $\boldsymbol{\theta}$. This joint distribution can always be factorized into the product of the *data-generating distribution*, $p(\mathbf{v}|\boldsymbol{\theta})$, and the prior distribution, $p(\boldsymbol{\theta})$. These quantities can be either *objective* (i.e, when the actual data-generation process is known, or can be estimated from data and experiments), or *subjective* (i.e., the decision-maker’s belief, which only reflects how strongly he believes a particular event/proposition). In most cases, the use of probabilities are more or less subjective, since *models* are merely mathematical simplifications of the world that rely on idealized assumptions.

Remark (Why probability: a brief review). The discussion of philosophy of statistics, i.e., how we justify and interpret the probabilities, is out of the scope of this thesis. However, we do like to point out that justifications for using probabilities to represent uncertainties have been extensively discussed in the literature. Here, we briefly review some of those important theoretical results. These results, while having their technical limitations, still provide invaluable insights and greatly strengthen the argument regarding the inevitability of probabilities.

One of the first justifications can be found in the theory of subjective probability of Ramsey [265] and De Finetti [53], namely the principle of coherence (of bets), which also relates to decision making. The principle of coherence states that, unless the bettor's assignment of beliefs to random events satisfies the rules of probabilities, he/she will take bets that return a definite loss. Such bets are also called the Dutch book, or in the terminology of economics, an arbitrage.

Alternatively, as described in Chapter 1, Savage [296] independently took an axiomatic approach to inference under uncertainty, firmly based on the context of decision making. Under certain axioms of an idealized rational agent (i.e, if the decision-maker has rational preferences among all possible decisions), then his assignment of uncertainty must satisfy the rules of probability.

Finally, another important axiomatic approach owes to Cox [45] and Jaynes [137], which extends logical reasoning under Boolean algebra to logical inference under uncertainty. Cox's theorem states that any system representing the strengths of belief (plausibility) of propositions will be inconsistent with true-false logic unless they satisfy the rules of probability. That is, logical reasoning under uncertainty must be implemented by statistical inference using probability.

supervised learning In the context of (supervised) machine learning, we will be given a training set $\mathbf{v} = \mathcal{D} := \{(\mathbf{x}_i, y_i)\}_{i=1:N}$, where $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ is the set of inputs, and $\mathbf{y} = \{y_1, y_2, ..., y_N\}$ is the set of corresponding outputs. We would like to fit a regression model parameterized by $p_{\boldsymbol{\theta}}(y|\mathbf{x})$, where the "state of the nature" $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ is now defined as the unknown parameters of the regression model. The term $p_{\boldsymbol{\theta}}(y|\mathbf{x})$ is often referred to as likelihood, which is a function of $\boldsymbol{\theta}$ given fixed y; and the prior term $p(\boldsymbol{\theta})$ reflects our belief of what the parameters $\boldsymbol{\theta}$ might look like.

Given the model $p(\mathcal{D}, \boldsymbol{\theta})$, we can perform Bayesian inference regarding the latent variable $\boldsymbol{\theta}$. That is, conditioning on training data $\mathcal{D}$, we may calculate the posterior distribution for $\boldsymbol{\theta}$, $p(\boldsymbol{\theta}|\mathcal{D})$, using Bayes' rule:

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{p(\boldsymbol{\theta})p(\mathcal{D}|\boldsymbol{\theta})}{\int_{\boldsymbol{\theta}\in\boldsymbol{\theta}} p(\boldsymbol{\theta})p(\mathcal{D}|\boldsymbol{\theta})d\boldsymbol{\theta}}. \tag{2.1}$$

Similarly, we can perform inference on an unknown observable variable. Given a test input $\mathbf{x}^*$, the Bayesian posterior predictive distribution for the corresponding output, y^*, is given by

$$p(y^*|\mathbf{x}^*,\mathcal{D}) = \int_{\boldsymbol{\theta}\in\boldsymbol{\theta}} p(y^*|\mathbf{x}^*,\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D})d\boldsymbol{\theta}, \tag{2.2}$$

which quantifies the uncertainty on the test output y^*, *after* observing the data $\mathcal{D}$. Note that both Equation (2.1) and Equation (2.2) require the marginalization of $\boldsymbol{\theta}$, which produces the so called marginal likelihood (or model evidence), given by

$$p(\mathcal{D}) = \int_{\boldsymbol{\theta}\in\boldsymbol{\theta}} p(\boldsymbol{\theta})p(\mathcal{D}|\boldsymbol{\theta})d\boldsymbol{\theta}.$$

Intuitively, $p(\mathcal{D})$ quantifies the average probability of a parameter randomly sampled from $p(\boldsymbol{\theta})$ generating $\mathcal{D}$. Since the parameters are integrated out, a model selection criterion based on $p(\mathcal{D})$ would naturally guard against over-fitting, and control the model complexity. To understand such effect, factorize $p(\mathcal{D})$ in a sequential way, $p(\mathcal{D}) = p(y_1)p(y_2|y_1)p(y_3|y_1,y_2)...p(y_N|y_{1:N-1})$ (with inputs omitted). If the model is too simple, it will fit each of those distributions poorly. If it is too complex, then it would over-fit the “early samples", and predict the other samples poorly [235]. This is referred as the Bayesian Occam’s Razor effect [205]. Therefore, the quantity $p(\mathcal{D})$ is often used to perform model comparison and model selection across different models.

Unfortunately, in the setting of many modern machine learning problems, the integration problem $\int_{\boldsymbol{\theta}\in\boldsymbol{\theta}} p(\boldsymbol{\theta})p(\mathcal{D}|\boldsymbol{\theta})d\boldsymbol{\theta}$ is often computationally intractable [176] except for a few special cases. This is often due to: 1), the integration problem often involves a high-dimensionality parameter space that is impractical for numerical integration methods; and 2), the functional form of the likelihood function, $p(y|\mathbf{x},\boldsymbol{\theta})$, can often be very complicated. To confront such challenges, we often resort to Approximate inference techniques [176], which will be introduced in the next section.

2.2 Approximate inference algorithms

2.2.1 Vanilla variational inference

Consider the problem of *approximating* the posterior distribution, $p(\boldsymbol{\theta}|\mathcal{D})$. In the supervised learning setting, given the model likelihood function $p(y|\mathbf{x},\boldsymbol{\theta})$ and the prior distribution $p(\boldsymbol{\theta})$, $p(\boldsymbol{\theta}|\mathcal{D})$ can be computed as

$$p(\boldsymbol{\theta}|\mathcal{D}) \propto p(\boldsymbol{\theta})\prod_n p(y_n|\mathbf{x}_n,\boldsymbol{\theta}). \tag{2.3}$$

Variational inference (VI) [140] converts the above inference problem into an optimization problem by first proposing a class of approximate posterior distributions $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ parameterized by variational parameters $\boldsymbol{\lambda}$, and then minimizing with respect to $\boldsymbol{\lambda}$ a certain divergence measure between the approximate posterior $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ and $p(\boldsymbol{\theta}|\mathcal{D})$. One common parameterization method is the so-called mean-field approximation [294]. For example, $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ can be parameterized by a full-factorized Gaussian, $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) = \prod_{1\le d\le|\boldsymbol{\theta}|}\mathcal{N}(\theta_d;m_d,\sigma_d^2)$, where $|\boldsymbol{\theta}|$ denotes the dimensionality of $\boldsymbol{\theta}$, θ_d is the d-th component of $\boldsymbol{\theta}$, and $\mathcal{N}(\theta_d;m_d,\sigma_d^2)$ is the Gaussian density function of θ_d with mean m_d and variance σ_d^2. The variational parameters $\boldsymbol{\lambda}$ are now given by a collection of mean and standard deviation parameters $\boldsymbol{\lambda} = \{(m_d,\sigma_d)|1\le d\le|\boldsymbol{\theta}|\}$.

Variational inference usually minimizes the (exclusive) Kullback–Leibler divergence [163] $\mathrm{D}_{\mathrm{KL}}[\cdot||\cdot]$ between the approximate posterior $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ and the true posterior $p(\boldsymbol{\theta}|\mathcal{D})$. It is defined as

$$\mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathcal{D})] = \int_{\boldsymbol{\theta}} \log\frac{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}{p(\boldsymbol{\theta}|\mathcal{D})} q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})d\boldsymbol{\theta}. \tag{2.4}$$

$\mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathcal{D})]$ has two nice properties that makes it a suitable objective function for VI. First, $\mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathcal{D})]$ is always non-negative; and second, $\mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathcal{D})] = 0$ if and only if $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\mathcal{D})$, that is, when the variational distribution is the perfect approximation to the ground truth.

Remark (Measure-theoretic definition of KL-divergence [92, 213]). Here we briefly introduce the measure-theoretic definition of KL-divergence, which will be useful for later chapters.

Definition 2.1 (KL-divergence)**.** *Consider two probability measures Q and P on some common measurable space $(\Omega,\mathcal{B})$. In other words, Q and P are two real-valued (measurable) functions on $(\Omega,\mathcal{B})$, that satisfy the Kolmogorov's axioms [155]. Then, the KL-divergence between Q and P is defined by*

$$\mathrm{D}_{KL}[Q||P] = \sup_{\mathcal{G}}\sum_i Q(\mathcal{G}_i)\log\frac{Q(\mathcal{G}_i)}{P(\mathcal{G}_i)}, \tag{2.5}$$

where $\mathcal{G} = \bigcup_i \mathcal{G}_i$ is a finite measurable partition of Ω, Q and P are the probability measures of Q and P on $(\Omega,\mathcal{B})$, respectively. In other words, the KL-divergence between two probability measures is the supreme of the relative entropies obtained on all possible (finite measurable) partitions of Ω.

One could further show that [92]:

Proposition 2.1. *The KL-divergence defined in Definition 2.1 satisfies the following properties:*

1. $\mathrm{D}_{KL}[Q||P] \geq 0$, *and* $\mathrm{D}_{KL}[Q||P] = 0$ *if and only if* $Q = P$*;*
2. *If Q is not absolutely continuous w.r.t. P, then* $\mathrm{D}_{KL}[Q||P] = \infty$*;*
3. *If Q is absolutely continuous w.r.t. P (i.e, $Q \ll P$), then the Radon-Nikodym derivative of Q w.r.t. P exists, and* $\mathrm{D}_{KL}[Q||P]$ *can be expressed as*

$$\mathrm{D}_{KL}[Q||P] = \int_{\Omega} \log\left[\frac{dQ}{dP}\right] dQ, \tag{2.6}$$

where $\frac{dQ}{dP}$ denotes the Radon-Nikodym derivative of Q w.r.t. P. This gives the alternative measure-theoretic definition of KL-divergence used in the literature [213].

However, we cannot directly compute $\mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathcal{D})]$ since the true posterior $p(\boldsymbol{\theta}|\mathcal{D})$ is unknown. To get around this, notice that the marginal likelihood $\log p(\mathcal{D})$ can be rewritten in terms of $\mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathcal{D})]$:

$$\begin{aligned}
\log p(\mathcal{D}) &= \log p(\mathcal{D}, \boldsymbol{\theta}) - \log p(\boldsymbol{\theta}|\mathcal{D}) \\
&= \int_{\boldsymbol{\theta}} q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})[\log p(\mathcal{D}, \boldsymbol{\theta}) - \log p(\boldsymbol{\theta}|\mathcal{D})] d\boldsymbol{\theta} \\
&= \int_{\boldsymbol{\theta}} q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})[\log \frac{p(\mathcal{D}, \boldsymbol{\theta})}{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})} - \log \frac{p(\boldsymbol{\theta}|\mathcal{D})}{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}] d\boldsymbol{\theta} \\
&= \underbrace{\mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathcal{D})]}_{\text{VI objective}} + \underbrace{\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}\left[\log \frac{p(\mathcal{D}, \boldsymbol{\theta})}{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}\right]}_{\text{free energy}}.
\end{aligned} \tag{2.7}$$

We call $\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}\left[\log \frac{p(\mathcal{D}, \boldsymbol{\theta})}{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}\right]$ the variational free energy. Since $\mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathcal{D})]$ is always non-negative, the term $\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}\left[\log \frac{p(\mathcal{D}, \boldsymbol{\theta})}{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}\right]$ forms a lower bound of the model evidence, $\log p(\mathcal{D})$. Therefore, $\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}\left[\log \frac{p(\mathcal{D}, \boldsymbol{\theta})}{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}\right]$ is also called the evidence lower bound (ELBO), often denoted by $\mathcal{L}(q)$. The relationships among $\log p(\mathcal{D})$, $\mathcal{L}(q)$, and $\mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathcal{D})]$ are depicted in Figure 2.1: the VI objective, $\mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathcal{D})]$ is essentially the gap between $\log p(\mathcal{D})$ and $\mathcal{L}(q)$. Therefore, in order to minimize $\mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathcal{D})]$, we can equivalently maximize the variational free energy, $\mathcal{L}(q)$.

In order to compute $\mathcal{L}(q)$, we often rewrite it into the following form:

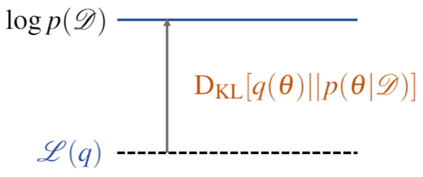


Figure 2.1 The relationships among $\log p(\mathcal{D})$, $\mathcal{L}(q)$, and $\mathrm{D_{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathcal{D})]$ in variational inference.

$$\mathcal{L}(q) = \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}\left[\log \frac{p(\mathcal{D},\boldsymbol{\theta})}{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}\right] = \underbrace{\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\log p(\mathcal{D}|\boldsymbol{\theta})]}_{\text{likelihood/predictive loss}} - \underbrace{\mathrm{D_{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta})]}_{\text{Regularization}}. \tag{2.8}$$

Intuitively, the first term $\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\log p(\mathcal{D}|\boldsymbol{\theta})]$, known as the *likelihood term* of the ELBO, quantifies the model's expected predictive performance on training data $\mathcal{D}$. The second term $\mathrm{D_{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta})]$ serves as the regularization term, which encourages $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ to be close to the prior, $p(\boldsymbol{\theta})$. Finally, we can optimize $\mathcal{L}(q)$ by performing gradient ascent using the gradient $\nabla_{\boldsymbol{\lambda}}\mathcal{L}(q)$.

2.2.2 Scalable variational inference

While optimizing the evidence lower bound (2.8) is usually easier than directly computing the high-dimensional integral in Equation (2.1), it does not tell us how to compute the expectation $\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\log p(\mathcal{D}|\boldsymbol{\theta})]$. In fact, two major difficulties arise when applying vanilla VI: 1), in many machine learning applications, evaluating $\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\log p(\mathcal{D}|\boldsymbol{\theta})]$ requires computing the likelihood $\log p(\mathcal{D}|\boldsymbol{\theta})$ over the entire dataset, which is computationally infeasible; 2), for many complicated models, $\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\log p(\mathcal{D}|\boldsymbol{\theta})]$ does not have an analytic form. Therefore, in the early history of variational inference, its application was often limited to specific models, specific variational distributions, and specific data volumes.

Consequently, scalable variational inference methods [266, 118] were developed to counter those difficulties. Consider again the following joint model:

$$p(\mathcal{D},\boldsymbol{\theta}) = p(\boldsymbol{\theta}) \prod_{1\leq n\leq N} p(y_n|\mathbf{x}_n,\boldsymbol{\theta}). \tag{2.9}$$

To perform VI, its ELBO can be written as

$$\mathcal{L}(q) = \sum_{n=1}^{N} \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\log p(y_n|\mathbf{x}_n, \boldsymbol{\theta})] - \mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta})]. \tag{2.10}$$

When N is large, stochastic optimization techniques [33] can be applied. Specifically, we sample a mini-batch $\mathcal{K} \subset \{1, ..., N\}$ of size K, and obtain a noisy estimate of the ELBO:

$$\hat{\mathcal{L}}(q) = \frac{N}{K} \sum_{k \in \mathcal{K}} \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\log p(y_k|\mathbf{x}_k, \boldsymbol{\theta})] - \mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})||p(\boldsymbol{\theta})], \tag{2.11}$$

which can then be used to perform stochastic gradient descent:

$$\boldsymbol{\lambda}^{(t)} = \boldsymbol{\lambda}^{(t-1)} + r_t \nabla_{\boldsymbol{\lambda}^{(t-1)}} \hat{\mathcal{L}}(q). \tag{2.12}$$

This will converge to a local optimum $\boldsymbol{\lambda}^{\star}$ of $\mathcal{L}(q)$ if the learning rate schedule $\{r_t\}$ satisfies the Robbins-Monro conditions [276]:

$$\sum_{t=1}^{\infty} r_t = \infty, \quad \sum_{t=1}^{\infty} r_t^2 < \infty. \tag{2.13}$$

Finally, to deal with the analytical intractability of $\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\log p(y_k|\mathbf{x}_k, \boldsymbol{\theta})]$, the following Monte Carlo approximation is often used:

$$\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\log p(y_k|\mathbf{x}_k, \boldsymbol{\theta})] \approx \frac{1}{M} \sum_{m=1}^{M} \log p(y_k|\mathbf{x}_k, \boldsymbol{\theta}^m), \quad \boldsymbol{\theta}^m \sim q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}), \tag{2.14}$$

which forms a unbiased estimator:

$$\mathbb{E}_{\boldsymbol{\theta}^1 \sim q(\boldsymbol{\theta}^1), ..., \boldsymbol{\theta}^M \sim q(\boldsymbol{\theta}^M)} \left[\frac{1}{M} \sum_{m=1}^{M} \log p(y_k|\mathbf{x}_k, \boldsymbol{\theta}^m) \right] = \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\log p(y_k|\mathbf{x}_k, \boldsymbol{\theta})]. \tag{2.15}$$

This allows us to estimate the expectations $\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\log p(y_k|\mathbf{x}_k, \boldsymbol{\theta})]$ without the need to obtain analytic solutions. By combining the noisy ELBO (2.11) and the MC approximation (2.14), variational inference can be easily applied to complicated models and large datasets, which forms an algorithmic foundation for modern Bayesian deep learning.

Remark (Monte Carlo gradient estimators). Although obtaining the MC approximation of $\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\log p(y_k|\mathbf{x}_k, \boldsymbol{\theta})]$ is straightforward, in practice we are more interested in estimating the *gradient*, $\nabla_{\boldsymbol{\lambda}} \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})} [\log p(y_k|\mathbf{x}_k, \boldsymbol{\theta})]$. However, note that the gradient is taken regarding the parameters $\boldsymbol{\lambda}$ of the probabilistic measure $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ in the expectation operator, $\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\cdot]$. Therefore, we cannot naively exchange the order of the derivative operator $\nabla_{\boldsymbol{\lambda}}$ and the expectation operator $\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\cdot]$. To properly compute $\nabla_{\boldsymbol{\lambda}} \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})} [\log p(y_k|\mathbf{x}_k, \boldsymbol{\theta})]$, several techniques can be adopted:

1. The REINFORCE gradient estimator [366, 73], also known as the score function estimator, is derived as follows:

$$\begin{aligned}
\nabla_{\boldsymbol{\lambda}} \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})} [\log p(y_k|\mathbf{x}_k, \boldsymbol{\theta})] &= \int_{\boldsymbol{\theta} \in \boldsymbol{\theta}} \nabla_{\boldsymbol{\lambda}} [q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) \log p(y_k|\mathbf{x}_k, \boldsymbol{\theta})] d\boldsymbol{\theta} \\
&= \int_{\boldsymbol{\theta} \in \boldsymbol{\theta}} q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) \nabla_{\boldsymbol{\lambda}} \log q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) \log p(y_k|\mathbf{x}_k, \boldsymbol{\theta}) d\boldsymbol{\theta} \\
&= \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})} [\underbrace{\nabla_{\boldsymbol{\lambda}} \log q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}_{\text{score function}} \log p(y_k|\mathbf{x}_k, \boldsymbol{\theta})] \\
&\approx \frac{1}{M} \sum_{m=1}^{M} \nabla_{\boldsymbol{\lambda}} \log q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}^m) \log p(y_k|\mathbf{x}_k, \boldsymbol{\theta}^m), \quad \boldsymbol{\theta}^m \sim q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}).
\end{aligned}$$

   Note that the above derivation does not require the gradient information of $\log p(y_k|\mathbf{x}_k, \boldsymbol{\theta})$. Therefore, the REINFORCE estimator can even be applied to the case where $\log p(y_k|\mathbf{x}_k, \boldsymbol{\theta})$ is not differentiable w.r.t. $\boldsymbol{\theta}$.

2. The path-wise estimator, and the reparameterization trick [231, 149]. Suppose $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ can be reparameterized into the following sampling procedure:

$$\boldsymbol{\theta} = h_{\boldsymbol{\lambda}}(\varepsilon), \quad \varepsilon \sim p(\varepsilon),$$

   where $p(\varepsilon)$ is some noise distribution, $h_{\boldsymbol{\lambda}}(\cdot)$ is a deterministic transformation. For example, if $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}; \mu, \sigma^2)$, then it can be reparameterized as

$$\boldsymbol{\theta} = \mu + \sigma\varepsilon, \quad \varepsilon \sim \mathcal{N}(\varepsilon; 0, 1).$$

Then, the path-wise gradient estimator is derived as follows:

$$\begin{aligned}\nabla_{\boldsymbol{\lambda}} \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}\left[\log p(y_k|\mathbf{x}_k, \boldsymbol{\theta})\right] &= \nabla_{\boldsymbol{\lambda}} \mathbb{E}_{p(\varepsilon)}\left[\log p(y_k|\mathbf{x}_k, h_{\boldsymbol{\lambda}}(\varepsilon))\right] \\ &= \mathbb{E}_{p(\varepsilon)}\left[\nabla_{\boldsymbol{\lambda}} \log p(y_k|\mathbf{x}_k, h_{\boldsymbol{\lambda}}(\varepsilon))\right] \\ &\approx \frac{1}{M}\sum_{m=1}^{M} \nabla_{\boldsymbol{\lambda}} \log p(y_k|\mathbf{x}_k, h_{\boldsymbol{\lambda}}(\varepsilon^m)), \quad \varepsilon^m \sim p(\varepsilon).\end{aligned}$$

The path-wise estimator does require to compute the gradient, $\nabla_{\boldsymbol{\lambda}} \log p(y_k|\mathbf{x}_k, h_{\boldsymbol{\lambda}}(\varepsilon^m))$. However, when the path-wise gradient estimator can be applied, it tends to have lower variance than the REINFORCE estimator [74, 73, 272], especially when 1), $\boldsymbol{\theta}$ is high-dimensional; and 2), $\log p(y_k|\mathbf{x}_k, \boldsymbol{\theta})$ is smooth with small magnitude of derivatives. Regarding the variance of the two estimators, [74] has shown the following result:

Proposition 2.2 (Variance analysis). *Let* $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) = \mathcal{N}(\mu, \sigma^2)$, $\boldsymbol{\lambda} = \{\mu, \sigma\}$, *and* $f(\boldsymbol{\theta})$ *be a real-valued function. Assume* $Var_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}((\boldsymbol{\theta} - \mu) f(\boldsymbol{\theta})) < \infty$, $Var_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}(f'(\boldsymbol{\theta})) < \infty$, $\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}(|(\boldsymbol{\theta} - \mu) f'(\boldsymbol{\theta}) + f(\boldsymbol{\theta})|) < \infty$ *and* $\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}(|f''(\boldsymbol{\theta})|) < \infty$. *If*

$$\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}((\boldsymbol{\theta} - \mu) f'(\boldsymbol{\theta}) + f(\boldsymbol{\theta}))^2 - \sigma^4 \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}(f''(\boldsymbol{\theta})^2) \geq 0$$

then path-wise derivative estimator of

$$\partial_\mu \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}(f(\boldsymbol{\theta})$$

has lower variance of score function estimator.

The intuition of this result is that path-wise estimators will exhibit lower variance than REINFORCE as long as the score function is better behaved (in the sense of expected change) than the original joint likelihood. As argued in [74], many functions $f(\boldsymbol{\theta})$ in VI satisfy such condition, hence the path-wise gradient estimator is often preferred. However, when the gradient information $\nabla_{\boldsymbol{\lambda}} \log p(y_k|\mathbf{x}_k, h_{\boldsymbol{\lambda}}(\varepsilon^m))$ is unavailable, one often need to resort to REINFORCE. In this scenario, various variance reduction methods can be applied, such as Rao-Blackwellization [266], the baseline approach [226], and more generally the control variates approach [266, 252], etc. There also exits a number of other VI variance reduction techniques for specific types of problems, for example Gumbel-Softmax [135] and VIMCO [227] for discrete variables.

In this thesis, unless specified, we will always use the path-wise gradient estimator for variational objectives.

2.2.3 Power expectation propagation, and α-divergence minimization

In certain cases, VI may suffer from certain pathologies such as under-estimating the uncertainties and introducing additional unwanted biases [344]. In this section, we introduce an extension of variational inference, namely the black-box alpha divergence minimization (BB-α) [113], that addresses some of the drawbacks of VI. Built upon ideas similar to VI, BB-α performs scalable approximate inference by working with the following α-divergence [380] $\mathrm{D}_\alpha[p||q]$:

$$q^\star = \arg\min_q \mathrm{D}_\alpha[p||q] = \arg\min_q \frac{1}{\alpha(1-\alpha)}\left(1 - \int_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} p(\boldsymbol{\theta})^\alpha q(\boldsymbol{\theta})^{1-\alpha} d\boldsymbol{\theta}\right). \tag{2.16}$$

The α-divergence is a generic class of divergences that includes as special cases the (exclusive) KL-divergence (α = 0, corresponds to VI) , the inclusive KL-divergence (α=1, corresponds to the EP objetive), and Hellinger distance (α=0.5), etc. Thus, BB-α naturally unifies VI [140] and expectation propagation (EP) [223, 178]. As $\alpha \to +\infty$, $\mathrm{D}_\alpha[p||q]$ tends to encourage mode-covering behaviour (i.e., q will tend to cover all local modes of p; on the contrary, as $\alpha \to -\infty$, $\mathrm{D}_\alpha[p||q]$ encourages mode-seeking behavior, and q will tend to place more mass on the area where p has the largest probability [225].

Before introducing further details regarding BB-α, we first briefly discuss the power expectation propagation (power EP) algorithm [224], since BB-α is largely inspired from the power EP. Unlike VI (which optimizes KL divergence *globally*), power EP parameterizes $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ via a set of local approximations $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) \propto \frac{1}{Z}p(\boldsymbol{\theta})\prod_n \tilde{f}_n(\boldsymbol{\theta})$, where each $\tilde{f}_n(\boldsymbol{\theta}) \propto \exp\left[\boldsymbol{\lambda}_n^T \boldsymbol{\phi}(\boldsymbol{\theta})\right]$ is an exponential family that captures the individual effect of $p(y_n|\mathbf{x}_n,\boldsymbol{\theta})$. Moreover, power EP optimizes the α-divergence $\mathrm{D}_\alpha[p||q]$ *locally* using message passing. If convergent, it converges to a fixed point of the following energy function, $\mathcal{L}_{\mathrm{PEP}}(\boldsymbol{\lambda}_0,\{\boldsymbol{\lambda}_n\})$:

$$\begin{aligned}\mathcal{L}_{\mathrm{PEP}}(\boldsymbol{\lambda}_0,\{\boldsymbol{\lambda}_n\}) &= \log Z(\boldsymbol{\lambda}_0) + (\frac{N}{\alpha} - 1)\log Z(\boldsymbol{\lambda}_q)\\ &- \frac{1}{\alpha}\sum_{n=1}^N \log \int p(y_n|\mathbf{x}_n,\boldsymbol{\theta})^\alpha \exp\left[(\boldsymbol{\lambda}_q - \alpha\boldsymbol{\lambda}_n)^T\boldsymbol{\phi}(\boldsymbol{\theta})\right] d\boldsymbol{\theta},\end{aligned} \tag{2.17}$$

where $\boldsymbol{\lambda}_q = \boldsymbol{\lambda}_0 + \sum_{n=1}^N \boldsymbol{\lambda}_n$ is the natural parameter of $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$.

Remark (Local approximations via power EP). We breifly describe here how iterative local message passing is done in power EP. For each iteration, power EP will pick a factor (say $\tilde{f}_n$) to refine. This is done by minimizing the α-divergence $\mathrm{D}_\alpha[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})\frac{p(y_n|\mathbf{x}_n,\boldsymbol{\theta})}{\tilde{f}_n(\boldsymbol{\theta})}||q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})]$, whose gradient with respect to $\boldsymbol{\lambda}_q$ is as follows:

$$\begin{aligned}
&\nabla_{\boldsymbol{\lambda}_q} \mathrm{D}_\alpha[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})\frac{p(y_n|\mathbf{x}_n,\boldsymbol{\theta})}{\tilde{f}_n(\boldsymbol{\theta})}||q] = \\
&\nabla_{\boldsymbol{\lambda}_q} \frac{1}{\alpha(1-\alpha)}\left[1 - \int_{\boldsymbol{\theta}\in\boldsymbol{\theta}} \left(q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})\frac{p(y_n|\mathbf{x}_n,\boldsymbol{\theta})}{\tilde{f}_n(\boldsymbol{\theta})}\right)^\alpha q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})^{1-\alpha} d\boldsymbol{\theta}\right] = \\
&-\frac{1}{\alpha}\int_{\boldsymbol{\theta}\in\boldsymbol{\theta}} \left(q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})\frac{p(y_n|\mathbf{x}_n,\boldsymbol{\theta})}{\tilde{f}_n(\boldsymbol{\theta})}\right)^\alpha q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})^{1-\alpha} \nabla_{\boldsymbol{\lambda}_q} \log q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) d\boldsymbol{\theta} \propto \\
&\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\boldsymbol{\phi}(\boldsymbol{\theta})] - \mathbb{E}_{\underbrace{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})\frac{p(y_n|\mathbf{x}_n,\boldsymbol{\theta})^\alpha}{\tilde{f}_n(\boldsymbol{\theta})^\alpha}}_{\text{tilted distribution}}}[\boldsymbol{\phi}(\boldsymbol{\theta})]
\end{aligned}$$

By setting $\nabla_{\boldsymbol{\lambda}_q} \mathrm{D}_\alpha[q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})\frac{p(y_n|\mathbf{x}_n,\boldsymbol{\theta})}{\tilde{f}_n(\boldsymbol{\theta})}||q]$ to zero, this gives us the fixed-point updates via moment matching:

$$\mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\boldsymbol{\phi}(\boldsymbol{\theta})] \leftarrow \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})\frac{p(y_n|\mathbf{x}_n,\boldsymbol{\theta})^\alpha}{\tilde{f}_n(\boldsymbol{\theta})^\alpha}}[\boldsymbol{\phi}(\boldsymbol{\theta})].$$

Note that power EP in general does not have any convergence guarantees. However, when convergent, it converges to a fixed point of $\mathcal{L}_{\mathrm{PEP}}$.

Now we are ready to describe BB-α. BB-α is hugely inspired by power EP, but took a different approach towards energy optimization. BB-α directly optimizes $\mathcal{L}_{\mathrm{PEP}}$ with tied factors $\tilde{f}_n = \tilde{f}$ to avoid prohibitive local factor updates and storage on the whole dataset. This means $\boldsymbol{\lambda}_n = \boldsymbol{\lambda}$ for all n and $\boldsymbol{\lambda}_q = \boldsymbol{\lambda}_0 + N\boldsymbol{\lambda}$. Therefore instead of parameterizing each factors, we can directly parameterize $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ and replace all the local factors in the power-EP energy function by $\tilde{f}(\boldsymbol{\theta}) \propto (q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})/p(\boldsymbol{\theta}))^{1/N}$. After re-arranging terms, this gives the BB-α energy:

$$\mathcal{L}_\alpha(q) = -\frac{1}{\alpha}\sum_n \log \mathbb{E}_q\left[\left(\frac{f_n(\boldsymbol{\theta})p(\boldsymbol{\theta})^{\frac{1}{N}}}{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})^{\frac{1}{N}}}\right)^\alpha\right], \tag{2.18}$$

which can be further approximated by the following if the dataset is large [177]:

$$\mathcal{L}_\alpha(q) = \mathrm{D}_{\mathrm{KL}}[q||p] - \frac{1}{\alpha}\sum_n \log \mathbb{E}_q\left[p(y_n|\mathbf{x}_n,\boldsymbol{\theta})^\alpha\right]. \tag{2.19}$$

The optimization of $\mathcal{L}_\alpha(q)$ could be performed in a black-box manner with mini-batch stochastic optimization and path-wise gradient estimator, as introduced in Section 2.2.2. Empirically, it has been shown that BB-α with $\alpha \neq 0$ can return significantly better uncertainty estimation than VI, and has been applied successfully in different scenarios [177, Depeweg et al.]. Moreover, from the perspective of model evidence approximation, it has been suggested [180] that the BB-α energy $\mathcal{L}_\alpha(q)$ forms a better estimation of log marginal likelihood, $\log p(\mathcal{D})$ when compared with the evidence lower bound, $\mathcal{L}(q)$. Therefore, in Chapter 3 of this thesis, BB-α energy is used for both Bayesian inference and model evidence approximation.

2.2.4 Implicit variational inference: beyond prescribed approximations

Due to the need for tractability and scalability are, many VI/EP methods rely on simple approximations such as mean-field approximations. However, mean-field approximations explicitly ignore correlations among different components of $\boldsymbol{\theta}$, which tends to under-estimate uncertainties [344], introduce multiple local minima and bring additional nonconvexity to the optimization problem [348]. These drawbacks of mean-field VI become more significant when applied to complicated models such as Bayesian neural networks [71, 127, 284, 251, 68]. Therefore, one promising direction for improving VI is to specify flexible approximations $q(\boldsymbol{\theta})$, which has received enormous attention from the community. Relevant approaches could be roughly categorized into structured mean field [348, 295, 364, 363], auxiliary/hierarchical VI [2, 291, 268], mixture VI [81, 96, 222, 10], copula VI [338, 100], normalizing flows [271], and implicit VI [125, 339, 191].

Among these approaches, we introduce recent works that consider implicit models/distributions as approximate posteriors [339, 191, 176, 182, 179] for more flexible VI. In contrast to *prescribed probabilistic models* [61] that assign *explicit* densities to possible outcomes, implicit distributions *implicitly assign* probability measures by the specification of the *data generating process*, also known as the *black-box simulator*. This means that we are only able to obtain Monte Carlo samples from such implicit distributions, but their probability density function might not be available for evaluation. In its most common example, an implicit distribution $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ can be defined by the following sampling process:

$$\boldsymbol{\theta} = g_{\boldsymbol{\lambda}}(\varepsilon), \quad \varepsilon \sim p(\varepsilon), \tag{2.20}$$

where $g_{\boldsymbol{\lambda}}(\cdot) : \mathbb{R}^{|\varepsilon|} \mapsto \mathbb{R}^{|\boldsymbol{\theta}|}$ is a multi-variate function parameterized by $\boldsymbol{\lambda}$, and ε is some simple noise variable (e.g., factorized standard normal distribution). The probability density function of $\boldsymbol{\theta}$ is implicitly defined as

$$q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) = \int_{\{\varepsilon | g(\varepsilon) = \boldsymbol{\theta}\}} p(\varepsilon) d\varepsilon \tag{2.21}$$

Implicit models provide exceptionally powerful tools for parameterizing and learning probabilistic distributions. One of the most well known implicit distributions in machine learning is the generative adversarial nets (GANs) [88, 11], where $g_{\boldsymbol{\lambda}}(\cdot) : \mathbb{R}^{|\varepsilon|} \mapsto \mathbb{R}^{|\boldsymbol{\theta}|}$ is defined by as a neural network (namely the *generator*) that maps the noise ε to higher dimensional observations. GANs have demonstrated their expressiveness and flexibility in complicated tasks such as image generation [262], protein modeling [270], lattice simulation in high-energy physics [256], weak lensing convergence map in cosmology [238], and Fermi-Hubbard model simulation in high-temperature superconductivity [40].

Remark (implicit distributions as pushforward measure). Defining implicit distributions as "distributions that allow sampling but not the evaluation of probabilities" is somewhat vague and unrigorous. With the help of measure-theoretic probability, we can define implicit distribution as a probabilistic measure defined via *push-forward measure*:

Definition 2.2 (Pushforward measure)**.** *Given a probability space* $(\Omega, \mathcal{B}, P)$*, then a measurable mapping* G *from* $(\Omega, \mathcal{B}, P)$ *to another measurable space* $(\mathcal{S}, \mathcal{M})$ *will induce a probability measure* G^**, called the pushforward of* P*, defined by*

$$G^*(M) := P(G^{-1}(M)), \quad \forall M \in \mathcal{M}. \tag{2.22}$$

The GAN example abve is a special of case of Equation (2.22) by considering $\Omega = \mathbb{R}^{|\varepsilon|}$, $\mathcal{S} = \mathbb{R}^{|\boldsymbol{\theta}|}$, and G as a measurable function defined by a neural network.

Although implicit distributions are very powerful, variational inference using such posterior approximators is quite difficult. This is due to that the KL-divergence term $\mathrm{D}_{\mathrm{KL}}[q||p]$ of the ELBO (2.8, 2.29) requires evaluation of the probability density function of q, which is unfortunately intractable if they are defined by implicit distributions. Therefore, several implicit variational inference methods have been proposed based on additional approximations to evaluate $\mathrm{D}_{\mathrm{KL}}[q||p]$. One common technique is called the density ratio estimator [328, 125, 339, 179]. To derive this technique, notice that $\mathrm{D}_{\mathrm{KL}}[q||p]$ can be written as

$$\mathrm{D}_{\mathrm{KL}}[q||p] = \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}\left[\log \frac{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}{p(\boldsymbol{\theta})}\right] = \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\log U(\boldsymbol{\theta})], \tag{2.23}$$

where $U(\boldsymbol{\theta}) = \frac{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}{p(\boldsymbol{\theta})}$ is the density ratio between $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ and $p(\boldsymbol{\theta})$. $U(\boldsymbol{\theta})$ can be estimated via a GAN-based idea by training the following binary classifier $D(\boldsymbol{\theta})$ that distinguish between samples from q and p

$$D(\boldsymbol{\theta}) := P(\boldsymbol{\theta} \text{ is sampled from } q|\boldsymbol{\theta}) = \frac{1}{1+\exp\{-\log U(\boldsymbol{\theta})\}}. \tag{2.24}$$

The scoring objective can be defined by the following loss function,

$$\max_{U} \ \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}[\log D(\boldsymbol{\theta})] + \mathbb{E}_{p(\boldsymbol{\theta})}[\log(1-D(\boldsymbol{\theta}))]. \tag{2.25}$$

Minimizing the above objective gives the optimal solution,

$$U^{\star} = \frac{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}{p(\boldsymbol{\theta})}. \tag{2.26}$$

Once $U^{\star}$ has been obtained, the implicit VI ELBO can be estimated via

$$\mathcal{L}(q) = \sum_{n=1}^{N} \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}\left[\log p(y_n|\mathbf{x}_n, \boldsymbol{\theta})\right] - \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})}\left[U^{\star}(\boldsymbol{\theta})\right]. \tag{2.27}$$

Remark (Other implicit VI approaches). Despite these successful applications in training generative adversarial networks, density ratio estimators have also been criticized for being an extremely challenging problem in high-dimensional settings [328], and introducing additional optimization complexities. They require a large number of Monte Carlo samples [176], often exhibiting high variances for high-dimensional $\boldsymbol{\theta}$ due to overfitting [328, 218], and lack of capabilities to process high-dimensional inputs such as neural network weights [308]. Apart from GAN-like density ratio estimators, there also exists a number of alternative estimators of (the gradients of) $\mathrm{D}_{\mathrm{KL}}[q||p]$. For instance, kernel estimators [308], Stein gradient estimators [181, 309], spectral approaches [309, 264], and nonparametric score estimators [379]. However, many of these methods still suffer from less efficiency for such high dimensional inputs to a certain extend [308, 379, 85].

Another closely related approach puts additional constraints on q, and assumes that q can be represented as mixtures of tractable distributions. These ideas originate from auxiliary variational inference [2], and has recently seen a revival in the context of *semi-implicit VI* [2, 291, 268, 371, 234, 336, 317, 229], which provides a nice balance between posterior flexibility and optimization simplicity.

2.3 Models

Many recent advances in Bayesian machine learning started from the developments of tools and techniques of approximate Bayesian inference, which have enabled the application of Bayesian principles to complicated models. In this section, we introduce some of these important models in Bayesian machine learning. Specifically, we will introduce Bayesian neural networks, Gaussian processes, and deep generative models, where elements of modern deep learning are combined with Bayesian modeling (i.e., Bayesian deep learning). We will also discuss the advantages and disadvantages behind those ideas, and thus motivate some new fresh ideas that will be developed later in the thesis.

2.3.1 Deep neural networks, and their Bayesianization

Deep neural networks There has been an explosion of deep machine learning models in the past decade [22, 115, 288, 162, 314, 62, 167, 230, 142, 43, 331], which have achieved state-of-the-art results on many large scale tasks. More recently, larger neural models such as ResNet [105], DenseNet [123], BERT [58], and GPT-3 [35] has been the driven force behind many real-world applications. Behind those successes, neural networks/multi-layer perceptions have been the core components of almost every deep learning model. Essentially, deep neural networks (DNNs) are collections of trainable units, organized in different layers of abstractions. They can learn features from raw, high-dimensional, noisy, ambiguous data at a large scale. The most common way to train a (L-layer) deep neural network is via the frequentist approach, namely the (regularized) empirical risk minimization (ERM):

$$\begin{aligned}\min_{\mathbf{w}} \quad & l(\mathbf{w},\mathcal{D}) = \sum_{1\leq n\leq N} ||y_n - g(\mathbf{x}_n,\mathbf{w})||^2 + \beta R(\mathbf{w}),\\ s.t. \quad & \mathbf{h}_n^l = \varsigma_{l-1}(\mathbf{h}_n^{l-1}\mathbf{w}_l + \mathbf{b}_l), l = 1,...,L-1,\\ & \mathbf{h}_n^0 = \mathbf{x}_n, \quad g(\mathbf{x}_n,\mathbf{w}) = \mathbf{h}_n^L,\end{aligned}$$

where $\mathbf{w} = \{(\mathbf{w}_1,\mathbf{b}_1),...,(\mathbf{w}_L,\mathbf{b}_L)\}$ is the set of neural network weights (and biases), $R(\mathbf{w})$ is the regularization term weighted by β (a common choice would be the Frobenius norm $r(\mathbf{w}) = ||\mathbf{w}||_F^2$), $\varsigma_{l-1}(\cdot)$ is the non-linear activation function for the l^{th} layer. The optimization problem $\min_{\mathbf{w}} \ l(\mathbf{w},\mathcal{D})$ is often solved by stochastic optimization methods.

Bayesian neural networks (BNNs) To apply the principles of Bayesian modeling to deep neural networks, one of the major ways is to introduce probability distributions to neural network weights, which will "Bayesianize" the DNNs and give us the *Bayesian neural networks* [205, 243]. This combines the best from both worlds: on one hand, modern approximate inference methods enable deep learning models to take advantage of Bayesian methods, and represent uncertainties in their predictions; on the other hand, the introduction of deep learning elements also harmonizes well with Bayesian approaches, since this allows one to use much more flexible probabilistic models.

Definition 2.3 (Bayesian neural neworks (BNN))**.** *Let $y = g(\mathbf{x}, \mathbf{w})$ be a neural network where $\mathbf{x}$ is the input, y is the output, and $\mathbf{w}$ denotes the weights. To build a Bayesian neural network (BNN), we place a prior $p(\mathbf{w})$ over $\mathbf{w}$. This prior models the* epistemic uncertainty *responsible for parameter uncertainties and can be explained away by observing more data. Furthermore, we add an observational noise $\varepsilon \sim \mathcal{N}(\varepsilon; 0, \sigma^2)$ to the output. This models the* aleatoric uncertainty*, which accounts for the intrinsic noise in the observation (i.e., data uncertainty). Then, a BNN is given by*

$$\log p(y|\mathbf{x}) = \log \mathbb{E}_{p(\mathbf{w})} p(y|\mathbf{x}, \mathbf{w}) = \log \int_{\mathbf{w}} \mathcal{N}(y; g(\mathbf{x}, \mathbf{w}), \sigma^2) p(\mathbf{w}) d\mathbf{w}, \tag{2.28}$$

where $\mathcal{N}(y; g(\mathbf{x}, \mathbf{w}), \sigma^2)$ is likelihood/data-generating distribution, $p(y|\mathbf{x}, \boldsymbol{\theta})$, as mentioned in Section 2.1.

Then, we can perform Bayesian inference with the BNN model: given a dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^n$, the goal of Bayesian inference is to compute the posterior $p(\mathbf{w}|\mathcal{D}) \propto p(\mathcal{D}|\mathbf{w})p(\mathbf{w})$. Since this is intractable, we often resort to approximate inference methods. In principle, all scalable approximate inference algorithms such as scalable VI, α-divergence minimization, (stochastic) expectation propagation can be applied. In particular, we may apply scalable variational inference by introducing a variational distribution $q(\mathbf{w})$ that approximates the posterior $p(\mathbf{w}|\mathcal{D})$. $q(\mathbf{w})$ is trained by maximizing (an noisy estimation of) the evidence lower bound (ELBO):

$$\mathcal{L}(q) := \mathbb{E}_q \log p(\mathcal{D}|\mathbf{w}) - \mathrm{D}_{\mathrm{KL}}[q(\mathbf{w})||p(\mathbf{w})]. \tag{2.29}$$

Once an optimal $q^\star$ has been found, we can approximate the Bayesian posterior predictive distribution on a test input $\mathbf{x}^*$ as follows:

$$p(y^*|\mathbf{x}^*, \mathcal{D}) \approx \frac{1}{M} \sum_{1 \le m \le M} \mathcal{N}(y^*; g(\mathbf{x}^*, \mathbf{w}^m), \sigma^2), \quad \mathbf{w}^m \sim q^\star(\boldsymbol{\theta}), \quad 1 \le m \le M. \tag{2.30}$$

In Equation (2.30), the approximation is twofold: 1), we have used $q^\star$ as an approximation to $p(\mathbf{w}|\mathcal{D})$; and 2), we have used MC samples to approximate the integration in Equation (2.2). It is now straightforward to see that Equation (2.30) performs (approximate) Bayesian model averaging (BMA) [138]: rather than working with a single point estimate of the weights $\hat{\mathbf{w}}$ and predicting using $\mathcal{N}(y^*; g(\mathbf{x}^*, \hat{\mathbf{w}}))$, BNNs with BMA considers all possible configurations of the weights (approximately represented by $\{\mathbf{w}^m\}_{1\leq m\leq M}$) simultaneously, and marginalises out the weights over $q^\star(\boldsymbol{\theta}) \approx p(\mathbf{w}|\mathcal{D})$.

Note that the objective in Equation (2.29) only provides a vague and general framework for variational inference in BNNs. Depending on how $q(\mathbf{w})$ is chosen, and how $\mathcal{L}(q)$ is estimated and optimized, we will arrive at different specific BNN algorithms. For the sake of scalability, many of these VI methods for BNNs rely on mean-field assumptions for $q(\mathbf{w})$ [117] as well as path-wise derivative estimators, which results in Bayes by Backprop (BBB) [29] and variational dropout [75]. Since the posterior $p(\mathbf{w}|\mathcal{D})$ is usually high multi-modal and complicated, the performance of VI is often sub-optimal, and suffers from over-confidence [344], variational over-pruning [341], over-parameterization [206], the cold-posterior effect [361], and other pathologies [70].

Remark (A brief history of VI for BNNs). Here we give a brief review of how VI methods for BNNs are developed in the literature. Variational inference for Bayesian neural network weights is first introduced by [117], which considered variational inference from an information-theoretic perspective (minimum description length). This idea is further developed by [17] in the case of non-factorial proxy in variational inference. More recently, [91] proposed the stochastic version of variational inference and derived one of the first scalable learning algorithms for Bayesian neural networks. This is later refined and extended by Bayes-by-Backprop (BBB) [29], which applied the reparameterization trick (path-wise gradient estimator) of [150] to obtain an unbiased estimate of the ELBO (and its gradients). This also allows the specification of non-Gaussian priors, which further boosted the performance of BNNs to a level comparable to other practical deep learning methods (such as dropout) at that time. Variational Bernoulli dropout (VDO) [75, 122] instead interprets dropout [323] as performing variational inference for BNNs. They also apply path-wise gradient estimator) of [150], but instead uses a mixture of Gaussians as $q(\mathbf{w})$, such that the columns of the weight matrices are randomly pruned to zero. This also introduces weight-correlations to $q(\mathbf{w})$, and halved the number of variational parameters. BBB and VDO are still among the most popular BNN algorithms to date.

2.3.2 Alternative BNN inference methods

The history of Bayesian neural networks is nothing but the history of the development of sophisticated approximate inference techniques. Apart from variational inference, several different approximate inference methods are proposed or applied to BNNs. We now give a brief review of some alternative Bayesian neural network inference methods developed prior to this thesis, including some recent advances in the past few years. Some of those methods will also be used as baselines in later chapters.

Modern Laplace approximations. The Laplace approximation [56, 204] is perhaps one of the first approximate inference methods in Bayesian neural networks, which approximates the BNN posterior by a multivariate Gaussian centered around its MAP mode. However, the need for calculating second-order derivatives (i.e., the Hessian matrix) made this vanilla approach computationally infeasible. Later, generalized Gauss-Newton (GGN) methods [301, 210] as well as its scalable block-diagonal/Kronecker-factored approximations [211, 32] have been introduced for the practical second-order optimization of deep neural networks. These approximation techniques have resulted in modern Laplace approximations for Bayesian neural networks [275, 71]. More recently, it has been argued that GGN approximation can be viewed as a linearized version of Bayesian neural networks, which popularized another branch of works that approximate the Bayesian posterior predictive distribution by that of the linearized model [71, 145, 127].

Expectation propagation, and other bounds/divergences. Probabilistic backpropagation (PBP) [111] is an approximate inference algorithm for Bayesian neural networks, that performs assumed density filtering (ADF) on training sets. Similar implementations of PBP based on expectation propagation and stochastic expectation propagation (SEP) were also proposed in [178]. In general, PBP has demonstrated superior performances over VI [91] in both accuracy and speed. Another EP-based method is the BB-α [113] introduced in Section 2.2.3, which directly minimizes the energy function $\mathcal{L}_\alpha$ of BNNs, instead of doing iterative message passing. In the context of variational Bernoulli dropout, an alternative reparameterization of the α-divergence objectives is also proposed and studied, which enables simple implementations for BNNs. Following this line of work, more variational bounds/divergences have been introduced for approximate inference. Well-known examples include Renyi's α-divergence [180], f-divergence [351], χ-divergence [60], and operator variational inference [267]. More recently, meta-learning strategies that directly learn to select the divergence metrics suitable for specific tasks have also been explored [377].

Flexible approximations for BNN posteriors. Most approximate inference methods (based on VI and EP) that we mentioned before rely on mean-field approximations to increase scalability. As observed both theoretically [71, 68] and empirically [127, 284, 251], mean-field approximations have limited expressiveness and tend to under-estimate uncertainties in regions where observations are sparse. Therefore, a promising line of research considers more expressive posterior approximations such as structured weight-matrix posterior approximations [195, 329, 275], normalizing flows [196, 271], or even variational dropouts that implicitly consider weight-space correlations [75, 77]. Another possibility is the application of implicit VI methods to BNNs. As reviewed in Section 2.2.4, implicit VI is a powerful tool for complicated approximate inference tasks. However, when applied to BNNs, implicit VI becomes computational infeasible: for any non-trivial BNNs, it is challenging to design implicit distributions for high-dimensional neural network weights [308, 218, 319]. More importantly, most existing gradient estimators for $\mathrm{D_{KL}}[q||p]$ are less efficient for such high dimensional distributions [308, 379, 85]. In this regard, implicit VI is mainly applied to BNNs when jointly used with mean-field assumptions [234, 181, 330]; only a handful of implicit VI based methods have been applied to BNNs with non-mean-field variational families [308].

Dimension reduction for BNN Since expressive VI with non-mean-field approximations is difficult for high-dimensional models like BNNs, various of methods are developed to perform inference in a lower-dimensional subspace of the neural network weights. A straightforward method is to perform inference only for the last layer of BNNs [274, 251, 248, 160, 34]. This idea was further extended to sub-network inference methods [51], which provide Bayesian inference over a subset of weights using certain selection strategies. Another similar idea is based on performing inference over a low-dimensional subspace of neural network weights, e.g. either two-stage procedure (via stochastic weight averaging + principal component projection [131]), or end-to-end optimization (low-rank assumptions on weight matrices [64]).

2.3.3 Gaussian Processes

Gaussian Processes [269], as a popular example of Bayesian nonparametrics, provide a principled probabilistic framework for non-parametric Bayesian inference over functions. This is achieved by imposing rich and flexible nonparametric priors over functions of interest. As flexible and interpretable function approximators, their Bayesian nature also enables GPs to provide valuable information of uncertainties regarding predictions for intelligence systems, all wrapped up in a single, *exact* closed-form solution to the posterior inference problem.

We briefly introduce GPs for regression. Consider again the scenario of having a training set $\mathbf{v} = \mathcal{D} := \{(\mathbf{x}_n, y_n)\}_{n=1:N}$, where $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ is the set of inputs, and $\mathbf{y} = \{y_1, y_2, ..., y_N\}$. A Gaussian Process model assumes that y_n is generated according the following procedure: firstly a function $f(\cdot)$ is drawn from a Gaussian Process $\mathcal{GP}(\mathfrak{M}(\cdot), \mathfrak{K}(\cdot,\cdot))$ (to be defined later). Then for each input data $\mathbf{x}_n$, the corresponding y_n is then drawn according to

$$y_n = f(\mathbf{x}_n) + \varepsilon_n, \;\; \varepsilon \sim \mathcal{N}(0, \sigma^2), \;\; n = 1, \cdots, N.$$

A Gaussian Process is then a nonparametric distribution defined over the space of functions, as defined next.

Definition 2.4 (Gaussian Processes)**.** *A Gaussian process (GP) is a collection of random variables, any finite number of which have a joint Gaussian distributions. A Gaussian Process is fully specified by its mean function* $\mathfrak{M}(\cdot) : \mathbb{R}^D \to \mathbb{R}$ *and covariance function* $\mathfrak{K}(\cdot,\cdot) : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$*, such that any finite collection of function values* $\mathbf{f}$ *are distributed as Gaussian distribution* $\mathcal{N}(\mathbf{f}; \mathbf{m}, \mathbf{K_{ff}})$*, where* $(\mathbf{m})_n = \mathfrak{M}(\mathbf{x}_n)$*,* $(\mathbf{K_{ff}})_{n,n'} = \mathfrak{K}(\mathbf{x}_n, \mathbf{x}_{n'})$*.*

Now, given a set of observational data $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$, we can perform probabilistic inference and assign posterior probabilities over all plausible functions that might have generated the data. Under the setting of regression, given a new test point input data $\mathbf{x}_*$, we are interested in posterior distributions over f_*. Fortunately, this posterior distribution of interest admits a closed-form solution $f_* \sim \mathcal{N}(\mu_*, \Sigma_*)$:

$$\mu_* = \mathbf{m} + \mathbf{K}_{\mathbf{x}_*\mathbf{f}}(\mathbf{K_{ff}} + \sigma^2\mathbf{I})^{-1}(\mathbf{y} - \mathbf{m}), \tag{2.31}$$

$$\Sigma_* = \mathbf{K}_{\mathbf{x}_*\mathbf{x}_*} - \mathbf{K}_{\mathbf{x}_*\mathbf{f}}(\mathbf{K_{ff}} + \sigma^2\mathbf{I})^{-1}\mathbf{K}_{\mathbf{f}\mathbf{x}_*}, \tag{2.32}$$

In our notation, $(\mathbf{y})_n = y_n$, $(\mathbf{K}_{\mathbf{x}_*\mathbf{f}})_n = \mathfrak{K}(\mathbf{x}_*, \mathbf{x}_n)$, and $\mathbf{K}_{\mathbf{x}_*\mathbf{x}_*} = \mathfrak{K}(\mathbf{x}_*, \mathbf{x}_*)$. Although the Gaussian Process regression framework is theoretically very elegant, in practice its computational burden is prohibitive for large datasets since the matrix inversion $(\mathbf{K_{ff}} + \sigma^2\mathbf{I})^{-1}$ takes $\mathcal{O}(N^3)$ time due to Cholesky decomposition. Once this matrix inversion is done, predictions at test time can be made with a cost of $\mathcal{O}(N)$ to compute the posterior mean μ_* and a cost of $\mathcal{O}(N^2)$ to compute the posterior variance Σ_*, respectively.

Despite the success and popularity of GPs (and other Bayesian non-parametric methods) in the past decades, their $\mathcal{O}(N^3)$ computation and $\mathcal{O}(N^2)$ storage complexities makes their application to large-scale datasets impractical. Therefore, people often resort to complicated approximate methods, e.g. see [304, 261, 315, 335, 109, 38, 37, 287, 46, 343].

Another critical issue to be addressed is the representational power of GP kernels. It has been argued that local kernels commonly used for nonlinear regressions are not able to obtain hierarchical representations for high dimensional data [23], which limits the usefulness of Bayesian nonparametric models in complicated tasks. A number of solutions were proposed, including deep GPs [49, 47, 36], the design of expressive kernels [346, 65, 337], and the hybrid model with features from deep neural nets as the input to a GP [116, 367]. However, the first two approaches still struggle to model complex high dimensional data such as text and images; and in the third approach, the advantages of fully Bayesian approaches are not fully exploited.

2.3.4 Generative models and variational EM

So far we have been discussing Bayesian machine learning in the context of supervised learning, which requires labeled data $\mathcal{D} = \{\mathbf{x}_n, y_n\}_{1 \leq n \leq N}$ to perform Bayesian inference and prediction. However, most data generated from the real world comes with no labels. It is therefore, very crucial for us to be able to make sense of those data in an *unsupervised* way. *Generative models*, also known as the latent variable models, provide a probabilistic approach to perform unsupervised learning with such unlabelled data.

Definition 2.5 (Generative models)**.** *A generative model is a probabilistic distribution $p_{\boldsymbol{\theta}}(\mathbf{x}_n)$ parameterized by learnable model parameters $\boldsymbol{\theta}$, that specifies the likelihood of observing a particular data sample, $\mathbf{x}_n$.*

Definition 2.6 (Latent variable models)**.** *A latent variable model is a generative model, where $p_{\boldsymbol{\theta}}(\mathbf{x}_n)$ is defined by assigning local latent variables $\mathbf{z}_n$ for each data instance $\mathbf{x}_n$. In other words, given a non-labelled dataset $\mathcal{D} = \{\mathbf{x}_n\}_{1 \leq n \leq N}$, we have that*

$$\log p_{\boldsymbol{\theta}}(\mathcal{D}) = \sum_n \log \int_{\mathbf{z}_n} p_{\boldsymbol{\theta}}(\mathbf{x}_n, \mathbf{z}_n) d\mathbf{z}_n. \tag{2.33}$$

Without loss of generality, in this thesis we assume all generative models are latent variable models.

Remark (Examples). In this thesis we constrain our discussions to a class of generative models defined by the following factorization:

$$\log p(\mathcal{D}) = \sum_n \log \int_{\mathbf{z}_n} p(\mathbf{x}_n|\mathbf{z}_n)p(\mathbf{z}_n)d\mathbf{z}_n. \tag{2.34}$$

An example of such model is probabilistic principle component analysis (PPCA), which is defined by the following linear model:

$$\begin{aligned} p(\mathbf{z}_n) &= \mathcal{N}(\mathbf{z}; 0, \mathbf{I}), \\ p(\mathbf{x}_n|\mathbf{z}) &= \mathcal{N}(\Lambda\mathbf{z}, \sigma\mathbf{I}), \\ n &= 1, ..., N, \end{aligned}$$

where Λ is a linear transformation matrix, σ is the standard deviation of the Gaussian noise, and the learnable model parameters are given by $\boldsymbol{\theta} = \{\Lambda, \sigma\}$.

Alternatively, another useful way to specify $p(\mathbf{x}_n, \mathbf{z}_n)$ is to use the so-called energy-based models

$$p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) = \frac{e^{-E_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z})}}{Z_{\boldsymbol{\theta}}}.$$

One famous example are Restricted Boltzmann machines (RBMs) [115], where $E_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})$ is given by

$$E_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T W\mathbf{z} + \mathbf{b}_{\mathbf{x}}^T\mathbf{x} + \mathbf{b}_{\mathbf{z}}^T\mathbf{z}.$$

One central task for unsupervised learning with generative models is *learning*, which is to perform maximum likelihood learning over $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}^\star = \arg\max_{\boldsymbol{\theta}} \sum_n \log p_{\boldsymbol{\theta}}(\mathbf{x}_n). \tag{2.35}$$

Note that since we only have access to $\{\mathbf{x}_n\}_{1\leq n\leq N}$, and the latent variables $\{\mathbf{z}_n\}_{1\leq n\leq N}$ are unobserved, they need to be *marginalized out* during maximum likelihood learning:

$$\boldsymbol{\theta}^\star = \arg\max_{\boldsymbol{\theta}} \sum_n \log p_{\boldsymbol{\theta}}(\mathbf{x}_n) = \sum_i \log \int_{\mathbf{z}_n} p_{\boldsymbol{\theta}}(\mathbf{x}_n|\mathbf{z}_n)p_{\boldsymbol{\theta}}(\mathbf{z}_n)d\mathbf{z}_n. \tag{2.36}$$

The above optimization problem requires to solve N integration problems $\int_{\mathbf{z}_n} p_{\boldsymbol{\theta}}(\mathbf{x}_n|\mathbf{z}_n)p_{\boldsymbol{\theta}}(\mathbf{z}_n)d\mathbf{z}_n$, which are usually analytically intractable except for certain special cases. To address this, a variational Expectation-Maximization (VEM) algorithm [19] can be applied, which combines the idea of variational inference (over $\mathbf{z}_n$) with maximum likelihood learning (over $\mathbf{z}$). Similar to VI, in VEM, a set of approximate posteriors $\{q_{\boldsymbol{\lambda}_n}(\mathbf{z}_n)\}_{1\leq n\leq N}$ are used to approximate each of the true posterior distributions, $\{p_{\boldsymbol{\theta}}(\mathbf{z}_n|\mathbf{x}_n)\}_{1\leq n\leq N}$, respectively. Then, variational EM optimizes the below evidence lower bound $\mathcal{L}$:

$$\sum_n \log p_{\boldsymbol{\theta}}(\mathbf{x}_n) \geq \mathcal{L}(\{\boldsymbol{\lambda}_n\}_{1\leq n\leq N}, \boldsymbol{\theta}) = \sum_n \mathbb{E}_{q_{\boldsymbol{\lambda}_n}(\mathbf{z}_n)}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_n, \mathbf{z}_n)}{q_{\boldsymbol{\lambda}_n}(\mathbf{z}_n)}\right]. \tag{2.37}$$

Remark (Global vs. local latent variables). Note that the ELBO in Equation (2.37) is different from that of a BNN in Equation (2.29). In a BNN, all data instances share the same latent variable, i.e. the neural network weights $\mathbf{w}$. We call $\mathbf{w}$ the *global latent variables*. Hence, for datasets of N data instances, the BNN bound in Equation (2.29) only has exactly one approximate factor (i.e., $q_{\boldsymbol{\lambda}}(\mathbf{w})$) and one KL-term. However, in a generative model, each data instance $\mathbf{x}_n$ in the dataset is assigned with different latent variables $\mathbf{z}_n$. This creates N different approximate factors $q_{\boldsymbol{\lambda}_n}(\mathbf{z}_n)$ as well as N KL terms in the ELBO 2.37.

The optimization is often done by recursively alternating between the following two steps:

- **E step (VI step):** $\boldsymbol{\lambda}_n^{new} = \arg\max_{\boldsymbol{\lambda}_n} \mathcal{L}(\{\boldsymbol{\lambda}_n\}_{1\leq n\leq N}, \boldsymbol{\theta}) = \arg\max_{\boldsymbol{\lambda}_n} \mathbb{E}_{q_{\boldsymbol{\lambda}_n}(\mathbf{z}_n)}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_n,\mathbf{z}_n)}{q_{\boldsymbol{\lambda}_n}(\mathbf{z}_n)}\right]$, for all $1 \leq n \leq N$. It is then trivial to verify that equality in (2.37) holds if and only if $q_{\boldsymbol{\lambda}}(\mathbf{z}_n) = p_{\boldsymbol{\theta}}(\mathbf{z}_n|\mathbf{x}_n)$ for all $1 \leq n \leq N$.
- **M step (learning step):** $\boldsymbol{\theta}^{new} = \arg\max_{\boldsymbol{\theta}} \mathcal{L}(\{\boldsymbol{\lambda}_n\}_{1\leq n\leq N}, \boldsymbol{\theta})$.

When exact inference is possible, i.e. $q_{\boldsymbol{\lambda}}(\mathbf{z}_n) = p_{\boldsymbol{\theta}}(\mathbf{z}_n|\mathbf{x}_n)$ for all $1 \leq n \leq N$, the above VEM procedure never decreases the likelihood since

$$\begin{aligned}&\mathcal{L}(\{\boldsymbol{\lambda}_n^{old}\}_{1\leq n\leq N}, \boldsymbol{\theta}^{old}) \leq \mathcal{L}(\{\boldsymbol{\lambda}_n^{new}\}_{1\leq n\leq N}, \boldsymbol{\theta}^{old}) = \sum_n \log p_{\boldsymbol{\theta}^{old}}(\mathbf{x}_n)\\ \leq &\mathcal{L}(\{\boldsymbol{\lambda}_n^{new}\}_{1\leq n\leq N}, \boldsymbol{\theta}^{new}) \leq \sum_n \log p_{\boldsymbol{\theta}^{new}}(\mathbf{x}_n).\end{aligned}$$

Unfortunately, exact inference is intractable in most cases, and the approximate inference used in the E step will break the condition $q_{\boldsymbol{\lambda}}(\mathbf{z}_n) = p_{\boldsymbol{\theta}}(\mathbf{z}_n|\mathbf{x}_n)$. Therefore, in practice, the likelihood of VEM is not guaranteed to always have non-negative increments. However, under mild conditions, the VEM procedure is still convergent and will converge to a local optimum of $\mathcal{L}$.

Note that variational inference is not the only candidate for E-step. An alternative inference method is Monte Carlo inference : we can draw MC samples from $p(\mathbf{z}_n|\mathbf{x}_n)$ using MCMC methods [7, 236, 244], and perform gradient estimation:

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} \sum_n \log p(\mathbf{x}_n) &= \sum_n \mathbb{E}_{p(\mathbf{z}|\mathbf{x}_n)}(\nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}_n, \mathbf{z})) \\ &\approx \sum_{n,m} \nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}_n, \mathbf{z}_{n,m}), \quad \mathbf{z}_{n,m} \sim p(\mathbf{z}_n|\mathbf{x}_n), \quad m = 1, ..., M.\end{aligned}$$

2.3.5 Deep generative models and amortized variational inference

The central challenge of generative modeling is how to learn *flexible* and *accurate* generative models, that can describe complicated patterns in the data. Following the recent advantages of deep learning, one promising approach is to combine generative models with deep learning models, which enables us to take advantage of the representation power of deep learning models. This is usually done by using deep neural networks to transform the latent variables $\mathbf{z}_n$ into the observables, $\mathbf{x}_n$:

$$p(\mathbf{z}_n) = \mathcal{N}(\mathbf{z}_n; \mathbf{0}, \mathbf{I}), \tag{2.38}$$

$$p_{\boldsymbol{\theta}}(\mathbf{x}_n|\mathbf{z}_n) = \mathcal{N}(\mathbf{x}_n; g_{\boldsymbol{\theta}}(\mathbf{z}_n), \sigma\mathbf{I}), \tag{2.39}$$

$$n = 1, ..., N,$$

where $g_{\boldsymbol{\theta}}(\cdot)$ is a deep neural network parameterized by $\boldsymbol{\theta}$ that transforms $\mathbf{z}_n$ into the mean parameter of the Gaussian distribution $p_{\boldsymbol{\theta}}(\mathbf{x}_n|\mathbf{z}_n)$. The above deep generative model can also be written in terms of the following sampling process:

$$\mathbf{x}_n = g_{\boldsymbol{\theta}}(\mathbf{z}_n) + \varepsilon_n, \quad \mathbf{z}_n \sim p(\mathbf{z}_n), \quad \varepsilon_n \sim \mathcal{N}(\varepsilon_n; 0, \sigma\mathbf{I}), \tag{2.40}$$

which takes the form of *implicit distributions*, introduced in Section 2.2.4, and that are known to excel at designing highly expressive probabilistic distributions. The distribution $p_{\boldsymbol{\theta}}(\mathbf{x}_n|\mathbf{z}_n)$ is often referred as the decoder in the context of autoencoders [150], or the *generator* in the context of GANs [88].

Just like any generative models, learning and inference for deep generative models, as defined in Equation (2.39) and (2.40), can also be performed by optimizing the evidence lower bound defined in Equation (2.37):

$$\sum_n \log p_{\boldsymbol{\theta}}(\mathbf{x}_n) \geq \mathcal{L}(\{\boldsymbol{\lambda}_n\}_{1\leq n\leq N}, \boldsymbol{\theta}) = \sum_n \mathbb{E}_{q_{\boldsymbol{\lambda}_n}(\mathbf{z}_n)}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_n, \mathbf{z}_n)}{q_{\boldsymbol{\lambda}_n}(\mathbf{z}_n)}\right] \tag{2.41}$$

In the context of deep generative models, variational EM is computationally infeasible. To begin with, it requires to store and optimize $\mathcal{O}(N)$ variational parameters, $\{\boldsymbol{\lambda}_n\}_{1\leq n\leq N}$, which clearly does not scale to big data. Moreover, it does not scale to new unseen data examples. Imagine that we have learned a deep generative model for images and, given a new image $\mathbf{x}^*$, we wish to infer its latent structure, $p(\mathbf{z}^*|\mathbf{x}^*)$. In this scenario, variational EM will create a new approximate distribution $q_{\boldsymbol{\lambda}^*}(\mathbf{z}^*)$, and optimize $\boldsymbol{\lambda}^*$ with gradient descent iterations. This procedure is slow in practice and does not handle a large number of new images.

The amortized inference method was proposed [150, 52] to address these practical challenges. In amortized inference, we can directly tie all local variational parameters $\boldsymbol{\lambda}_n$ to be the same, $\boldsymbol{\lambda}$. To compensate for this parameter-tying, the observables $\mathbf{x}_n$ are used as additional input to the approximate posteriors. This gives the following amortized approximation

$$q_{\boldsymbol{\lambda}^n}(\mathbf{z}_n) \approx q_{\boldsymbol{\lambda}}(\mathbf{z}_n|\mathbf{x}_n).$$

In other words, $q_{\boldsymbol{\lambda}}(\mathbf{z}_n|\mathbf{x}_n)$ tries to *predict* (the sufficient statistics of) the approximate posterior distribution $q_{\boldsymbol{\lambda}^n}(\mathbf{z}_n)$ by taking as input the observable $\mathbf{x}_n$. $q_{\boldsymbol{\lambda}}(\mathbf{z}_n|\mathbf{x}_n)$ is often referred as the encoder, or *recognition net/inference net*. An important way to parameterize $q_{\boldsymbol{\lambda}}(\mathbf{z}_n|\mathbf{x}_n)$ is via neural networks:

$$q_{\boldsymbol{\lambda}}(\mathbf{z}_n|\mathbf{x}_n) = \mathcal{N}(\mathbf{z}_n; \mu_{\boldsymbol{\lambda}}(\mathbf{x}_n), \operatorname{diag}(\sigma_{\boldsymbol{\lambda}}(\mathbf{x}_n))), \tag{2.42}$$

where both $\mu_{\boldsymbol{\lambda}}(\cdot)$ and $\sigma_{\boldsymbol{\lambda}}(\cdot)$ are deep neural nets that maps the observables $\mathbf{x}_n$ to the mean and standard deviations of the distribution $q_{\boldsymbol{\lambda}}(\mathbf{z}_n|\mathbf{x}_n)$.

Now, with the amortized approximation $q_{\boldsymbol{\lambda}^n}(\mathbf{z}_n) \approx q_{\boldsymbol{\lambda}}(\mathbf{z}_n|\mathbf{x}_n)$, the amortized variational lower bound becomes:

$$\sum_n \log p_{\boldsymbol{\theta}}(\mathbf{x}_n) \geq \mathcal{L}(\boldsymbol{\lambda}, \boldsymbol{\theta}) = \sum_n \mathbb{E}_{q_{\boldsymbol{\lambda}}(\mathbf{z}_n|\mathbf{x}_n)}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_n, \mathbf{z}_n)}{q_{\boldsymbol{\lambda}}(\mathbf{z}_n|\mathbf{x}_n)}\right]. \tag{2.43}$$

Finally, instead of the two-step recursive procedure in variational EM, we can now apply the advanced techniques of scalable variational inference (Section 2.2.2) to the amortized objective (2.43), and perform gradient descent on both $\boldsymbol{\theta}$ and $\boldsymbol{\lambda}$ simultaneously. This reproduces the variational auto-encoder model of [150].

Amortized inference certainly sacrifices representation power due to the parameter tying approximations $\boldsymbol{\lambda}_n \approx \boldsymbol{\lambda}$. However, it brings several major advantages for generative modeling. The most direct impact is the drastically reduced memory cost, making this new formulation highly scalable to large datasets. Also, it allows for fast down-top inference on new data, without worrying about the mixing problem of MCMC or the additional cost of optimization steps in variational EM. Moreover, the parameterization $q_{\boldsymbol{\lambda}}(\mathbf{z}_n|\mathbf{x}_n)$ decouples the local structures (represented by $\mathbf{x}_n$) and the global structure (represented by $\boldsymbol{\lambda}$ shared across all data instances) of q [176]. This potentially helps the optimization of the variational parameters $\boldsymbol{\lambda}$ as well as the model parameters $\boldsymbol{\theta}$.

2.3.6 Helmholtz machines and the wake-sleep algorithm

The idea of amortized inference is not a new invention of deep generative models/variational autoencoders. It originally appeared in the wake-sleep algorithm of the Helmholtz machine [52, 114], which has modeling assumptions very similar to those of modern variational auto-encoders. The wake-sleep algorithm in the Helmholtz machine is an alternative method for learning deep generative models. Similar to variational EM, it recursively alternates between two phases called the wake phase and the sleep phase:

- **The wake phase:** in this phase we optimize the model parameters $\boldsymbol{\theta}$, by maximizing the amortized ELBO of Equation (2.43) w.r.t. $\boldsymbol{\theta}$ only.
- **The sleep phase:** Recall that in normal VI/variational EM, we train the recognition network $q_{\boldsymbol{\lambda}}$ by optimizing

$$\begin{aligned}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\mathbf{z}|\mathbf{x})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})] \approx \\ \sum_n \mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\mathbf{z}_n|\mathbf{x}_n)||p_{\boldsymbol{\theta}}(\mathbf{z}_n|\mathbf{x}_n)], \quad \mathbf{x}_n \sim p_{\text{data}}(\mathbf{x}),\end{aligned} \tag{2.44}$$

  where $p_{\text{data}}(\mathbf{x})$ is the ground truth data distribution that generated the training set, $\mathcal{D}$. On the contrary, the sleep phase of the wake-sleep algorithm optimizes the following reverse KL-divergence:

$$\mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{x})}\mathrm{D}_{\mathrm{KL}}[p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})||q_{\boldsymbol{\lambda}}(\mathbf{z}|\mathbf{x})]. \tag{2.45}$$

Note that the outer expectation is taken using the model distribution $p_{\boldsymbol{\theta}}(\mathbf{x})$ instead of the data distribution $p_{\text{data}}(\mathbf{x})$. This quantity can be optimized using gradient descent, which computes the gradients as below:

$$\begin{aligned}\nabla_{\boldsymbol{\lambda}}\mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{x})}\mathrm{D}_{\mathrm{KL}}[p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})||q_{\boldsymbol{\lambda}}(\mathbf{z}|\mathbf{x})] &= \nabla_{\boldsymbol{\lambda}}\mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) - \log q_{\boldsymbol{\lambda}}(\mathbf{z}|\mathbf{x})\right] \quad (2.46)\\ &= \mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z})}\left[-\nabla_{\boldsymbol{\lambda}}\log q_{\boldsymbol{\lambda}}(\mathbf{z}|\mathbf{x})\right]. \quad (2.47)\end{aligned}$$

Therefore, the intuition behind the sleep phase is to perform maximum likelihood learning of the recognition model $q_{\boldsymbol{\lambda}}(\mathbf{z}|\mathbf{x})$, using "dreamed" samples obtained from the model, $p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z})$. This avoids taking a derivative w.r.t. the probability measure of the expectation operator $\mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z})}$, which makes the gradient estimation become more well-behaved. However, it comes with a cost of breaking a unified single objective, i.e. the evidence lower bound $\mathcal{L}$.

Remark (Sleep phase $\boldsymbol{\lambda}$ updates and wake phase $\boldsymbol{\lambda}$ updates). The sleep phase update described above is usually referred to as the sleep phase $\boldsymbol{\lambda}$ update [31]. An alternative update rule is the so called *wake phase* $\boldsymbol{\lambda}$ *updates*, which optimizes

$$\begin{aligned}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathrm{D}_{\mathrm{KL}}[p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})||q_{\boldsymbol{\lambda}}(\mathbf{z}|\mathbf{x})] &= \mathbb{E}_{p_{\text{data}}(\mathbf{x})p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) - \log q_{\boldsymbol{\lambda}}(\mathbf{z}|\mathbf{x})\right] \quad (2.48)\\ &\approx \mathbb{E}_{p_{\text{data}}(\mathbf{x})q_{\boldsymbol{\lambda}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) - \log q_{\boldsymbol{\lambda}}(\mathbf{z}|\mathbf{x})\right]. \quad (2.49)\end{aligned}$$

That is, it performs maximum likelihood learning of the recognition model $q_{\boldsymbol{\lambda}}(\mathbf{z}|\mathbf{x})$ using samples from the data distribution. This approach is sometimes more advantageous than the sleep phase $\boldsymbol{\lambda}$ update. Since $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ is unknown, it is often replaced by the biased approximation $q_{\boldsymbol{\lambda}}(\mathbf{z}|\mathbf{x})$ (which needs further bias-reduction techniques such as importance sampling).

2.4 Conclusion

In this chapter, we gave a systematic review of the basic techniques for uncertainty quantification in machine learning, including Bayesian modeling, approximate inference, and important models for both supervised and unsupervised learning. Some of these techniques will be further applied, developed, and compared in the next chapters (3, 4, and 5). During this process, we hope to give the readers a better idea of existing advancements and challenges in Bayesian machine learning and Bayesian deep learning, with most of them usually centered around the scalability, flexibility, and accuracy of approximate inference methods. In the rest of the thesis, we will present new contributions along those dimensions.

Part A

Approximate Inference in Function Space

Chapter 3

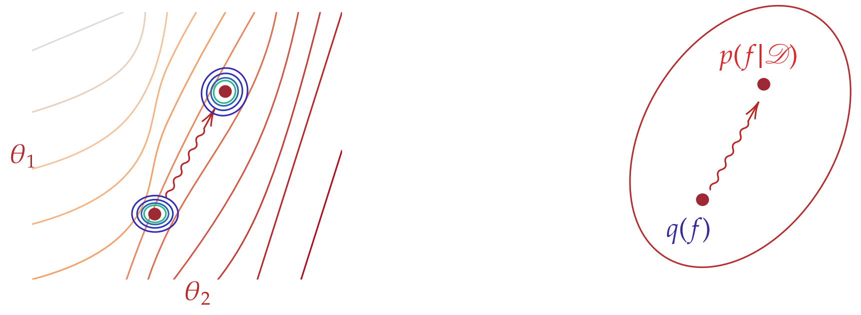
Why Function Space Inference?

In Part A (Chapter 3, 4 and 5) of the thesis, we will first focus on supervised learning problems, and try to address Challenge I identified Chapter 1, i.e., obtaining efficient and accurate *model uncertainty* in supervised learning problems. Estimating model uncertainty in a proper way is a crucial task, as given a set of observations, there might exist a number of potential models that could fit the data equally well. Therefore, we are uncertain about which predictive model to choose in the end, which will effect the decisions that we will be making based on such models. As introduced in Chapter 2, this type of uncertainty can in principle be quantified by performing Bayesian inference over model parameters.

In this chapter, we will first take Bayesian neural networks as an example, and review some of the difficulties of performing Bayesian inference in parameter space. Undoubtedly, model unidentifiability/overparameterization is one of the major obstacles to improving the quality of approximate inference. Thus, we will argue that we may consider performing inference in function space, as it is equivalent to performing inference on *minimal sufficient parameters*, hence resolving the problem of model non-identifiability and posterior inconsistency (see Figure 3.1 for preview).

3.1 Challenges in parameter-space inference

As introduced in Chapter 2, approximate inference for high dimensional, complicated models is still very challenging and problematic. To kickstart the discussion, let us consider Bayesian neural networks as a motivating example, where a number of pathologies/challenges have been observed when performing scalable approximate inference. Below we only list a few of them:



$q^\star(\theta) = \arg\min_q \mathrm{D}_{\mathrm{KL}}[q(\theta)||p(\theta|\mathscr{D})]$ parameter space (*over-parameterized*)

$q^\star(f) = \arg\min_q \mathrm{D}_{\mathrm{KL}}[q(f)||p(f|\mathscr{D})]$ function space (*minimal sufficient*)

Figure 3.1 Chapter 3 preview: parameter space inference versus function space inference. **Left**: parameter space VI minimizes KL divergence over the parameters $\boldsymbol{\theta} = \{\theta_1, \theta_2\}$, which suffers from overparameterization and model unidentifiability. **Right**: on the contrary, function space inference directly performs Bayesian inference in the space of functions, and minimizes KL divergence over functions. This is equivalent to perform inference on *minimal sufficient parameters*, which forms an identifiable reparameterization of the regression model, and resolves the pathologies of posterior inconsistency of parameter space inference.

- Overparameterization and unidentifiability. Deep neural networks are over-parameterized models with up to billions of parameters [35]. This often leads to model unidentifiability, meaning that there exist multiple modes that will give rise to similar predictive distributions. A special case of neural network unidentifiability is the so-called weight symmetries in neural networks [258, 206]. One example is the *discrete symmetries*: permuting two hidden units of the same layer will leave the neural network output invariant. Another example is *continuous symmetries*: in *ReLU* networks, the output of a hidden unit will be invariant if the input weights are scaled by a factor α while the output is scaled by $1/\alpha$ simultaneously. These symmetries (especially continuous symmetries) will create a desperate situation for Bayesian inference: they generate an exponentially large number of modes in the BNN posterior distribution, which makes it very difficult to perform approximate inference accurately.

- Expressiveness of variational family. When performing VI, it has been observed that mean-field BNNs have limited expressiveness, and tend to underestimate in-between uncertainty [70, 68]. While it is generally possible to introduce non-factored or non-

Gaussian approximations (see Section 2.3.2), it is much more difficult to design such posterior approximations for BNNs due to their high volume of parameters.

- Specification of priors. Neural network weights usually do not have scientific implications, which makes it difficult to specify meaningful priors that will induce certain favorable predictive behaviors [69]. What makes it even worse is, the choice of priors has a substantial impact on Bayesian neural networks [313, 361, 72], especially for out-of-distribution detection [313].

- Cold posterior effect. It has been empirically observed that BNN posteriors tend to achieve better performance when the posterior over weights are sharpened/tempered by a temperature $T < 1$ [361, 359, 94]:

$$\log p(\mathbf{w}|\mathcal{D})^{\frac{1}{T}} \propto \frac{1}{T}\log p(\mathcal{D}|\mathbf{w}) + \frac{1}{T}\log p(\mathbf{w}). \tag{3.1}$$

  The cold posterior could be caused by a number of issues, for instance inaccurate inference [132], model mis-specification (especially priors [72, 361]), data curation [3], data augmentation [132], etc. We point out that a similar effect has been discovered not only in BNNs, but also for other model classes as well [94].

Although the challenges listed above are quite prevalent in Bayesian neural network applications, we note that they are not tied to BNNs only. Many of these problems are in part caused by more general properties of probabilistic models, namely, *model non-identifiability* [156, 278], which is quite commonly observed for both parametric and non-parametric models. We will discuss identifiability and related theoretical results in the next sections and will show that performing inference in *function space* will resolve the problem of identifiability and posterior consistency.

3.2 Interlude: indentifiability, minimal sufficiency, and inference consistency

Roughly speaking, the identifiability of a probabilistic model $P_\theta(\mathbf{x})$ characterizes the property that the correspondence between parameters $\theta \in \Theta$ and corresponding distributions P_θ is one-to-one. Its formal definition is given in the following remark:

Remark (Identifiability) The formal definition of identifiability has been presented and generalized in different contexts, such as [278, 255, 6]. Here, we stick to the widely used definition based on observational equivalence [278]:

Definition 3.1 (Observational equivalence)**.** *Assume $\mathcal{P} = \{P_\theta | \theta \in \Theta\}$ is a collection of probability measures defined on some well behaved measurable space $(\mathcal{X}, \mathcal{B})$ [a], and indexed by elements in some (possibly infinite dimensional) measurable parameter space $\theta \in \Theta$ [b] [c]. Then, two parameter values θ_1 and θ_2 are said to be* observational equivalent*, if $P_{\theta_1}(\mathbf{x}) = P_{\theta_1}(\mathbf{x})$ a.s. in $\mathbf{x} \in \mathcal{X}$.*

Definition 3.2 (Identifiability)**.** *The model $\mathcal{P}$ is said to be identifiable (on Θ), if all observationally equivalent parameter pairs have the same value.*

[a]For instance, in Bayesian analysis we often assume $\mathcal{X}$ to be polish space, with $\mathcal{B}$ to be a Borel σ-algebra, which ensures that $P_\theta(\cdot)$ is well defined, i.e., a regular conditional probability distribution.

[b]Similarly, we will also assume Θ to be a polish space with some Borel σ-algebra. This will ensure the regularity on the posterior, $P(\theta|\mathbf{x})$.

[c]In probability theory, both $\mathcal{P}$ and Θ are referred as *the model*.

The reason that we bring up the concept of identifiability is that model identifiability relates to one of the most important asymptotic properties of statistical inference: the *posterior consistency.* That is, whether the posterior distribution will robustly contract to the *true parameter* of data distribution under perfect information, regardless of the choice of prior belief. In the language of probability theory, it can be formally described as follows:

Definition 3.3 (Posterior consistency/convergence)**.** [1] *Assume $\mathcal{D}_n = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ is a collection of n i.i.d. random observational variables distributed according to a probability measure (called the data distribution) $P_{\theta^\star}$, parameterized by a ground truth parameter $\theta^\star$. Additionally, assume that $\mathcal{P}$ is equipped with a metric, d.* [2] *Then, the posterior $P(\theta|\mathcal{D}_n)$ is said to be consistent if $P(\theta|\mathcal{D}_n)$ weakly converges to Dirac measure $\delta_{\theta^\star}$ when $n \to \infty$, $P_{\theta^\star}$-almost surely.*

In other words, Definition 3.3 formalizes the idea that "perfect knowledge should be able to override prior beliefs asymptotically" [153]. This property provides an important theoretical justification for Bayesian approaches to practical problems, such as regression and classification. Due to the crucial importance of the posterior consistency of $P(\theta|\mathcal{D}_n)$, its

[1]Depending on the context, there are different definitions of posterior consistency [345, 168]. Here we only present one of the mostly adopted ones.

[2]This assumption can be further removed. If we do so, the definition of consistency becomes: $P(\theta|\mathcal{D}_n)$ is consistent, if for all neighbourhood U of $P_{\theta^\star}$ in $\mathcal{P}$, $P(\theta \in U|\mathcal{D}_n) \to 1$,, $P_{\theta^\star}$-a.s. as $n \to \infty$.

sufficient conditions have been studied broadly in the statistics literature, and a wide spectrum of posterior consistency theorems have been derived. Informally speaking, posterior consistency theorems usually take the following form:

Theorem 3.1 (Consistency theorems (informal statement))**.** *Under suitable settings, we have*

$$\textit{model identifiability} + \textit{regularity conditions} \Rightarrow \textit{posterior consistency}.$$

A number of consistency theorems in this form have been derived under different scenarios, for example the Bernstein-Von Mises (BvM) theorems [164, 345] for parametric models, Doob's theorem [63] and Schwartz theorem [302] for nonparametric models and consistency results under misspecification (i.e., $P_{\theta^\star} \notin \mathcal{P}$) [25, 152, 151]. The formal statements of those theorems are quite technical and out of the scope of this thesis. However, we note that model identifiability assumptions often appear as a core part of the sufficient conditions. In Doob's theorem, identifiability is directly assumed; in Schwartz theorem and BvM theorems, the identifiability condition is replaced by stronger conditions called testability [302].

When the identifiability assumption is violated, the corresponding posterior consistency no longer holds. In other words, the posterior distribution will be substantially affected by the prior distribution, which makes Bayesian inference in this case problematic. An interesting observation is, even when the model parameter is non-identifiable, the corresponding posterior consistency may still be achieved on a *reparameterized version* of the original parameters, called the *minimal sufficient parameters* [16]. The concept of minimal sufficient parameter is defined in a similar way as sufficient statistics:

Definition 3.4 (Minimal sufficient parameters [249])**.** *Given a model $p_\theta(\mathbf{x})$, a quantity ϕ is called a sufficient parameter, if i), $\phi = \Lambda(\theta)$ for some function Λ, and ii), $\mathbf{x} \perp \theta|\phi$. Furthermore, ϕ is said to be a minimal sufficient parameter, if it is a sufficient parameter and can be expressed as a function of any sufficient parameter.*

Remark (Non-uniqueness and existence). Note that the minimal sufficient parameter of a model is not unique: they can be only determined up to a one-to-one transformation. Moreover, minimal sufficient parameters always exist, since the likelihood function itself constitutes a minimal sufficient parameter of the model.

One can show that both the likelihood function $P_\theta(\mathbf{x})$ and the posterior $P(\theta|\mathbf{x})$ depend on θ only through its minimal sufficient parameters, and these minimal sufficient parameters are always identifiable [16, 13, 249]. Therefore, a more intuitive way to think about minimal sufficient parameters is as an “identifiable version” of the original θ, which ensures posterior convergence:

Theorem 3.2 (Posterior consistency on minimal sufficient parameters)**.** *Let ϕ be a minimal sufficient parameter of P_θ. Then under the regularity conditions of [25], the posterior distribution $P(\phi|\mathcal{D}_n)$ of ϕ will weakly converge to the true value $\phi^\star$, almost surely in P.*

The effectiveness of minimal sufficient parameters raises a new direction for approximate Bayesian inference: when the regression model is non-identifiable in the original parameter space Θ, can we find its minimal sufficient parameter $\phi = \Lambda(\theta)$, and perform inference on the space of $\Phi = \{\Lambda\theta|\theta \in \Theta\}$ instead (Figure 3.1)? In fact, for regression problems, this is equivalent to performing inference in *function space*, which will be discussed in the next section.

Remark (Posterior correlations on θ). Apart from posterior consistency, Theorem 3.2 also provides another perspective on the challenges of performing *approximate* inference on $\theta \in \Theta$. Since the posterior on $\phi = \lambda(\theta)$ asymptotically approaches the point mass, it implies that the constraint of $\lambda(\theta) \approx \phi^\star$ will approximately hold, which might potentially introduce stronger pairwise correlations between different components of θ as n increases. This might cause mean-field VI to under-estimate the posterior variance on θ [344].

Remark (Bayesian or frequentist?). The setting of Definition 3.3 and Theorems 3.1 and 3.2 is different from the usual Bayesian inference problem that we have seen. On one hand, we have assumed the existence of a ground truth parameter value $\theta^\star$, and treat each data sample $\mathbf{x}_n$ as i.i.d. random variables, which corresponds to a frequentist setting. On the other hand, we perform parameter estimation using a Bayesian approach, i.e., via the posterior $P(\theta|\mathcal{D}_n)$. Therefore, the setting of posterior consistency in Definition 3.3 is in fact a mixture of both frequentist and Bayesian paradigms.

3.3 Function space inference: performing inference over minimal sufficient parameters

To finally motivate performing inference in function space, let us consider the Bayesian neural network example mentioned earlier. Given a dataset $\mathcal{D} := \{(\mathbf{x}_i, y_i)\}_{i=1:N}$, the BNN regression log-likelihood is defined as (Section 2.3.1)

$$\log p(\mathcal{D}_n|\mathbf{w}) = \sum_n \log \mathcal{N}(y_n; g(\mathbf{x}_n, \mathbf{w}), \sigma^2). \tag{3.2}$$

This is unfortunately non-identifiable except for few special cases[258] according to Definition 3.2, which may cause a number of issues. However, we may reparameterize the likelihood function, using the predictive mean function $f(\cdot) := g(\cdot, \mathbf{w})$:

$$\log p(\mathcal{D}_n|\mathbf{w}) = \sum_n \log p(\mathcal{D}_n|f). \tag{3.3}$$

Now, according to Definition 3.4, the predictive function $f(\cdot)$ is a *minimal sufficient parameter* of the BNN (this can be seen by noticing that the likelihood function $p(\mathcal{D}_n|f)$ is itself a minimum sufficient parameter, and it corresponds to $f(\cdot)$ one-to-one). Therefore, we can imagine that the resulting *function space posterior*,

$$p(f|\mathcal{D}_n) = \frac{p(\mathcal{D}_n|f)p(f)}{p(\mathcal{D}_n)}, \tag{3.4}$$

will converge to the true function asymptotically under certain conditions. Indeed, in the context of Bayesian neural networks for example, such function-space posterior convergence has been proved [172] under certain priors. [3] *This motivates us to perform inference directly in function space,* which gives us a number of advantages. In particular:

- Function space inference directly performs Bayesian inference in the space of minimal sufficient parameters, which forms an identifiable reparameterization of the regression model. Therefore, if done properly, this theory will help us bypass the problems of parameter space inference mentioned in Section 3.1. In particular, unidentifiability, over-parameterization, symmetries, and (asymptotic) sensitivity to weigh-space prior beliefs.

- In the case where the model parameters do not have specific scientific implications (such as in BNNs), we may directly design flexible and/or meaningful variational families for $q(f)$, without needing to derive first variational distributions on weight-space. This also helps us bypass the challenge of expressiveness of MFVI, which was described in Section 3.1.

[3]Technically speaking, these BNN posterior convergence results are proved in the sense of Hellinger neighbourhood (over $\mathcal{P}$) on the predictive joint density functions $p(y_n|f(\mathbf{x}_n))p(\mathbf{x}_n)$ (with $p(\mathbf{x}_n)$ fixed as $p(\mathbf{x}_n) \propto 1$), which is also a minimum sufficient parameter (therefore equivalent to the predictive function f).

- As it will be shown in Chapter 4, when performing VI in function space, the corresponding ELBO will be tighter than the ELBO in weight space;
- It potentially allows us to deal with non-parametric structured priors on $p(f)$ that may not have any convenient weigh-space expressions in the first place.

Of course, the idea of introducing function space inference will also introduce new challenges, which will be discussed in further detail in Chapter 4 and 5. In one word, the research presented in Part A are all about:

- How to specify/learn an appropriate $p(f)$?
- How to practically perform approximate inference for $p(f|\mathcal{D}_n)$.

Part A addresses these issues in a progressive manner:

- In Chapter 4, we will first propose a method called *variational implicit process*, which approximates $p(f|\mathcal{D}_n)$ via Gaussian process (GP) approximations and learns $p(f)$ via a wake-sleep procedure;
- Then, in Chapter 5, we will propose a more sophisticated variational inference framework that approximates $p(f|\mathcal{D}_n)$ via more flexible non-Gaussian processes.

Chapter 4

Variational Implicit Processes

In Chapter 3, we have motivated the advantages of performing inference in function space. The concept of function space inference itself is not a newly invented method — it is an old problem that is deeply rooted in the world of Bayesian nonparametrics. As introduced in Section 2.3.3 of Chapter 2, one of the most important example for Bayesian nonparametric is Gaussian processes for regression. In short, recall that GP performs function space inference in two steps:

1. **Construction of** $p(f)$, i.e., a prior $p(f)$ over the space of functions. In GPs, this is done by assuming that any finite collection of function values $\mathbf{f}$ follows a multivariate Gaussian distribution.

2. **Inference.** Given a collection of training data $\mathcal{D}$, the exact posterior $p(f|\mathcal{D})$ is then given by closed-form expressions.

To a certain extent, the concept of function space inference considered in this paper can be seen as the *algorithmic abstraction* of the above procedure of GP inference. Instead of *constructing* a specific infinite-dimensional $p(f)$, in function space inference we are *given* a predictive model $f_{\boldsymbol{\theta}}(\mathbf{x})$ parameterized by some parameter, $\boldsymbol{\theta}$ (could be finite dimensional). We will treat them *as if they are infinite-dimensional objects*, and computes the posterior in function space, $p(f|\mathcal{D})$. There is no doubt that Gaussian processes themselves do not immediately lead to such algorithmic abstractions. This is due to the fact that Gaussian processes assume a specific form of function space prior (which is Gaussian), and its inference method (based on closed form expression) can not be directly applied to other priors. Therefore, to find such algorithmic abstractions, we may have two potential options:

I The first option is *model-driven*. That is, we extend the existing Bayesian nonparametric priors such as GPs, to some class of more general and more flexible priors. Ideally, this should be able to cover many interesting Bayesian models as special cases. Then, we need to develop a specific method for optimizing variational approximations under this new prior.

II The other option is *algorithm-driven*, which is much more difficult. Instead of considering a specific class of functional priors, we will take an existing approximate inference method of parameter space, for example variational inference, and extend it to its function space counterpart. This should give us a general purpose function space approximate inference method.

In this chapter, we will first explore the first route (and leave the second one to Chapter 5). Our key idea is to draw inspirations from recent advancements of implicit models. As introduced in Section 2.2.4 of Chapter 2), probabilistic models with *implicit distributions* as core components have recently attracted enormous interest in both deep learning and the approximate Bayesian inference communities. In contrast to *prescribed probabilistic models* [61] that assign *explicit* densities to possible outcomes of the model, implicit models *implicitly assign* probability measures by the specification of the *data generating process*. One of the most well-known implicit distributions is the generator of generative adversarial nets (GANs) [88, 11] that transforms isotropic noise into high dimensional data, using neural networks. In approximate inference context, implicit distributions have also been used as flexible approximate posterior distributions [271, 191, 340, 182].

This chapter explores the extension of implicit models to Bayesian modeling of *random functions*. Similar to the construction of Gaussian processes (GPs), we develop *implicit process* (IP), which assigns implicit distributions over any finite collections of random variables. Therefore, IPs can be much more flexible than GPs when complicated models like neural networks are used for implicit distributions. With an IP as the prior, we can directly perform (variational) posterior inference *over functions* in a non-parametric fashion. This is beneficial for better-calibrated uncertainty estimates like GPs [36]. It also avoids typical issues of inference in parameter space, that is, symmetric modes in the posterior distribution of Bayesian neural network *weights*. The function-space inference for IPs is achieved by our proposed *variational implicit process* (VIP) algorithm, which addresses the intractability issues of implicit distributions.

Concretely, the contributions of this Chapter are threefold:

- We formalize implicit stochastic process priors over *functions*, and prove its well-definedness in both finite and infinite-dimensional cases. By allowing the usage of IPs with rich structures as priors (e.g., data simulators and Bayesian LSTMs), our approach provides a unified and powerful Bayesian inference framework for these important but challenging deep models.

- We derive a novel and efficient variational inference framework that gives a closed-form approximation to the IP posterior. It does not rely on e.g. density ratio/gradient estimators in implicit variational inference literature which can be inaccurate in high dimensions. Our inference method is computationally cheap, and it allows scalable hyperparameter learning in IPs.
- We conduct extensive comparisons between IPs trained with the proposed inference method, and GPs/BNNs/Bayesian LSTMs trained with existing variational approaches. Our method consistently outperforms other methods and achieves state-of-the-art results on a large-scale Bayesian LSTM inference task.

This chapter is arranged as follows. In Section 4.1, we will first introduce the notion of stochastic processes (i.e., distributions over functions), and how to construct them via Kolmogorov extension theorem. Then, in Section 4.2, we will utilize such tools to generalize Gaussian processes to implicit processes. We will provide concrete examples, and derive theoretical results regarding its well-definedness. In Section 4.3, we will introduce an approximate inference method for implicit processes. In Section 4.3.3, we will discuss computational complexities and scalable methods for predictive inference. Then, we perform experimental evaluations of the proposed method in Section 4.4, and finally, in Section 4.5, we review the related works from different areas.

4.1 Stochastic processes

We first introduce the notion of stochastic processes, and why it relates to function space inference. A stochastic process is nothing more than a collection of random variables, the formal definition of which is given as the following:

Definition 4.1 (Stochastic processes)**.** *Suppose that we are given a probability space* $(\Omega, \mathcal{F}, P)$*, and an non-empty set,* $\mathcal{T}$*. Then, any collection of random variables* $\{v_x : \omega \in \Omega \mapsto v_x(\omega) \in \mathcal{Y} | x \in \mathcal{T}\}$ *is called a stochastic process, and* $\mathcal{T}$ *is called the index set.*

Remark (Random variables and induced measure). Here we take this chance to briefly review the measure theoretic definition of random variables. A random variable v on a probability space $(\Omega, \mathcal{F}, P)$ is a measurable function that maps Ω to another measurable space, $(\mathcal{Y}, \mathcal{B})$. therefore, it defines a induced probability measure on $\mathcal{Y}$, denote by P_v, given by:

$$P_v(B \in \mathcal{B}) := P(v^{-1}(B)). \tag{4.1}$$

Therefore, whenever we talk about a random variable on $(\Omega, \mathcal{F}, P)$, we can also interpret it as a (induced) measure on $(\mathcal{Y}, \mathcal{B})$.

In most applications in machine learning, we would consider the special case of $\mathcal{T} = \mathbb{R}^d$, $\mathcal{Y} = \mathbb{R}$, and $\mathcal{B}$ being a Borel σ-algebra. A closely related concept is the sample paths of a stochastic processes:

Definition 4.2 (Sample path). *The $\mathcal{Y}$-valued function defined on $\mathcal{T}$:*

$$x \in \mathcal{T} \mapsto v_x(\omega) \in \mathcal{Y} \tag{4.2}$$

is called a sample path of the stochastic process $\{v_x | x \in \mathcal{T}\}$ on $\mathcal{T}$.

Therefore, for each possible outcome $\omega \in \Omega$, it is mapped to a sample path $x \in \mathcal{T} \mapsto v_x\omega \in \mathcal{Y}$. Hence, we may also interpret a stochastic process as a $\mathcal{Y}^{\mathcal{T}}$-valued random variable, where $\mathcal{Y}^{\mathcal{T}} := \{f : \mathcal{T} \to \mathcal{Y}\}$ is the set of $\mathcal{Y}$-valued functions (here, we have used f in its scalar form to denote a scalar function, $f(\cdot) : \mathcal{T} \to \mathcal{Y}$). In other words, a stochastic process defines a (induced) distribution of random functions. For this reason, we denote a stochastic process on $(\Omega, \mathcal{F}, P)$ by $p(f)$, and will use it to specify priors over function space.

Remark (Three interpretations of stochastic processes). To summarize, we have three equivalent interpretations of stochastic processes:

1. Stochastic process as a collection of random variables, $\{v_x : \omega \in \Omega \mapsto v_x(\omega) \in \mathcal{Y} | x \in \mathcal{T}\}$;

2. Stochastic process as a $\mathcal{Y}^{\mathcal{T}}$-valued random variable, i.e., as a mapping from an outcome $\omega \in \Omega$ to a sample path, $x \in \mathcal{T} \mapsto v_x(\omega) \in \mathcal{Y}$;

3. and finally, we can also interpret a stochastic process as the induced probability measure (by the $\mathcal{Y}^{\mathcal{T}}$-valued random variable) on the measurable space $\mathcal{Y}^{\mathcal{T}}$, i.e., a distribution over random functions.

The second and third interpretations gives us two alternative definitions of stochastic processes:

Definition 4.3 (Stochastic processes, ii)**.** *Suppose we are given a measurable space* $(\Omega, \mathcal{F}, P)$*, and an non-empty set,* $\mathcal{T}$*. Then, a stochastic process is a measurable function from* Ω *to some measurable space,* $\mathcal{Y}^{\mathcal{T}}$ *with some suitable* σ*-algebra denoted by* $\mathcal{B}^{\mathcal{T}}$[a]*.*

Definition 4.4 (Stochastic processes, iii)**.** *Suppose we are given a measurable space* $(\mathcal{Y}, \mathcal{B})$ *(usually we assume* $\mathcal{Y} = \mathbb{R}$*,* $\mathcal{B}$ *is the corresponding Borel algebra), and an non-empty set,* $\mathcal{T}$*. Then, a stochastic process is a probability measure over* $\mathcal{Y}^{\mathcal{T}}$*, with some suitable* σ*-algebra, denoted by* $\mathcal{B}^{\mathcal{T}}$*.*

[a]Technically, this is in fact the cylinder σ-algebra of $\mathcal{Y}$, given in Theorem 4.1 [129].

Given the definition of stochastic processes, how do we specify a useful stochastic process? As conceptually simple as they are, we cannot directly specify the exact probability for each possible sample paths on $\mathcal{T}$. It would be much more convenient if we could work with a finite subset $\{x_1, x_2, ..., x_n\} \subset \mathcal{T}$. Fortunately, Kolmogorov extension theorem guarantees that if we can consistently define a set of self-consistent probability distributions $p_{x_1,x_2,...,x_n}$ on any finite subset $\{x_1, x_2, ..., x_n\} \subset \mathcal{T}$, then there exist a unique stochastic process that shares the same marginal distributions as $p_{x_1,x_2,...,x_n}$.

Theorem 4.1 (Kolmogorov extension theorem)**.** *Without loss of generality, suppose for any finite subset* $\{x_1, x_2, ..., x_n\} \subset \mathcal{T}$*, we have a corresponding finite dimensional probability measure* $p_{x_1,x_2,...,x_n}$ *(called marginal distributions) on the measurable space* $\mathcal{Y}^n := \mathbb{R}^n$*. If they satisfies the following consistency condition for all finite subset* $\{x_1, x_2, ..., x_n\} \subset \mathcal{T}$ *and integer n:*

1. *for any permutation* $\pi : \{1, ..., n\} \to \{1, ..., n\}$*, we have*

$$p_{x_1,x_2,...,x_n}(B_1 \times B_2 \times ... \times B_n) = p_{x_{\pi(1)},x_{\pi(2)},...,x_{\pi(n)}}(B_{\pi(1)} \times B_{\pi(2)} \cdots \times B_{\pi(n)}), \quad \forall B_1, ..., B_n \subset \mathcal{B}(\mathbb{R}); \tag{4.3}$$

2. *for all integer n and all* $\{x_1, ..., x_n, x_{n+1}\} \in \mathcal{T}$*, we have*

$$p_{x_1,x_2,...,x_n}(B_1 \times ... \times B_n) = p_{x_1,x_2,...,x_n,x_{n+1}}(B_1 \times ... \times B_n \times \mathbb{R}). \tag{4.4}$$

Then, there exists a unique stochastic process $p^\star(f)$ *on* $\mathbb{R}^{\mathcal{T}}$ *(under the so called cylinder algebra, denoted by* $\mathcal{B}^{\mathcal{T}}$*), such that*

$$p^\star(f_{x_1} \in B_1, f_{x_2} \in B_2, ..., f_{x_n} \in B_n) = p_{x_1,x_2,...,x_n}(B_1 \times B_2 \times ... \times B_n) \tag{4.5}$$

for all integer n, all finite subsets $x_1, x_2, ..., x_n \subset \mathcal{T}$, and all measurable sets $B_1, ..., B_n \subset \mathcal{B}(\mathbb{R})$.

Being able to define a stochastic processes through marginal distributions $p_{x_1,x_2,...,x_n}$ allows us to define flexible function space priors $p(f)$, by combining the ideas of implicit distributions discussed in Section 2.2.4.

4.2 Implicit Stochastic Processes

In this section, we generalize GPs to implicit stochastic processes. Readers are referred to Section 2.3.3 for a detailed introduction, but briefly speaking, a GP defines the distribution of a random function f by placing a multivariate Gaussian distribution $\mathcal{N}(\mathbf{f}; \mathbf{m}, \mathbf{K_{ff}})$ over any finite collection of function values $\mathbf{f} = (f(\mathbf{x}_1), ..., f(\mathbf{x}_N))^\top$ evaluated at any given finite collection of input locations $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N$. Here $(\mathbf{m})_n = \mathfrak{M}(\mathbf{x}_n)$ and $(\mathbf{K_{ff}})_{n,n'} = \mathfrak{K}(\mathbf{x}_n, \mathbf{x}_{n'})$, and following Kolmogorov consistency theorem [129], the mean and covariance functions $\mathfrak{M}(\cdot)$, $\mathfrak{K}(\cdot,\cdot)$ are shared across all such finite collections. An alternative parameterization of GPs defines the sampling process as $\mathbf{f} \sim \mathcal{N}(\mathbf{f}; \mathbf{m}, \mathbf{K_{ff}}) \Leftrightarrow \mathbf{z} \sim \mathcal{N}(\mathbf{z}; 0, \mathbf{I})$, $\mathbf{f} = \mathbf{Bz} + \mathbf{m}$, with $\mathbf{K_{ff}} = \mathbf{BB}^\top$ the Cholesky decomposition of the covariance matrix. Observing this, we propose a generalization of the generative process by replacing the linear transform of the latent variable $\mathbf{z}$ with a nonlinear one. This gives the following formal definition of implicit stochastic process. Note that it is not just to non-linearise the mapping $\mathbf{B}$ (in that case it would only be a GP with a different kernel), but also the change of dependencies to induce non-Gaussian distributions.

Definition 4.5 (noiseless implicit stochastic processes)**.** *An implicit stochastic process (IP) is a collection of random variables $f(\cdot)$, such that any finite collection $\mathbf{f} = (f(\mathbf{x}_1), ..., f(\mathbf{x}_N))^\top$ has joint distribution implicitly defined by the following generative process:*

$$\mathbf{z} \sim p(\mathbf{z}), \;\; f(\mathbf{x}_n) = g_{\boldsymbol{\theta}}(\mathbf{x}_n, \mathbf{z}), \;\; \forall\, \mathbf{x}_n \in \mathbf{X}. \tag{4.6}$$

A function distributed according to the above IP is denoted as $f(\cdot) \sim \mathcal{IP}(g_{\boldsymbol{\theta}}(\cdot,\cdot), p_{\mathbf{z}})$.

Note that $\mathbf{z} \sim p(\mathbf{z})$ could be infinite dimensional (such as samples from a Gaussian Process). Definition 4.5 is validated by the following propositions.

Proposition 4.1 (Finite dimension case)**.** *Let $\mathbf{z}$ be a finite dimensional vector. Then there exists a unique stochastic process on index set $\mathcal{T} = \mathbf{X}$, such that any finite collection of random variables has distribution implicitly defined by (4.6).*

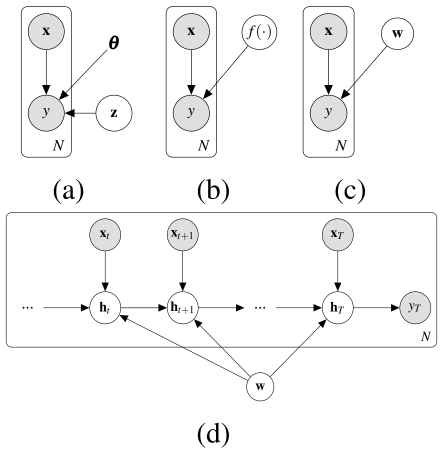


Figure 4.1 Examples of IPs: (a) Neural samplers; (b) Warped GPs (c) Bayesian neural networks; (d) Bayesian RNNs.

Proposition 4.2 (Infinite dimension case)**.** *Let $z(\cdot) \sim \mathcal{SP}(0, \mathfrak{C})$ be a centered continuous stochastic process on $\mathcal{L}^2(\mathbb{R}^d)$ with covariance function $\mathfrak{C}(\cdot,\cdot)$. Then the operator $g(\mathbf{x}, z) = O_{\mathfrak{K}}(z)(\mathbf{x}) := h(\int_{\mathbf{x}} \sum_{l=0}^{M} \mathfrak{K}_l(\mathbf{x}, \mathbf{x}')z(\mathbf{x}')d\mathbf{x}')$, $0 < M < +\infty$ defines a stochastic process if $\mathfrak{K}_l \in \mathcal{L}^2(\mathbb{R}^d \times \mathbb{R}^d)$, h is a Borel measurable, bijective function in $\mathbb{R}$ and there exist $0 \leq A < +\infty$ such that $|h(x)| \leq A|x|$ for $\forall x \in \mathbb{R}$.*

Proposition 4.1 is proved in appendix 4.A.1 using the Kolmogorov extension theorem. Proposition 4.2 considers *random functions* as the latent input $z(\cdot)$, and introduces a specific form of the transformation/operator g, so that the resulting collection of variables $f(\cdot)$ is still a valid stochastic process (see appendix 4.A.2 for a proof). Note this operator can be recursively applied to build highly non-linear operators over functions [97, 365, 327, 169, 84]. These two propositions indicate that IPs form a rich class of priors over functions. Indeed, we visualize some examples of IPs in Figure 4.1 with discussions as follows:

Remark (Examples of implicit processes).

Example 4.1 (Data simulators)**.** *Simulators, e.g. physics engines and climate models, are omnipresent in science and engineering. These models encode laws of physics in $g_{\boldsymbol{\theta}}(\cdot,\cdot)$, use $\mathbf{z} \sim p(\mathbf{z})$ to explain the remaining randomness, and evaluate the function at input locations $\mathbf{x}$: $f(\mathbf{x}) = g_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})$. We define the **neural sampler** as a specific instance of this class. In this case $g_{\boldsymbol{\theta}}(\cdot,\cdot)$ is a neural network with weights $\boldsymbol{\theta}$, i.e., $g_{\boldsymbol{\theta}}(\cdot,\cdot) = \mathrm{NN}_{\boldsymbol{\theta}}(\cdot,\cdot)$, and $p(\mathbf{z}) = Uniform([-a,a]^d)$.*

Example 4.2 (Warped Gaussian Processes)**.** *Warped Gaussian Processes [316] is also an interesting example of IPs. Let $z(\cdot) \sim p(z)$ be a sample from a GP prior, and $g_{\boldsymbol{\theta}}(\mathbf{x}, z)$ is defined as $g_{\boldsymbol{\theta}}(\mathbf{x}, z) = h(z(\mathbf{x}))$, where $h(\cdot)$ is a one dimensional monotonic function.*

Example 4.3 (Bayesian neural network)**.** *In a Bayesian neural network, the synaptic weights W with prior $p(W)$ play the role of $\mathbf{z}$ in (4.6). A function is sampled by $W \sim p(W)$ and then setting $f(\mathbf{x}) = g_{\boldsymbol{\theta}}(\mathbf{x}, W) = \mathrm{NN}_{\mathrm{W}}(\mathbf{x})$ for all $\mathbf{x} \in \mathbf{X}$. In this case $\boldsymbol{\theta}$ could be the hyper-parameters of the prior $p(W)$ to be tuned.*

Example 4.4 (Bayesian RNN)**.** *Similar to Example 4.3, a Bayesian recurrent neural network (RNN) can be defined by considering its weights as random variables, and taking as function evaluation an output value generated by the RNN after processing the last symbol of an input sequence.*

4.3 Variational Implicit Processes

Consider the following regression model with an IP prior over the regression function:

$$f(\cdot) \sim \mathcal{IP}(g_{\boldsymbol{\theta}}(\cdot,\cdot), p_{\mathbf{z}}),\ y = f(\mathbf{x}) + \varepsilon,\ \varepsilon \sim \mathcal{N}(0, \sigma^2). \tag{4.7}$$

Equation (4.7) defines an implicit model $p(\mathbf{y}, \mathbf{f}|\mathbf{x})$, which is intractable in most cases. Note that it is common to add Gaussian noise ε to an implicit model, e.g. see the noise smoothing trick used in GANs [318, 290]. Given an observed dataset $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$ and a set of test inputs $\mathbf{X}_*$, Bayesian predictive inference computes the predictive distribution $p(\mathbf{y}_*|\mathbf{X}_*, \mathbf{X}, \mathbf{y}, \boldsymbol{\theta})$, which itself requires interpolating over posterior $p(f|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta})$. Besides prediction, we also want to learn the model parameters $\boldsymbol{\theta}$ and σ by maximizing the marginal likelihood: $\log p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}) = \log \int_{\mathbf{f}} p(\mathbf{y}|\mathbf{f}) p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta}) d\mathbf{f}$, with $\mathbf{f} = f(\mathbf{X})$ being the evaluation of f on the points in $\mathbf{X}$. Unfortunately, both the prior $p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta})$ and the posterior $p(f|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta})$ are intractable as the implicit process does not allow point-wise density evaluation, let alone the marginalization tasks. Therefore, to address these, we must resort to approximate inference.

We propose a generalization of the *wake-sleep* algorithm [114] to handle both intractabilities. This method returns (i) an approximate posterior distribution $q(f|\mathbf{X}, \mathbf{y})$ which is later used for predictive inference, and (ii) an approximation to the marginal likelihood $p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})$ for hyper-parameter optimization. We use the posterior of a GP to approximate the posterior of the IP, i.e. $q(f|\mathbf{X}, \mathbf{y}) = q_{\mathcal{GP}}(f|\mathbf{X}, \mathbf{y})$, since GP is one of the few existing tractable distributions over functions. A high-level summary of our algorithm is the following:

- ***Sleep phase***: sample function values $\mathbf{f}$ and noisy outputs $\mathbf{y}$ as indicated in (4.7). This *dreamed* data is then used as the *maximum-likelihood (ML)* target to fit a GP. This is equivalent to minimizing $\mathrm{D}_{\mathrm{KL}}[p(\mathbf{y},\mathbf{f}|\mathbf{X},\boldsymbol{\theta})||q_{\mathcal{GP}}(\mathbf{y},\mathbf{f}|\mathbf{X})]$ for any possible $\mathbf{X}$.
- ***Wake phase***: The optimal GP posterior approximation $q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X},\mathbf{y})$ obtained in the sleep phase is used to construct a variational approximation to $\log p(\mathbf{y}|\mathbf{X},\boldsymbol{\theta})$, which is then optimized with respect to $\boldsymbol{\theta}$.

Our approach has two key advantages. First, the algorithm has no explicit sleep phase computation, since the sleep phase optimization has an analytic solution that can be directly plugged into the wake-phase objective. Second, the proposed wake phase update is highly scalable, as it is equivalent to a Bayesian linear regression task with random features sampled from the implicit process. With our wake-sleep algorithm, the evaluation of the implicit prior density is no longer an obstacle for approximate inference. We call this inference framework the *variational implicit process* (VIP). In the following sections we give specific details on both the wake and sleep phases.

4.3.1 Sleep phase: GP posterior as variational distribution

This section proposes an approximation to the IP posterior $p(\mathbf{f}|\mathbf{X},\mathbf{y},\boldsymbol{\theta})$. The naive variational inference [140] would require computing the joint distribution $p(\mathbf{y},\mathbf{f}|\mathbf{X},\boldsymbol{\theta})$ which is intractable. However, sampling from this joint distribution is straightforward. We leverage this idea in the *sleep phase* of our wake-sleep algorithm to approximate the joint distribution $p(\mathbf{y},\mathbf{f}|\mathbf{X},\boldsymbol{\theta})$ instead.

Precisely, for any finite collection of variables $\mathbf{f}$ with their input locations $\mathbf{X}$, we approximate $p(\mathbf{y},\mathbf{f}|\mathbf{X},\boldsymbol{\theta})$ with a simpler distribution $q(\mathbf{y},\mathbf{f}|\mathbf{X}) = q(\mathbf{y}|\mathbf{f})q(\mathbf{f}|\mathbf{X})$ instead. We choose $q(\mathbf{f}|\mathbf{X})$ to be a GP with mean and covariance functions $\mathfrak{M}(\cdot)$ and $\mathfrak{K}(\cdot,\cdot)$, respectively, and write the prior as $q(\mathbf{f}|\mathbf{X}) = q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X},\mathfrak{M},\mathfrak{K})$. The sleep-phase update minimizes the following KL divergence:

$$\begin{aligned} q^{\star}_{\mathcal{GP}} &= \arg\min_{\mathfrak{M},\mathfrak{K}} \mathcal{U}(\mathfrak{M},\mathfrak{K}), \qquad (4.8)\\ \text{with} \quad \mathcal{U}(\mathfrak{M},\mathfrak{K}) &= \mathrm{D}_{\mathrm{KL}}[p(\mathbf{y},\mathbf{f}|\mathbf{X},\boldsymbol{\theta})||q_{\mathcal{GP}}(\mathbf{y},\mathbf{f}|\mathbf{X},\mathfrak{M},\mathfrak{K})]. \end{aligned}$$

We further assume $q(\mathbf{y}|\mathbf{f}) = p(\mathbf{y}|\mathbf{f})$, which reduces $\mathcal{U}(\mathfrak{M},\mathfrak{K})$ to $\mathrm{D}_{KL}[p(\mathbf{f}|\mathbf{X},\boldsymbol{\theta})||q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X},\mathfrak{M},\mathfrak{K})]$. In this case the optimal $\mathfrak{M}(\cdot)$ and $\mathfrak{K}(\cdot,\cdot)$ are equal to the mean and covariance functions of the IP, respectively:

$$\begin{aligned} \mathfrak{M}^\star(\mathbf{x}) &= \mathbb{E}[f(\mathbf{x})], \\ \mathfrak{K}^\star(\mathbf{x}_1,\mathbf{x}_2) &= \mathbb{E}[(f(\mathbf{x}_1)-\mathfrak{M}^\star(\mathbf{x}_1))(f(\mathbf{x}_2)-\mathfrak{M}^\star(\mathbf{x}_2))]. \end{aligned} \tag{4.9}$$

Below we also write the optimal solution as $q^\star_{\mathcal{GP}}(\mathbf{f}|\mathbf{X},\boldsymbol{\theta}) = q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X},\mathfrak{M}^\star,\mathfrak{K}^\star)$ to explicitly specify the *dependency on prior parameters* $\boldsymbol{\theta}$ [1]. In practice, the mean and covariance functions are estimated by by Monte Carlo, which leads to *maximum likelihood* training (MLE) for the GP with *dreamed* data from the IP. Assume S functions are drawn from the IP: $f^{\boldsymbol{\theta}}_s(\cdot) \sim \mathcal{IP}(g_{\boldsymbol{\theta}}(\cdot,\cdot),p_{\mathbf{z}}), s=1,\dots,S$. The optimum of $\mathcal{U}(\mathfrak{M},\mathfrak{K})$ is then estimated by the *MLE solution*:

$$\mathfrak{M}^\star_{\text{MLE}}(\mathbf{x}) = \frac{1}{S}\sum_s f^{\boldsymbol{\theta}}_s(\mathbf{x}), \tag{4.10}$$

$$\mathfrak{K}^\star_{\text{MLE}}(\mathbf{x}_1,\mathbf{x}_2) = \frac{1}{S}\sum_s \Delta_s(\mathbf{x}_1)\Delta_s(\mathbf{x}_2), \tag{4.11}$$

$$\Delta_s(\mathbf{x}) = f^{\boldsymbol{\theta}}_s(\mathbf{x}) - \mathfrak{M}^\star_{\text{MLE}}(\mathbf{x}).$$

To reduce computational costs, the number of dreamed samples S is often small. Therefore, we perform *maximum a posteriori* instead of MLE, by putting an inverse Wishart process prior [306] $\mathcal{IWP}(\nu,\Psi)$ over the GP covariance function $\mathfrak{K}$ (Appendix 4.A.3).

The original sleep phase algorithm in [114] also finds a posterior approximation by minimizing (4.9). However, the original approach would define the q distribution as $q(\mathbf{y},\mathbf{f}|\mathbf{X}) = p(\mathbf{y}|\mathbf{X},\boldsymbol{\theta})q_{\mathcal{GP}}(\mathbf{f}|\mathbf{y},\mathbf{X})$, which builds a *recognition model* that can be directly transfered for later inference. By contrast, we define $q(\mathbf{y},\mathbf{f}|\mathbf{X}) = p(\mathbf{y}|\mathbf{f})q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X})$, which corresponds to an approximation of the IP prior. In other words, we approximate an intractable generative model using another generative model with a GP prior and later, the resulting GP posterior $q^\star_{\mathcal{GP}}(\mathbf{f}|\mathbf{X},\mathbf{y})$ is employed as the variational distribution. Importantly, we never explicitly perform the sleep phase updates, that is, the optimization of $\mathcal{U}(\mathfrak{M},\mathfrak{K})$, as there is an analytic solution readily available, which can potentially save a significant amount of computation.

[1]This allows us to compute gradients w.r.t. $\boldsymbol{\theta}$ through $\mathfrak{M}^\star$ and $\mathfrak{K}^\star$ using reparameterization trick (by definition of IP, $f(\mathbf{x}) = g_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z})$), during the wake phase in Section 4.3.2.

Another interesting observation is that the sleep phase's objective $\mathcal{U}(\mathfrak{M},\mathfrak{K})$ also provides an upper-bound to the KL divergence between the posterior distributions,

$$\mathcal{J} = \mathrm{D}_{\mathrm{KL}}[p(\mathbf{f}|\mathbf{X},\mathbf{y},\boldsymbol{\theta})||q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X},\mathbf{y})].$$

One can show that $\mathcal{U}$ is an upper-bound of $\mathcal{J}$ according to the non-negativity and chain rule of the KL divergence:

$$\mathcal{U}(\mathfrak{M},\mathfrak{K}) = \mathcal{J} + \mathrm{D}_{\mathrm{KL}}[p(\mathbf{y}|\mathbf{X},\boldsymbol{\theta})||q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X})] \geq \mathcal{J}. \tag{4.12}$$

Therefore, $\mathcal{J}$ is also decreased when the mean and covariance functions are optimized during the sleep phase. This bounding property justifies $\mathcal{U}(\mathfrak{M},\mathfrak{K})$ as a appropriate variational objective for posterior approximation.

4.3.2 Wake phase: a scalable approach to learning the model parameters θ

In the wake phase of the original wake-sleep algorithm, the IP model parameters $\boldsymbol{\theta}$ are optimized by maximizing a variational lower-bound on the log marginal likelihood $\log p(\mathbf{y}|\mathbf{X},\boldsymbol{\theta})$. Unfortunately, this requires evaluating the IP prior $p(\mathbf{f}|\mathbf{X},\boldsymbol{\theta})$ which is intractable. But recall from (4.12) that during the sleep phase $\mathrm{D}_{\mathrm{KL}}[p(\mathbf{y}|\mathbf{X},\boldsymbol{\theta})||q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X})]$ is also minimized. Therefore we directly approximate the log marginal likelihood using the *optimal* GP from the sleep phase, i.e.

$$\log p(\mathbf{y}|\mathbf{X},\boldsymbol{\theta}) \approx \log q^{\star}_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\boldsymbol{\theta}). \tag{4.13}$$

This again demonstrates the key advantage of the proposed sleep phase update via generative model matching. Also it is a sensible objective for predictive inference as the GP returned by wake-sleep will be used for making predictions.

Similar to GP regression, optimizing $\log q^{\star}_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\boldsymbol{\theta})$ can be computationally expensive for large datasets. Therefore sparse GP approximation techniques [315, 335, 109, 38] are applicable, but we leave them to future work and consider an alternative approach that is related to random feature approximations of GPs [263, 78, 75, 15, 166].

Note that $\log q^{\star}_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\boldsymbol{\theta})$ can be approximated by the log marginal likelihood of a Bayesian linear regression model with S randomly sampled dreamed functions, and a

Algorithm 1 Variational Implicit Processes (VIP)

```
Require: data $\mathcal{D} = (\mathbf{X}, \mathbf{y})$; IP $\mathcal{IP}(g_\theta(\cdot,\cdot), p_\mathbf{z})$; variational distribution $q_\lambda(\mathbf{a})$; hyper-parameter $\alpha$
1: while not converged do
2:     sample a mini-batch $\mathcal{K} \subset \{1,...,N\}$ of size $K$
3:     sample $S$ function values:
       $\mathbf{z}_s \sim p(\mathbf{z}), f_s^\theta(\mathbf{x}_m) = g_\theta(\mathbf{x}_m, \mathbf{z}_s)$
4:     solutions of sleep phase:
       $\mathfrak{m}^\star(\mathbf{x}_m) = \frac{1}{S}\sum_{s=1}^S f_s^\theta(\mathbf{x}_m)$,
       $\Delta_s(\mathbf{x}_m) = f_s^\theta(\mathbf{x}_m) - \mathfrak{m}^\star(\mathbf{x}_m)$
5:     compute the wake phase energy $\mathcal{L}_{\mathcal{GP}}^\alpha(\theta, \lambda)$ in (4.16) using (4.15)
6:     gradient descent on $\mathcal{L}_{\mathcal{GP}}^\alpha(\theta, \lambda)$ w.r.t $\theta, \lambda$, via reparameterization tricks
7: end while
```

coefficient vector $\mathbf{a} = (a_1, ..., a_S)$:

$$\log q_{\mathcal{GP}}^\star(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}) \approx \log \int \prod_n q^\star(y_n|\mathbf{x}_n, \mathbf{a}, \boldsymbol{\theta}) p(\mathbf{a}) d\mathbf{a}, \tag{4.14}$$

$$\begin{aligned} q^\star(y_n|\mathbf{x}_n, \mathbf{a}, \boldsymbol{\theta}) &= \mathcal{N}\left(y_n; \mu(\mathbf{x}_n, \mathbf{a}, \boldsymbol{\theta}), \sigma^2\right), \\ \mu(\mathbf{x}_n, \mathbf{a}, \boldsymbol{\theta}) &= \mathfrak{M}^\star(\mathbf{x}_n) + \frac{1}{\sqrt{S}} \sum_s \Delta_s(\mathbf{x}_n) a_s, \\ \Delta_s(\mathbf{x}_n) &= f_s^{\boldsymbol{\theta}}(\mathbf{x}_n) - \mathfrak{M}^\star(\mathbf{x}_n), \; p(\mathbf{a}) = \mathcal{N}(\mathbf{a}; 0, \mathbf{I}). \end{aligned} \tag{4.15}$$

For scalable inference, we follow Li and Gal [177] to approximate (4.14) by the α-energy (see Section 2.2.3), with $q_{\boldsymbol{\lambda}}(\mathbf{a}) = \mathcal{N}(\mathbf{a}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ and sample a mini-batch $\mathcal{K} \subset \{1,...,N\}$ of size K:

$$\begin{aligned} &\log q_{\mathcal{GP}}^\star(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}) \approx \mathcal{L}_{\mathcal{GP}}^\alpha(\boldsymbol{\theta}, \boldsymbol{\lambda}) \\ &= \frac{N}{\alpha K} \sum_{k \in \mathcal{K}} \log \mathbb{E}_{q_{\boldsymbol{\lambda}}(\mathbf{a})} \left[q^\star(y_k|\mathbf{x}_k, \mathbf{a}, \boldsymbol{\theta})^\alpha\right] \\ &- \mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\lambda}}(\mathbf{a}) || p(\mathbf{a})]. \end{aligned} \tag{4.16}$$

See Algorithm 1 for the full algorithm. When $\alpha \to 0$ the α-energy reduces to the variational lower-bound, and empirically the α-energy returns better approximations when $\alpha > 0$. For Bayesian linear regression (4.15) the exact posterior of $\mathbf{a}$ is a multivariate Gaussian, which justifies our choice of $q_{\boldsymbol{\lambda}}(\mathbf{a})$. Stochastic optimization is applied to optimize $\boldsymbol{\theta}$ and $\boldsymbol{\lambda}$ jointly, making our method highly scalable.

4.3.3 Computational complexity and scalable predictive inference

Assume the evaluation of a sampled function value $f(\mathbf{x}) = g_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})$ for a given input $\mathbf{x}$ takes $\mathcal{O}(C)$ time. The VIP has time complexity $\mathcal{O}(CKS + KS^2 + S^3)$ in training, where K is the size of a mini-batch, and S is the number of random functions sampled from $\mathcal{IP}(g_{\boldsymbol{\theta}}(\cdot,\cdot), p_{\mathbf{z}})$. Note that approximate inference techniques in $\mathbf{z}$ space, e.g. mean-field Gaussian approximations to the posterior of Bayesian neural network weights [29, 113, 177], also take $\mathcal{O}(CKS)$ time. Therefore when C is large (typically the case for neural networks) the additional cost is often negligible, as S is usually significantly smaller than the typical number of inducing points in sparse GP ($S = 20$ in the experiments).

Predictive inference follows the standard GP equations to compute $q^{\star}_{\mathcal{GP}}(\mathbf{f}_*|\mathbf{X}_*, \mathbf{X}, \mathbf{y}, \boldsymbol{\theta}^{\star})$ on the test set $\mathbf{X}_*$ with L datapoints: $\mathbf{f}_*|\mathbf{X}_*, \mathbf{X}, \mathbf{y} \sim \mathcal{N}(\mathbf{f}_*; \mathbf{m}_*, \boldsymbol{\Sigma}_*)$,

$$\begin{aligned} \mathbf{m}_* &= \mathfrak{M}^{\star}(\mathbf{X}_*) + \mathbf{K}_{*\mathbf{f}}(\mathbf{K}_{\mathbf{ff}} + \sigma^2\mathbf{I})^{-1}(\mathbf{y} - \mathfrak{M}^{\star}(\mathbf{X})), \\ \boldsymbol{\Sigma}_* &= \mathbf{K}_{**} - \mathbf{K}_{*\mathbf{f}}(\mathbf{K}_{\mathbf{ff}} + \sigma^2\mathbf{I})^{-1}\mathbf{K}_{\mathbf{f}*}. \end{aligned} \tag{4.17}$$

Recall that the optimal variational GP approximation has mean and covariance functions defined as (4.10) and (4.11), respectively, which means that $\mathbf{K}_{\mathbf{ff}}$ has rank S. Therefore predictive inference requires both function evaluations and matrix inversion, which costs $\mathcal{O}(C(L+N)S + NS^2 + S^3)$ time. This complexity can be further reduced: note that the computational cost is dominated by $(\mathbf{K}_{\mathbf{ff}} + \sigma^2\mathbf{I})^{-1}$. Denote the Cholesky decomposition of the kernel matrix $\mathbf{K}_{\mathbf{ff}} = \mathbf{B}\mathbf{B}^{\top}$. It is straightforward to show that in the Bayesian linear regression problem (4.15) the exact posterior of $\mathbf{a}$ is $q(\mathbf{a}|\mathbf{X}, \mathbf{y}) = \mathcal{N}(\mathbf{a}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$, with $\boldsymbol{\mu} = \frac{1}{\sigma^2}\boldsymbol{\Sigma}\mathbf{B}^{\top}(\mathbf{y} - \mathbf{m}), \sigma^2\boldsymbol{\Sigma}^{-1} = \mathbf{B}^{\top}\mathbf{B} + \sigma^2\mathbf{I}$. Therefore the parameters of the GP predictive distribution in (4.17) are reduced to:

$$\mathbf{m}_* = \mathfrak{M}^{\star}(\mathbf{X}_*) + \boldsymbol{\phi}_*^{\top}\boldsymbol{\mu}, \ \boldsymbol{\Sigma}_* = \boldsymbol{\phi}_*^{\top}\boldsymbol{\Sigma}\boldsymbol{\phi}_*, \tag{4.18}$$

with the elements in $\boldsymbol{\phi}_*$ as $(\boldsymbol{\phi}_*)_s = \Delta_s(\mathbf{x}_*)/\sqrt{S}$. This reduces the prediction cost to $\mathcal{O}(CLS + S^3)$, which is on par with e.g. conventional predictive inference techniques for Bayesian neural networks that also cost $\mathcal{O}(CLS)$. In practice we use the mean and covariance matrix from $q(\mathbf{a})$ to compute the predictive distribution. Alternatively one can directly sample $\mathbf{a} \sim q(\mathbf{a})$ and compute $\mathbf{f}_* = \sum_{s=1}^{S} a_s f_s^{\boldsymbol{\theta}}(\mathbf{X}_*)$, which is also an $\mathcal{O}(CKS + S^3)$ inference approach but would have higher variance.

4.4 Experiments

In this section, we test the capability of VIPs with various tasks, including time series interpolation, Bayesian NN/LSTM inference, and Approximate Bayesian Computation (ABC) with simulators,etc. When the VIP is applied to Bayesian NN/LSTM (Example 4.3-4.4), the prior parameters over each weight are tuned individually. We use $S = 20$ for VIP unless noted otherwise. We focus on comparing VIPs as an *inference method* to other Bayesian approaches, with detailed experimental settings presented in Appendix 4.D.

4.4.1 Synthetic example

We first assess the behaviours of VIPs, including its quality of uncertainty estimation and the ability to discover structures under uncertainty. The synthetic training set is generated by first sampling 300 inputs x from $\mathcal{N}(0,1)$. Then, for each x obtained, the corresponding target y is simulated as $y = \frac{\cos 5x}{|x|+1} + \varepsilon,\ \varepsilon \sim \mathcal{N}(0, 0.1)$. The test set consists of 10^3 evenly spaced points on $[-3,3]$. We use an IP with a Bayesian neural network (1-10-10-1 architecture) as the prior. We use $\alpha = 0$ for the wake-step training. We also compare VIP with the exact full GP with *optimized* compositional kernel (RBF+Periodic), and another BNN with identical architecture but trained using variational dropout (VDO) with dropout rate $p = 0.99$ and length scale $l = 0.001$. The (hyper-)parameters are optimized using 500 epochs (batch training) with Adam optimizer (learning rate = 0.01).

Figure 4.2 visualizes the results. Compared with VDO and the full GP, the VIP predictive mean recovers the ground truth function better. Moreover, VIP provides the best predictive uncertainty, especially when compared with VDO: it increases smoothly when $|x| \to 3$, where training data is sparse around there. Although the composition of periodic kernel helps the full GP to return a better predictive mean than VDO (but worse than VIP), it still over-fits to the data and returns a poor uncertainty estimate around $|x| \approx 2.5$.

Test Negative Log-likelihood (NLL) and RMSE results reveal similar conclusions (see the left two plots in Figure 4.3), where VIP significantly outperforms VDO and GP.

4.4.2 Solar irradiance interpolation under missingness

Time series interpolation is an ideal task to evaluate the quality of uncertainty estimate. We compare the VIP ($\alpha = 0$) with a variationally sparse GP (SVGP, 100 inducing points), an exact GP and VDO on the solar irradiance dataset [170]. The dataset is constructed following [78], where 5 segments of length 20 are removed for interpolation. All the inputs

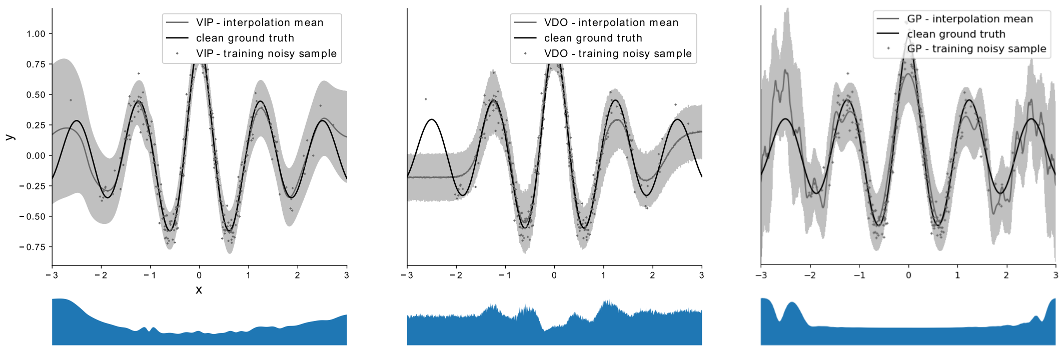


Figure 4.2 First row: Predictions returned from VIP (**left**), VDO (**middle**) and exact GP with RBF + Periodic kernel (**right**), respectively. **Dark grey dots**: noisy observations; **dark line**: clean ground truth function; **dark gray line**: predictive means; **Gray shaded area**: confidence intervals with 2 standard deviations. **Second row**: Corresponding predictive uncertainties.

are then centered, and the targets are standardized. We use the same settings as in Section 4.4.1, except that we run Adam with learning rate = 0.001 for 5000 iterations. Note that GP/SVGP predictions are reproduced directly from [78].

Predictive interpolations are shown in Figure 4.5. We see that VIP and VDO give similar interpolation behaviors. However, VDO overall under-estimates uncertainty when compared with VIP, especially in the interval $[-100, 200]$. VDO also incorrectly estimates the mean function around $x = -150$ where the ground truth there is a constant. On the contrary, VIP is able to recover the correct mean estimation around this interval with high confidence. GP methods recover the exact mean of the training data with high confidence, but they return poor estimates of predictive means for interpolation. Quantitatively, the right two plots in Figure 4.3 show that VIP achieves the best NLL/RMSE performance, again indicating that its returns high-quality uncertainties and accurate mean predictions.

4.4.3 Predictive Performance: Multivariate regression

We apply the VIP inference to a Bayesian neural network (VIP-BNN, example 4.3) and a neural sampler (VIP-NS, example 4.1) , using real-world multivariate regression datasets from the UCI data repository [185]. We mainly compare with the following BNNs baselines: variational Gaussian inference with reparameterization tricks [VI, 29], variational dropout [VDO, 75], and variational alpha dropout [177]. We also include the variational GP (SVGP,

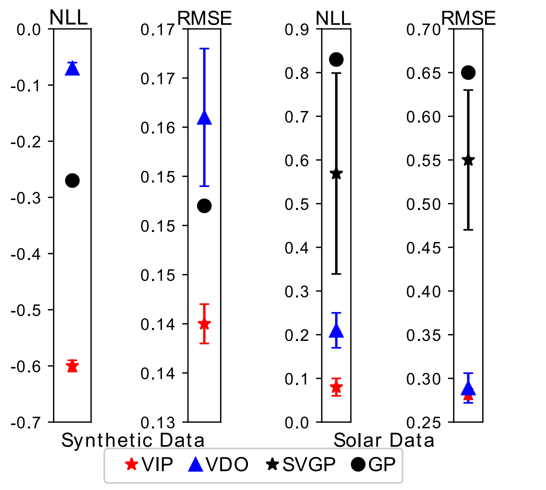


Figure 4.3 Test performance on synthetic example (**left two**) and solar irradiance interpolation (**right two**)

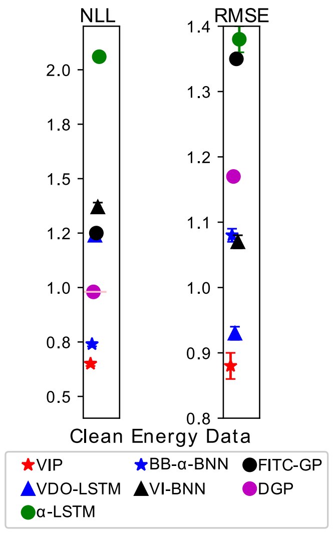


Figure 4.4 Test performance on clean energy dataset

[335]), exact GP and the functional BNNs (fBNN)[2], and the results for fBNN is quoted from Sun et al. [330]. All neural networks have two hidden layers of size 10, and are trained for 1,000 (except for fBNNs where the results cited use 2,000 epochs). The observational noise variance for VIP and VDO is tuned over a validation set, as detailed in Appendix 4.D. The α value for both VIP and alpha-variational inference are fixed to 0.5, as suggested in [113]. The experiments are repeated for 10 times on all datasets except *Protein*, on which we report an averaged results across 5 repetitive runs.

Results are shown in Table 4.1 and 4.2 with the best performances boldfaced. Note that our method is not directly comparable to exact (full) GP and fBNN in the last two columns. They are only trained on small datasets since they require the computation of the *exact* GP likelihood, and fBNNs are trained for longer epochs. Therefore they are not included for the overall ranking shown in the last row of the tables. VIP methods consistently outperform other methods, obtaining the best test-NLL in 7 datasets, and the best test RMSE in 8 out of the 9 datasets. In addition, VIP-BNN obtains the best ranking among 6 methods. Note also that VIP marginally outperforms exact GPs and fBNNs (4 of 5 in NLLs), despite the comparison is not even fair. Finally, it is encouraging to see that, despite its general form, the VIP-NS achieves the second best average ranking in RMSE, outperforming many specifically designed BNN algorithms.

[2]fBNN is a recent inference method designed for BNNs, where functional priors (GPs) are used to regularize BNN training. See related work for further discussions.

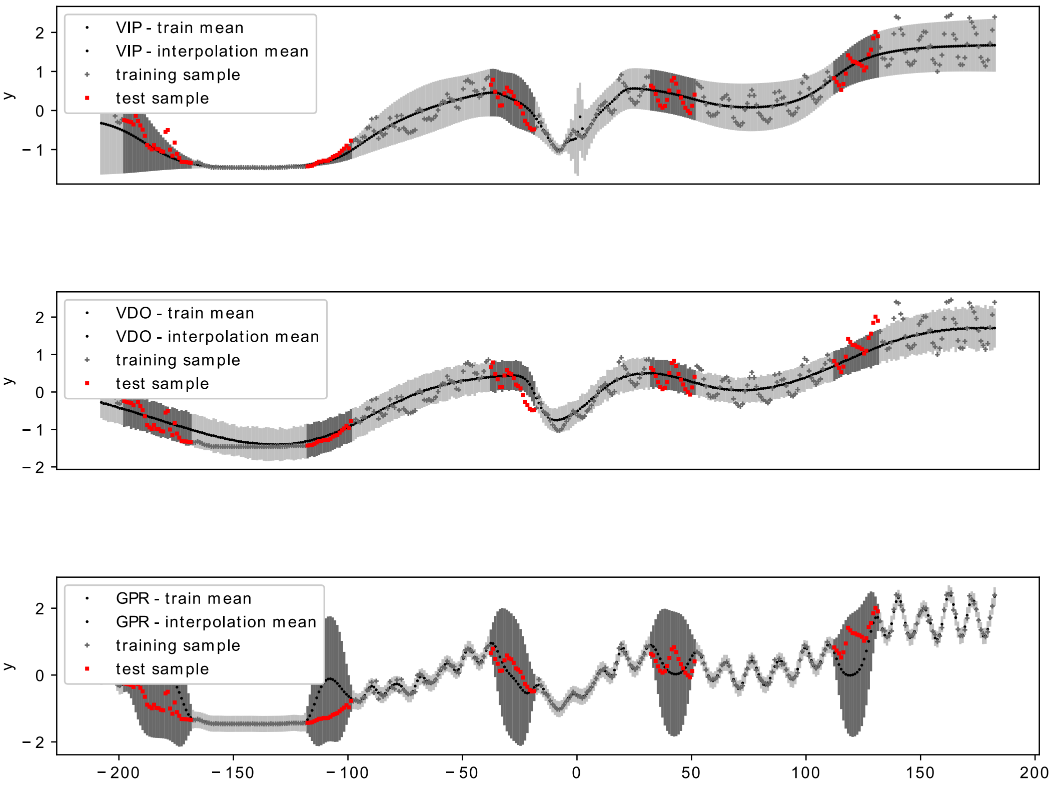


Figure 4.5 Interpolations returned by VIP (**top**), variational dropout (**middle**), and exact GP (**bottom**), respectively. SVGP visualization is omitted as it looks nearly the same. Here **grey dots**: training data, **red dots**: test data, **dark dots**: predictive means, **light grey and dark grey areas**: Confidence intervals with 2 standard deviations of the training and test set, respectively. Note that our GP/SVGP predictions reproduces [78].

4.4.4 Bayesian LSTM for predicting power conversion efficiency of organic photovoltaics molecules

To demonstrate the scalability and flexibility of VIP, we perform experiments with the Harvard Clean Energy Project Data, the world's largest materials high-throughput virtual screening effort [98]. A large number of molecules of organic photovoltaics are scanned to find those with high power conversion efficiency (PCE) using quantum-chemical techniques. The target value of the dataset is the PCE of each molecule, and the input is the variable-length character sequence of the molecule structures. Previous studies have handcrafted [259, 36, 113] or learned fingerprint features [66] that transforms the raw string data into fixed-size features for prediction.

Table 4.1 Regression experiment: Average test negative log likelihood

Dataset	N	D	VIP-BNN	VIP-NS	VI	VDO	$\alpha = 0.5$	SVGP	exact GP	fBNN
boston	506	13	**2.45±0.04**	**2.45±0.03**	2.76±0.04	2.63±0.10	**2.45±0.02**	2.63±0.04	2.46±0.04	2.30±0.10
concrete	1030	8	**3.02±0.02**	3.13±0.02	3.28±0.01	3.23±0.01	3.06±0.03	3.4±0.01	3.05±0.02	3.09±0.01
energy	768	8	0.60±0.03	**0.59±0.04**	2.17±0.02	1.13±0.02	0.95±0.09	2.31±0.02	0.57±0.02	0.68±0.02
kin8nm	8192	8	**-1.12±0.01**	-1.05±0.00	-0.81±0.01	-0.83±0.01	-0.92±0.02	-0.76±0.00	N/A±0.00	N/A±0.00
power	9568	4	2.92±0.00	2.90±0.00	2.83±0.01	2.88±0.00	**2.81±0.00**	2.82±0.00	N/A±0.00	N/A±0.00
protein	45730	9	**2.87±0.00**	2.96±0.02	3.00±0.00	2.99±0.00	2.90±0.00	3.01±0.00	N/A±0.00	N/A±0.00
red wine	1588	11	**0.97±0.02**	1.20±0.04	1.01±0.02	**0.97±0.02**	1.01±0.02	0.98±0.02	0.26±0.03	1.04±0.01
yacht	308	6	**-0.02±0.07**	0.59±0.13	1.11±0.04	1.22±0.18	0.79±0.11	2.29±0.03	0.10±0.05	1.03±0.03
naval	11934	16	-5.62±0.04	-4.11±0.00	-2.80±0.00	-2.80±0.00	-2.97±0.14	**-7.81±0.00**	N/A±0.00	N/A±0.00
Avg.Rank			**1.77±0.54**	2.77±0.57	4.66±0.28	3.88±0.38	2.55±0.37	4.44±0.66	N/A±0.00	N/A±0.00

Table 4.2 Regression experiment: Average test RMSE

Dataset	N	D	VIP-BNN	VIP-NS	VI	VDO	$\alpha = 0.5$	SVGP	exact GP	fBNN
boston	506	13	2.88±0.14	**2.78±0.12**	3.85±0.22	3.15±0.11	3.06±0.09	3.30±0.21	2.95±0.12	2.37±0.101
concrete	1030	8	**4.81±0.13**	5.54±0.09	6.51±0.10	6.11±0.10	5.18±0.16	7.25±0.15	5.31±0.15	4.93±0.18
energy	768	8	**0.45±0.01**	**0.45±0.05**	2.07±0.05	0.74±0.04	0.51±0.03	2.39±0.06	0.45±0.01	0.41±0.01
kin8nm	8192	8	**0.07±0.00**	0.08±0.00	0.10±0.00	0.10±0.00	0.09±0.00	0.11±0.01	N/A±0.00	N/A±0.00
power	9568	4	4.11±0.05	4.11±0.04	4.11±0.04	4.38±0.03	4.08±0.00	**4.06±0.04**	N/A±0.00	N/A±0.00
protein	45730	9	**4.25±0.07**	4.54±0.03	4.88±0.04	4.79±0.01	4.46±0.00	4.90±0.01	N/A±0.00	N/A±0.00
red wine	1588	11	**0.64±0.01**	0.66±0.01	0.66±0.01	**0.64±0.01**	0.69±0.01	0.65±0.01	0.62±0.01	0.67±0.01
yacht	308	6	**0.32±0.06**	0.54±0.09	0.79±0.05	1.03±0.06	0.49±0.04	2.25±0.13	0.35±0.04	0.60±0.06
naval	11934	16	**0.00±0.00**	**0.00±0.00**	0.38±0.00	0.01±0.00	0.01±0.00	**0.00±0.00**	N/A±0.00	N/A±0.00
Avg.Rank			**1.33±0.23**	2.22±0.36	4.66±0.33	4.00±0.44	3.11±0.42	4.44±0.72	N/A±0.00	N/A±0.00

We use a VIP with a prior defined by a Bayesian LSTM (200 hidden units) and $\alpha = 0.5$. We replicate the experimental settings in Bui et al. [36], Hernández-Lobato et al. [113], except that our method directly takes raw sequential molecule structure data as input. We compare our approach with a deep GP trained with expectation propagation [DGP, 36], variational dropout for LSTM [VDO-LSTM, 76], alpha-variational inference LSTM [α-LSTM, 177], BB-α on BNN [113], VI on BNN [29], and FITC GP [315]. Results for the latter 4 methods are quoted from Hernández-Lobato et al. [113], Bui et al. [36]. Results in Figure 4.4 show that VIP significantly outperforms other baselines and hits a state-of-the-art result in test likelihood and RMSE.

4.4.5 ABC example: the Lotka–Volterra model

Finally, we apply the VIP on an Approximate Bayesian Computation (ABC) example with the Lotka–Volterra (L-V) model that models the continuous dynamics of stochastic population of a predator-prey system. An L-V model consists of 4 parameters $\theta = \{\theta_1, \theta_2, \theta_3, \theta_4\}$ that controls the rate of four possible random events in the model:

$$\dot{y} = \theta_1 xy - \theta_2 y, \quad \dot{x} = \theta_3 x - \theta_4 xy,$$

Table 4.3 ABC with the Lotka–Volterra model

Method	VIP-BNN	VDO-BNN	SVGP	MCMC-ABC	SMC-ABC
Test NLL	**0.485**	1.25	1.266	0.717	0.588
Test RMSE	**0.094**	0.80	0.950	0.307	0.357

where x is the population of the predator, and y is the population of the prey. Therefore the L-V model is an implicit model, which allows the simulation of data but not the evaluation of model density. We follow the setup of [253] to select the ground truth parameter of the L-V model, so that the model exhibit a oscillatory behavior which makes posterior inference difficult. Then the L-V model is simulated for 25 time units with a step size of 0.05, resulting in 500 training observations. The prediction task is to extrapolate the simulation to the $[25,30]$ time interval.

We consider (approximate) posterior inference using two types of approaches: regression-based methods (VIP-BNN, VDO-BNN and SVGP), and ABC methods (MCMC-ABC [207] and SMC-ABC [20, 30]). ABC methods first perform posterior inference in the parameter space, then use the L-V simulator with posterior parameter samples for prediction. By contrast, regression-based methods treat this task as an ordinary regression problem, where VDO-BNN fits an approximate posterior to the NN weights, and VIP-BNN/SVGP perform predictive inference directly in function space. Results are shown in Table 4.3, where VIP-BNN outperforms others by a large margin in both test NLL and RMSE. More importantly, VIP is the only regression-based method that outperforms ABC methods, demonstrating its flexibility in modeling implicit systems.

4.5 Related works

In the world of nonparametric models, Gaussian Processes [GPs, 269] provide accurate uncertainty estimates on unseen data, making them popular choices for Bayesian modelling in the past decades. Unfortunately, the $\mathcal{O}(N^3)$ time and $\mathcal{O}(N^2)$ space complexities make GPs impractical for large-scale datasets, therefore people often resort to approximations [261, 315, 335, 109, 38, 287]. Another intrinsic issue is the limited representational power of GPs with stationary kernels, limiting the applications of GP methods to high dimensional data [23].

In the world of parametric modeling, deep neural networks are extremely flexible function approximators that enable learning from very high-dimensional and structured data [22, 115, 288, 162, 314]. As people starts to apply deep learning techniques to critical applications such as health care, uncertainty quantification of neural networks has become increasingly important. Although decent progress has been made for Bayesian neural networks (BNNs) [56, 117, 17, 243, 91, 29, 111, 177], uncertainty in deep learning still remains an open challenge.

Research in the *GP-BNN correspondance* has been extensively explored in order to improve the understandings of both worlds [242, 243, 365, 104, 75, 173, 214]. Notably, in Neal [242], Gal and Ghahramani [75] a one-layer BNN with non-linearity $\sigma(\cdot)$ and mean-field Gaussian prior is approximately equivalent to a GP with kernel function

$$\mathfrak{K}_{\mathrm{VDO}}(\mathbf{x}_1, \mathbf{x}_2) = \mathbb{E}_{p(\mathbf{w})p(b)}[\sigma(\mathbf{w}^\top \mathbf{x}_1 + b)\sigma(\mathbf{w}^\top \mathbf{x}_2 + b)].$$

Later Lee et al. [173] and Matthews et al. [214] showed that a deep BNN is approximately equivalent to a GP with a *compositional kernel* [42, 108, 50, 257] that mimic the deep net. These approaches allow us to construct expressive kernels for GPs [159], or conversely, exploit the *exact* Bayesian inference on GPs to perform exact Bayesian prediction for BNNs [173]. The above kernel is compared with equation (4.11) in Appendix 4.C.

Alternative schemes have also been investigated to exploit deep structures for GP model design. These include: (1) *deep GPs* [49, 36], where compositions of GP priors are proposed to represent prior over compositional functions; (2) the search and design of kernels for accurate and efficient learning [346, 65, 337, 21, 293], and (3) *deep kernel learning* that uses deep neural net features as the inputs to GPs [116, 367, 4, 34, 130]. Frustratingly, the first two approaches still struggle to model high-dimensional structured data such as texts and images; and the third approach is only Bayesian w.r.t. the last output layer.

The intention of our work is not to understand BNNs as GPs, nor to use deep learning to help GP design. Instead we directly treat a BNN as an instance of implicit processes (IPs), and the GP is used as a *variational distribution* to assist predictive inference. This approximation does not require previous assumptions in the GP-BNN correspondence literature [173, 214] nor the conditions in compositional kernel literature. Therefore the VIP approach also retains some of the benefits of Bayesian nonparametric approaches, and avoids issues of weight-space inference such as symmetric posterior modes.

To certain extent, the approach in Flam-Shepherd et al. [69] resembles an inverse of VIP by encoding properties of GP priors into BNN weight priors, which is then used to regularize BNN inference. This idea is further investigated by a concurrent work on functional BNNs [330], where GP priors are directly used to regularize BNN training through gradient estimators [309].

Concurrent work of neural process [80] resembles the neural sampler, a special case of IPs. However, it performs inference in **z** space using the variational auto-encoder approach [149, 273], which is not applicable to other IPs such as BNNs. By contrast, the proposed VIP approach applies to any IPs, and performs inference in function space. In the experiments we also show improved accuracies of the VIP approach on neural samplers over many existing Bayesian approaches.

4.6 Conclusions

We presented a variational approach for learning and Bayesian inference over function space based on implicit process priors. It provides a powerful framework that combines the rich flexibilities of implicit models with the well-calibrated uncertainty estimates from (parametric/nonparametric) Bayesian models. As an example, with BNNs as the implicit process prior, our approach outperformed many existing GP/BNN methods and achieved significantly improved results on molecule regression data. Many directions remain to be explored. Better posterior approximation methods beyond GP prior matching in function space will be designed. Classification models with implicit process priors will be developed. Implicit process latent variable models will also be derived in a similar fashion as Gaussian process latent variable models. A promising direction of application would be investigating novel inference methods for models equipped with other implicit process priors, e.g. data simulators in astrophysics, ecology and climate science.

Appendix for Chapter 4

Appendix 4.A Derivations

4.A.1 Proof of Proposition 1 (finite dimensional case)

Proposition 1. *If* $\mathbf{z}$ *is a finite dimensional random variable, then there exists a unique stochastic process, with finite marginals that are distributed exactly according to Definition 4.5.*

Proof Generally, consider the following noisy IP model:

$$f(\cdot) \sim \mathcal{IP}(g_{\boldsymbol{\theta}}(\cdot,\cdot), p_{\mathbf{z}}), \;\; y_n = f(\mathbf{x}_n) + \varepsilon_n, \; \varepsilon_n \sim \mathcal{N}(0, \sigma^2).$$

For any finite collection of random variables $y_{1:n} = \{y_1, ..., y_n\}$, $\forall n$ we denote the induced distribution as $p_{1:n}(y_{1:n})$. Note that $p_{1:n}(y_{1:n})$ can be represented as $\mathbb{E}_{p(\mathbf{z})}[\prod_{i=1}^{n} \mathcal{N}(y_i; g(\mathbf{x}_i; \mathbf{z}), \sigma^2)]$. Therefore for any $m < n$, we have

$$\begin{aligned}
&\int p_{1:n}(y_{1:n}) dy_{m+1:n} \\
&= \int\int \prod_{i=1}^{n} \mathcal{N}(y_i; g(\mathbf{x}_i, \mathbf{z}), \sigma^2) p(\mathbf{z}) d\mathbf{z} dy_{m+1:n} \\
&= \int\int \prod_{i=1}^{n} \mathcal{N}(y_i; g(\mathbf{x}_i, \mathbf{z}), \sigma^2) p(\mathbf{z}) dy_{m+1:n} d\mathbf{z} \\
&= \int \prod_{i=1}^{m} \mathcal{N}(y_i; g(\mathbf{x}_i, \mathbf{z}), \sigma^2) p(\mathbf{z}) d\mathbf{z} = p_{1:m}(y_{1:m}).
\end{aligned}$$

Note that the swap of the order of integration relies on that the integral is finite, which is true when the prior $p(\mathbf{z})$ is proper. Therefore, the marginal consistency condition of Kolmogorov extension theorem is satisfied. Similarly, the permutation consistency condition of Kolmogorov extension theorem can be proved as follows: assume $\pi(1:n) = \{\pi(1), ..., \pi(n)\}$

is a permutation of the indices $1:n$, then

$$\begin{aligned}
& p_{\pi(1:n)}(y_{\pi(1:n)}) \\
&= \int \prod_{i=1}^{n} \mathcal{N}(y_{\pi(i)}; g(\mathbf{x}_{\pi(i)}, \mathbf{z}), \sigma^2) p(\mathbf{z}) d\mathbf{z} \\
&= \int \prod_{i=1}^{n} \mathcal{N}(y_i; g(\mathbf{x}_i, \mathbf{z}), \sigma^2) p(\mathbf{z}) d\mathbf{z} = p_{1:n}(y_{1:n}).
\end{aligned}$$

Therefore, by Kolmogorov extension theorem, there exists a unique stochastic process, with finite marginals that are distributed exactly according to Definition 4.5.

□

4.A.2 Proof of Proposition 2 (infinite dimensional case)

Proposition 2. *Let $z(\cdot) \sim \mathcal{SP}(0, \mathfrak{C})$ be a centered continuous stochastic process on $\mathcal{L}^2(\mathbb{R}^d)$ with covariance function $\mathfrak{C}(\cdot,\cdot)$. Then the operator $g(\mathbf{x}, z) = O(z)(\mathbf{x}) := h(\int \sum_{l=0}^{M} \mathfrak{K}_l(\mathbf{x}, \mathbf{x}')z(\mathbf{x}')d\mathbf{x}')$, $0 < M < +\infty$ defines a stochastic process if $\mathfrak{K}_l \in \mathcal{L}^2(\mathbb{R}^d \times \mathbb{R}^d)$, h is a Borel measurable, bijective function in $\mathbb{R}$ and there exist $0 \leq A < +\infty$ such that $|h(x)| \leq A|x|$ for $\forall x \in \mathbb{R}$.*

Proof Since $\mathcal{L}^2(\mathbb{R}^d)$ is closed under finite summation, without loss of generality, we consider the case of $M = 1$ where $O(z)(\mathbf{x}) = h(\int \mathfrak{K}(\mathbf{x}, \mathbf{x}')z(\mathbf{x}')d\mathbf{x}')$. According to Karhunen-Loeve expansion (K-L expansion) theorem [194], the stochastic process z can be expanded as the *stochastic* infinite series,

$$z(\mathbf{x}) = \sum_{i}^{\infty} Z_i \phi_i(\mathbf{x}), \quad \sum_{i}^{\infty} \lambda_i < +\infty.$$

Where Z_i are zero-mean, uncorrelated random variables with variance λ_i. Here $\{\phi_i\}_{i=1}^{\infty}$ is an orthonormal basis of $\mathcal{L}^2(\mathbb{R}^d)$ that are also eigen functions of the operator $O_{\mathfrak{C}}(z)$ defined by $O_{\mathfrak{C}}(z)(\mathbf{x}) = \int \mathfrak{C}(\mathbf{x}, \mathbf{x}')z(\mathbf{x}')d\mathbf{x}'$. The variance λ_i of Z_i is the corresponding eigen value of $\phi_i(\mathbf{x})$.

Apply the linear operator

$$O_{\mathfrak{K}}(z)(\mathbf{x}) = \int \mathfrak{K}(\mathbf{x}, \mathbf{x}')z(\mathbf{x}')d\mathbf{x}'$$

on this K-L expansion of z, we have:

$$\begin{aligned} O_{\mathfrak{K}}(z)(\mathbf{x}) &= \int \mathfrak{K}(\mathbf{x},\mathbf{x}')z(\mathbf{x}')d\mathbf{x}' \\ &= \int \mathfrak{K}(\mathbf{x},\mathbf{x}')\sum_i^{\infty} Z_i\phi_i(\mathbf{x}')d\mathbf{x}' \\ &= \sum_i^{\infty} Z_i \int \mathfrak{K}(\mathbf{x},\mathbf{x}')\phi_i(\mathbf{x}')d\mathbf{x}', \end{aligned} \tag{4.A.1}$$

where the exchange of summation and integral is guaranteed by Fubini's theorem. Therefore, the functions $\{\int_{\mathbf{x}} \mathfrak{K}(\mathbf{x},\mathbf{x}')\phi_i(\mathbf{x}')d\mathbf{x}'\}_{i=1}^{\infty}$ forms a new basis of $\mathcal{L}^2(\mathbb{R}^d)$. To show that the stochastic series 4.A.1 converge:

$$\begin{aligned} &||\sum_i^{\infty} Z_i \int \mathfrak{K}(\mathbf{x},\mathbf{x}')\phi_i(\mathbf{x}')d\mathbf{x}||_{\mathcal{L}^2}^2 \\ &\leq ||O_K||^2 ||\sum_i^{\infty} Z_i\phi_i(\mathbf{x}')||_{\mathcal{L}^2}^2 \\ &= ||O_K||^2 \sum_i^{\infty} ||Z_i||_2^2, \end{aligned}$$

where the operator norm is defined by

$$||O_{\mathfrak{K}}|| := \inf\{c \geq 0 : ||O_{\mathfrak{K}}(f)||_{\mathcal{L}^2} \leq c||f||_{\mathcal{L}^2}, \forall f \in \mathcal{L}^2(\mathbb{R}^d)\}.$$

This is a well defined norm since O_K is a bounded operator ($\mathfrak{K} \in \mathcal{L}^2(\mathbb{R}^d \times \mathbb{R}^d)$). The last equality follows from the orthonormality of $\{\phi_i\}$. The condition $\sum_i^{\infty} \lambda_i < \infty$ further guarantees that $\sum_i^{\infty} ||Z_i||^2$ converges almost surely. Therefore, the random series (4.A.1) converges in $\mathcal{L}^2(\mathbb{R}^d)$ a.s..

Finally we consider the nonlinear mapping $h(\cdot)$. With $h(\cdot)$ a Borel measurable function satisfying the condition that there exist $0 \leq A < +\infty$ such that $|h(x)| \leq A|x|$ for $\forall x \in \mathbb{R}$, it follows that $h \circ O_{\mathfrak{K}}(z) \in \mathcal{L}^2(\mathbb{R}^d)$. In summary, $g = O_{\mathfrak{K}}(z) = h \circ O_{\mathfrak{K}}(z)$ defines a well-defined stochastic process on $\mathcal{L}^2(\mathbb{R}^d)$.

□

Despite of its simple form, the operator $g = h \circ O_{\mathfrak{K}}(z)$ is in fact the building blocks for many flexible transformations over functions [97, 365, 327, 169, 84] . Recently Guss [97] proposed the so called Deep Function Machines (DFMs) that possess universal approximation ability to nonlinear operators:

Definition 4.6 (Deep Function Machines [97]). *A deep function machine $g = O_{DFM}(z, S)$ is a computational skeleton S indexed by I with the following properties:*

- *Every vertex in S is a Hilbert space $\mathbb{H}_l$ where $l \in I$.*
- *If nodes $l \in A \subset I$ feed into l' then the activation on l' is denoted $y^l \in \mathbb{H}_l$ and is defined as*
$$y^{l'} = h \circ (\sum_{l \in A} O_{\mathfrak{K}_l}(y^l))$$

Therefore, by Proposition 2, we have proved:
Corollary 2 *Let $z(\cdot) \sim \mathcal{SP}(0, \mathfrak{C})$ be a centered continuous stochastic process on $\mathbb{H} = \mathcal{L}^2(\mathbb{R}^d)$. Then the Deep function machine operator $g = O_{DFM}(z, S)$ defines a well-defined stochastic process on $\mathbb{H}$.*

4.A.3 Inverse Wishart process as a prior for kernel functions

Definition 4.7 (Inverse Wishart processes [306]). *Let Σ be random function $\Sigma(\cdot,\cdot) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. A stochastic process defined on such functions is called the inverse Wishart process on $\mathcal{X}$ with parameter ν and base function $\Psi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, if for any finite collection of input data $\mathbf{X} = \{\mathbf{x}_s\}_{1 \le s \le N_s}$, the corresponding matrix-valued evaluation $\Sigma(\mathbf{X}, \mathbf{X}) \in \Pi(N_s)$ is distributed according to an inverse Wishart distribution $\Sigma(\mathbf{X}, \mathbf{X}) \sim IW_S(\nu, \Psi(\mathbf{X}, \mathbf{X}))$. We denote $\Sigma \sim \mathcal{IWP}(\nu, \Psi(\cdot,\cdot))$.*

Consider the problem in Section 4.3.1 of minimizing the objective

$$\mathcal{U}(\mathfrak{M}, \mathfrak{K}) = \mathcal{D}_{\mathrm{KL}}[p(\mathbf{f}, \mathbf{y}|\mathbf{X}, \boldsymbol{\theta})||q_{\mathcal{GP}}(\mathbf{f}, \mathbf{y}|\mathbf{X}, \mathfrak{M}(\cdot), \mathfrak{K}(\cdot,\cdot))]$$

Since we use $q(\mathbf{y}|\mathbf{f}) = p(\mathbf{y}|\mathbf{f})$, this reduces $\mathcal{U}(\mathfrak{M}, \mathfrak{K})$ to $\mathrm{D}_{KL}[p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta})||q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X}, \mathfrak{M}, \mathfrak{K})]$. In order to obtain optimal solution wrt. $\mathcal{U}(\mathfrak{M}, \mathfrak{K})$, it sufficies to draw S fantasy functions (each sample is a random function $f_s(\cdot)$) from the prior distribution $p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta})$, and perform moment matching, which gives exactly the MLE solution, i.e., empirical mean and covariance functions

$$\mathfrak{M}^{\star}_{\text{MLE}}(\mathbf{x}) = \sum_s \frac{1}{S} f_s(\mathbf{x}), \tag{4.A.2}$$

$$\mathfrak{K}^{\star}_{\text{MLE}}(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{S} \sum_s \Delta_s(\mathbf{x}_1)\Delta_s(\mathbf{x}_2), \tag{4.A.3}$$

$$\Delta_s(\mathbf{x}) = f_s(\mathbf{x}) - \mathfrak{M}^{\star}_{\text{MLE}}(\mathbf{x}). \tag{4.A.4}$$

In practice, in order to gain computational advantage, the number of fantasy functions S is often small, therefore we further put an inverse wishart process prior over the GP covariance function, i.e. $\mathfrak{K}(\cdot,\cdot) \sim \mathcal{IWP}(\nu, \Psi)$. By doing so, we are able to give MAP estimation instead of MLE estimation. Since inverse Wishart distribution is conjugate to multivariate Gaussian distribution, the maximum a posteriori (MAP) solution is given by

$$\begin{aligned} &\mathfrak{K}^{\star}_{\text{MAP}}(\mathbf{x}_1, \mathbf{x}_2) \\ &= \frac{1}{\nu + S + N + 1} \{ \sum_s \Delta_s(\mathbf{x}_1)\Delta_s(\mathbf{x}_2) + \Psi(\mathbf{x}_1, \mathbf{x}_2) \}. \end{aligned} \tag{4.A.5}$$

Where N is the number of data points in the training set $\mathbf{X}$ where $m(\cdot)$ and $\mathfrak{K}(\cdot,\cdot)$ are evaluated. Alternatively, one could also use the posterior mean Estimator (PM) that minimizes posterior expected squared loss:

$$\begin{aligned} &\mathfrak{K}^{\star}_{\text{PM}}(\mathbf{x}_1, \mathbf{x}_2) \\ &= \frac{1}{\nu + S - N - 1} \{ \sum_s \Delta_s(\mathbf{x}_1)\Delta_s(\mathbf{x}_2) + \Psi(\mathbf{x}_1, \mathbf{x}_2) \}. \end{aligned} \tag{4.A.6}$$

In the implementation of this paper, we choose $\mathfrak{K}_{\text{PM}}$ estimator with $\nu = N$ and $\Psi(\mathbf{x}_1, \mathbf{x}_2) = \psi\delta(\mathbf{x}_1, \mathbf{x}_2)$. The hyper parameter ψ is trained using fast grid search using the same procedure for the noise variance parameter, as detailed in Appendix 4.D.

4.A.4 Derivation of the upper bound $\mathcal{U}(m, \mathfrak{K})$ for sleep phase

Applying the chaine rule of KL-divregence, we have

$$\begin{aligned}\mathcal{J}(\mathfrak{M}, \mathfrak{K}) =& \mathcal{D}_{\mathrm{KL}}[p(\mathbf{f}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta})||q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X}, \mathbf{y}, \mathfrak{M}(\cdot), \mathfrak{K}(\cdot, \cdot))] \\ =& \mathcal{D}_{\mathrm{KL}}[p(\mathbf{f}, \mathbf{y}|\mathbf{X}, \boldsymbol{\theta})||q_{\mathcal{GP}}(\mathbf{f}, \mathbf{y}|\mathbf{X}, \mathfrak{M}(\cdot), \mathfrak{K}(\cdot, \cdot))] \\ & - \mathcal{D}_{\mathrm{KL}}[p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})||q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X}, \mathfrak{M}(\cdot), \mathfrak{K}(\cdot, \cdot))] \\ =& \mathcal{U}(\mathfrak{M}, \mathfrak{K}) - \mathcal{D}_{\mathrm{KL}}[p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})||q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X}, \mathfrak{M}(\cdot), \mathfrak{K}(\cdot, \cdot))].\end{aligned}$$

Therefore, by the non-negative property of KL divergence, we have $\mathcal{J}(\mathfrak{M}, \mathfrak{K}) < \mathcal{U}(\mathfrak{M}, \mathfrak{K})$. Since we select $q(\mathbf{y}|\mathbf{f}) = p(\mathbf{y}|\mathbf{f})$, the optimal solution of $\mathcal{U}(\mathfrak{M}, \mathfrak{K})$ also minimizes

$$\mathcal{D}_{\mathrm{KL}}(p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})||q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X}, \mathfrak{M}(\cdot), \mathfrak{K}(\cdot, \cdot)))$$

. Therefore not only the upper bound $\mathcal{U}$ is optimized in sleep phase, the gap

$$-\mathcal{D}_{\mathrm{KL}}(p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})||q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X}, \mathfrak{M}(\cdot), \mathfrak{K}(\cdot, \cdot)))$$

is also decreased when the mean and covariance functions are optimized.

4.A.5 Empirical Bayes approximation for VIP with a hierarchical prior on θ

The implicit processes (such as Bayesian neural networks and GPs) could be sensitive to the choice of the model parameters (that is, parameters $\boldsymbol{\theta}$ of the prior). To make our variational implicit process more robust we further present an empirical Bayesian treatment, by introducing an extra hierarchical prior distribution $p(\boldsymbol{\theta})$ on the prior parameters $\boldsymbol{\theta}$, and fitting a variational approximation $q(\boldsymbol{\theta})$ to the posterior. Sleep phase updates remain the same when conditioned on a given configuration of $\boldsymbol{\theta}$. The α-energy term in wake phase learning becomes

$$\begin{aligned}
&\log q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X}) \\
&= \log \int_{\boldsymbol{\theta}} q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta} \approx \mathcal{L}^{\alpha}_{\mathcal{GP}}(q(\mathbf{a}),q(\boldsymbol{\theta})), \\
&\mathcal{L}^{\alpha}_{\mathcal{GP}}(q(\mathbf{a}),q(\boldsymbol{\theta})) \\
&= \frac{1}{\alpha}\sum_{n}^{N} \log \mathbb{E}_{q(\mathbf{a})q(\boldsymbol{\theta})}\left[q^{\star}(y_n|\mathbf{x}_n,\mathbf{a},\boldsymbol{\theta})^{\alpha}\right] \\
&- \mathrm{D}_{\mathrm{KL}}[q(\mathbf{a})||p(\mathbf{a})] - \mathrm{D}_{\mathrm{KL}}[q(\boldsymbol{\theta})||p(\boldsymbol{\theta})].
\end{aligned} \tag{4.A.7}$$

Compared with the approximate MLE method, the only extra term needs to be estimated is $-\mathcal{D}_{\mathrm{KL}}[q(\boldsymbol{\theta})||p(\boldsymbol{\theta})]$. Note that, introducing $q(\boldsymbol{\theta})$ will double the number of parameters. In the case of Bayesian NN as an IP, where $\boldsymbol{\theta}$ contains means and variances for weight priors, then a simple Gaussian $q(\boldsymbol{\theta})$ will need two sets of means and variances variational parameters (i.e., posterior means of means, posterior variances of means,posterior means of variances, posterior variances of variances). Therefore, to make the representation compact, we choose $q(\boldsymbol{\theta})$ to be a Dirac-delta function $\delta(\boldsymbol{\theta}_q)$, which results in an *empirical Bayesian solution.*

Another possible alternative approach is, instead of explicitly specifying the form and hyperparameters for $p(\boldsymbol{\theta})$,we can notice that from standard variational lower bound

$$\log q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X}) \approx \mathbb{E}_{q(\boldsymbol{\theta})}[\log q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\boldsymbol{\theta})] - \mathcal{D}_{\mathrm{KL}}[q(\boldsymbol{\theta})||p(\boldsymbol{\theta})].$$

Then $\mathcal{D}_{\mathrm{KL}}[q(\boldsymbol{\theta})||p(\boldsymbol{\theta})]$ can be approximated by

$$\begin{aligned}
-\mathcal{D}_{\mathrm{KL}}[q(\boldsymbol{\theta})||p(\boldsymbol{\theta})] &\approx -\mathbb{E}_{q(\boldsymbol{\theta})}[\log q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\boldsymbol{\theta})] + \text{constant} \\
&= -\log q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\boldsymbol{\theta}_q) + \text{constant}
\end{aligned}$$

Therefore, we can use $-\log q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\boldsymbol{\theta}_q)$ as the regularization term instead, which penalizes the parameter configurations that returns a *full* marginal log likelihood (as opposed to the diagonal likelihood in the original BB-α energy $\frac{1}{\alpha}\sum_n^N \log \mathbb{E}_{q(\mathbf{z})q(\boldsymbol{\theta})} q_{\mathcal{GP}}(y_n|\mathbf{x}_n,\mathbf{z},\boldsymbol{\theta})^{\alpha})$ that is too high, especially the contribution from non-diagonal covariances. We refer this as *likelihood regularization.* In practice, $-\log q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\boldsymbol{\theta}_q)$ is estimated on each mini-batch.

Appendix 4.B KL divergence on function space v.s. KL divergence on weight space

We briefly discuss KL divergence on function space in finite dimensional case. In the sleep phase of VIP, we have proposed minimizing the following KL divergence in function space:

$$\mathcal{U}(\mathfrak{M}, \mathfrak{K}) = \mathrm{D}_{\mathrm{KL}}[p(\mathbf{y}, \mathbf{f}|\mathbf{X}, \boldsymbol{\theta})||q_{\mathcal{GP}}(\mathbf{y}, \mathbf{f}|\mathbf{X}, \mathfrak{M}, \mathfrak{K})]. \tag{4.B.1}$$

This is an example of KL divergence in function space (i.e., the output **f**). Generally speaking, we may assume that $p(\mathbf{f}) = \int_{\mathbf{W}} p(\mathbf{f}|\mathbf{W})p(\mathbf{W})d\mathbf{W}$, and $q(\mathbf{f}) = \int_{\mathbf{W}} p(\mathbf{f}|\mathbf{W})q(\mathbf{W})$, where $q(\mathbf{W})$ is weight-space variational approximation. That is to say, both stochastic processes p and q can be generated by finite dimensional weight space representation **W**. This can be seen as a one-step Markov chain with preivious state $s_t = \mathbf{W}$, new state $s_{t+1} = \mathbf{f}$, and probability transition function $r(s_{t+1}|s_t) = p(\mathbf{f}|\mathbf{W})$. Then, by applying the second law of thermodynamics of Markov chains(Cover and Thomas [44]), we have:

$$\mathrm{D}_{\mathrm{KL}}[p(\mathbf{f})||q(\mathbf{f})] \leq \mathrm{D}_{\mathrm{KL}}[p(\mathbf{W})||q(\mathbf{W})] \tag{4.B.2}$$

This shows that the KL divergence in function space forms a tighter bound than the KL divergence on weight space, which is one of the merits of function space inference.

Appendix 4.C Further discussions on Bayesian neural networks

We provide a comparison between our kernel in equation (4.11), and the kernel proposed in Gal and Ghahramani [75]. Notably, consider the following Gaussian process:

$$\begin{aligned}
&y(\cdot) \sim \mathcal{GP}(0, \mathfrak{K}_{\mathrm{VDO}}(\cdot,\cdot)),\\
&\mathfrak{K}_{\mathrm{VDO}}(\mathbf{x}_1, \mathbf{x}_2) = \\
&\int p(\mathbf{w})p(b)\sigma(\mathbf{w}^\top \mathbf{x}_1 + b)\sigma(\mathbf{w}^\top \mathbf{x}_2 + b)d\mathbf{w}db.
\end{aligned} \tag{4.C.1}$$

Here $\sigma(\cdot)$ is a non-linear activation function, **w** is a vector of length D, b is the bias scaler, and $p(\mathbf{w})$, $p(b)$ the corresponding prior distributions. Gal and Ghahramani [75] considered approximating this GP with a one-hidden layer BNN $\hat{y}(\cdot) = \mathrm{BNN}(\cdot, \boldsymbol{\theta})$ with $\boldsymbol{\theta}$ collecting the weights and bias vectors of the network. Denote the weight matrix of the first layer as $\boldsymbol{W} \in \mathbb{R}^{D\times K}$, i.e. the network has K hidden units, and the kth column of $\boldsymbol{W}$ as $\mathbf{w}_k$. Similarly the bias vector is $\boldsymbol{b} = (b_1, ..., b_K)$. We further assume the prior distributions of the first-layer parameters are $p(\boldsymbol{W}) = \prod_{k=1}^K p(\mathbf{w}_k)$ and $p(\boldsymbol{b}) = \prod_{k=1}^K p(b_k)$, and use mean-field Gaussian prior for the output layer. Then this BNN constructs an approximation to the GP kernel as:

$$\tilde{\mathfrak{K}}_{\mathrm{VDO}}(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{K}\sum_k \sigma(\mathbf{w}_k^\top \mathbf{x}_1 + b_k)\sigma(\mathbf{w}_k^\top \mathbf{x}_2 + b_k),$$
$$\mathbf{w}_k \sim p(\mathbf{w}), \quad b_k \sim p(b).$$

This approximation is equivalent to the empirical estimation (4.11), if $S = K$ and the IP is defined by

$$\begin{gathered} g_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) = \sigma(\mathbf{w}^\top \mathbf{x} + b), \mathbf{z} = \{\mathbf{w}, b\}, p(\mathbf{z}) = p(\mathbf{w})p(b), \\ p(\mathbf{z}),\ \sigma(\cdot) \text{ satisfy } \mathbb{E}_{p(\mathbf{z})}[\sigma(\mathbf{w}^\top \mathbf{x} + b)] = 0. \end{gathered} \tag{4.C.2}$$

In such case, the output layer of that one-hidden layer BNN corresponds to the Bayesian linear regression "layer" in our final approximation. However, the two methods are motivated in different ways. Gal and Ghahramani [75] used this interpretation to approximate a GP with kernel (4.C.1) using a one-hidden layer BNN, while our goal is to approximate the IP 4.C.2 by a GP (note that the IP is defined as the output of the *hidden* layer, not the output of the BNN). Also this coincidence only applies when the IP is defined by a Bayesian logistic regression model, and our approximation is applicable to BNN and beyond.

Appendix 4.D Further experimental details

We provide further experimental details in this section. We opensource the code of VIP for UCI experiments at https://github.com/LaurantChao/VIP.

4.D.1 General settings for VIP

For small datasets we use the posterior GP equations for prediction, otherwise we use the $\mathcal{O}(S^3)$ approximation. We use $S = 20$ for VIP unless noted otherwise. When the VIP is equipped with a Bayesian NN/LSTM as prior over functions (Example 4.3-4.4), the prior parameters over each weight are untied, thus can be individually tuned. Empirical Bayesian estimates of the prior parameters are used in 4.4.3 and 4.4.4.

4.D.2 Further experimental details of synthetic example

The compositional kernel for GP is the summation of RBF and Periodic kernels. In this toy experiment, both VDO and VIP use a BNN as the underlying model. Note that it appears that the GP slightly overfits. It is possible to hand-pick the kernel parameters for a smoother fit of GP. However, we have found that quantitatively this will result in a decrease in test predictive likelihood and an increase of RMSE. Therefore, we chose to optimize the kernel parameters by maximizing the marginal likelihood.

4.D.3 Further implementation details for multivariate regression experiments

- Variational Gaussian inference for BNN (VI-BNN): we implement VI for BNN using the Bayesian deep learning library, ZhuSuan [307]. VI-BNN employs a mean-field Gaussian variational approximation but evaluates the variational free energy using the reparameterisation trick [149]. We use a diagonal Gaussian prior for the weights and fix the prior variance to 1. The noise variance of the Gaussian noise model is optimized together with the means and variances of the variational approximation using the variational free energy.

- Variational implicit process-Neural Sampler regressor (VIP-NS): we use neural sampler with two hidden layers of 10 hidden units. The input noise dimension is 10 or 50, which is determined using validation set.

- Variational dropout (VDO) for BNN: similar to Gal and Ghahramani [75], we fix the length scale parameter $0.5 * l^2 = 10e^{-6}$. Since the network size is relatively small, dropout probability is set as 0.005 or 0.0005. We use 2000 forward passes to evaluate posterior likelihood.

- α-dropout inference for BNN: suggested by Li and Gal [177], we fix $\alpha = 0.5$ which often gives high quality uncertainty estimations, possibility due to it is able to achieve a balance between reducing training error and improving predictive likelihood. We use $K = 10$ for MC sampling.

- Variational sparse GPs and exact GPs: we implement the GP-related algorithms using GPflow [215]. variational sparse GPs uses 50 inducing points. Both GP models use the RBF kernel.

- About noise variance parameter grid search for VIPs (VIP-BNN and VIP-NS), VDOs and α-dropout: we start with random noise variance parameter, run optimization on the model parameters, and then perform a (thick) grid search over noise variance parameter on validation set. Then, we train the model on the entire training set using this noise variance parameter value. This coordinate ascent like procedure does not require training the model for multiple times as in Bayesian optimization, therefore can speed up the learning process. The same procedure is used to search for optimal hyperparameter ψ of the inverse-Wishart process of VIPs.

4.D.4 Additional implementation details for ABC experiment

Following the experimental setting of Papamakarios and Murray [253], we set the ground truth L-V model parameter to be $\boldsymbol{\theta}_1 = 0.01, \boldsymbol{\theta}_2 = 0.5, \boldsymbol{\theta}_3 = 1.0, \boldsymbol{\theta}_4 = 0.01$. We simulate population data in the range of $[0, 30]$ with step size 0.05, which result in 600 gathered measurements. We use the first 500 measurements as training data, and the remaining as test set. For MCMC-ABC and SMC-ABC setup, we also follow the implementation of Papamakarios and Murray [253].[3] MCMC-ABC is ran for 10000 samples with tolerance ε set to be 2.0 which is manually tuned to give the best performance. In MCMC-ABC, last 100 samples are taken as samples. Likewise SMC-ABC uses 100 particles. Model likelihood is calculated based on Gaussian fit. VIP ($\alpha = 0$) is trained for 10000 iterations with Adam optimizer using 0.001 learning rate.

4.D.5 Additional implementation details for predicting power conversion efficiency of organic photovoltaics molecules

For Bayesian LSTMs, we put Gaussian prior distributions over LSTM weights. The output prediction is defined as the final output at the last time step of the input sequence. We use $S = 10$ for VIP. All methods use Adam with a learning rate of 0.001 for stochastic optimization. Noise variance parameter are not optimized, but set to suggested value according to Hernández-Lobato et al. [113].To match the run time of the fingerprint-based methods, all LSTM methods are trained for only 100 epochs with a batch size of 250. Among different models in the last few iterations of optimization, we choose the one with the best training likelihood for testing. Note that in the original paper of variational dropout and α-dropout inference, K sample paths ($K = 1$ for VDO and $K = 10$ for α-dropout) are

[3]https://github.com/gpapamak/epsilon_free_inference

created for *each* training data, which is too prohibitive for memory storage. Therefore, in our implementation, we enforce all training data to share K sample paths. This approximation is accurate since we use a small dropout rate, which is 0.005.

4.D.6 Additional Tables

Table 4.4 Interpolation performance on toy dataset.

Method	VIP	VDO	GP
Test NLL	**-0.60±0.01**	-0.07 ± 0.01	-0.27 ± 0.00
Test RMSE	**0.140±0.00**	0.161±0.00	0.152±0.00

Table 4.5 Interpolation performance on solar irradiance.

Method	VIP	VDO	SVGP	GP
Test NLL	**0.08±0.02**	0.21 ± 0.04	0.56 ± 0.23	0.832 ± 0.00
Test RMSE	**0.28±0.00**	0.29±0.01	0.55±0.08	0.650±0.0

Table 4.6 Performance on clean energy dataset

Metric	**VIP**	**VDO-LSTM**	**α-LSTM**	**BB-α**	**VI-BNN**	**FITC-GP**	**EP-DGP**
Test NLL	**0.65±0.01**	1.24±0.01	2.06±0.02	0.74±0.01	1.37±0.02	1.25±0.00	0.98±0.00
Test RMSE	**0.88±0.02**	0.93±0.01	1.38±0.02	1.08±0.01	1.07±0.01	1.35±0.00	1.17±0.00

Chapter 5

Functional Variational Inference

In Chapter 4, we have described two different approaches to function space inference: the model-driven approach and the algorithm-driven approach. The model-driven approach starts from an existing example of Bayesian nonparametric priors (in our case, the Gaussian process), extends it to a more flexible class of priors, and then develops the corresponding approximate inference algorithms. On the other hand, the algorithm-driven approach starts from an existing inference method for parameter-space, and develops its function-space counterpart.

So far we have presented our first approach to function space inference (VIP), which is based on GP posterior approximations using implicit process priors. Roughly speaking, in the VIP method the true posterior of a IP $p(f|\mathcal{D})$ is approximated by an approximate posterior of following form (denoted by $q_{\text{VIP}}(f|\mathcal{D})$):

$$q_{\text{VIP}}(f|\mathcal{D}) = \sum_s a_s \phi_s, \mathbf{a} \sim \mathcal{N}(\mathbf{a}; \boldsymbol{\mu}, \boldsymbol{\Sigma}), \tag{5.0.1}$$

where the basis functions $\{\phi_s\}_{s=1}^S$ are random samples drawn directly from $p(f)$.

Despite having demonstrated empirical advantage over weight space inference methods, VIPs still suffer from a number of issues, detailed in the following remarks.

Remark (Limitations of VIPs). VIP clearly has a few limitations that need to be addressed. To start with, its approximate posterior, $q_{\text{VIP}}(f|\mathcal{D})$ resembles a GP approximation to $p(f|\mathcal{D})$, whereas the true posterior in function space might be arbitrarily complex. Therefore, GP approximations might not be able to capture non-GP behaviors.

Also, VIP requires the implicit prior $p(f)$ to be reparameterizable, that is: 1), $p(f)$ should take the form of $f(\mathbf{x}) = g_\theta(\mathbf{x},\mathbf{z}), \mathbf{z} \sim p(\mathbf{z})$; 2), the function $g_\theta(\cdot,\cdot)$ must be differentiable; and 3), the functional form of $g_\theta(\cdot,\cdot)$ must be fully known in advance. These assumptions may limit its applicability to more complicated priors such as structured implicit priors [330]. Lastly, the wake-sleep procedure of VIP does not optimize a coherent function-space objective function (such as ELBO usually used in parameter-space VI).

Therefore, it is an open challenge how to improve and justify (variational) inference in the space of functions using priors given by stochastic processes. In this chapter, we investigate this old but important problem and propose a new solution called Functional Variational Inference (FVI). Our contributions are as follows:

- We propose a new objective function for variational inference in function space, as an alternative to functional KL divergence between stochastic processes [330]. We show that this new objective is a valid divergence, and can avoid some of the problems that the functional KL divergence has.
- We propose a new class of flexible variational distributions in function space, called stochastic process generators (SPGs). SPGs are non-Gaussian generalizations of the VIP family [198], and can help avoid the fundamental limitation of mean-field Gaussians [70] used in BNNs and functional BNNs. A theorem regarding the expressiveness of SPGs is proved (Proposition 5.4).
- Based on SPGs, our proposed functional divergence between stochastic processes can be estimated efficiently using mini-batch sampling (Proposition 5.5, 5.6 and 5.7), which achieves a significant speed-up against the gradient estimator approach in [330].
- We compare our methods against existing weight-space and function-space inference methods in several tasks. Our method consistently outperforms the baselines, and is much faster than f-BNN, which validates the effectiveness of our approach.

This chapter is organized as follows. In Section 5.1, we will formalize the framework of functional variational inference, and explain basic concepts for functional KL-divergences between stochastic processes, functional ELBO, as well as functional Bayesian neural networks. We will also introduce the theoretical pathologies of functional KL divergence minimization. In Section 5.2, we propose a more well-behaved functional divergence called grid-functional KL divergence, and use it as the objective function for function space inference. We derive a number of theoretical results and showcase how the corresponding

ELBO can be calculated. In Section 5.3, we introduce the definition of Stochastic Process Generators (SPGs), and derive efficient estimations of functional ELBO based on SPGs (Section 5.4). Finally, in Section 5.7, we apply FVI to several tasks and showcase their experimental performances.

5.1 Problem setting, and the functional KL divergence

In this chapter, we consider Bayesian inference problem in function space. Let $p(f)$ be a stochastic process defined on the probability space $(\Omega, \mathcal{B})$. Note that we use f in its scalar form to denote a scalar function $f(\cdot): \mathcal{T} \mapsto \mathbb{R}$. Here $\mathcal{T}$ is the index set of $p(f)$ (assumed to be a compact subset of $\mathbb{R}^d$). For example, $p(f)$ could be a Gaussian process $p(f) = \mathcal{GP}(\mathfrak{M}(\cdot), \mathfrak{K}(\cdot,\cdot))$, a Bayesian neural network, or any other suitable stochastic process. We use $p(f)$ to model the uncertainty in function space. Then, a likelihood function $p_\pi(y|f(\cdot))$ is defined on top of f to generated observable data y.

Given observed data $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$, our goal is to infer the posterior process $p(f|\mathcal{D})$ conditioned on the observations $\mathcal{D}$. If $p(f)$ is a GP, then $p(f|\mathcal{D})$ can be computed analytically. However, in most cases this is intractable. Therefore, following [213, 330], we define another stochastic process $q(f)$ on $(\Omega, \mathcal{B})$ as our variational family to approximate $p(f|\mathcal{D})$. $q(f)$ can be optimized by minimizing the variational objective

$$\mathrm{D}_{\mathrm{KL}}[q(f)||p(f|\mathcal{D})], \tag{5.1.1}$$

Note that $\mathrm{D}_{\mathrm{KL}}[q(f)||p(f|\mathcal{D})]$ is the functional KL-divergence between stochastic processes, $q(f)$ and $p(f|\mathcal{D})$. Unfortunately, both measures $q(f)$ and $p(f|\mathcal{D})$ does not have a convenient density form[1]. Therefore, $\mathrm{D}_{\mathrm{KL}}[q(f)||p(f|\mathcal{D})]$ can only be defined by following measure-theoretic definition.

Remark (Formal definition of functional KL divergence between stochastic processes). The concept of functional KL is nothing but the special case of the measure-theoretic definition of KL divergence $\mathrm{D}_{\mathrm{KL}}[Q||P]$ (Definition 2.1), by noticing that both $Q = q(f)$ and $P = p(f)$ are just measures over $\Omega^{\mathcal{T}}$, defined by

$$\Omega^{\mathcal{T}} = \{f(\cdot)|f(\mathbf{x}) \in \Omega, \mathbf{x} \in \mathcal{T}\}. \tag{5.1.2}$$

[1]Since there does not exist "useful" infinite dimensional Lebesgue measures [213].

Proposition 2.1 shows that when Q is absolutely continuous w.r.t. P (i.e, $Q \ll P$), $\mathrm{D_{KL}}[Q||P]$ can be expressed as

$$\mathrm{D_{KL}}[Q||P] = \int_{\Omega} \log \left[\frac{dQ}{dP}\right] dQ, \tag{5.1.3}$$

where $\frac{dQ}{dP}$ denotes the Radon-Nikodym derivative of Q w.r.t. P.

In practice, the definition given by Equation (5.1.3) is not very convenient to work with. Fortunately, as shown by [330], $\mathrm{D_{KL}}[q(f)||p(f)]$ can be expressed in the form of finite dimensional densities:

$$\mathrm{D_{KL}}[q(f)||p(f)] = \sup_{n,\mathbf{X}_n} \mathrm{D_{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})], \tag{5.1.4}$$

where $\mathbf{X}_n$ denote a set of n measure points $\{\mathbf{x}_k\}_{1\le k\le n}$ in the domain/index set of $f(\cdot)$, which can be treated as an element of the product space $\mathcal{T}^n$; and $\mathbf{f}^{\mathbf{X}_n}$ denotes the vector of function values evaluated on $\mathbf{X}_n$. Since $\mathbf{f}^{\mathbf{X}_n}$ is a finite dimensional vector, the densities functions $q(\mathbf{f}^{\mathbf{X}_n})$ and $p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})$ exist under mild conditions, and $\mathrm{D_{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})]$ can be computed using those densities. Here, we have slightly abused the notations, and use $p(\cdot)$ to denote both the *probability measure* $p(f)$ itself, and the corresponding density functions $q(\mathbf{f}^{\mathbf{X}_n})$ over its finite dimensional function values $\mathbf{f}^{\mathbf{X}_n}$. In other words, the KL-divergence between stochastic processes is the supreme of the relative entropies obtained on all possible measure points in $\mathcal{T}^{\mathbb{Z}^+}$ (Figure 5.1).

Functional ELBO Similar to the parameter space VI, minimizing the functional KL-divergence (5.1.4) is equivalent to maximizing the evidence lower bound (ELBO) in *function space*:

$$\mathcal{L}_q^{functional} := \mathbb{E}_{q(f)}[\log p_{\pi}(\mathcal{D}|f)] - \mathrm{D_{KL}}[q(f)||p(f)] \tag{5.1.5}$$

where $\mathrm{D_{KL}}[q(f)||p(f)]$ is the functional KL divergence between $q(f)$ and $p(f)$. We call $\mathcal{L}_q^{functional}$ the *functional ELBO*. In the context of machine learning, the above formulation of functional ELBO maximization is first used in the concurrent work of functional Bayesian neural networks (f-BNNs) [330]. To a certain extend, f-BNNs is the reverse of our VIP method: instead of using GPs as posterior approximations, they use GPs as priors in function space, and performs VI using BNNs as approximate posterior. That is, f-BNNs optimizes

$$\mathcal{L}_q^{functional} := \mathbb{E}_{q_{\mathrm{BNN}}(f)}[\log p_{\pi}(\mathcal{D}|f)] - \mathrm{D_{KL}}[q_{\mathrm{BNN}}(f)||p_{GP}(f)] \tag{5.1.6}$$

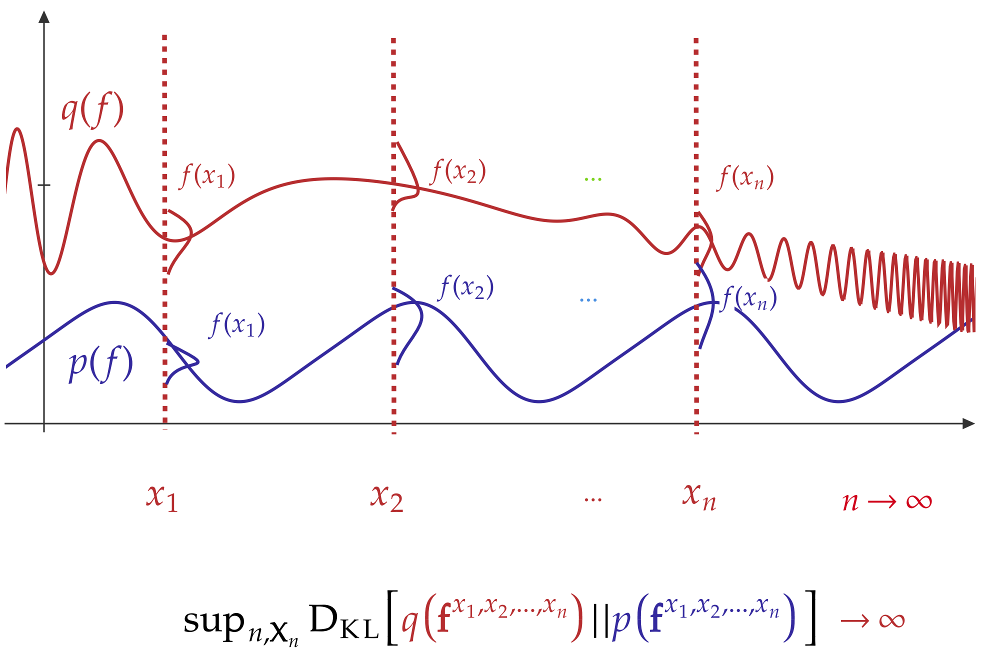

$q(f)$
$f(x_1)$
$f(x_2)$
...
$f(x_n)$
$p(f)$
$f(x_1)$
$f(x_2)$
...
$f(x_n)$
x_1
x_2
...
x_n
$n \to \infty$
$\sup_{n,\mathbf{X}_n} \mathrm{D_{KL}}\left[q\left(\mathbf{f}^{x_1,x_2,\dots,x_n}\right)||p\left(\mathbf{f}^{x_1,x_2,\dots,x_n}\right)\right] \to \infty$

Figure 5.1 Illustration of functional KL divergence. The KL-divergence between stochastic processes is the supremum of the relative entropies obtained on all possible measure points locations ($\mathbf{X}_n$) and sizes (n). As notice by [39], such supremum value may be infinite.

where $p_{GP}(f)$ is a GP and $q_{\mathrm{BNN}}(f)$ is a BNN. The gradients of $\mathrm{D_{KL}}[q_{\mathrm{BNN}}(f)||p_{GP}(f)]$ is estimated via Stein gradient estimator [181].

Remark (Gradient estimation of functional BNNs). F-BNNs proposes to estimate the gradient of $\mathrm{D_{KL}}[q_{\mathrm{BNN}}(f)||p_{GP}(f)]$ via spectral Stein gradient estimator (SSGEs) [309]. First, assuming that $q(f)$ can be reparameterized as $f(\cdot) = g(\cdot, \mathbf{z})$, $\mathbf{z} \sim p(\mathbf{z})$. Then, given a fixed measure points $\mathbf{X}_n$, the gradient of $\mathrm{D_{KL}}[q_{\boldsymbol{\lambda}}(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]$ w.r.t. $\boldsymbol{\lambda}$ can be rewritten as:

$$\nabla_{\boldsymbol{\lambda}} \mathrm{D_{KL}}[q_{\boldsymbol{\lambda}}(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})] = \mathbb{E}_{\mathbf{z}}\left\{\nabla_{\boldsymbol{\lambda}} \mathbf{f}^{\mathbf{X}_n}[\nabla_{\mathbf{f}^{\mathbf{X}_n}} \log q_{\boldsymbol{\lambda}}(\mathbf{f}^{\mathbf{X}_n}) - \nabla_{\mathbf{f}^{\mathbf{X}_n}} \log p(\mathbf{f}^{\mathbf{X}_n})]\right\}. \tag{5.1.7}$$

For implicit processes, both $\nabla_{\mathbf{f}^{\mathbf{X}_n}} \log q_{\boldsymbol{\lambda}}(\mathbf{f}^{\mathbf{X}_n})$ and $\nabla_{\mathbf{f}^{\mathbf{X}_n}} \log p(\mathbf{f}^{\mathbf{X}_n})]$ can be intractable. To solve this problem, Spectral stein gradient estimators SSGE is able to estimate $\nabla_{\mathbf{f}^{\mathbf{X}_n}} \log q_{\boldsymbol{\lambda}}(\mathbf{f}^{\mathbf{X}_n})$ (or $\nabla_{\mathbf{f}^{\mathbf{X}_n}} \log p(\mathbf{f}^{\mathbf{X}_n})]$) using only samples from $q_{\boldsymbol{\lambda}}(\mathbf{f}^{\mathbf{X}_n}$ (or $p(\mathbf{f}^{\mathbf{X}_n})$). Given a kernel function $\mathfrak{K}(\mathbf{f}^{\mathbf{X}_n}, \mathbf{f}'^{\mathbf{X}_n})$ satisfying

$$\int_{\mathbf{f}^{\mathbf{X}_n} \in \mathbb{R}^n} \nabla_{\mathbf{f}^{\mathbf{X}_n}}[\mathfrak{K}(\mathbf{f}^{\mathbf{X}_n}, \mathbf{f}'^{\mathbf{X}_n}) p(\mathbf{f}^{\mathbf{X}_n})] d\mathbf{f}^{\mathbf{X}_n}, \quad \forall \mathbf{f}'^{\mathbf{X}_n} \in \mathbb{R}^n, \tag{5.1.8}$$

[309] shows that

$$\nabla_{\mathbf{f}^{\mathbf{X}_n}} \log p(\mathbf{f}^{\mathbf{X}_n})] = -\sum_{j=1}^{\infty} [\mathbb{E}_{\mathbf{f}^{\mathbf{X}_n} \sim q(\mathbf{f}^{\mathbf{X}_n})} \nabla_{\mathbf{f}^{\mathbf{X}_n}} \psi_j(\mathbf{f}^{\mathbf{X}_n})] \psi_j(\mathbf{f}^{\mathbf{X}_n}), \tag{5.1.9}$$

where $\{\psi_j(\cdot)\}_{1 \leq j}$ are the eigenfunctions of $\mathfrak{K}$, estimated by Nyström method [247]. In practice, Equation 5.1.9 is estimated by finite truncation of the infinite sumation, and MC estimation of the expectation operator $\mathbb{E}_{\mathbf{f}^{\mathbf{X}_n} \sim q(\mathbf{f}^{\mathbf{X}_n})}[\cdot]$.

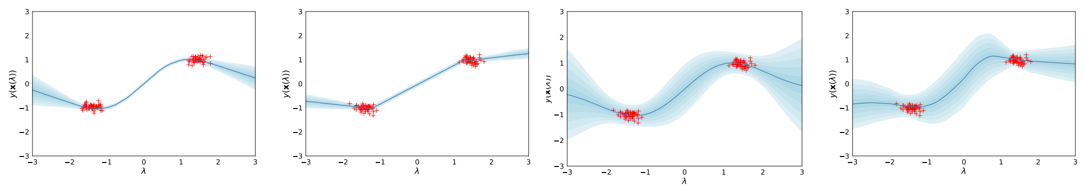
While f-BNNs is one of the first methods that highlight the importance of function space inference, it suffers from the limitations of the SSGE approach to functional ELBOs.

- As introduced before, f-BNNs optimizes the functional KL divergence between BNNs and GPs. Unfortunately, functional KL divergence not always well-defined . As shown by [39], the functional KL divergence between a BNN and a GP is infinite. Also, the functional KL divergence between two parameteric models could also be infinite. Therefore, if the original objective function is ill-defined, it does not make sense to apply SSGEs for gradient estimations in the first place.

- Second, even we assume that the functional KL divergence is finite, it still brings up another computational issue. In Equation (5.1.4), computing the functional KL divergence requires to find the maximum over *all possible* measure point locations and sizes. This process is unfortunately very difficult, and has found to be prone to overfitting [330].

- Another issue is, it has been shown that (spectral) stein gradient estimators are less efficient for high dimensional distributions [379, 85]. When estimating $\nabla_{\mathbf{f}^{\mathbf{X}_n}} \log q_{\boldsymbol{\lambda}}(\mathbf{f}^{\mathbf{X}_n})$, the dimensionality of $\mathbf{f}^{\mathbf{X}_n}$ could be very large (depending on the optimal n and $\mathbf{X}_n$ that maximizes Equation 5.1.4). Technically, the size of optimal $\mathbf{X}_n$ should be larger than the size of $\mathcal{D}$, which makes SSGE not scalable to large data setting (as the computational cost of SSGE scales linearly to the dimensionality of $\mathbf{f}^{\mathbf{X}_n}$).

- Finally, the q distributions of f-BNNs rely on a mean-field paramaterization of BNNs, which often lacks predictive in-between uncertainty on test data (see Figure 5.2). This issue was observed both with single layer BNNs [70], and deeper BNNs (able to represent in-between uncertainty, but more over-confident than HMC empirically for certain settings [71, 68]).



(a) functional BNN (b) Mean field VI BNN (c) HMC BNN (d) Ours (FVI on BNN)

Figure 5.2 A regression task on a synthetic dataset (red crosses) from [70]. We plot predictive mean and uncertainties for each algorithms. This tasks is used to demonstrate the theoretical pathologies of weight-space VI for single-layer BNNs: there is *no* setting of the variational parameters that can model the in-between uncertainty between two data clusters. The functional BNNs [330] also have this problem, since mean-field BNNs are used as part of the model. On the contrary, our FVI method can produce sensible uncertainty estimates. See Appendix 5.C.2 for more details.

Remark (Ad-hoc solutions to f-BNN limitations). In practice, the supremum operator $\sup_{n,\mathbf{X}_n}[\cdot]$ is replaced by $\mathbb{E}_{\mathbf{X}_n \sim \mathcal{U}(\mathbb{R}^n)}[\cdot]$, i.e. the expectation over n uniformly distributed random measure points, with n being fixed. This partially addresses some of the above listed limitations. However, under such approximation, the resulting objective function is not a valid functional divergence anymore. More importantly, for each randomly sampled measure points $\mathbf{X}_n \sim \mathcal{U}(\mathbb{R}^n)$, we need to re-run Nyström procedures in SSGE to estimate the corresponding $\nabla_{\mathbf{f}^{\mathbf{X}_n}} \log q_{\boldsymbol{\lambda}}(\mathbf{f}^{\mathbf{X}_n})$ individually. This creates a loop of SSGE estimators, which is computationally intensive.

Therefore, those limitations motivates us to propose new objective functions for function-space VI, as well as more scalable gradient estimation methods, which will be introduced in the next section.

5.2 The grid-functional KL divergence: a new objective for functional VI

As pointed out by [39], the functional KL divergence defined in Eq 5.1.4 is not always well-defined. For example, the functional KL divergence between two BNNs with different network architectures can be infinite. To address this issue, we propose to optimize a new divergence measure, called the *grid-functional KL divergence*. Figure 5.3 illustrates the idea of grid functional KL divergence, which we now proceed to describe. Instead of taking the supremum as in the original functional KL divergence, we take an expectation over *both* n and $\mathbf{X}_n$. We specify a probability distribution $\{p_n\}_{1\leq n<\infty}$ over $n \leq 1$ assigning low probability values p_n to larger n. This way, we can trim down the contributions from the KL terms $\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})]$ that correspond to large n, and hope fully arrive at a finite expectation value.

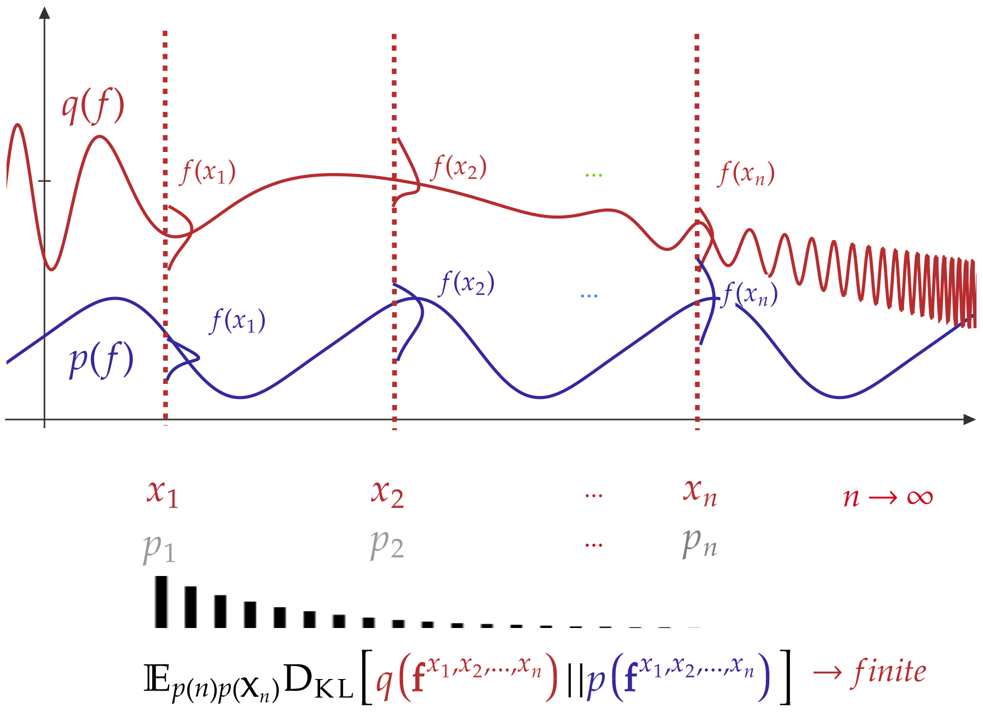

$q(f)$
$f(x_1)$
$f(x_2)$
...
$f(x_n)$
$p(f)$
$f(x_1)$
$f(x_2)$
...
$f(x_n)$
x_1
x_2
...
x_n
$n \rightarrow \infty$
p_1
p_2
...
p_n
$\mathbb{E}_{p(n)p(\mathbf{X}_n)}\mathrm{D}_{\mathrm{KL}}\left[q\left(\mathbf{f}^{x_1,x_2,\ldots,x_n}\right)||p\left(\mathbf{f}^{x_1,x_2,\ldots,x_n}\right)\right] \rightarrow finite$

Figure 5.3 Illustration of grid functional KL divergence.

Following this idea, we give the formal definition of grid-functional KL divergence as:

Definition 5.1 (Grid-functional KL divergence)**.** *The grid-functional KL divergence between two stochastic processes $q(f)$ and $p(f|\mathcal{D})$ is given by*

$$\mathrm{D}_{grid}[q(f)||p(f|\mathcal{D})] := \mathbb{E}_{n,\mathbf{X}_n\sim c}\mathrm{D}_{KL}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})], \tag{5.2.1}$$

where $\mathbf{X}_n$ is a set of n measurement points $\{\mathbf{x}_k\}_{1\leq k\leq n}$ sampled from $\mathcal{T}$, according to some sampling distribution c.

Here, c is defined on the product space $\mathcal{T}^{\mathbb{Z}^+}$, and the number of sampled measure points n is also random. One may recognize that [330] proposed a similar objective as an approximation to 5.1.4, in which the number of measure points n is a fixed constant instead of a random variable. Note that $\mathrm{D}_{\mathrm{grid}}[q(f)||p(f|\mathcal{D})]$ is *not* an approximation to

$\mathrm{D}_{\mathrm{KL}}[q(f)||p(f|\mathcal{D})]$: it is a valid functional divergence at its own, and we propose to use it as a first principle for function space VI. Indeed, we can prove that $\mathrm{D}_{\mathrm{grid}}[q(f)||p(f|\mathcal{D})]$ is a valid divergence, under mild conditions:

Proposition 5.1. *Suppose c has full support on $\mathcal{T}^{\mathbb{Z}^+}$. Then, $\mathrm{D}_{grid}[q(f)||p(f|\mathcal{D})]$ satisfies the following conditions: i), $\mathrm{D}_{grid}[q(f)||p(f|\mathcal{D})] \geq 0$; ii), $\mathrm{D}_{grid}[q(f)||p(f|\mathcal{D})] = 0$ if and only if $q(f) = p(f|\mathcal{D})$.*

In other words, if c has full support on $\mathcal{T}^{\mathbb{Z}^+}$ (we will give an example of such c later), then $\mathrm{D}_{\mathrm{grid}}[q(f)||p(f|\mathcal{D})]$ is a valid divergence in function space. Therefore, we can use $\mathrm{D}_{\mathrm{grid}}$ as an alternative objective for function space inference. Now, the problem becomes: is $\mathrm{D}_{\mathrm{grid}}[q(f)||p(f|\mathcal{D})]$ more well behaved than $\mathrm{D}_{\mathrm{KL}}[q(f)||p(f|\mathcal{D})]$? Indeed it is. In fact, we can show that for certain scenarios, $\mathrm{D}_{\mathrm{grid}}$ can avoid some of the issues the original functional KL divergence has (see Appendix 5.A.2 for details):

Proposition 5.2. *Let $p(f)$ and $q(f)$ be two distributions for random functions. Assume that $p(f)$ is parameterized by the following sampling processes:*

$$f = h + \varepsilon, h(\mathbf{x}) \sim p(h|\mathbf{x};\Theta), \Theta \sim p(\Theta), \varepsilon \sim \mathcal{N}(0, \sigma^2)$$

, And $q(f)$ is parameterized by:

$$f = h + \eta, h(\mathbf{x}) \sim q(h|\mathbf{x};\Gamma), \Gamma \sim q(\Gamma), \eta \sim \mathcal{N}(0, \sigma^2)$$

. Here, $\mathbf{x} \in \mathcal{T} \subset \mathbb{R}^d$, h is the random latent function, $\Theta \in \mathbb{R}^I$, $\Gamma \in \mathbb{R}^J$ are the parameters of each random function distributions, respectively. Suppose that: 1, $q(\mathbf{h}^{\mathbf{x}})$ has a compact support, denoted by $\mathcal{B}$; 2, $q(\Gamma)$ has a compact support, denote by $\mathcal{V}$; and 3, $p(h|\mathbf{x};\Theta) > 0$ for $\forall h \in \mathcal{B}, \mathbf{x} \in \mathcal{T}$, and $\Theta \in \mathbb{R}^I$. Then, there exist a sampling distribution c such that: 1, c has full support on $\mathcal{T}^{\mathbb{Z}^+}$, and 2, $\mathrm{D}_{grid}[q(f)||p(f)]$ is finite.

This result shows that the grid-functional KL divergence allows us to perform VI between p and q even if they have *different* parametric forms. Moreover, in Appendix 5.A.3 Corollary 5.1, we have shown that if one of the distributions is replaced by a Gaussian process (of certain kernel function), then under some additional assumptions, the grid-functional KL is still finite. In those cases, the original functional KL is no longer finite. This validates our choice of grid-functional KL divergence. To use $\mathrm{D}_{\mathrm{grid}}[q(f)||p(f|\mathcal{D})]$ for VI, we further derive a new ELBO based on $\mathrm{D}_{\mathrm{grid}}$ (Appendix 5.A.1):

Proposition 5.3. *Let $n, \mathbf{X}_n \sim c$ be a set of random measure points such that $\mathbf{X}_{\mathcal{D}} \subset \mathbf{X}_n$. Define:*

$$\mathcal{L}_q^{grid} := \log p(\mathcal{D}) - \mathrm{D}_{grid}[q(f)||p(f|\mathcal{D})]. \tag{5.2.2}$$

Then we have:

$$\mathcal{L}_q^{grid} = \mathbb{E}_{q(f)}[\log p(\mathcal{D}|f)] - \mathrm{D}_{grid}[q(f)||p(f)] \tag{5.2.3}$$

and $\log p(\mathcal{D}) \geq \mathcal{L}_q^{grid} \geq \mathcal{L}_q^{functional}$.

Proposition 5.3 shows that $\mathcal{L}_q^{grid}$ is a valid variational objective function: it is a lower bound for $\log p(\mathcal{D})$, and also upper-bounds $\mathcal{L}_q^{functional}$. For the rest of the chapter, we will discuss how to perform functional VI based on $\mathcal{L}_q^{grid}$. We will focus on how to propose a expressive variational family $q(f)$, and how to efficiently estimate $D_{grid}[q(f)||p(f)]$.

Remark (Choice of c). One example of c that satisfies the requirement of Propositions 5.1, 5.2, and 5.3 takes the following form (which will be used throughout the chapter):

$$(n-|\mathcal{D}|) \sim \mathrm{Geom}(p), \mathbf{x}_k \sim \mathcal{U}(\mathcal{T}), \ \forall 1 \leq k \leq n-|\mathcal{D}|, \mathbf{X}_n := \mathbf{X}_{\mathcal{D}} \bigcup \{\mathbf{x}_k\}_{1\leq k \leq n-|\mathcal{D}|}, \tag{5.2.4}$$

where we first sample n from a geometric distribution, such that $(n-|\mathcal{D}|) \sim \mathrm{Geom}(p)$ with parameter p (see Appendix 5.A.2 for more discussion). Then, $(n-|\mathcal{D}|)$ out of distribution (OOD) measure points are sampled independently from a uniform distribution on $\mathcal{T}$.

5.3 Choosing $q(f)$: stochastic process generators (SPGs)

In order to obtain good performance in FVI, it is crucial to pick an expressive variational family for $q(f)$. Here, we propose a new class of variational distributions called stochastic process generators (SPGs), which is motivated by the *Karhunen-Loeve expansion theorem* [194] for stochastic processes. According to this theorem, the true posterior $f \sim p(f|\mathcal{D})$[2] can be expanded as the *stochastic* infinite series (Figure 5.4a),

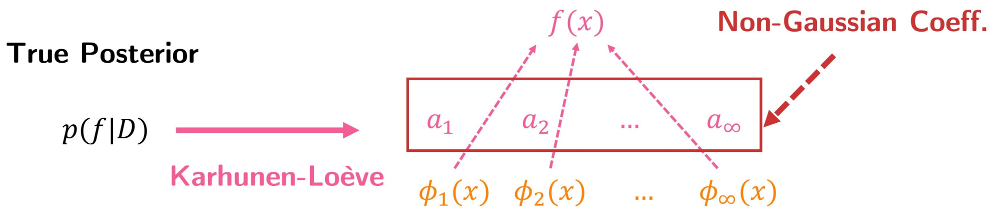
$$f = \sum_{i}^{\infty} a_i \phi_i(\mathbf{x}) \sim p(f|\mathcal{D}), \tag{5.3.1}$$

where $\{a_i\}_{1\leq i}$ are set of zero-mean random variables (which can be non-Gaussian), $\{\phi_i\}_{i=1}^{\infty}$ forms a set of orthonormal basis functions of $\mathcal{L}^2(\mathbb{R}^d)$. Furthermore, we can show that $\{\phi_i\}_{i=1}^{\infty}$

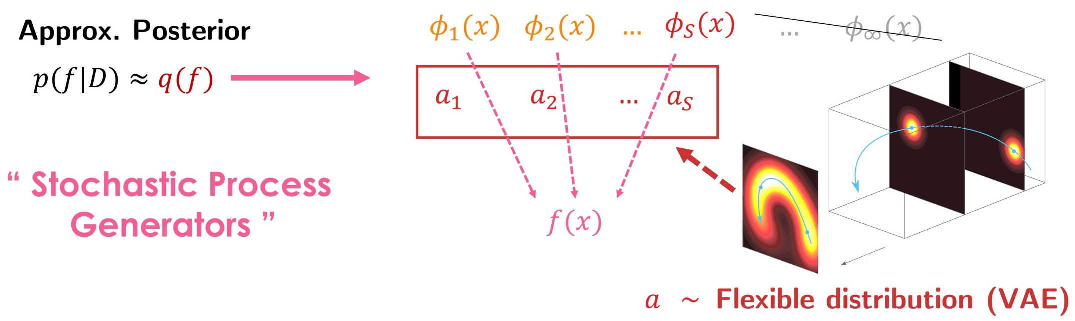
[2]Assuming it is ($\mathcal{L}^2(\mathbb{R}^d)$-).

are exactly the eigen functions of the operator $O_{\mathfrak{C}}(f)(\mathbf{x}) = \int \mathfrak{C}(\mathbf{x}, \mathbf{x}')f(\mathbf{x}')d\mathbf{x}'$, where $\mathfrak{C}(\cdot,\cdot)$ is the covariance function of $p(f|\mathcal{D})$.

sub


True Posterior
$p(f|D)$
Karhunen-Loève
$f(x)$
Non-Gaussian Coeff.
a_1 a_2 ... a_∞
$\phi_1(x)$ $\phi_2(x)$ ... $\phi_\infty(x)$

(a) Illustration of the Karhunen-Loeve expansion theorem.


Approx. Posterior
$p(f|D) \approx q(f)$
" Stochastic Process Generators "
$\phi_1(x)$ $\phi_2(x)$ ... $\phi_S(x)$ ... $\phi_\infty(x)$
a_1 a_2 ... a_S
$f(x)$
a ~ Flexible distribution (VAE)

(b) Illustration of the stochastic process generators.

Figure 5.4 Illustration of stochastic process generators. According to the Karhunen-Loeve expansion theorem, the true posterior $f \sim p(f|\mathcal{D})$ can be expanded as the stochastic infinite series. This motivates us to uses finite truncation approximations, and parameterizes the non-Gaussian coefficients with powerful implicit models.

Motivated by Karhunen-Loeve expansion theorem, we propose to parameterize our approximate posterior distribution via the following S-term finite truncation approximation:

$$f = \sum_s a_s \phi_s(\cdot, \mathbf{w}_s) + \nu, \quad \nu \sim \mathcal{GP}(\nu; 0, \delta(\cdot,\cdot)\sigma_\nu^2), \tag{5.3.2}$$

where $\{\phi_s(\cdot, \mathbf{w}_s)\}_{1\le s\le S}$ are a set of real-valued functions parameterized by some parameters, $\{\mathbf{w}_s\}_{1\le s\le S}$. ν is a white noise process that models additive aleatoric uncertainty (as well as the approximation error). For the practical parameterization of $\phi_s(\cdot, \mathbf{w}_s)$, since the covariance function of $p(f|\mathcal{D})$ is unknown, we may assume that each $\phi_s : \mathbb{R}^d \mapsto \mathbb{R}^1$ is defined by an individual flexible deep neural network with weights, $\mathbf{w}_s$. In order to compensate for the finite truncation approximation, we assume that these weights $\mathbf{w}_s$ are learnable via optimization (which will be introduced later). The only missing part is the parameterization of the non-Gaussian variables, $\mathbf{a} = (a_1, ..., a_S)$. This is done by specifying a implicit distribution

(Section 2.2.4) on $\mathbf{a}$:

$$q(\mathbf{a}) = \int_{\mathbf{h}} p_\theta(\mathbf{a}|\mathbf{h}) q_{\boldsymbol{\eta}}(\mathbf{h}), \tag{5.3.3}$$

where $\mathbf{h}$ is the latent variable of $q(\mathbf{a})$, the conditional distribution $p_\theta(\mathbf{a}|\mathbf{h})$ is parameterized by θ, and $q_{\boldsymbol{\eta}}(\mathbf{h})$ is some distribution over the latent space. One can immediately recognize that $q(\mathbf{a})$ is nothing more than the variational auto-encoder (VAE) [150], by noticing that $p_\theta(\mathbf{a}|\mathbf{h})$ is just the decoder, and $q(\mathbf{a})$ is the prior on the latent space. We can finally give the definition of our SPG variational family $q_{\text{SPG}}(f|q_{\boldsymbol{\eta}}(\mathbf{h}))$ (Figure 5.4b):

$$f = \sum_s a_s \phi_s(\cdot, \mathbf{w}_s) + \nu, \quad \mathbf{a} \sim \int_{\mathbf{h}} p_\theta(\mathbf{a}|\mathbf{h}) q_{\boldsymbol{\eta}}(\mathbf{h}), \quad \nu \sim \mathcal{GP}(\nu; 0, \delta(\cdot,\cdot)\sigma_\nu^2). \tag{5.3.4}$$

Intuitively, q_{SPG} serves as a generator for stochastic processes. $q_{\text{SPG}}(f|\cdot)$ maps any given $q_{\boldsymbol{\eta}}(\mathbf{h})$ to a stochastic process $q_{\text{SPG}}(f|q_{\boldsymbol{\eta}}(\mathbf{h}))$, hence the name. Regarding the expressiveness of SPGs, we have the following result:

Proposition 5.4 (Expressiveness of SPGs)**.** *Let $p(f)$ be a square-integrable stochastic process on a probability space $(\mathcal{X}, \mathcal{B})$, and its index set $\mathcal{T}$ is a compact subset of $\mathbb{R}^d$. Suppose $\mathcal{X}$ is a compact metric space, $\mathcal{B}$ is the Borel set on $\mathcal{X}$. Then, for $\forall \varepsilon > 0$, there exists a SPG $q^{\varepsilon}_{SPG}(f)$, such that:*

$$\text{MMD}(p, q^{\varepsilon}_{SPG}; \mathcal{F}) < \varepsilon \quad \textit{for} \quad \forall \mathbf{x} \in \mathcal{T}, \tag{5.3.5}$$

where MMD *is the maximum mean discrepancy, $\mathcal{F}$ is the MMD function class defined to be a unit ball in a RKHS with a universal kernel [324] $k(\cdot,\cdot)$ as its reproducing kernel.*

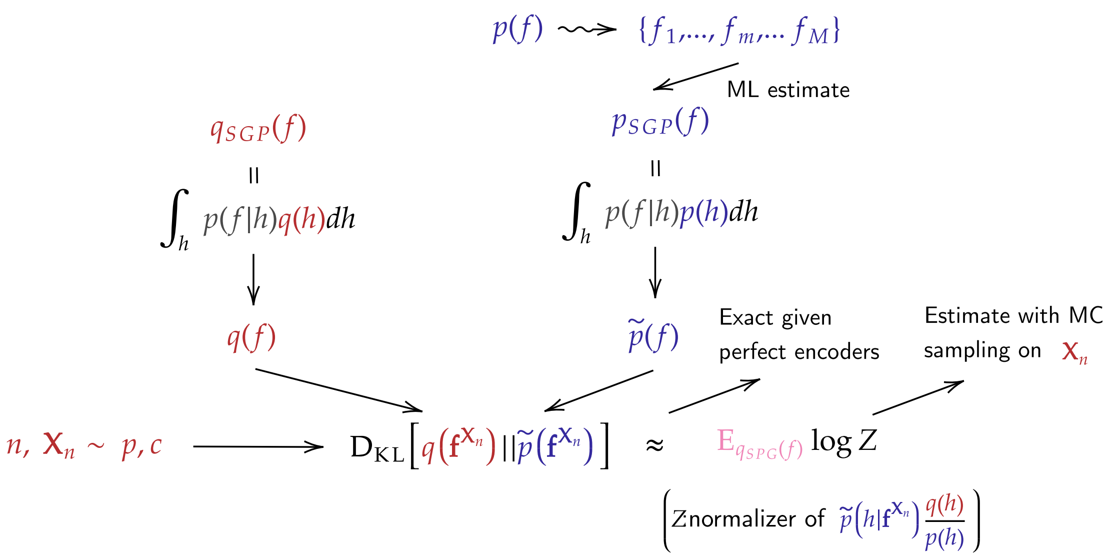
Next we will discuss how $\{\mathbf{w}_s\}$, θ and $q_{\boldsymbol{\eta}}(\mathbf{h})$ can be estimated, and how can we use SPG to estimate the function space KL-divergence in Equation 5.1.4.

Remark (Relations to VIP). SPGs can be seen as the non-Gaussian extension of the variational approximation used in VIP. Recall that in VIP, the variational family $q_{\text{VIP}}(f)$ is defined by the following sampling process:

$$f = \sum_s a_s \phi_s(\cdot), \quad \mathbf{a} \sim \mathcal{N}(\mathbf{a}; \boldsymbol{\mu}, \boldsymbol{\Sigma}), \tag{5.3.6}$$

where $\{\phi_s\}_{s=1}^S$ are random paths sampled from the prior process $p(f)$. In SPG, we essentially removed the Gaussian assumption on $\mathbf{a}$, by specifying the non-Gaussian implicit model, $q(\mathbf{a})$. Furthermore, SPG removed the constraint that $\{\phi_s\}_{s=1}^S$ need to be sampled from $p(f)$.

5.4 Efficient estimation of grid-functional KL divergence



$p(f) \rightsquigarrow \{f_1,\dots,f_m,\dots f_M\}$
ML estimate
$q_{SGP}(f)$
$p_{SGP}(f)$
$\int_h p(f|h)q(h)dh$
$\int_h p(f|h)p(h)dh$
$q(f)$
$\tilde{p}(f)$
Exact given perfect encoders
Estimate with MC sampling on X_n
$n, X_n \sim p, c$
$D_{KL}\left[q(f^{X_n})||\tilde{p}(f^{X_n})\right] \approx E_{q_{SPG}(f)} \log Z$
Znormalizer of $\tilde{p}(h|f^{X_n})\frac{q(h)}{p(h)}$

Figure 5.5 Illustration of grid functional KL divergence estimation procedure.

In order to estimate the grid-functional KL-divergence in Equation 5.2.1, we propose to use a two-step method for a scalable approximation (depicted in Figure 5.5). In the first step, we distill $p(f)$ by fitting a stochastic process generator $\tilde{p}_{\text{SPG}}(f)$ to $p(f)$. In the second step, we calculate the KL-divergence between $q_{\text{SPG}}(f)$ and $\tilde{p}_{\text{SPG}}(f)$ as the surrogate for the KL-divergence between $q_{\text{SPG}}(f)$ and $p(f)$.

Distilling $p(f)$ via a stochastic process generator. Assume that we can draw M random functions $[f_1(\cdot), f_2(\cdot), \dots, f_m(\cdot), \dots, f_M(\cdot)]$ from the prior process, $p(f)$. Let $\tilde{p}_{\text{SPG}}(f)$ be a SPG given by

$$f = \sum_s a_s \phi_s(\cdot, \mathbf{w}_s) + \nu, \quad \mathbf{a} \sim \int_{\mathbf{h}} p_\theta(\mathbf{a}|\mathbf{h}) p_0(\mathbf{h}), \quad \nu \sim \mathcal{GP}(\nu; 0, \delta(\cdot,\cdot)\sigma_\nu^2), \tag{5.4.1}$$

where $p_0(\mathbf{h})$ is a fixed standard normal distribution. We denote the above process by $\tilde{p}_{\text{SPG}}(f|p_0(\mathbf{h}))$. We can train $\tilde{p}_{\text{SPG}}(f)$ on $f_1, f_2, \dots, f_m, \dots, f_M$, by optimizing the aggregated

ELBO on $\tilde{p}_{\text{SPG}}(f)$:

$$\max_{\{\mathbf{w}_s\},\theta,\boldsymbol{\lambda}} \mathbb{E}_{\mathbf{X}_O} \sum_m \log \tilde{p}_{\text{SPG}}(\mathbf{f}_m^{\mathbf{X}_O}) \geq \max_{\{\mathbf{w}_s\},\theta,\boldsymbol{\lambda}} \mathbb{E}_{\mathbf{X}_O} \mathbb{E}_{\tilde{q}_{\boldsymbol{\lambda}}(\mathbf{h}|\mathbf{f}_m^{\mathbf{X}_O})} \log \frac{\tilde{p}_{\text{SPG}}(\mathbf{f}_m^{\mathbf{X}_O}|\mathbf{h}) p_0(\mathbf{h})}{\tilde{q}_{\boldsymbol{\lambda}}(\mathbf{h}|\mathbf{f}_m^{\mathbf{X}_O})}, \tag{5.4.2}$$

where $\mathbf{f}_m^{\mathbf{X}}$ are the function values of f_m evaluated on $\mathbf{X}_O$, $\mathbf{X}_O$ are $|O| \leq |\mathcal{D}|$ measure points independently sampled from the training set $\mathbf{X}_{\mathcal{D}} = \{\mathbf{x}_i\}_{i=1}^N$. $\mathbf{f}_m^{\mathbf{X}}$ are the function values of f_m evaluated on $\mathbf{X}_O$, and $\tilde{q}_{\boldsymbol{\lambda}}(\mathbf{h}|\mathbf{f}_m^{\mathbf{X}_O})$ is an encoder network that approximates the true posterior $\tilde{p}_{\text{SPG}}(\mathbf{h}|\mathbf{f}_m^{\mathbf{X}_O})$.

Product of Experts (PoE) encoder. When sampling $\mathbf{X}_O$, since its size might vary each time, we would need to set up 2^N inference nets, one for each possible subsets. To overcome this issue, we adopt the Product of Experts encoder [369], a simple and flexible approach for such scenario, given by:

$$\tilde{q}_{\boldsymbol{\lambda}}(\mathbf{h}|\mathbf{f}_m^{\mathbf{X}_O}) \propto p_0(\mathbf{h}) \prod_{i=1}^{|O|} \tilde{q}_{\boldsymbol{\lambda}}(\mathbf{h}|f_m(\mathbf{x}_i), \mathbf{x}_i), \tag{5.4.3}$$

where $\tilde{q}_{\boldsymbol{\lambda}}(\mathbf{h}|f_m(\mathbf{x}_i), \mathbf{x}_i)$ is an inference network representing the expert associated with the i-th measurement point. Now the we can use a single encoder $\tilde{q}_{\boldsymbol{\lambda}}(\mathbf{h}|\mathbf{f}_m^{\mathbf{X}_O})$ to handle all the possible inputs $\mathbf{f}_m^{\mathbf{X}_O}$. In practice, we let $\tilde{q}_{\boldsymbol{\lambda}}(\mathbf{h}|f_m(\mathbf{x}_i), \mathbf{x}_i)$ to be a Gaussian expert that maps $[f_m(\mathbf{x}_i), \mathbf{x}_i]$ to a factorized Gaussian in latent space. Since the product of Gaussian experts is still Gaussian, $\tilde{q}_{\boldsymbol{\lambda}}(\mathbf{h}|\mathbf{f}_m^{\mathbf{X}_O})$ is a Gaussian distribution whose statistics can be computed analytically.

Estimating the grid-functional KL-divergence given $\mathbf{X}_n$. In order to estimate the grid-functional KL divergence between $q_{\text{SPG}}(f)$ and $p(f)$, we first discuss how this divergence can be estimated on *measurement points* $\mathbf{X}_n$, i.e., $\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]$ where $\mathbf{f}^{\mathbf{X}_n}$ is the vector of function values evaluated on $\mathbf{X}_n$. We then discuss how this can be used to estimate the grid-functional divergence in Equation 5.2.1. To begin with, as in Section 5.3, our variational family is given by

$$f = \sum_s a_s \phi_s(\cdot, \mathbf{w}_s) + \nu, \quad \mathbf{a} \sim q(\mathbf{a}) = \int_{\mathbf{h}} p_\theta(\mathbf{a}|\mathbf{h}) q_{\boldsymbol{\eta}}(\mathbf{h}), \quad \nu \sim \mathcal{GP}(\nu; 0, \sigma_\nu^2). \tag{5.4.4}$$

We denote the above variational family by $q_{\text{SPG}}(f|q_{\boldsymbol{\eta}}(\mathbf{h}))$. The key ingredient of our estimation method is that, we force $q_{\text{SPG}}(f|q_{\boldsymbol{\eta}}(\mathbf{h}))$ and $\tilde{p}_{\text{SPG}}(f|p_0(\mathbf{h}))$ to share the same basis functions (or weights $\{\mathbf{w}_s\}$) and decoder parameters θ. That is, once optimal $\{\mathbf{w}_s\}$ and θ are obtained by fitting $\tilde{p}_{\text{SPG}}(f)$ to $p(f)$, these are frozen and will be reused in $q_{\text{SPG}}(f)$.

This makes sense, since according to our definition in Section 5.1, $q_{\mathrm{SPG}}(f)$ and $\tilde{p}_{\mathrm{SPG}}(f)$ share the same measurable space, $(\mathbb{R}^{\mathcal{T}}, \mathcal{B}_{\mathbb{R}}^{\mathcal{T}})$.

Therefore, the only difference between $q_{\mathrm{SPG}}(f)$ and $\tilde{p}_{\mathrm{SPG}}(f)$ is the choice of the prior distributions on $\mathbf{h}$, which is $q_{\boldsymbol{\eta}}(\mathbf{h})$ and $p_0(\mathbf{h})$, respectively. Given this property, we can compute the KL divergence between $q_{\mathrm{SPG}}(f)$ and $\tilde{p}_{\mathrm{SPG}}(f)$ given *measurement points* $\mathbf{X}_n$ (Appendix 5.A.5):

Proposition 5.5 (KL divergence on measurement points between SPGs)**.** *Let $q_{SPG}(f)$ and $\tilde{p}_{SPG}(f)$ be the SPGs defined in Equation 5.4.1 and 5.4.4. Then we have*

$$\mathrm{D}_{KL}[q_{SPG}(\mathbf{f}^{\mathbf{X}_n})||\tilde{p}_{SPG}(\mathbf{f}^{\mathbf{X}_n})] = \mathbb{E}_{f\sim q_{SPG(f)}} \log \mathcal{Z}(\mathbf{f}^{\mathbf{X}_n}), \tag{5.4.5}$$

where $\mathcal{Z}(\mathbf{f}^{\mathbf{X}_n})$ is the partition function, $\mathcal{Z}(\mathbf{f}^{\mathbf{X}_n}) = \int_{\mathbf{h}} \tilde{p}_{SPG}(\mathbf{h}|\mathbf{f}^{\mathbf{X}_n}) \frac{q_{\boldsymbol{\eta}}(\mathbf{h})}{p_0(\mathbf{h})} d\mathbf{h}$.

Note that $\mathcal{Z}(\mathbf{f}^{\mathbf{X}_n})$ is intractable to compute due to the intractability of the posterior $\tilde{p}_{\mathrm{SPG}}(\mathbf{h}|\mathbf{f}^{\mathbf{X}_n})$. Fortunately, this is already approximated by the PoE inference net $\tilde{q}_{\boldsymbol{\lambda}}(\mathbf{h}|\mathbf{f}^{\mathbf{X}_n})$ given by Equation 5.4.3:

$$\mathcal{Z}(\mathbf{f}^{\mathbf{X}_n}) \approx \tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}_n}) := \int_{\mathbf{h}} \tilde{q}_{\boldsymbol{\lambda}}(\mathbf{h}|\mathbf{f}^{\mathbf{X}_n}) \frac{q_{\boldsymbol{\eta}}(\mathbf{h})}{p_0(\mathbf{h})} d\mathbf{h}. \tag{5.4.6}$$

Since $q_{\boldsymbol{\eta}}(\mathbf{h})$, $p_0(\mathbf{h})$, and $\tilde{q}_{\boldsymbol{\lambda}}(\mathbf{h}|\mathbf{f}^{\mathbf{X}_n})$ are all Gaussian distributions, $\tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}_n})$ can be computed using analytic solutions. Note also that thanks to the VAE-like structure in SPGs, all the calculations are performed in the latent space, whose dimensionality is much lower than $\mathbf{f}^{\mathbf{X}}$. With the additional help of analytic solutions for $\tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}})$, the estimation of (5.4.6) is very efficient and scalable.

5.5 Functional Variational Inference: the final algorithm

So far, we have introduce a new objective function, a new variational family, and the corresponding estimation method of the KL-divergence *on measurement points*, $\mathbf{X}_n$. Finally, we are able to derive a practical functional variational inference algorithm based on the grid-functional ELBO $\mathcal{L}_q^{grid}$ defined in Equation 5.2.3. Applying the approximation (5.4.6) to Equation 5.2.3, our final variational objective becomes

$$\begin{aligned}\log p(\mathcal{D}) &\geq \sum_i^{|\mathcal{D}|} \mathbb{E}_{q(f)}[\log p_\pi(y_i|f(\mathbf{x}_i))] - \mathbb{E}_{n,\mathbf{X}_n\sim c} \mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})] \\ &\approx \sum_i^{|\mathcal{D}|} \mathbb{E}_{q(f)}[\log p_\pi(y_i|f(\mathbf{x}_i))] - \mathbb{E}_{n,\mathbf{X}_n\sim c} \mathbb{E}_{f\sim q_{\mathrm{SPG}}(f)} \log \tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}_n}).\end{aligned} \tag{5.5.1}$$

To make Equation 5.5.1 scalable to large data, we can apply mini-batch sampling to the likelihood term $\sum_i^{|\mathcal{D}|} \mathbb{E}_{q(f)}[\log p(y_i|f(\mathbf{x}_i))]$. Then the only bottleneck of Equation 5.5.1 is that the input $\mathbf{X}_n$ to the inference net $\tilde{q}_{\boldsymbol{\lambda}}$ (used in $\tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}_n})$) can be very high dimensional due to the condition $\mathbf{X}_\mathcal{D} \subset \mathbf{X}_n$ required by Proposition 5.3. Fortunately, we can derive the following mini-batch estimators for $\mathbb{E}_{n,\mathbf{X}_n\sim c}\mathbb{E}_{f\sim q_{\mathrm{SPG}}(f)} \log \tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}_n})$ (Appendix 5.A.6 and 5.A.7):

Proposition 5.6. $\mathbb{E}_{n,\mathbf{X}_n\sim c}\mathbb{E}_{f\sim q_{SPG}(f)} \log \tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}_n})$ *can be estimated by the mini-batch estimator*

$$\begin{aligned}\mathcal{J}_K := \frac{1}{2}\sum_{i=1}^{H} \mathbb{E}_{f\sim q_{SPG}(f)} \Big[&\log \sigma_{\boldsymbol{\eta}_i}^{-2} + \log \hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2} - \log(\sigma_{\boldsymbol{\eta}_i}^{-2} + \hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2} - 1) - \hat{\mu}_{\boldsymbol{\lambda}_i}^2 \hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2} - \boldsymbol{\mu}_{\boldsymbol{\eta}_i}^2 \sigma_{\boldsymbol{\eta}_i}^{-2} \\ &+ (\hat{\sigma}_{\boldsymbol{\eta}_i}^{-2}\hat{\mu}_{\boldsymbol{\eta}_i} + \hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2}\hat{\mu}_{\boldsymbol{\lambda}_i})^2(\sigma_{\boldsymbol{\eta}_i}^{-2} + \hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2} - 1)^{-1}\Big],\end{aligned} \tag{5.5.2}$$

where H is the dimensionality of $\mathbf{h}$, $\mathcal{N}(\mathbf{h};\mu_{\boldsymbol{\eta}_i},\sigma_{\boldsymbol{\eta}_i}^2) = q_{\boldsymbol{\eta}}(h_i)$, $\mathcal{N}(\mathbf{h};\mu_{\boldsymbol{\lambda}_i},\sigma_{\boldsymbol{\lambda}_i}^2) = \tilde{q}_{\boldsymbol{\lambda}}(h_i|\mathbf{f}^{\mathbf{X}})$. $\hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2}$ *and* $\hat{\mu}_{\boldsymbol{\lambda}_i}$ *are the mini-batch approximators for* $\mu_{\boldsymbol{\lambda}_i}$ *and* $\sigma_{\boldsymbol{\lambda}_i}^2$*, respectively:*

$$\begin{aligned}\hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2} &:= \sum_{k\in\mathcal{K}} \frac{|\mathcal{D}|}{K} \sigma_{h_i|f^{\mathbf{x}_k}}^{-2} + \sum_{\mathbf{x}_l\in\mathbf{X}_n\setminus\mathbf{X}_\mathcal{D}} \sigma_{h_i|f^{\mathbf{x}_l}}^{-2} \\ \frac{\hat{\mu}_{\boldsymbol{\lambda}_i}}{\hat{\sigma}_{\boldsymbol{\lambda}_i}^2} &:= \sum_{k\in\mathcal{K}} \frac{|\mathcal{D}|}{K} \sigma_{h_i|f^{\mathbf{x}_k}}^{-2} \mu_{h_i|f^{\mathbf{x}_b}} + \sum_{\mathbf{x}_l\in\mathbf{X}_n\setminus\mathbf{X}_\mathcal{D}} \sigma_{h_i|f^{\mathbf{x}_l}}^{-2} \mu_{h_i|f^{\mathbf{x}_l}},\end{aligned}$$

where $\mathcal{K}$ is a mini-batch of size K sampled from $\{1,...,|\mathcal{D}|\}$, $\mathbf{x}_l \in \mathbf{X}_n \setminus \mathbf{X}_\mathcal{D}$ *is a set of OOD samples sampled from $\mathcal{T}$ using c in Eq. 5.2.4, and $\mu_{h_i|f^{\mathbf{x}_k}}$ and $\sigma_{h_i|f^{\mathbf{x}_k}}^2$ are the mean and variance parameter returned from $\tilde{q}_{\boldsymbol{\lambda}}(h_i|f(\mathbf{x}_k))$.*

The estimation in Eq. 5.5.2 is biased (but consistent). To remove the bias, we propose to debias Eq. 5.5.2 based on the Russian Roulette estimator [141]:

Proposition 5.7 (Russian Roulette estimator)**.** *Let R be a random integer from a distribution $\mathbb{P}(N)$ with support over the integers larger than K.* $\mathbf{x}_0$ *is a random location sampled from*

Algorithm 2 Functional Variational Inference (FVI)

```
Require: data $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$; prior $p(f)$; surrogate $\tilde{p}_{\text{SPG}}(f)$, variational process $q_{\text{SPG}}(f)$, likelihood function $p_\pi(y|f)$, mini-batch sizes $I$ and $K$
1: while not converged do
2:     Sample $[f_1(\cdot), f_2(\cdot), ..., f_M(\cdot)]$ from $p(f)$.
3:     Improve $\tilde{p}_{\text{SPG}}(f)$ by optimizing the aggregated ELBO in Equation 5.4.2 w.r.t. $\{\mathbf{w}_s\}, \theta, \boldsymbol{\lambda}$.
4: end while
5: while not converged do
6:     sample mini-batch $\mathcal{I}$ from $\{1, ..., |\mathcal{D}|\}$, and a set of measure points $\mathbf{X}_n$ via c.
7:     Optimize $\hat{\mathcal{L}}_{\text{FVI}}$ in Equation 5.5.3 w.r.t. $\boldsymbol{\eta}$ and $\pi$, via reparameterization tricks
8: end while
```

$\mathcal{T}$. Then $\mathbb{E}_{n,\mathbf{X}_n \sim c}\mathbb{E}_{f \sim q_{SPG}(f)} \log \tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}_n})$ can be estimated by $\mathbb{E}\left[\mathcal{J}_K + \sum_{k=K}^{R} \frac{\Delta_k}{\mathbb{P}(N \geq k)}\right]$, where $\Delta_k = \mathcal{J}_{\mathbf{k+1}} - \mathcal{J}_{\mathbf{k}}$, and the expectation is taken over R, n, $\mathbf{X}_n$, and all mini-batches used by each $\mathcal{J}_k$ terms.

This will enable us to also perform mini-batch sampling on the measurement points when performing FVI. Our final optimization objective function is

$$\hat{\mathcal{L}}_{\text{FVI}} := \frac{|\mathcal{D}|}{I} \sum_{i \in \mathcal{I}} \mathbb{E}_{q(f)}[\log p(y_i | f(\mathbf{x}_i))] - \mathcal{J}_K - \sum_{k=B}^{R} \frac{\Delta_k}{\mathbb{P}(N \geq k)}, \tag{5.5.3}$$

where $\mathcal{I}$ is a mini-batch of size I for the likelihood terms, R is an integer sampled from $\mathbb{P}(N)$, which is set to be $(R-K) \sim \text{Geom}(0.5)$. Finally, the full algorithm is sketched in Algorithm 2. We call this proposed method Functional Variational Inference (FVI).

Remark (Scalability). Our method is empirically much faster than f-BNN (Appendix 5.C.3). When estimating $\text{D}_{\text{grid}}[q(f)||p(f)]$, our method scales as $\mathcal{O}(M_q)$, where M_q is the number of samples sampled from $q_{\text{SPG}}(f)$, that are used in $\mathcal{J}_K$. In practice, we use $M_q = 1$. On the contrary, the SSGE estimator used in f-BNN scales as $\mathcal{O}(M_q^3 + M_q^2 |\mathbf{f}^{\mathbf{X}_n}|)$, where $|\mathbf{f}^{\mathbf{X}_n}|$ is the dimensionality of $\mathbf{f}^{\mathbf{X}_n}$. Usually in SSGEs, much larger value of M_q needs to used (e.g., $M_q = 100$). Also, as analyzed in Section 5.1, SSGE scales linearly to the dimensionality of $\mathbf{f}^{\mathbf{X}_n}$, and does not allow mini-batch estimation. Those factors will limit the applicability of SSGEs to large scale function space inference. In Appendix 5.C.3, we provide further results on the run-time performance of these methods.

5.6 Related works

Since BNNs and VIPs are discussed in Chapter 2 and 4 respectively, here we only address the rest of the related works, including f-BNNs, functional priors, and Bayesian non-parameterics.

Functional Bayesian neural networks (F-BNNs) and functional particle optimization (F-POVI). The functional BNN [330] is proposed to address the issue of specifying meaningful priors to BNN weights. It matches a BNN to a GP prior, by minimizing the functional KL divergence estimated by score function estimators [181]. As discussed in [39], this objective is not well-defined for a wide class of distributions. Also, the score function estimators used in f-BNNs often perform poorly in high dimensional spaces and are less efficient. In practice, we found that the f-BNN computational time is prohibitive. On the contrary, our FVI estimator avoids these issues by making use of the grid-functional divergence, which can be efficiently estimated using the latent representation of SPGs. More recently, the concurrent work of [285] proposes a tractable function space VI method for BNNs, in which the functional KL divergence is approximated via linearization. Similarly, f-POVI performs inference on BNN priors by performing particle optimization in function space. One limitation of f-POVI is that it requires the prior to be reparameterizable and differentiable while our method does not have this issue.

Function space priors. Another line of work directly defines distributions over functions by combining stochastic processes with neural networks. For example, neural processes (NPs) [80] and its variants [147, 90] focus on meta-learning scenarios and propose to use set encoders to model all possible posterior distributions of the form $\{p(\mathbf{f}|\mathcal{C})|\mathcal{C} \subset \mathcal{D}\}$, where $\mathcal{C}$ is the so-called "context points" in neural processes. This could be inefficient for large datasets, since it needs to feed all data points to the set encoder, which scales linearly w.r.t. the dataset size. More importantly, NPs still use for learning and inference an ELBO defined on parameter space instead of function space. On the contrary, our method focuses on the functional VI for supervised learning scenarios and does not need to model all possible conditionals. When computing predictive distribution , we only need to evaluate $q_\eta(\mathbf{h})$, which is a simple Gaussian distribution (no set encoders involved).

Bayesian non-parametrics. In the field of Bayesian non-parameterics, Gaussian Processes (GPs) [269] and deep GPs [49] are great examples of using function-space priors to produce

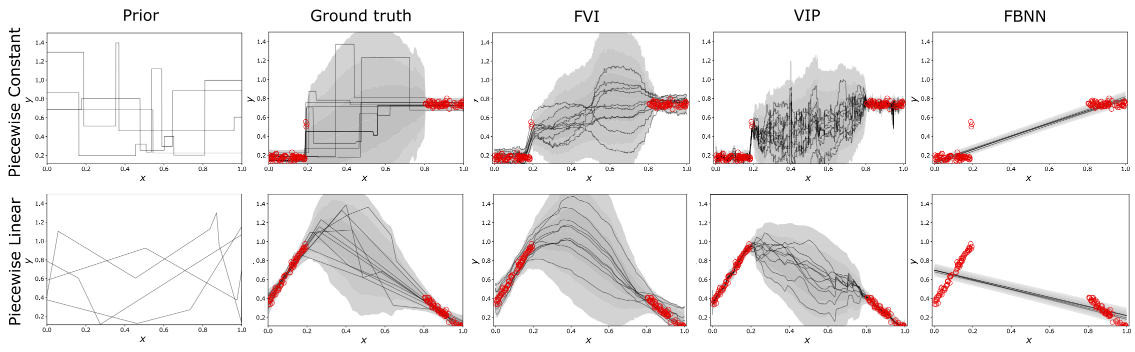


Figure 5.6 Implicit function prior and posterior samples from ground truth, FVI, VIP, and f-BNN, respectively. **The first row** corresponds to a piecewise constant prior, and **the second row** corresponds to a piecewise linear prior. **The leftmost column** shows 5 prior samples. **From the second column to the rightmost column** we show posterior samples generated by ground truth (returned by SIR), FVI, VIP and f-BNN, respectively. Red dots denote the training data. We plot 10 posterior samples in black lines and show predictive uncertainty as grey shaded areas.

models that are reliable under uncertainty. To reduce the prohibitive computation cost of exact/deep GPs, various VI methods [335, 109, 213, 292] have been studied. These methods share a similar principle with our work, that is, to minimize the functional divergence between the posterior and variational processes. Nevertheless, the GP components of the functional prior play a critical role in this line of work, which makes them less applicable to general non-GP based priors.

5.7 Experiments

In this section, we evaluate the performance of FVI using a number of tasks, including interpolation with structured implicit priors, multivariate regression with BNN priors, contextual bandits, and image classification. We mainly compare FVI with other weight-space and function-space Bayesian inference methods *using the same priors*. For more implementation details, please refer to Appendix 5.B. Additional experiments can be found in Appendix 5.C.

5.7.1 Interpolation with non-Gaussian priors: structured implicit priors

An advantage of FVI is that it can be applied to implicit (and non-Gaussian) priors over functions, where typical GPs do not apply. In this experiment, we evaluate the interpolation task as [330]. We consider two 1-D implicit priors on $[0,1]$: 1), piecewise constant random

Table 5.1 Regression experiment: Average test negative log likelihood

Dataset	N	D	FVI	VIP-BNN	VIP-NP	BBB	VDO	$\alpha = 0.5$	fBNN	exact GP
boston	506	13	**2.33±0.04**	2.45±0.04	2.45±0.03	2.76±0.04	2.63±0.10	2.45±0.02	2.30±0.10	2.46±0.04
concrete	1030	8	**2.88±0.06**	3.02±0.02	3.13±0.02	3.28±0.01	3.23±0.01	3.06±0.03	3.09±0.01	3.05±0.02
energy	768	8	0.58±0.05	**0.56±0.04**	0.60±0.03	2.17±0.02	1.13±0.02	0.95±0.09	0.68±0.02	0.54±0.02
kin8nm	8192	8	**-1.15±0.01**	-1.12±0.01	-1.05±0.00	-0.81±0.01	-0.83±0.01	-0.92±0.02	N/A±0.00	N/A±0.00
power	9568	4	**2.69±0.00**	2.92±0.00	2.90±0.00	2.83±0.01	2.88±0.00	2.81±0.00	N/A±0.00	N/A±0.00
protein	45730	9	**2.85±0.00**	2.87±0.00	2.96±0.02	3.00±0.00	2.99±0.00	2.90±0.00	N/A±0.00	N/A±0.00
red wine	1588	11	**0.97±0.06**	**0.97±0.02**	1.20±0.04	1.01±0.02	**0.97±0.02**	1.01±0.02	1.04±0.01	0.26±0.03
yacht	308	6	0.59±0.11	**-0.02±0.07**	0.59±0.13	1.11±0.04	1.22±0.18	0.79±0.11	1.03±0.03	0.10±0.05
naval	11934	16	**-7.21±0.06**	-5.62±0.04	-4.11±0.00	-2.80±0.00	-2.80±0.00	-2.97±0.14	-7.13±0.02	N/A±0.00
Mean rank	N/A	N/A	**1.33**	2.11	3.56	5.22	4.56	3.33	N/A	N/A

functions, and 2), piecewise linear random functions. Please refer to Appendix 5.B.2 for details. For each prior, we first sample a random function from the prior; then, 100 observed data points are sampled as $\mathcal{D}$, half of which are sampled from $[0, 0.2]$ and the other half are sampled from $[0.8, 1]$. Finally, we ask the algorithms to perform inference using the prior, i.e., producing samples from $p(f|\mathcal{D})$.

We compare the performance of FVI with ground truth, f-BNN and VIPs. The ground truth posterior samples are generated by sampling importance re-sampling. F-BNNs are based on the code kindly open-sourced by [330]. As we found that the training time required by f-BNN is prohibitive, we only trained f-BNN for 100 epochs for fairness. For VIPs (Gaussian approximations), we use an empirical covariance kernel, which is estimated from random function samples of the implicit priors. For FVI, implementation details can be found in Appendix 5.B.2.

Results are displayed in Figure 5.6. FVI can successfully generate samples that mimic the piecewise constant/linear behaviors. The posterior uncertainty returned by FVI is also close to the ground truth estimates. On the other hand, f-BNNs severely under-fit the data and provide very poor in-between uncertainties. Note that, although f-BNNs are only trained for 100 epochs, their running time is still 100x higher than that of FVI (Appendix 5.C.3). VIP performs better than f-BNNs, but fails to mimic the behaviour of the priors: the posterior samples from VIP are very noisy. This is due to the prior function samples violating the Gaussian assumption, with the correlation level between points being lower than expected. This results in very noisy VIP posterior samples that are hard to interpret.

5.7.2 Multivariate regression with BNNs priors

In this experiment, we test if the proposed FVI can perform accurate posterior inference with BNNs as functional priors. We consider multivariate regression tasks based on 9 different UCI datasets. We mainly compare with the following weight-space VI baselines for BNNs:

Table 5.2 Contextual bandits performance comparison. Results are relative to the cumulative regret of the worst algorithm on each dataset. Numbers after the algorithm are the network sizes. The best methods are boldfaced, and the second best methods are highlighted in brown.

	MEAN RANK	MUSHROOM	STATLOG	COVERTYPE	JESTER	ADULT	WHEEL	CENSUS
FVI 2×50	**2.11**	16.46 ± 2.04	$\mathbf{7.95 \pm 2.92}$	$\mathbf{49.59 \pm 1.61}$	$\mathbf{68.59 \pm 6.87}$	$\mathbf{90.33 \pm 0.86}$	41.44 ± 9.28	51.77 ± 3.06
UNIFORM	10.45	100.0 ± 0.00	99.85 ± 0.36	99.49 ± 0.62	100.0 ± 0.00	99.60 ± 0.53	94.04 ± 11.9	99.30 ± 0.55
RMS	5.68	17.74 ± 7.65	10.36 ± 2.51	69.72 ± 7.23	75.07 ± 5.50	97.65 ± 1.48	70.39 ± 19.7	94.55 ± 3.60
DROPOUT 2×50	5.54	19.84 ± 6.46	15.53 ± 4.50	67.72 ± 2.32	75.04 ± 4.66	97.44 ± 0.98	59.40 ± 10.8	86.60 ± 0.52
BBB 2×50	4.88	23.18 ± 5.90	30.90 ± 3.29	63.91 ± 1.96	72.93 ± 5.69	95.49 ± 2.03	56.38 ± 11.3	70.68 ± 2.32
BBB 1×50	8.22	$\mathbf{15.52 \pm 4.40}$	80.25 ± 18.6	94.80 ± 4.84	83.30 ± 5.26	99.24 ± 0.66	58.12 ± 18.0	99.46 ± 0.37
NEURALLINEAR	6.94	19.04 ± 2.96	21.22 ± 1.98	75.34 ± 1.00	86.86 ± 3.61	97.93 ± 1.37	$\mathbf{37.41 \pm 8.86}$	83.75 ± 1.44
BOOTRMS	4.51	17.11 ± 5.99	9.47 ± 2.03	63.27 ± 1.35	74.66 ± 3.87	96.11 ± 1.02	63.15 ± 25.9	90.47 ± 3.40
PARAMNOISE	5.94	17.76 ± 4.14	20.95 ± 3.07	78.08 ± 5.66	76.95 ± 5.84	96.23 ± 1.81	41.26 ± 6.48	96.34 ± 4.56
BBα 2×50	9.45	68.45 ± 6.05	95.22 ± 4.88	98.60 ± 1.45	94.29 ± 2.69	98.72 ± 1.28	80.50 ± 7.96	97.94 ± 2.01
FBNN 2×50	3.17	16.55 ± 2.41	10.01 ± 1.39	50.10 ± 5.70	70.82 ± 3.27	90.72 ± 3.18	77.70 ± 21.2	$\mathbf{51.22 \pm 2.55}$

Bayes-by-Backprop [29], variational dropout [75], and variational alpha dropout [177] ($\alpha = 0.5$). We also compare with three function-space BNN inference methods: VIP-BNNs, VIP-Neural processes [198], and f-BNNs. Finally, we include comparisons to function space particle optimization [357] in Appendix 5.C.7 for reference purpose. All inference methods are based on the same BNN priors whenever applicable. For experimental settings, we follow [198]. Each dataset was randomly split into train (90%) and test sets (10%). This was repeated 10 times and results were averaged.

Results are shown in Table 5.1. Overall, FVI consistently outperforms other VI-based inference methods for BNNs and achieves the best result in 7 datasets (out of 9). FVI also outperforms f-BNNs (in 5 datasets out of 6), despite the fact that they are more expensive to train. Note that exact GPs and f-BNNs are not directly comparable to other methods, since i), they perform inference over different priors; and ii), they are much more expensive as they require the evaluation of the *exact* GP likelihood. Thus, their results are only available for smaller datasets, and are not included for ranking.

5.7.3 Contextual Bandits

Uncertainty estimates are important for downstream decision-making scenarios, since exploration-exploitation is a common dilemma that must be addressed. In this section, we consider a classic task called contextual bandits, where the agent is asked to make decisions that maximize the reward given some contexts (inputs). For this, Thompson sampling [334] is an elegant approach to guide exploration, where a model configuration is first sampled from the posterior, and then an optimal action is choosen based on the sampled configuration.

Table 5.3 Image classification and OOD detection performance. Accuracy, negative log-likelihood (NLL) and area-under-the-curve (AUC) of OOD detection are reported. Our method outperforms all baselines in terms of classification accuracy and OOD-AUC, and performs competitively on NLL for CIFAR10. Results for MAP, KFAC and Ritter et al. are obtained from [127].

	FMNIST			CIFAR10		
Model	Accuracy	NLL	OOD-AUC	Accuracy	NLL	OOD-AUC
FVI	**91.60±0.14**	**0.254±0.05**	**0.956±0.06**	**77.69 ±0.64**	0.675±0.03	**0.883±0.04**
MFVI	91.20±0.10	0.343±0.01	0.782±0.02	76.40±0.52	1.372±0.02	0.589±0.01
MAP	91.39±0.11	0.258±0.00	0.864±0.00	77.41±0.06	0.690±0.00	0.809±0.01
KFAC-Laplace	84.42±0.12	0.942±0.01	0.945±0.00	72.49±0.20	1.274±0.01	0.548±0.01
Ritter et al.	91.20±0.07	0.265±0.00	0.947±0.00	77.38±0.06	**0.661±0.00**	0.796±0.00

We compare FVI with several NN-related baselines, on datasets benchmarked by [274]. The hyperparameter settings are consistent with [330], except that we used a smaller batch-size (32). The learning rates for each baseline are tuned from $[0.001, 0.05]$. We report the cumulative regret, as well as the mean ranks. Experiments are repeated for 10 runs. As shown in Table 5.2, no single algorithm always outperforms the others in all bandit problems. However, FVI tends to give better performance than the baselines (ranks the first overall, performs the best on 4 out of 7 datasets, and ranked top 2 on 6 out of 7 datasets), indicating that FVI can provide reliable uncertainty estimates for decision making. Moreover, FVI is much more efficient than f-BNN (nearly 500 times faster, c.f. Appendix 5.C.4).

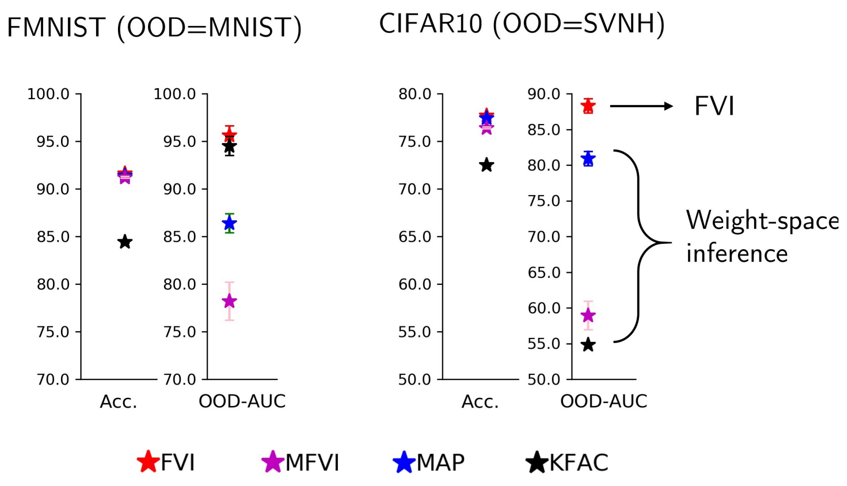


Figure 5.7 Image classification and OOD detection performance.

5.7.4 Image classification and out-of-distribution detection

To demonstrate the scalability of our method to higher dimensional data, we consider image classification tasks on Fashion MINIST [370] and CIFAR10 [161] with BNN priors. We compare our method to the following baselines: mean field VI (MFVI), maximum a posteriori (MAP), KFAC Laplace-GNN approximation [211] and its dampened version [275]. For all models, we use Bayesian CNNs with the same mixed CNN-fully connected structure as in [127, 300]. Apart from test accuracy and negative log likelihood (NLL), we also perform out-of-distribution detection using in-distribution (ID) / out-of-distribution (OOD) pairs including FashionMNIST/MNIST and CIFAR10/SVNH. Following the settings of [250, 127], we calculate the area under the curve (AUC) of out-of-distribution detection based on predictive entropies. Results are shown in Table 5.3. On both datasets, our proposed FVI method consistently outperforms all baselines in terms of (in-distribution) classification accuracy and OOD detection AUC. Although FVI does not achieve the best NLL on CIFAR10, it still performs competitively to MAP and dampened KFAC. This demonstrates that our method is able to scale to high dimensional data, and produce accurate predictions with well-calibrated uncertainties.

5.8 Conclusion

In this chapter, we took a algorithm-driven approach for function space inference, and proposed Functional Variational Inference (FVI).. It optimizes a grid-based functional divergence, which can be estimated based on our proposed SPG model. We demonstrated that FVI works well with implicit priors, scales well to high dimensional data and provides reliable uncertainty estimates. Possible directions for future work might include developing grid-function KL estimation method without surrogate models, and improving the theoretical understanding of functional space VI.

Appendix for Chapter 5

Appendix 5.A Proof of Theoretical results

5.A.1 Proof of Proposition 5.1 and 5.3

To prove Proposition 5.1, we first need the following lemma:

Lemma 5.1 (Alternative equivalent definition of functional KL divergence [330])**.** *The KL-divergence between two stochastic processes can be estimated by the supremum of marginal KL divergences over all finite subset of inputs:*

$$\mathrm{D}_{KL}[q(f)||p(f)] = \sup_{n,\mathbf{X}_n} \mathrm{D}_{KL}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})], \tag{5.A.1}$$

where $\mathbf{X}_n$ is the so called measurement points, $\mathbf{f}^{\mathbf{X}_n}$ is the vector of function values evaluated on $\mathbf{X}_n$, and $\mathrm{D}_{KL}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]$ is the KL-divergence over random vectors typically used in machine learning community.

Readers may refer to [330] for the proof of this lemma.

Proposition 5.1. *Suppose c has full support on $\mathcal{T}^{\mathbb{Z}^+}$. Then, $\mathrm{D}_{grid}[q(f)||p(f|\mathcal{D})]$ Satisfies the following conditions:*

- $\mathrm{D}_{grid}[q(f)||p(f|\mathcal{D})] \geq 0$
- $\mathrm{D}_{grid}[q(f)||p(f|\mathcal{D})] = 0$ *if and only if* $q(f) = p(f|\mathcal{D})$

Proof: First, according the the definition of

$$\mathrm{D}_{\mathrm{grid}}[q(f)||p(f|\mathcal{D})] = \mathbb{E}_{n,\mathbf{X}_n\sim c}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})]$$

, the positivity property holds since $\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})] \geq 0$.

Next, to prove $\mathrm{D}_{\mathrm{grid}}[q(f)||p(f|\mathcal{D})] = 0$ if and only if $q(f) = p(f|\mathcal{D})$, we first show that

$$\arg\min_{q(f)} \mathrm{D}_{\mathrm{KL}}[q(f)||p(f|\mathcal{D})] = \arg\min_{q(f)} \mathbb{E}_{n,\mathbf{X}_n\sim c}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})],$$

Let's first consider the left handside, $\arg\min_{q(f)} \mathrm{D}_{\mathrm{KL}}[q(f)||p(f|\mathcal{D})]$. When it reaches the optimum, we have a unique solution, $q^{\star}_{L}(f) = p(f|\mathcal{D})$. According to Equation 5.A.1, we have:

$$\arg\min_{q(f)} \sup_{n,\mathbf{X}_n} \mathrm{D_{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})] = \arg\min_{q(f)} \mathrm{D_{KL}}[q(f)||p(f|\mathcal{D})] = q_L^\star(f)$$

Also, notice that

$$\mathbb{E}_{n,\mathbf{X}_n\sim c}\mathrm{D_{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})] \leq \sup_{n,\mathbf{X}_n} \mathrm{D_{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})]$$

At $q_L^\star(f)$, we have

$$0 \leq \mathbb{E}_{n,\mathbf{X}_n\sim c}\mathrm{D_{KL}}[q_L^\star(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})] \leq \sup_{n,\mathbf{X}_n} \mathrm{D_{KL}}[q_L^\star(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})] = 0$$

Therefore, we have

$$q_L^\star(f) \in \arg\min_{q(f)} \mathbb{E}_{n,\mathbf{X}_n\sim c}\mathrm{D_{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})]$$

On the other hand, assume that $\mathbb{E}_{n,\mathbf{X}_n\sim c}\mathrm{D_{KL}}[q_L^\star(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})]$ reaches its optimum 0 at some optimal solution $q_R^\star(f)$. Since $\mathrm{D_{KL}}[q_R^\star(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})]$ is non-negative and c has full support, we have $\mathrm{D_{KL}}[q_R^\star(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})] = 0$ for all possible $\mathbf{X}_n \subset \mathrm{supp}(c) = \mathcal{T}^{\mathbb{Z}^+}$. Therefore, we have

$$\mathrm{D_{KL}}[q_R^\star(f)||p(f|\mathcal{D})] = \sup_{n,\mathbf{X}_n} \mathrm{D_{KL}}[q_R^\star(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})] = 0$$

Therefore, we have

$$q_R^\star(f) = p(f|\mathcal{D}) = q_L^\star(f)$$

That is,

$$\arg\min_{q(f)} \mathrm{D_{KL}}[q(f)||p(f|\mathcal{D})] = \arg\min_{q(f)} \mathbb{E}_{n,\mathbf{X}_n\sim c}\mathrm{D_{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})] = p(f|\mathcal{D})$$

In other words, both the functional KL divergence and grid-functional KL divergence have the same unique global optimal solution, $q(f|\mathcal{D})$. At $q(f|\mathcal{D})$, both divergence achieves minimum value, 0. Therefore, $\mathrm{D_{KL}}[q(f)||p(f|\mathcal{D})] = 0$ implies that $q(f)$ must be $p(f|\mathcal{D})$.

□

Proposition 5.3. *Let $n, \mathbf{X}_n \sim c$ be a set of random measure points such that $\mathbf{X}_n$ always contains $\mathbf{X}_\mathcal{D}$. Define:*

$$\mathcal{L}_q^{grid} := \log p(\mathcal{D}) - \mathrm{D}_{grid}[q(f)||p(f|\mathcal{D})]. \tag{5.A.2}$$

Then we have:

$$\mathcal{L}_q^{grid} = \mathbb{E}_{q(f)}[\log p(\mathcal{D}|f)] - \mathrm{D}_{grid}[q(f)||p(f)] \tag{5.A.3}$$

and $\log p(\mathcal{D}) \geq \mathcal{L}_q^{grid} \geq \mathcal{L}_q^{functional}$.

Proof: Since $\mathrm{D}_{\mathrm{grid}} \geq 0$, the the statement $\log p(\mathcal{D}) \geq \mathcal{L}_q^{grid}$ obviously holds. Then, notice that:

$$\begin{aligned}
&\mathcal{L}_q^{grid}\\
&= \log p(\mathcal{D}) - \mathbb{E}_{n,\mathbf{X}_n\sim c}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})]\\
&= \mathbb{E}_{n,\mathbf{X}_n\sim c}\{\log p(\mathcal{D}) - \mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})]\}\\
&= \mathbb{E}_{n,\mathbf{X}_n\sim c}\{\log p(\mathcal{D}) - \mathbb{E}_q[\log \frac{q(\mathbf{f}^{\mathbf{X}_n})}{p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})}]\}\\
&= \mathbb{E}_{n,\mathbf{X}_n\sim c}\{\log p(\mathcal{D}) - \mathbb{E}_q[\log \frac{q(\mathbf{f}^{\mathbf{X}_n})p(\mathcal{D})}{p(\mathbf{f}^{\mathbf{X}_n},\mathcal{D})}]\}\\
&= \mathbb{E}_{n,\mathbf{X}_n\sim c}\{\mathbb{E}_q[-\log q(\mathbf{f}^{\mathbf{X}_n}) + \log p(\mathbf{f}^{\mathbf{X}_n},\mathcal{D})]\}\\
&= \mathbb{E}_{n,\mathbf{X}_n\sim c}\{\mathbb{E}_{q(\mathbf{f}^{\mathcal{D}})}\log p(\mathcal{D}|\mathbf{f}^{\mathcal{D}}) - \mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]\}\\
&= \mathbb{E}_{q(f)}[\log p_\pi(\mathcal{D}|f)] - \mathbb{E}_{n,\mathbf{X}_n\sim c}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]
\end{aligned}$$

This proves the statement that $\mathcal{L}_q^{grid} = \mathbb{E}_{q(f)}[\log p_\pi(\mathcal{D}|f)] - \mathrm{D}_{\mathrm{grid}}[q(f)||p(f)]$.

Finally, since

$$\begin{aligned}
&\mathbb{E}_{n,\mathbf{X}_n\sim c}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]\\
\leq&\mathbb{E}_{n,\mathbf{X}_n\sim c}\sup_{n,\mathbf{X}_n}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]\\
=&\sup_{n,\mathbf{X}_n}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]\\
=&\mathrm{D}_{\mathrm{KL}}[q(f)||p(f)],
\end{aligned}$$

Therefore we also have:

$$\mathbb{E}_{q(f)}[\log p_\pi(\mathcal{D}|f)] - \mathbb{E}_{n,\mathbf{X}_n\sim c}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})] \geq \mathcal{L}_q^{functional},$$

which concludes the first part of the proposition.

□

5.A.2 Proof of Proposition 5.2

Proposition 5.2. *Let $p(f)$ and $q(f)$ be two distributions for random functions. Assume that $p(f)$ is parameterized by the following sampling processes:*

$$f = h + \varepsilon, h(\mathbf{x}) \sim p(h|\mathbf{x};\Theta), \Theta \sim p(\Theta), \varepsilon \sim \mathcal{N}(0,\sigma^2)$$

, And $q(f)$ is parameterized by:

$$f = h + \eta, h(\mathbf{x}) \sim q(h|\mathbf{x};\Gamma), \Gamma \sim q(\Gamma), \eta \sim \mathcal{N}(0,\sigma^2)$$

. Here,$\mathbf{x} \in \mathcal{T} \subset \mathbb{R}^d$, h is the random latent function, $\Theta \in \mathbb{R}^I$, $\Gamma \in \mathbb{R}^J$ are the parameters of each random function distributions, respectively. Suppose that: 1, $q(\mathbf{h}^{\mathbf{x}})$ has a compact support, denoted by $\mathcal{B}$; 2, $q(\Gamma)$ has a compact support, denote by $\mathcal{V}$; and 3, $p(h|\mathbf{x};\Theta) > 0$ for $\forall h \in \mathcal{B}, \mathbf{x} \in \mathcal{T}$, and $\Theta \in \mathbb{R}^I$. Then, there exist a sampling distribution c such that: 1, c has full support on $\mathcal{T}^{\mathbb{Z}^+}$, and 2, $\mathrm{D}_{grid}[q(f)||p(f)]$ is finite.

Proof : Let $\mathbf{X}_n$ denote a set of n measure points $\{\mathbf{x}_k\}_{1\leq k\leq n}$ in $\mathcal{T}^n$. Also, let the sampling distribution c to have the following form:

$$n \sim p(n), \mathbf{x}_k \sim \mathcal{U}(\mathcal{T}), \ \forall 1 \leq k \leq n$$

That is, c first samples a positive integer n from the distribution $p(n)$, and then draw n samples from $\mathcal{T}$ independently and uniformly. Let us consider $\mathrm{D}_{\mathrm{grid}}[q(f)||p(f)]$. According to the definition

$$\begin{aligned}
&\mathrm{D}_{\mathrm{grid}}[q(f)||p(f)]\\
=&\mathbb{E}_{n,\mathbf{X}_n\sim c}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n}|\mathcal{D})]\\
=&\mathbb{E}_{n\sim p(n)}\mathbb{E}_{\mathbf{X}_n\sim U(\mathcal{T}^n)}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]\\
=&\sum_{n=1}^{\infty}p(n)\mathbb{E}_{\mathbf{X}_n\sim U(\mathcal{T}^n)}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]
\end{aligned}$$

Therefore, we only need to show that the series $\sum_{n=1}^{\infty}p(n)\mathbb{E}_{\mathbf{X}_n\sim U(\mathcal{T}^n)}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]$ converges. Notice that

$$\begin{aligned}
&\mathbb{E}_{\mathbf{X}_n\sim U(\mathcal{T}^n)}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]\\
\leq&\mathbb{E}_{\mathbf{X}_n\sim U(\mathcal{T}^n)}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{h}^{\mathbf{X}_n})||p(\mathbf{h}^{\mathbf{X}_n})]\\
=&\mathbb{E}_{\mathbf{X}_n\sim U(\mathcal{T}^n)}\int_{\mathbf{h}^{\mathbf{X}_n}}q(\mathbf{h}^{\mathbf{X}_n})\log\frac{q(\mathbf{h}^{\mathbf{X}_n})}{p(\mathbf{h}^{\mathbf{X}_n})}d\mathbf{h}^{\mathbf{X}_n}\\
\leq&\mathbb{E}_{\mathbf{X}_n\sim U(\mathcal{T}^n)}\left[\log\bar{q}-\log\underline{p}\right]\\
\leq&\sup_{\mathbf{X}_n\in\mathcal{T}^n}\left[\log\bar{q}-\log\underline{p}\right]
\end{aligned}\tag{5.A.4}$$

The first inequality is due to information processing inequality. The $\bar{q}$ and $\underline{p}$ in the second inequality is defined as

$$\bar{q}=\sup_{\mathbf{h}^{\mathbf{X}_n}\in\mathcal{B}^n\subset\mathbb{R}^n}q(\mathbf{h}^{\mathbf{X}_n})>0$$

$$\underline{p}=\inf_{\mathbf{h}^{\mathbf{X}_n}\in\mathcal{B}^n\subset\mathbb{R}^n}p(\mathbf{h}^{\mathbf{X}_n})>0$$

. Note that both $\bar{q}$ and $\underline{p}$ are strictly greater than 0, due to the the fact that $\mathcal{B}^n$ is the support of $q(\mathbf{h}^{\mathbf{X}_n})$, and $p(h|\mathbf{x};\Theta) > 0$ for $\forall h \in \mathcal{B}$. Next, notice that

$$\begin{aligned}
&\bar{q} \\
=& \sup_{\mathbf{h}^{\mathbf{X}_n} \in \mathcal{B}^n} q(\mathbf{h}^{\mathbf{X}_n}) \\
=& \sup_{\mathbf{h}^{\mathbf{X}_n} \in \mathcal{B}^n} \int_{\Gamma} \prod_{1 \le k \le n} q(h_k|\mathbf{x}_k;\Gamma) q(\Gamma) d\Gamma \\
\le& \sup_{\mathbf{h}^{\mathbf{X}_n} \in \mathcal{B}^n} \sup_{\Gamma \in \mathcal{V}} \sup_{\mathbf{X}_n \in \mathcal{T}^n} \prod_{1 \le k \le n} q(h_k|\mathbf{x}_k;\Gamma) \\
\le& \prod_{1 \le k \le n} \left(\sup_{h_k \in \mathcal{B}} \sup_{\Gamma \in \mathcal{V}} \sup_{\mathbf{x}_k \in \mathcal{T}} q(h_k|\mathbf{x}_k;\Gamma) \right) \\
=& (q^\star)^n > 0
\end{aligned}$$

Where we have used $(q^\star)$ to denote $\left(\sup_{h_k \in \mathcal{B}} \sup_{\Theta \in \mathcal{V}} \sup_{\mathbf{x}_k \in \mathcal{T}} q(h_i|\mathbf{x_k};\Gamma)\right)$. The second equality is given by the definition of $q(h;\Gamma)$; the first inequality is due to that the expectation is replaced by $\sup_{\Gamma \in \mathcal{V}} \sup_{\mathbf{X}_n \in \mathcal{T}^n}$, and the fact that $q(\Gamma)$ has compact support; and in the last inequality $q^\star > 0$ since otherwise, $q(h_k|\mathbf{x}_k;\Gamma) \equiv 0$ which contradicts with that fact that $q(\mathbf{h}^{\mathbf{x}_k}) = \int_\Gamma q(h_k|\mathbf{x}_k;\Gamma) q(\Gamma) d\Gamma$ has compact support. Similarly, we also have:

$$\begin{aligned}
&\underline{p} \\
=& \inf_{\mathbf{h}^{\mathbf{X}_n} \in \mathcal{B}^n} p(\mathbf{h}^{\mathbf{X}_n}) \\
=& \inf_{\mathbf{h}^{\mathbf{X}_n} \in \mathcal{B}^n} \int_{\Theta} \prod_{1 \le k \le n} p(h_k|\mathbf{x}_k;\Theta) p(\Theta) d\Theta \\
\ge& \inf_{\mathbf{h}^{\mathbf{X}_n} \in \mathcal{B}^n} \inf_{\Theta} \inf_{\mathbf{X}_n \in \mathcal{T}^n} \prod_{1 \le k \le n} p(h_k|\mathbf{x}_k;\Theta) \\
\ge& \prod_{1 \le k \le n} \left(\inf_{h_k \in \mathcal{B}} \inf_{\Theta} \inf_{\mathbf{x}_k \in \mathcal{T}} p(h_k|\mathbf{x}_k;\Theta) \right) \\
=& (p^\star)^n > 0
\end{aligned}$$

, where we have used $p^\star$ to denote $\left(\inf_{h_k \in \mathcal{B}} \inf_{\Theta} \inf_{\mathbf{x}_k \in \mathcal{T}} p(h_k|\mathbf{x}_k;\Theta)\right)$. Note that the second inequality is given by our assumption that $p(h|\mathbf{x};\Theta) > 0$ for $\forall h \in \mathcal{B}, \mathbf{x} \in \mathcal{T}$, and $\Theta \in \mathbb{R}^I$.

Now, back to inequality 5.A.4, we have:

$$\begin{aligned}&\mathbb{E}_{\mathbf{X}_n\sim U(\mathcal{T}^n)}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]\\ \leq&\sup_{\mathbf{X}_n\in\mathcal{T}^n}\left[\log\bar{q}-\log\underline{p}\right]\\ \leq&n\left[\log q^{\star}-\log p^{\star}\right]\end{aligned}$$

Finally, let us consider the series $\sum_{n=1}^{\infty}p(n)n\left[\log q^{\star}-\log p^{\star}\right]$. Apparently, based on d'Alembert's criterion, this series is absolute convergent if we can choose $p(n)$ such that $\lim_{n\to\infty}p(n+1)/p(n)<1$. For example,$p(n)$ could be a geometric distribution with a success probability that is strictly greater than 0 and smaller than 1. Since geometric distribution has full support in $\mathbb{Z}^{+}$, it satisfies the claim of this proposition. Finally, given such $p(n)$ distribution, $\sum_{n=1}^{\infty}p(n)\mathbb{E}_{\mathbf{X}_n\sim U(\mathcal{T}^n)}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]$ is also convergent due to direct comparison test.

□

Remark (grid-functional KL using BNN as priors). We here note that the proof applies to BNN priors. Assume $p(h_i|\mathbf{x}_i;\Theta=\mathbf{w})=\mathcal{N}(h_i;g_{\mathbf{w}}(\mathbf{x}_i),\varsigma^2)$, where $g_{\mathbf{w}}(\cdot)$ is a Bayesian neural network parameterized by $\mathbf{w}$, and $p(\mathbf{w})$ is some suitable prior on weights such as factorized Gaussians. In this case, it is trivial to verify that given any compact set $\mathcal{B}$, $p(h|\mathbf{x};\Theta)>0$ for $\forall h\in\mathcal{B},\mathbf{x}\in\mathcal{T}$, and $\Theta\in\mathbb{R}^{I}$ holds, hence the assumptions in Proposition 5.A.2 is satisfied.

5.A.3 Grid-Functional KL between a parametric model and a Gaussian process

In this section, we discuss the non-parametric counter part of Proposition 5.2, i.e., is the grid functional KL between a parametric model and a Gaussian process is still finite? Assume that $q(f)$ is a parametric model parameterized as in Proposition 1, and $p(f)$ is a zero mean Gaussian process with kernel function $K(\cdot,\cdot)$. Assume that $K(\cdot,\cdot)$ is a stationary kernel, i.e., $K(\mathbf{x}_1,\mathbf{x}_2)=\Phi(||\mathbf{x}_1-\mathbf{x}_2||)$ for some function Φ (e.g., radial basis function). In fact, we have the following Corollary:

Corollary 5.1. *Let $p(f)$ and $q(f)$ be two distributions for random functions. Assume that $q(f)$ is parameterized by the following sampling processes:*

$$f=h+\varepsilon,h(\mathbf{x})\sim q(h|\mathbf{x};\Gamma),\Gamma\sim p(\Gamma),\varepsilon\sim\mathcal{N}(0,\sigma^2)$$

, And $p(f)$ is parameterized by a zero mean Gaussian process with kernel function $K(\cdot,\cdot)$.

Assume further that: i), $q(f)$ satisfies the assumptions in Proposition 5.2; ii), $K(\cdot,\cdot)$ is a stationary kernel, i.e., $K(\mathbf{x}_1,\mathbf{x}_2) = \Phi(\|\mathbf{x}_1 - \mathbf{x}_2\|)$ for some function Φ (e.g., radial basis function). and iii), the smallest eigen value of $\mathbf{K}_{\mathbf{X}_n,\mathbf{X}_n}$, denoted by $\boldsymbol{\lambda}_n$, decays in the order of $\mathcal{O}(n^{-\gamma})$ for some constant $\gamma > 1$ (see the literature of eigen value distribution/lower bounding smallest eigen value of kernel matrices, and/or norm estimation for inverse matrices. For example, [358, 14, 360, 18, 297, 239] to name a few).

Then, there exist a sampling distribution c such that: 1, c has full support on $\mathcal{T}^{\mathbb{Z}^+}$, and 2, $\mathrm{D}_{grid}[q(f)||p(f)]$ is finite.

Proof We can basically apply most of the proof of Proposition 5.2. In our case, the key ingredient is to derive a lower bound for

$$\underline{p} = \inf_{\mathbf{h}^{\mathbf{X}_n} \in \mathcal{A}^n \subset \mathbb{R}^n} p(\mathbf{h}^{\mathbf{X}_n})$$

. Since $p(\mathbf{h}^{\mathbf{X}_n})$ is a GP as described before, its likelihood function is given by

$$\log p(\mathbf{h}^{\mathbf{X}_n}) = -\frac{{\mathbf{h}^{\mathbf{X}_n}}^T \mathbf{K}^{-1}_{\mathbf{X}_n,\mathbf{X}_n} \mathbf{h}^{\mathbf{X}_n}}{2} - \frac{n}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}_{\mathbf{X}_n,\mathbf{X}_n}|$$

Without loss of generality, assume that $\|\mathbf{h}^{\mathbf{X}_n}\| \leq A$ for some constant A. Then, we have

$${\mathbf{h}^{\mathbf{X}_n}}^T \mathbf{K}^{-1}_{\mathbf{X}_n,\mathbf{X}_n} \mathbf{h}^{\mathbf{X}_n} \leq \frac{1}{\boldsymbol{\lambda}_n}\|\mathbf{h}^{\mathbf{X}_n}\| \leq \frac{A}{\boldsymbol{\lambda}_n}$$

, where $\boldsymbol{\lambda}_n$ denotes the smallest eigen value for $\mathbf{K}_{\mathbf{X}_n,\mathbf{X}_n}$ (or equivalently, $\frac{1}{\boldsymbol{\lambda}_n}$ is the largest eigen value for $\mathbf{K}^{-1}_{\mathbf{X}_n,\mathbf{X}_n}$).

Notice also that

$$\log|\mathbf{K}_{\mathbf{X}_n,\mathbf{X}_n}| \leq n\log\frac{1}{n}\mathrm{Tr}(\mathbf{K}_{\mathbf{X}_n,\mathbf{X}_n}) = n\log\Phi(0)$$

.

Therefore, we can write

$$\log\underline{p} \geq -\frac{n}{2}(\log 2\pi + \log\Phi(0)) - \frac{A}{2\boldsymbol{\lambda}_n}$$

By the same argument used in Proposition 5.2, we have

$$\begin{aligned}&\mathbb{E}_{\mathbf{X}_n\sim U(\mathcal{T}^n)}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]\\ \le& n\left[\log q^\star+\frac{1}{2}(\log 2\pi+\log\Phi(0))\right]+\frac{A}{2\boldsymbol{\lambda}_n}\end{aligned}$$

Since $\boldsymbol{\lambda}_n$ decays in the order of $\mathcal{O}(n^{-\gamma})$ for some constant $\gamma>1$, by running the same argument as in the proof of Proposition 5.2, $\sum_{n=1}^{\infty}p(n)\mathbb{E}_{\mathbf{X}_n\sim U(\mathcal{T}^n)}\mathrm{D}_{\mathrm{KL}}[q(\mathbf{f}^{\mathbf{X}_n})||p(\mathbf{f}^{\mathbf{X}_n})]$ is absolute convergent if $\lim_{n\to\infty}p(n+1)/p(n)<1$. □

5.A.4 Proof of Proposition 5.4

Proposition 5.3 (Expressiveness of SPGs). *Let $p(f)$ be a square-integrable stochastic process defined on probability space $(\mathcal{X},\mathcal{B})$, and its index set $\mathcal{T}$ is a compact subset of $\mathbb{R}^d$. Here, $\mathcal{X}$ is a compact metric space, $\mathcal{B}$ is the Borel set on $\mathcal{X}$. Then, for $\forall\varepsilon>0$, there exists a SPG $q^{\varepsilon}_{SPG}(f)$ with a Gaussian prior on latent space, such that:*

$$\mathrm{MMD}(p,q^{\varepsilon}_{SPG};\mathcal{F})<\varepsilon\quad \textit{for}\quad \forall\mathbf{x}\in\mathcal{T},$$

where MMD *is the maximum mean discrepancy between p and q, $\mathcal{F}$ is the MMD function class defined to be a unit ball in a reproducing kernel Hilbert space (RKHS) with a universal kernel [324] $k(\cdot,\cdot)$ as its reproducing kernel.*

Proof Since $p(f)$ is a stochastic process defined on $\mathcal{L}^2(\mathcal{T})$, we can apply Karhunen–Loeve expansion to $f(\mathbf{x})$. Specifically, we have:

$$f(\mathbf{x})=\lim_{N\to\infty}L_N,\quad L_N:=\sum_i^N Z_i\phi_i(\mathbf{x}),\quad \sum_i^{\infty}\boldsymbol{\lambda}_i<+\infty.$$

Where the limit is in the sense of (uniform) convergence in $\mathcal{L}^2(\mathcal{T})$, Z_i are zero-mean, uncorrelated random variables with variance $\boldsymbol{\lambda}_i$. Here $\{\phi_i\}_{i=1}^{\infty}$ is an orthonormal basis of $\mathcal{L}^2(\mathbb{R}^d)$ that are also eigen functions of the operator $O_C(f)$ defined by $O_C(f)(\mathbf{x})=\int C(\mathbf{x},\mathbf{x}')z(\mathbf{x}')d\mathbf{x}'$, $C(\mathbf{x},\mathbf{x}')$ is the covariance function of $f(\cdot)$. The variance $\boldsymbol{\lambda}_i$ of Z_i is the corresponding eigen value of $\phi_i(\mathbf{x})$.

Then, notice that since we have assumed that k is universal and $\mathcal{T}$ is a compact metric space, by Theorem 23 of [322] we have that $\mathrm{MMD}(\cdot,\cdot;\mathcal{F})$ metrizes the weak convergence of probability measures on $\mathcal{P}$, where $\mathcal{P}$ is the set of all Borel measures on $(\mathcal{X},\mathcal{B})$. Here, "metrization" means that for any sequence of measures $\mathbb{P}_1, \mathbb{P}_2, ..., \mathbb{P}_n, ... \in \mathcal{P}$, we have

$$\mathbb{P}_n \xrightarrow{w} \mathbb{P} \Leftrightarrow \lim_{n \to \infty} \mathrm{MMD}(\mathbb{P}_n, \mathbb{P}; \mathcal{F}) = 0.$$

Since convergence of $L_N \to f$ in $\mathcal{L}^2$ implies weak convergence, we can apply this theorem to $p(f)$, and show that:

$$\lim_{n \to \infty} \mathrm{MMD}(p_{L_n}, p; \mathcal{F}) = 0$$

holds uniformly in $\mathcal{L}^2(\mathcal{T})$. Next, given a SPG q_{SPG}, we have:

$$\mathrm{MMD}(q_{\mathrm{SPG}}, p; \mathcal{F}) \leq \mathrm{MMD}(p_{L_n}, p; \mathcal{F}) + \mathrm{MMD}(q_{\mathrm{SPG}}, p_{L_n}; \mathcal{F}), \quad \forall n \in \mathbb{Z}_+$$

The above triangle inequality holds since k is universal [93]. Hence, to prove our theorem, it sufficies to show that there exits a sequence of SPGs $q_{\mathrm{SPG},1}, ..., q_{\mathrm{SPG},n'}, ...$ such that $\lim_{n' \to \infty} \mathrm{MMD}(q_{\mathrm{SPG},n'}, p_{L_n}; \mathcal{F}) = 0, \quad \forall n \in \mathbb{Z}_+, \, \mathbf{x} \in \mathcal{T}$. To prove this, let us fix n for now, and consider the random coefficients $\{Z_i\}_{i=1}^n$ of L_n. Based on the results from [48], there exists a sequence of Gaussian VAEs $q_{\mathrm{VAE},1}(\{Z_i\}_{i=1}^n), ..., q_{\mathrm{VAE},n''}(\{Z_i\}_{i=1}^n), ...$ of latent size n, such that

$$q_{\mathrm{VAE},n''}(\{Z_i\}_{i=1}^n) \xrightarrow{w} p(\{Z_i\}_{i=1}^n)$$

. Then, define our sequence of SPGs to be:

$$q_{\mathrm{SPG},n'} = \sum_i^n \tilde{Z}_i \phi_i, \quad , \{\tilde{Z}_i\}_{i=1}^n \sim q_{\mathrm{VAE},n'}(\{Z_i\}_{i=1}^n)$$

. Based on our definition in Section 5.3, $q_{\mathrm{SPG},n'}$ is indeed a SPG. Since the linear summation over ϕ_i using linear weights $\{Z_i\}_{i=1}^n$ is a continuous mapping, we also have:

$$q_{\mathrm{SPG},n'} \xrightarrow{w} p_{L_n}, \quad \forall n \in \mathbb{Z}_+, \, \mathbf{x} \in \mathcal{T}$$

due to continuous mapping theorem. Again, from the MMD metrization, we have

$$\lim_{n' \to \infty} \mathrm{MMD}(q_{\mathrm{SPG},n'}, p_{L_n}; \mathcal{F}) = 0, \quad \forall n \in \mathbb{Z}_+, \, \mathbf{x} \in \mathcal{T}$$

. To finally prove our theorem, consider an arbitrary error ε. Then, there exists L_n such that $\mathrm{MMD}(p_{L_n}, p; \mathcal{F}) < \varepsilon/2$. Next, given this particular L_n, there exits n' such that

$\mathrm{MMD}(p_{L_n}, q_{\mathrm{SPG},n'}; \mathcal{F}) < \varepsilon/2$. Together, we have:

$$\mathrm{MMD}(q_{\mathrm{SPG},n'}, p; \mathcal{F}) \leq \mathrm{MMD}(p_{L_n}, p; \mathcal{F}) + \mathrm{MMD}(q_{\mathrm{SPG},n'}, p_{L_n}; \mathcal{F}) < \varepsilon/2 + \varepsilon/2 = \varepsilon$$

which completes the proof our theorem.

□

5.A.5 Proof of Proposition 5.5

Proposition 5.4 (functional KL divergence on measurement points for SPGs)**.** *Let $q_{SPG}(f)$ and $\tilde{p}_{SPG}(f)$ be the SPGs defined in Equation 5.4.1 and 5.4.4. Then we have:*

$$\mathrm{D}_{KL}[q_{SPG}(\mathbf{f}^{\mathbf{X}_n})||\tilde{p}_{SPG}(\mathbf{f}^{\mathbf{X}_n})] = \mathbb{E}_{f\sim q_{SPG(f)}} \log \mathcal{Z}(\mathbf{f}^{\mathbf{X}_n}),$$

where $\mathcal{Z}(\mathbf{f}^{\mathbf{X}_n})$ is the partition function, $\mathcal{Z}(\mathbf{f}^{\mathbf{X}_n}) = \int_{\mathbf{h}} \tilde{p}_{SPG}(\mathbf{h}|\mathbf{f}^{\mathbf{X}_n})\frac{q_{\eta}(\mathbf{h})}{p_0(\mathbf{h})}d\mathbf{h}$.

Proof First, we have

$$\begin{aligned}
&\mathrm{D}_{\mathrm{KL}}[q_{\mathrm{SPG}}(\mathbf{f}^{\mathbf{X}_n})||\tilde{p}_{\mathrm{SPG}}(\mathbf{f}^{\mathbf{X}_n})]\\
=&\mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\eta}}(\mathbf{h})||p_0(\mathbf{h})] - \mathbb{E}_{f\sim q_{\mathrm{SPG}(f)}}\mathrm{D}_{\mathrm{KL}}[q_{\mathrm{SPG}}(\mathbf{h}|\mathbf{f}^{\mathbf{X}_n})||\tilde{p}_{\mathrm{SPG}}(\mathbf{h}|\mathbf{f}^{\mathbf{X}_n})]\\
=&\mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\eta}}(\mathbf{h})||p_0(\mathbf{h})] - \mathbb{E}_{f\sim q_{\mathrm{SPG}(f)}}\mathrm{D}_{\mathrm{KL}}[\tilde{p}_{\mathrm{SPG}}(\mathbf{h}|\mathbf{f}^{\mathbf{X}_n})\frac{q_{\boldsymbol{\eta}}(\mathbf{h})}{\mathcal{Z}(\mathbf{f}^{\mathbf{X}_n})p_0(\mathbf{h})}||\tilde{p}_{\mathrm{SPG}}(\mathbf{h}|\mathbf{f}^{\mathbf{X}_n})]\\
=&\mathrm{D}_{\mathrm{KL}}[q_{\boldsymbol{\eta}}(\mathbf{h})||p_0(\mathbf{h})] - \mathbb{E}_{f\sim q_{\mathrm{SPG}(f)},\mathbf{h}\sim q_{\mathrm{SPG}}(\mathbf{h}|\mathbf{f}^{\mathbf{X}_n})}\log\frac{q_{\boldsymbol{\eta}}(\mathbf{h})}{p_0(\mathbf{h})} + \mathbb{E}_{f\sim q_{\mathrm{SPG}(f)}}\log\mathcal{Z}(\mathbf{f}^{\mathbf{X}_n})\\
=&\mathbb{E}_{f\sim q_{\mathrm{SPG}(f)}}\log\mathcal{Z}(\mathbf{f}^{\mathbf{X}_n})
\end{aligned}$$

where the first equality directly follows from the chain rule of KL-divergence, and the second equality follows from the fact that $q_{\mathrm{SPG}}(\mathbf{h}|\mathbf{f}^{\mathbf{X}_n})) \propto q_{\mathrm{SPG}}(\mathbf{h})\tilde{p}_{\mathrm{SPG}}(\mathbf{f}^{\mathbf{X}_n}|\mathbf{h}), \tilde{p}_{\mathrm{SPG}}(\mathbf{h}|\mathbf{f}^{\mathbf{X}_n}) \propto p_0(\mathbf{h})\tilde{p}_{\mathrm{SPG}}(\mathbf{f}^{\mathbf{X}_n}|\mathbf{h})$.

□

5.A.6 Proof of Proposition 5.6

Proposition 5.5 (Biased Mini-batch estimation of log-partition function)**.** $\mathbb{E}_{n,\mathbf{X}_n\sim c}\mathbb{E}_{f\sim q_{SPG}(f)}\log\tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}_n})$ *can be estimated by the following mini-batch estima-*

tor:

$$\begin{aligned}\mathcal{J}_K := \frac{1}{2}\sum_{i=1}^{H} \mathbb{E}_{f\sim q_{SPG}(f)} \Big[&\log\sigma_{\boldsymbol{\eta}_i}^{-2} + \log\hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2} \\ &- \log(\sigma_{\boldsymbol{\eta}_i}^{-2} + \hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2} - 1) - \hat{\boldsymbol{\mu}}_{\boldsymbol{\lambda}_i}^{2}\hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2} - \boldsymbol{\mu}_{\boldsymbol{\eta}_i}^{2}\sigma_{\boldsymbol{\eta}_i}^{-2} \\ &+ (\hat{\sigma}_{\boldsymbol{\eta}_i}^{-2}\hat{\boldsymbol{\mu}}_{\boldsymbol{\eta}_i} + \hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2}\hat{\boldsymbol{\mu}}_{\boldsymbol{\lambda}_i})^2(\sigma_{\boldsymbol{\eta}_i}^{-2} + \hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2} - 1)^{-1}\Big],\end{aligned} \tag{5.A.5}$$

where H is the dimensionality of $\mathbf{h}$, $\mathcal{N}(\mathbf{h};\boldsymbol{\mu}_{\boldsymbol{\eta}_i},\sigma_{\boldsymbol{\eta}_i}^2) = q_{\boldsymbol{\eta}}(h_i)$, $\mathcal{N}(\mathbf{h};\boldsymbol{\mu}_{\boldsymbol{\lambda}_i},\sigma_{\boldsymbol{\lambda}_i}^2) = \tilde{q}_{\boldsymbol{\lambda}}(h_i|\mathbf{f}^{\mathbf{X}_n})$. $\hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2}$ and $\hat{\boldsymbol{\mu}}_{\boldsymbol{\lambda}_i}$ are the mini-batch approximators for $\boldsymbol{\mu}_{\boldsymbol{\lambda}_i}$ and $\sigma_{\boldsymbol{\lambda}_i}^2$, respectively:

$$\begin{aligned}\hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2} &:= \sum_{k\in\mathcal{K}} \frac{|\mathcal{D}|}{K}\sigma_{h_i|f^{\mathbf{x}_k}}^{-2} + \sum_{\mathbf{x}_l\in\mathbf{X}_n\setminus\mathbf{X}_{\mathcal{D}}} \sigma_{h_i|f^{\mathbf{x}_l}}^{-2} \\ \frac{\hat{\boldsymbol{\mu}}_{\boldsymbol{\lambda}_i}}{\hat{\sigma}_{\boldsymbol{\lambda}_i}^2} &:= \sum_{k\in\mathcal{K}} \frac{|\mathcal{D}|}{K}\sigma_{h_i|f^{\mathbf{x}_k}}^{-2}\boldsymbol{\mu}_{h_i|f^{\mathbf{x}_b}} + \sum_{\mathbf{x}_l\in\mathbf{X}_n\setminus\mathbf{X}_{\mathcal{D}}} \sigma_{h_i|f^{\mathbf{x}_l}}^{-2}\boldsymbol{\mu}_{h_i|f^{\mathbf{x}_l}}\end{aligned}$$

where $\mathcal{K}$ is a mini-batch of size K sampled from $\{1,...,|\mathcal{D}|\}$, $\mathbf{x}_l \in \mathbf{X}_n \setminus \mathbf{X}_{\mathcal{D}}$ is a set of OOD samples sampled from $\mathcal{T}$ using c in Eq. 5.2.4, and $\boldsymbol{\mu}_{h_i|f^{\mathbf{x}_k}}$ and $\sigma_{h_i|f^{\mathbf{x}_k}}^2$ are the mean and variance parameter returned from $\tilde{q}_{\boldsymbol{\lambda}}(h_i|f(\mathbf{x}_k))$.

Proof To derive the mini-batch estimator, we first compute the expression for $\tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}_n})$. Since $\tilde{q}_{\boldsymbol{\lambda}}(\mathbf{h}|\mathbf{f}^{\mathbf{X}_n})$ is a product of Gaussian encoder, its mean and variance can be computed by:

$$\begin{aligned}\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}^{-1} &= \sum_{\mathbf{x}\in\mathbf{X}_n} \boldsymbol{\Sigma}_{h_i|f^{\mathbf{x}}}^{-1} \\ \boldsymbol{\mu}_{\boldsymbol{\lambda}} &= \boldsymbol{\Sigma}_{\boldsymbol{\lambda}} \sum_{\mathbf{x}\in\mathbf{X}_n} \boldsymbol{\Sigma}_{h_i|f^{\mathbf{x}}}^{-1}\boldsymbol{\mu}_{h_i|f^{\mathbf{x}}}\end{aligned}$$

where $\boldsymbol{\Sigma}_{h_i|f^{\mathbf{x}}}$ is a diagonal matrix with component $\left(\boldsymbol{\Sigma}_{h_i|f^{\mathbf{x}}}\right)_{ii} = \sigma_{h_i|f^{\mathbf{x}}}^2$. Let $\boldsymbol{\Sigma}_{\boldsymbol{\eta}}$ and $\boldsymbol{\mu}_{\boldsymbol{\eta}}$ be the covariance and mean of $q_{\boldsymbol{\eta}}(\mathbf{h})$. By our assumptions, $\boldsymbol{\Sigma}_{\boldsymbol{\eta}}$ is also a diagonal matrix with $(\boldsymbol{\Sigma}_{\boldsymbol{\eta}})_{ii} = \sigma_{\boldsymbol{\eta}_i}^2$. Since $\tilde{q}_{\boldsymbol{\lambda}}(\mathbf{h}|\mathbf{f}^{\mathbf{X}_n})\frac{q_{\boldsymbol{\eta}}(\mathbf{h})}{p_0(\mathbf{h})}$ is a product of three Gaussian distributions, its log normalization constant $\log\tilde{\mathcal{Z}}$ can be computed using the results from, for example Appendix

A.2 of [110]:

$$
\begin{aligned}
&\mathbb{E}_{n,\mathbf{X}_n\sim c}\mathbb{E}_{f\sim q_{\text{SPG}}(f)}\log\tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}_n})\\
=&\mathbb{E}_{n,\mathbf{X}_n\sim c}\mathbb{E}_{f\sim q_{\text{SPG}}(f)}\Big[\frac{1}{2}\log|\boldsymbol{\Sigma}_{\boldsymbol{\eta}}^{-1}|+\frac{1}{2}\log|\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}^{-1}|-\frac{1}{2}\log|\boldsymbol{\Sigma}_{\boldsymbol{\eta}}^{-1}+\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}^{-1}-I|\\
&-\frac{1}{2}\boldsymbol{\mu}_{\boldsymbol{\lambda}}^T\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}^{-1}\boldsymbol{\mu}_{\boldsymbol{\lambda}}-\frac{1}{2}\boldsymbol{\mu}_{\boldsymbol{\eta}}^T\boldsymbol{\Sigma}_{\boldsymbol{\eta}}^{-1}\boldsymbol{\mu}_{\boldsymbol{\eta}}+\frac{1}{2}(\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}^{-1}\boldsymbol{\mu}_{\boldsymbol{\lambda}}+\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}^{-1}\boldsymbol{\mu}_{\boldsymbol{\lambda}})^T(\boldsymbol{\Sigma}_{\boldsymbol{\eta}}^{-1}+\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}^{-1}-I)^{-1}(\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}^{-1}\boldsymbol{\mu}_{\boldsymbol{\lambda}}+\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}^{-1}\boldsymbol{\mu}_{\boldsymbol{\lambda}})\Big]\\
=&\frac{1}{2}\sum_{i=1}^{H}\mathbb{E}_{n,\mathbf{X}_n\sim c}\mathbb{E}_{f\sim q_{\text{SPG}}(f)}\Big[\log\sigma_{\boldsymbol{\eta}_i}^{-2}+\log\sigma_{\boldsymbol{\lambda}_i}^{-2}-\log(\sigma_{\boldsymbol{\eta}_i}^{-2}+\sigma_{\boldsymbol{\lambda}_i}^{-2}-1)\\
&-\boldsymbol{\mu}_{\boldsymbol{\lambda}_i}^2\sigma_{\boldsymbol{\lambda}_i}^{-2}-\boldsymbol{\mu}_{\boldsymbol{\eta}_i}^2\sigma_{\boldsymbol{\eta}_i}^{-2}+(\sigma_{\boldsymbol{\lambda}_i}^{-2}\boldsymbol{\mu}_{\boldsymbol{\lambda}_i}+\sigma_{\boldsymbol{\lambda}_i}^{-2}\boldsymbol{\mu}_{\boldsymbol{\lambda}_i})^2(\sigma_{\boldsymbol{\eta}_i}^{-2}+\sigma_{\boldsymbol{\lambda}_i}^{-2}-1)^{-1}\Big]
\end{aligned}
$$

where $\sigma_{\boldsymbol{\eta}_i}^2, \sigma_{\boldsymbol{\lambda}_i}^2, \boldsymbol{\mu}_{\boldsymbol{\eta}_i}, \boldsymbol{\mu}_{\boldsymbol{\lambda}_i}$ are the ith element of $\text{diag}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{\eta}}$, $\text{diag}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}$, $\boldsymbol{\mu}_{\boldsymbol{\eta}}$, $\boldsymbol{\mu}_{\boldsymbol{\lambda}}$, respectively. To effectively estimate $\sigma_{\boldsymbol{\lambda}_i}^{-2}=\sum_{\mathbf{x}\in\mathbf{X}_n}\sigma_{h_i|f^{\mathbf{x}}}^{-2}$ and $\boldsymbol{\mu}_{\boldsymbol{\lambda}_i}=\sigma_{\boldsymbol{\lambda}_i}^2\sum_{\mathbf{x}\in\mathbf{X}_n}\sigma_{h_i|f^{\mathbf{x}}}^{-2}\boldsymbol{\mu}_{h_i|f^{\mathbf{x}}}$, we can uniformly sample a mini-batch $\mathbf{X}_{\mathcal{K}}$ of size K from $\mathbf{X}_{\mathcal{D}}$, and then compute the following noisy mini-batch estimation:

$$
\begin{aligned}
\sigma_{\boldsymbol{\lambda}_i}^{-2}&=\sum_{\mathbf{x}\in\mathbf{X}_n}\sigma_{h_i|f^{\mathbf{X}_n}}^{-2}=N\mathbb{E}_{\mathbf{x}\in\mathbf{X}_{\mathcal{D}}}\sigma_{h_i|f^{\mathbf{x}}}^{-2}+\sum_{\mathbf{x}_l\in\mathbf{X}_n\backslash\mathbf{X}_{\mathcal{D}}}\sigma_{h_i|f^{\mathbf{x}_l}}^{-2}\\
&\approx\sum_{k\in\mathcal{K}}\frac{N}{K}\sigma_{h_i|f^{\mathbf{x}_k}}^{-2}+\sum_{\mathbf{x}_l\in\mathbf{X}_n\backslash\mathbf{X}_{\mathcal{D}}}\sigma_{h_i|f^{\mathbf{x}_l}}^{-2},\\
\boldsymbol{\mu}_{\boldsymbol{\lambda}_i}\sigma_{\boldsymbol{\lambda}_i}^{-2}&=\sum_{\mathbf{x}\in\mathbf{X}_n}\sigma_{h_i|f^{\mathbf{x}}}^{-2}\boldsymbol{\mu}_{h_i|f^{\mathbf{x}}}=N\mathbb{E}_{\mathbf{x}\in\mathbf{X}_{\mathcal{D}}}\sigma_{h_i|f^{\mathbf{x}}}^{-2}\boldsymbol{\mu}_{h_i|f^{\mathbf{x}}}+\sum_{\mathbf{x}_l\in\mathbf{X}_n\backslash\mathbf{X}_{\mathcal{D}}}\sigma_{h_i|f^{\mathbf{x}_l}}^{-2}\boldsymbol{\mu}_{h_i|f^{\mathbf{x}_l}}\\
&\approx\sum_{k\in\mathcal{K}}\frac{N}{K}\sigma_{h_i|f^{\mathbf{x}_k}}^{-2}\boldsymbol{\mu}_{h_i|f^{\mathbf{x}_k}}+\sum_{\mathbf{x}_l\in\mathbf{X}_n\backslash\mathbf{X}_{\mathcal{D}}}\sigma_{h_i|f^{\mathbf{x}_l}}^{-2}\boldsymbol{\mu}_{h_i|f^{\mathbf{x}_l}},
\end{aligned}
$$

We denote the estimators for $\sigma_{\boldsymbol{\lambda}_i}^{-2}$ and $\boldsymbol{\mu}_{\boldsymbol{\lambda}_i}$ by $\hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2}$ and $\hat{\boldsymbol{\mu}}_{\boldsymbol{\lambda}_i}$, respectively. Then, applying these noisy estimations to $\mathbb{E}_{n,\mathbf{X}_n\sim c}\mathbb{E}_{f\sim q_{\text{SPG}}(f)}\log\tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}_n})$, we have

$$
\begin{aligned}
&\mathbb{E}_{n,\mathbf{X}_n\sim c}\mathbb{E}_{f\sim q_{\text{SPG}}(f)}\log\tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}_n})\\
\approx&\frac{1}{2}\sum_{i=1}^{H}\mathbb{E}_{f\sim q_{\text{SPG}}(f)}\Big[\log\sigma_{\boldsymbol{\eta}_i}^{-2}+\log\hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2}-\log(\sigma_{\boldsymbol{\eta}_i}^{-2}+\hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2}-1)\\
&-\hat{\boldsymbol{\mu}}_{\boldsymbol{\lambda}_i}^2\hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2}-\boldsymbol{\mu}_{\boldsymbol{\eta}_i}^2\sigma_{\boldsymbol{\eta}_i}^{-2}+(\hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2}\hat{\boldsymbol{\mu}}_{\boldsymbol{\lambda}_i}+\hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2}\hat{\boldsymbol{\mu}}_{\boldsymbol{\lambda}_i})^2(\sigma_{\boldsymbol{\eta}_i}^{-2}+\hat{\sigma}_{\boldsymbol{\lambda}_i}^{-2}-1)^{-1}\Big],\mathbf{X}_n\sim c(\mathcal{T}^{\mathbb{Z}^+})
\end{aligned}
$$

The symbol $\approx$ in the last line means that it is a consistent estimator, due to multivariate continuous mapping theorem. $\square$

5.A.7 Proof of Proposition 5.7

Proposition 5.6 (Debiasing)**.** *Let R be a random integer from a distribution $\mathbb{P}(N)$ that has support over the integers larger than K, and $\mathbf{x}_0$ is a random location sampled from $\mathcal{T}$. Then $\mathbb{E}_{n,\mathbf{X}_n\sim c}\mathbb{E}_{f\sim q_{SPG}(f)}\log\tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}_n})$ can be estimated by:*

$$\mathbb{E}\left[\mathcal{J}_K+\sum_{k=K}^{R}\frac{\Delta_k}{\mathbb{P}(N\geq k)}\right] \tag{5.A.6}$$

where $\Delta_k=\mathcal{J}_{\mathbf{k+1}}-\mathcal{J}_{\mathbf{k}}$, and the expectation $\mathbb{E}$ is taken over R, n, $\mathbf{X}_n$, and all mini-batches used by each $\mathcal{J}_k$ terms.

Proof By definition, we have $\lim_{k\to\infty}\mathbb{E}\mathcal{J}_K=\mathbb{E}\mathcal{J}_N=\mathbb{E}_{n,\mathbf{X}_n\sim c}\mathbb{E}_{f\sim q_{\text{SPG}}(f)}\log\tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}_n})$. Apparently, $\mathbb{E}\sum_{k=0}^{R}\frac{\Delta_k}{\mathbb{P}(N\geq n)}$ constructs an Russian Roulette estimator [141]. Based on lemma 3 from [41], in order prove our result we only have to show that $\mathbb{E}\sum_{k=0}^{\infty}|\Delta_k|<\infty$. In fact, since the data distribution is assumed to be an empirical distribution, we have

$$\begin{aligned}\sum_{k=0}^{\infty}|\Delta_k|&=\sum_{k=0}^{\infty}|\mathcal{J}_{\mathbf{k+1}}-\mathcal{J}_{\mathbf{k}}|\\&=\sum_{k=0}^{N-1}|\mathcal{J}_{\mathbf{k+1}}-\mathcal{J}_{\mathbf{k}}|<\infty\end{aligned}$$

holds for all possible mini-batches used by each Δ_k. The second equality is based on the fact that $\mathcal{J}_{\mathbf{k+1}}=\mathcal{J}_{\mathbf{k}}=\log\tilde{\mathcal{Z}}(\mathbf{f}^{\mathbf{X}_n})$ for all $k\geq N$. Therefore, we have $\mathbb{E}\sum_{k=0}^{\infty}|\Delta_k|=\sum_{k=0}^{N-1}\mathbb{E}|\Delta_k|<\infty$.

Appendix 5.B Further details of experiments

5.B.1 General settings

Data split/Cross-validation schemes For UCI experiments, each dataset was randomly split into train (90%) and test sets (10%). This was repeated 10 times. In contextual bandits, we used the code open-sourced by [330], therefore the data sampling process described in [330] was exactly executed. For classification experiment for MNIST and CIFAR 10, since the train/test set are predefined, we have only run experiments with 5 different random seeds (for initialization). For interpolation with implicit prior experiment, see 5.B.2 for details.

Choice of sampling distribution 3 One example of c that satisfies the requirement of Propositions 5.1, 5.2, and 5.3 takes the following form:

$$\begin{aligned} &(n-|\mathcal{D}|) \sim \text{Geom}(p), \mathbf{x}_k \sim \mathcal{U}(\mathcal{T}), \ \ \forall 1 \leq k \leq n-|\mathcal{D}|, \\ &\mathbf{X}_n := \mathbf{X}_{\mathcal{D}} \bigcup \{\mathbf{x}_k\}_{1\leq k \leq n-|\mathcal{D}|} \end{aligned} \tag{5.B.1}$$

where we first sample n from a geometric distribution, such that $(n-|\mathcal{D}|) \sim \text{Geom}(p)$ with parameter p. Here, we use the parameter $p = 0.5$. Then, $(n-|\mathcal{D}|)$ out of distribution (OOD) measure points are sampled independently from a uniform distribution on $\mathcal{T}$.

Construction of the compact space $\mathcal{T}$ The construction of $\mathcal{T}$ depends on specific tasks. If we know the range of the input , we can directly set to be such interval. For example, in synthetic datasets of Experiment 5.7.1, we already know that the input lies in the interval between 0 and 1, therefore $\mathcal{T} = [0,1]$. If we don't know the range of the inputs, then we can simply set $\mathcal{T}$ to be a hyperrectangle, with each $x_i \in [x^i_{min}, x^i_{max}]$, where x^i_{min} and x^i_{max} are the empirical min and max of the i-th variable of input dataset.

Choice of prior processes $p(f)$ Note that since FVI is an *inference method* instead of a new *model*, we will assume that FVI and most of the baselines will be using the same priors, whenever applicable. For example, in the interpolation structured prior tasks, both FVI and f-BNN will use the same piecewise implicit prior. In multivariate regression and image classification tasks, all algorithms will use the same BNN prior with the same structures, therefore we can isolate the difference caused by inference algorithms.

Structure of SPGs Unless specified otherwise, we use 10 basis functions for our SPGs, and each basis function is a two-layer neural network that maps from R^d to R^1. The structure of these networks is input-100-100-output. Note that these neural network parameters are *not* part of the variational parameters, since they are frozen forever after we have finished distilling $p(f)$ using $\tilde{p}_{\text{SPG}}(f)$. To further reduce the number of free parameters, the parameters of the first two layers of all basis functions can be shared (this is applied only to larger scale experiments such as image classification). The encoder $\tilde{q}_{\boldsymbol{\lambda}}(\mathbf{h}|f)$ for $\tilde{p}_{\text{SPG}}(f)$ is also a two-layer neural network (input-500-200-latent statistics), whose parameters are also fixed after distilling $p(f)$. The decoders also have two hidden layers (latent variables-50-100-output). The latent dimension is different depending on the tasks so that the comparison between baselines will be fair. This will be detailed later. For the stationary GP white noise process used in SPGs, we assume that they have isotropic noise level $\sigma_v^2 = 0.1$.

Optimization Unless otherwise specified, we use Adam optimizer with learning rate $lr = 0.001$. We use a slightly larger learning rate in the contextual Bandit experiment since the learning rates used for each baseline is tuned from $[0.001, 0.05]$, as specified in the experiment section. When training $\tilde{p}_{\mathrm{SPG}}(f)$, we use 5k epochs unless otherwise noticed. For the inference phase where $q_{\mathrm{SPG}}(f)$ is optimized to maximize the functional ELBO, the number of iterations is determined by the other baselines. For example in contextual Bandits, all baselines are trained for 100 epochs, so is FVI. In terms of batch size, unless specified otherwise, we choose the batch size to be 100. This batch size is also used to perform MC estimation of the likelihood term in Equation 5.5.3, and training $\tilde{p}_{SPG}$ in Equation 5.4.2.

Hyperparameters of the likelihood function $p_\pi(y|f)$ Regarding the likelihood function $p_\pi(y|f)$, since we only deal with continuous outputs in this chapter, we simply choose $p_\pi(y|f)$ to be a Gaussian likelihood with noise standard deviation σ and mean $\boldsymbol{\mu} = 0$ just like all the other baselines. The value of σ is set to be 0.02 except for multivariate regression experiments, since we follow the setting of [198], where the noise variance is determined individually for all BNN baselines, including FVI. For FVI, we found that making the mean parameter $\boldsymbol{\mu}$ of $p_\pi(y|f)$ to be optimizable will accelerate the convergence (which is equivalent to adding an additional basis/bias in SPGs).

5.B.2 Individual settings for each experiments

Interpolation with structured implicit priors To sample a random function from piecewise constant priors, we first sample the number of change points n from a Poisson distribution Poisson(3). The exact location of each change point is uniformly sampled from $[0,1]$. Then, for each interval specified by the change points, we sample $n+1$ function values uniformly to specify constant function values in each interval. This results in piecewise constant functions. For the piecewise linear prior, the function values are sampled similarly and we draw straight lines to connect each function value. In this experiment, for FVI, we draw 1k samples from the implicit priors that are used to optimize the FVI parameters. The basis function for FVI are directly obtained by sampling from the implicit prior. We use 200 basis functions, with a latent dimension of 10. To train our $\tilde{p}_{\mathrm{SPG}}(f)$, we sample 1k random function samples and optimize $\tilde{p}_{\mathrm{SPG}}(f)$ for 5k epochs. For inference, the variational parameters are trained for 1k epochs.

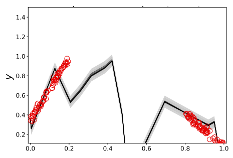
UCI Multivariate regression For this experiment, we follow [198]. The functional prior is a fully connected ReLU BNN with two hidden layers (input-10-10-output). We train FVI for 1k epochs. We use 10 basis functions for FVI and a batch size of 100. The latent size is set to 100.

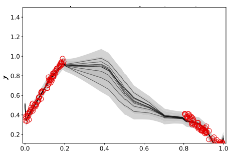
Gaussian Processes in UCI regression On UCI datasets, variational sparse GPs and exact GPs are implemented using GPflow. VSGPs uses 50 inducing points. Both variations of GP models use the RBF kernel.

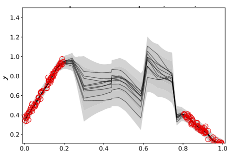
VIPs, Bayes-by-Backprop, variational dropout, α-dropout for Bayesian neural networks on UCI VIP and Weight-space inference methods for BNNs are based on the same implementations used in Chapter 4. For details of othese, readers may refer to Appendix of Chapter 4. Results in Table 5.1 are also taken from Chapter 4, as they have used exactly the same data split scheme and BNN prior structures.

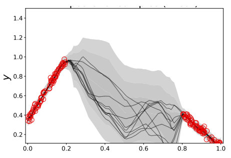
Contextual Bandits We use similar settings to [330], where we use a batchsize = 32, training epochs = 100, training frequency = 50, and contexual points = 2000. We use ReLU BNNs as functional priors. It has two hidden layers, each with 50 hidden units. For FVI, we use 100 basis functions (with shared weights until the last layer), a learning rate of 0.005. For details of the algorithms mentioned in Table 5.2, readers may refer to [274] for details. Here we briefly explain the meaning of the abbreviations. **FVI**: functional variational inference; **FBNN**: functional Bayesian neural networks; **Uniform**: uniform sampling; **RMS**: trains a neural network and acts greedily using RMSprop; **Boot RMS**: Bootsrapped RMS; **Neural Linear**: Bayesian linear regression over deep NN features; **ParamNoise**: just a regular DNN, but when making decisions, an isotropic Gaussian perturbation is added to the NN weights; **Dropout**: variational dropout BNNs; **BBB**: Bayes-by-Backprop BNNs; **BB α**: Black-box alpha divergence minimization.

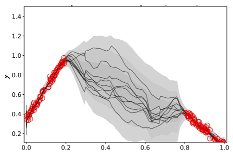
Image classification and OOD detection For all models in this experiment, the CNN structures are the same as in [127]. That is, the 3 convolutional layers plus 3 fully connected layers in the DeepOBS benchmark [300]. Similar to [127], we apply standard isotropic Gaussian prior on all weight parameters. We use Adam with learning rate of 0.001 and

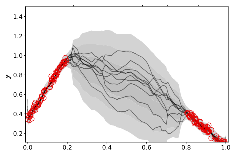

(a) VIP, 5 basis functions

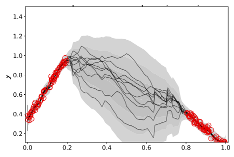

(b) VIP, 10 basis functions


(c) VIP, 20 basis functions


(d) VIP, 50 basis functions


(e) VIP, 100 basis functions


(f) VIP, 150 basis functions


(g) VIP, 200 basis functions

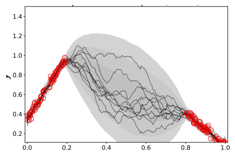

(h) VIP, 500 basis functions

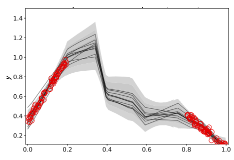
Figure 5.8 The posterior samples from VIPs with different number of basis functions. As more basis functions are used, the posterior samples from VIP become more and more noisy, and finally converges to GP-like behaviour when 500 basis functions are used. Compared to the ground truth estimate from Figure 5.6, VIP clearly under-estimates the predictive uncertainties in-between the training samples.

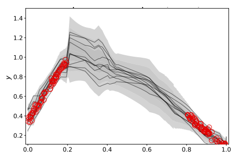
batch size of 100, and run the training procedure for 100 epochs. For FVI, we use 100 basis functions in the SPGs on both datasets. Note that each basis function is a there-layer convolutional network that maps from R^d to R^{10}. To significantly reduce the memory usage, the parameters of the convolutional layers of all basis functions are shared.

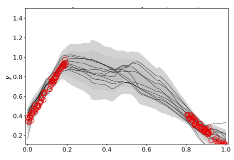
Appendix 5.C Additional Experiments

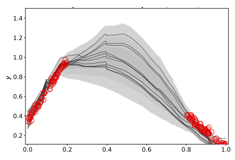
5.C.1 Impact of number of basis functions on SPGs and VIPs

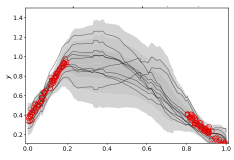
As discussed in Section 5.3, the SPGs used the proposed FVI can be treated as a non-Gaussian extension of the VIP variational family, by removing the Gaussian assumption on $\mathbf{a}$. One natural question that arises in this setting will be that, does the advantage of FVI over VIP vanish as the number of basis function increases? How does the number of basis functions affect the performance of each method? To provide more intuition for FVI and VIPs, we consider again the 1-D function interpolation task with piecewise-linear implicit prior. In Figure 5.8 and Figure 5.9, we show how the posterior samples from FVI and VIPs evolve when the number of basis functions gradually increase from 5 to 500. Note that for a fair comparison, the basis functions for both FVI and VIP are obtained by drawing random function samples from the implicit prior. Since the piecewise-linear implicit functional prior

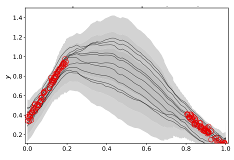

(a) FVI, 5 basis functions

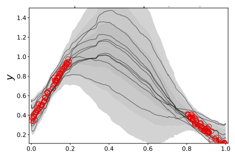

(b) FVI, 10 basis functions


(c) FVI, 20 basis functions


(d) FVI, 50 basis functions


(e) FVI, 100 basis functions


(f) FVI, 150 basis functions


(g) FVI, 200 basis functions

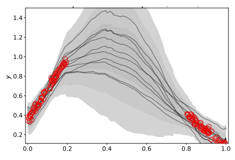
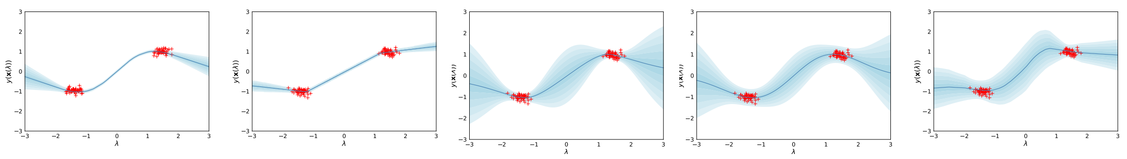

(h) FVI, 500 basis functions

Figure 5.9 The posterior samples from FVI with different number of basis functions. FVI is still able to learn the piecewise linear behaviour from the prior as more basis functions are used. As the number of basis functions is increased to 500, FVI converges to a solution that is much closer to the ground truth (compared with VIP), and is still able to exhibit non-Gaussian behaviours from the prior.

is not reparameterizable, once the basis functions for FVI and VIP are sampled, they will be frozen forever (in contrast, when the prior is reparameterizable, both FVI and VIP can optimize the basis functions, therefore the number of basis functions required will be much smaller than this experiment).

From Figure 5.8 and Figure 5.9, we can first observe that as the number of basis function increases, the predictive uncertainty of both FVI and VIP also increase, until around when 200 basis functions are reached. However, as more basis functions are used, the posterior samples from VIP become noisier, and finally converges to GP-like behavior when 500 basis functions are used. Compared to the ground truth estimate from Figure 5.6, VIP under-estimates the predictive uncertainties in-between the training samples. This is due to that the piecewise linear behavior of the function samples violates the Gaussian assumption of VIP, such that the correlation level between points will be lower than expected. On the other hand, FVI is still able to learn the piecewise linear behavior from the prior as more basis functions are used. As the number of basis functions is increased to 500, FVI converges to a solution that is much closer to the ground truth (compared with VIP) and is still able to exhibit non-Gaussian behaviors from the prior. We can conclude that the advantage of FVI over VIP does not vanish as the number of basis functions increases. In contrast, the difference between FVI and VIP becomes even more distinct and recognizable.



(a) functional BNN (b) MFVI BNN (c) GP (d) HMC (e) Ours

Figure 5.10 A regression task on a synthetic dataset (red crosses) reproduced from [70]. We plot predictive mean and uncertainties for each algorithms. This tasks is used to demonstrate the theoretical finding on the pathologies of weight-space VI for single-layer BNNs: there is *no* setting of the variational parameters that can model the in-between uncertainty between two data clusters. The functional BNNs [330] also has this problem, since BNNs are use as part of the model. On the contrary, our functional VI method can produce sensible in-between uncertainties for out-of-distribution data. See Appendix 5.C.2 for more details.

5.C.2 On in-between uncertainty pathologies of BNNs

In figure 5.10, we presented a 2-D regression tasks on a synthetic dataset (red crosses), reproduced from [70]. This tasks is used to demonstrate the pathologies of weight-space inference for single-layer BNNs (including f-BNNs where BNNs are use as part of the model): there is *no* setting of the variational parameters that can model the in-between uncertainty between two data clusters. To be concrete, we have the following proposition:

Proposition 5.7 (Limitations for single-hidden layer BNNs [70])**.** *Consider any single-hidden layer fully-connected ReLU NN $f : \mathbb{R}^D \to \mathbb{R}$. Let x_d denote the d^{th} element of the input vector $\mathbf{x}$. Suppose we have a fully factorised Gaussian distribution over the weights and biases in the network. Consider any points $\mathbf{p}, \mathbf{q}, \mathbf{r} \in \mathbb{R}^D$ such that $\mathbf{r} \in \overrightarrow{\mathbf{pq}}$ and either:*

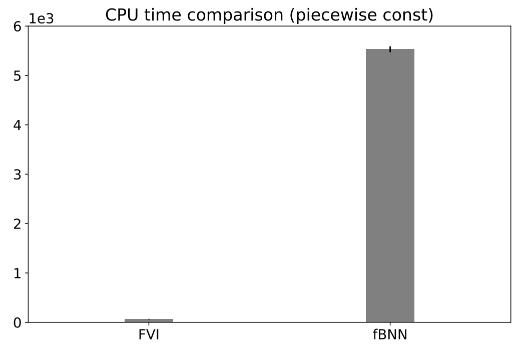
1. *$\overrightarrow{\mathbf{pq}}$ contains $\mathbf{0}$* and *$\mathbf{r}$ is closer to $\mathbf{0}$ than both $\mathbf{p}$ and $\mathbf{q}$.*
2. *$\overrightarrow{\mathbf{pq}}$ is orthogonal to and intersects the plane $x_d = 0$,* and *$\mathbf{r}$ is closer to the plane $x_d = 0$ than both $\mathbf{p}$ and $\mathbf{q}$.*

Then $\mathrm{Var}[f(\mathbf{r})] \leq \mathrm{Var}[f(\mathbf{p})] + \mathrm{Var}[f(\mathbf{q})]$.

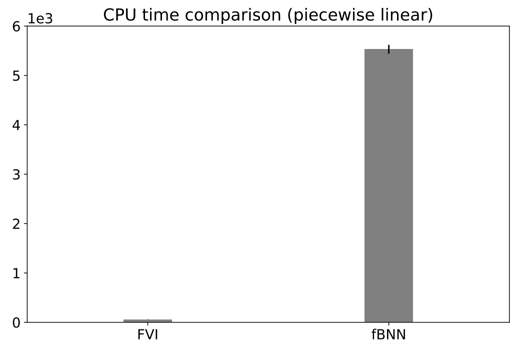
That is, the weight space inference of a single-hidden layer variational BNNs (using mean-field VI) fails to represent the in-between uncertainty, and become over-confident on out-of-distribution data. In this experiment, the training data is sampled as follows: the 2-D input locations of training data are generated by sampling 100 points, 50 each from two separate clusters that follow Gaussian distributions. The inputs of the cluster on the left of Figure 5.2 around $(-1,-1)$, and the other cluster is centered around $(1,1)$. Both have isotropic Gaussian noise with zero mean and variance 0.01. The outputs (y) are -1 and 1 for the left and right clusters, respectively. We further add a Gaussian observational noise of variance 0.1 to the outputs. To test whether the baselines can learn the in-between uncertainties between clusters, we use a fully connected ReLU BNN of a single hidden layer (50 units). The FVI also uses this prior as functional prior and has 50 basis functions and 50 latent dimensions. The settings of MFVI are determined according to [70]. The settings of F-BNN are determined similarily.

In figure 1, the $\boldsymbol{\lambda}$ axis is the 1-D parameter that parameterizes the 1-dimensional straight line embedded in the 2-D plane, that connects $(-3,-3)$ and $(3,3)$. The value of the $\boldsymbol{\lambda}$-coordinate implies that its actual 2-D coordinate in the 2-D plane is $(\boldsymbol{\lambda},\boldsymbol{\lambda})$. The results in Figure 5.2 show that both Mean-field variational BNN and functional BNN suffers from the limitations of single hidden layer BNN. On the contrary, FVI can produce a sensible in-between uncertainty that is similar to GPs and HMC. For GPs, we use infinite-width BNN kernel following [70]

5.C.3 CPU time comparison, FVI vs f-BNN on implicit priors



(a) CPU time (s), piecewise constant prior



(b) CPU time (s), piecewise linear prior

Figure 5.11 CPU time comparison, FVI vs f-BNN on implicit priors. Although f-BNNs are only trained for 100 epochs, its running time is still 100x slower than FVI.

5.C.4 CPU time comparison, FVI vs f-BNN on Census

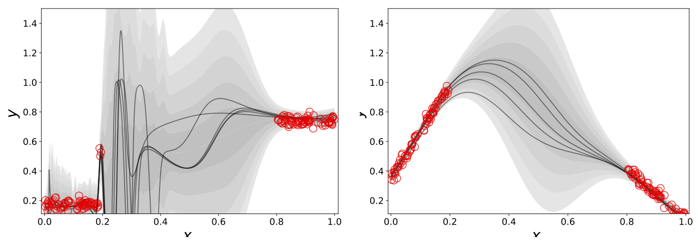
In order to compare the efficiency between FVI and f-BNN, we provide the CPU time comparison of running contextual bandits on Census dataset [12], one of the largest contextual bandits dataset that we have tested. It has more than 2 million data points, each with 389 dimensional features as input (including dummy binary variables for categorical variables). The output has 9 different classes (actions). The CPU time consumed by each algorithm on Census is listed as follows:

Table 5.4 CPU time performance comparison of running contextual bandit on Census dataset

	FVI	f-BNN	BBB
Run time (s)	28.14 $\pm$ 2.604	9648.19 $\pm$ 957.3	19.98 $\pm$ 1.238

Based on Table 5.4, we can see that FVI is nearly 500 times faster than f-BNN. The run time of FVI is similar to Bayes-by-Backprop, indicating that FVI is very efficient and scalable.

5.C.5 Improved results of f-BNN on implicit priors



(a) F-BNN, piecewise constant prior (b) F-BNN, pieacewise linear prior

Figure 5.12 F-BNN on structured implicit priors, trained with 10k epochs

In experiment 5.7.1, we have only run f-BNNs for 100 epochs due to its computational costs. Here, we provide improved results of fully-trained f-BNNs after 10k epochs. Note that this epoch number is much larger than the FVI setting (5k), since we found that after 5k epochs, the f-BNN posteriors do not seem to improve over the results in experiment 5.7.1. As shown in Figure 5.12, after 10k epochs, the posterior uncertainty estimates of f-BNNs become much loser to the ground truth in Figure compared with its 100 epochs version. However, this comes with the cost of significantly increased computational time. Moreover, f-BNNs seem to provide less convincing posterior samples in terms of mimicking the piece-wise constant/linear behaviours.

5.C.6 Large scale experiments on deeper Bayesian neural networks

To demonstrated the scalability and applicability of FVIs to larger datasets and deeper Bayesian neural networks, in this section we perform regression experiments using a Bayesian DNN with 5 hidden layers of 100 units. We compare our results to f-BNNs and BBB with the same DNN structure, which are cited directly from [330]. For fair comparison, we increase the size of the basis functions used by SPGs to neural networks with 5 hidden units and 100 units. An additional hidden layer is added to the decoder and encoder of the VAE used by SPGs. We follow the settings of Section 5.7.2, except that we train FVI using 30000 iterations using mini-batch stochastic optimization. We report results on Naval dataset, protein datset, and GPU dataset. We also include results of FVI on shallow networks used in Section 5.7.2. From Table 5.6, we notice that the performance of FVI on deeper networks is generally competitive to f-BNNs and BBBs, indicating that FVI is scalable to larger datasets and deeper neural networks.

5.C.7 Comparison to function space particle optimization (f-SVGD) and GPs

In this section, we further compare FVI to function space particle optimization (f-SVGD) and GPs:

Table 5.5 Regression experiment: Average test negative log likelihood

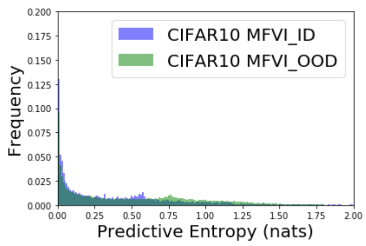
Dataset	N	**FVI**	**f-SVGD**	**GP**
boston	506	2.33±0.04	**2.30±0.05**	2.63±0.04
concrete	1030	**2.88±0.06**	2.90±0.02	3.4±0.01
energy	768	**0.58±0.05**	0.69±0.03	2.31±0.02
kin8nm	8192	**-1.15±0.01**	-1.11±0.01	-0.76±0.00
power	95684	**2.69±0.00**	2.73±0.00	2.82±0.00
protein	45730	**2.85±0.00**	2.85±0.00	3.01±0.00
red wine	1588	0.97±0.06	**0.89±0.01**	0.98±0.02
yacht	308	**0.59±0.11**	0.75±0.01	2.29±0.03
naval	11934	-7.21±0.06	-4.82±0.10	**-7.81±0.00**

Note that f-SVGD is not included in our main experiments in Table 5.1, since it is a particle optimization-based inference method. On the other hand, GP is not included since it is not a BNN-based model. For GPs, we used variational sparse GP with 50 inducing points plus an RBF kernel. The additional results in Table 5.5 shows that FVI performs the best in 6 out of 9 datasets. Moreover, FVI outperforms f-SVGD in 6 out of 9 datasets and outperforms GP in 8 out of 9 datasets in terms of NLLs.

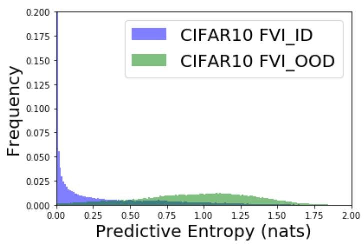
Table 5.6 larger scale regression experiment: Average test negative log likelihood

Dataset	N	FVI	f-BNNs	BBB	FVI shallow
GPU	241600	**2.93±0.03**	2.97±0.02	2.99±0.01	3.10±0.04
Protein	45730	2.82±0.01	2.72±0.01	**2.72±0.01**	2.85±0.00
Naval	11934	**-7.42±0.01**	-7.24±0.01	-6.96±0.01	-7.38±0.04

5.C.8 Out-of-distribution detection visualization on CIFAR10



(a) MFVI - Bayesian CNN



(b) FVI - Bayesian CNN

Figure 5.13 Histograms of predictive entropies on CIFAR10/SVHN OOD detection. Left: MFVI. Right: FVI.

Part B

Unsupervised learning and decision making under missing data uncertainty

Chapter 6

Overview: missing data uncertainty, decision making, and identifiability

In Part A, we presented our novel contributions to quantify model uncertainty in supervised learning problems. However, most data generated from the real world comes with no labels. Most of the time, we are only able to observe unlabelled datasets, $\mathcal{D} = \{\mathbf{x}_n\}_{1 \leq n \leq N}$. It is therefore, very crucial for us to be able to make sense of such unlabeled data efficiently.

Thus in Part B, we now move into the field of *unsupervised learning*, and address the research question of performing (unsupervised) learning, inference, and high-value information acquisition under the presence of missing data uncertainty (i.e., Challenges II and III in Chapter 1). Recall that in Chapter 2, Section 2.3.4, we introduced the idea that unsupervised learning can be naturally performed by (deep) generative models, which take the following form:

$$\log p_{\boldsymbol{\theta}}(\mathcal{D}) = \sum_n \log \int_{\mathbf{z}_n} p_{\boldsymbol{\theta}}(\mathbf{x}_n|\mathbf{z}_n) p(\mathbf{z}_n) d\mathbf{z}_n. \tag{6.0.1}$$

From the perspective of uncertainty, the latent variable $\mathbf{z}_n$ (or more precisely, the posterior $p_{\boldsymbol{\theta}}(\mathbf{z}_n|\mathbf{x}_n)$) quantifies the data uncertainty of each data point $\mathbf{x}_n$ (e.g., if θ is sufficient statistics and if the model complexity is correct). This type of uncertainty is categorized as the *aleatoric uncertainty*, since $\mathbf{z}_n$ cannot be reduced by collecting more data. This is because each data instance $\mathbf{x}_n$ is assigned with different latent variables $\mathbf{z}_n$, and $p_{\boldsymbol{\theta}}(\mathbf{z}_n|\mathbf{x}_n, \mathbf{x}_m) = p_{\boldsymbol{\theta}}(\mathbf{z}_n|\mathbf{x}_n)$ for all $m \neq n$.

Using the methods (variational EM/amortized VI/wake-sleep) introduced in Chapter 2, learning and inference of $\log p_{\boldsymbol{\theta}}(\mathcal{D})$ can be performed efficiently. However, all these methods are built upon an important assumption: the data set $\mathcal{D} = \{\mathbf{x}_n\}_{1 \leq n \leq N}$ must be fully observed. Unfortunately, in many real-world applications, this assumption does not hold: when we collect datasets, they often contain missing data. This can be caused by human errors, physical constraints, non-response, etc. Since we are uncertain regarding the missing entries in our dataset, the failure to account for this uncertainty may compromise the performance of machine learning models, as well as downstream tasks based on these models. Therefore, it is important to be able to perform learning and inference under missing data and quantify the uncertainties caused by missing data.

6.1 Missing mechanism, and traditional methods for handling missing data

Suppose that we have a partially observed dataset, $\mathcal{D}_O = \{\mathbf{x}_O^{(n)}\}_{1\leq n\leq N}$, where $\mathbf{x}_O^{(n)}$ denotes the partially observed subset of features of the n-th data point, and $\mathbf{x}_U^{(n)}$ denotes the unobserved variables. Furthermore, let $\mathbf{r}^{(n)}$ denote the missing mask indicator. $r_i^{(n)} = 1$ indicates x_i is observed, and $r_i^{(n)} = 0$ indicates x_i is missing. The conditional distribution $p(\mathbf{r}|\mathbf{x})$ is called the missing mechanisms. Depending on the dependency structure between X and R, there are three types of missing mechanisms [279]:

1. If $p(\mathbf{r}|\mathbf{x}) = p(\mathbf{r})$, the data is *missing completely at random* (MCAR);
2. If $p(\mathbf{r}|\mathbf{x}) = p(\mathbf{r}|\mathbf{x}_o)$, the data is *missing at random* (MAR). That is, the cause of missingness $\mathbf{r}$ is observed;
3. Otherwise, the data is missing not at random (MNAR). That is, the cause of missingness is unobserved.

MCAR and MAR are the most used assumptions in practice due to their technical simplicity: under MCAR and MAR, we can ignore the missing mechanism and only focus on $p(\mathbf{x})$ [279]. However, MCAR and MAR do not hold in many real-world applications where, by contrast, MNAR mechanisms are more common. In MNAR scenarios, we must explicitly model the joint distribution $p(\mathbf{x},\mathbf{r})$. In Chapter 8, we will discuss these assumptions in more details.

Methods for handling missing data have been extensively studied in the past few decades. These methods can be roughly classified into two categories: complete case analysis (CCA) based methods and imputation based methods. CCA-based methods, such as listwise deletion [5] and pairwise deletion [209] directly delete data instances that contain missing entries and only keep those that are complete for subsequent data analysis. Listwise/pairwise deletion methods are known to be unbiased under MCAR and will be biased under MAR/MNAR. On the contrary, imputation-based methods try to replace missing values with imputed/predicted values. One popular imputation technique is called single imputation, where only one single set of imputed values for each data instance is produced. Standard techniques of single imputation include mean/zero imputation, regression-based imputation [5] and non-parametric methods [143, 325].

Unfortunately, single imputation methods only return point estimates of the missing values. Hence, they cannot quantify the missing data uncertainty. As opposed to single imputation, multiple imputation (MI) methods [280, 281, 119, 237] such as MICE [362] are simulation-based methods that return multiple imputation values for subsequent statistical analysis. Unlike single imputation, the standard errors of estimated parameters produced with MI are known to be unbiased [282]. Apart from MI, there exist other methods such as full information maximum likelihood [9, 67] and inverse probability weighting [277, 120], which can be directly applied to MAR without introducing additional bias. However, these methods assume a MAR missing data mechanism, and cannot be directly applied to MNAR without introducing bias.

6.2 Generative models for missing data uncertainty and decision making

Deep generative models (Chapter 2 Section 2.3.4) are flexible and scalable probabilistic models that excel at density estimation tasks. Thus, they are a natural choice for modelling missing data uncertainties at scale. If we have an accurate generative model $p_{\boldsymbol{\theta}}(\mathbf{x})$ for the complete data, then in theory, we can infer the posterior distribution on the rest of the missing variables, i.e., $p_{\boldsymbol{\theta}}(\mathbf{x}_U|\mathbf{x}_O)$. However, there are several technical difficulties:

1. **Learning.** How can we efficiently estimate the parameters of the complete data model $\log p_{\boldsymbol{\theta}}(\mathbf{x})$ (or $\log p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{r})$ under MNAR), given only $\mathcal{D}_O = \{\mathbf{x}_O^{(n)}\}_{1 \leq n \leq N}$? Furthermore, can this be done in the large data/large model regime?

2. **Inference.** How can we quantify the missing data uncertainty? That is, given a partially observed $\mathbf{x}_O$, how can we compute $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}_O)$? Equivalently, how can we perform missing data imputation, $p_{\boldsymbol{\theta}}(\mathbf{x}_U|\mathbf{x}_O)$? There are many possible partitions of complete data $\mathbf{x}$ into $\mathbf{x}_O$ and $\mathbf{x}_U$. For d dimensional observable variables, there exists 2^d different combinations of the observed subset, $\mathbf{x}_O$. Therefore, there are 2^d different

posterior distributions of the form $p_{\boldsymbol{\theta}}(\mathbf{x}_U|\mathbf{x}_O)$; performing Bayesian inference for each different combination presents a significant computational challenge. How can we efficiently address this challenge?

3. **Decision making.** The aleatoric uncertainty (represented by $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$) can not be fully eliminated by collecting more data points. However, the missing data uncertainty (represented by $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}_O)$) is still partially reducible in the sense that, if we are able to actively observe more features from *the same data point* $\mathbf{x}_O$, the posterior $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}_O)$ will finally approach the complete data posterior $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$. As a matter of fact, in many applications, it is possible to acquire additional information (sometimes at specific costs). For example, in assessing the health status of a patient we may decide to take additional measurements such as diagnostic tests or imaging scans before making a final assessment.

   Therefore, this brings up an interesting decision making problem: suppose that in a given prediction task we are interested in some task-specific variables $\mathbf{x}_\phi \subset \mathbf{x}_U$ (which we call the target variables). Then, can we optimally choose which feature $\mathbf{x}_i \in \mathbf{x}_U \setminus \mathbf{x}_\phi$ to observe next, so that $\mathbf{x}_i \in \mathbf{x}_U \setminus \mathbf{x}_\phi$ is provides the most informative knowledge about the target $\mathbf{x}_\phi$? Or equivalently, can the missing data uncertainty of $\mathbf{x}_\phi$, represented as $p_{\boldsymbol{\theta}}(\mathbf{x}_\phi|\mathbf{x}_O)$, be maximally reduced? [1] If these questions can be answered, we expect that the decision quality of $\mathbf{x}_\phi$ can be improved (evaluated by certain metrics such as likelihood and/or accuracy).

As analyzed in Chapter 1 Section 1.3, all these research questions are important not only for technical advancements of generative models but also for building practical systems to replicate human expert's decision-making behaviors. In Chapter 7, we will first work under the assumptions of MCAR and MAR, and present our original contributions to these research questions in the context of unsupervised learning and active information acquisition.

6.3 Model non-identifiability in unsupervised learning under missing data

In Chapter 3, we investigated the notion of model identifiability and its implications for Bayesian inference under supervised learning settings. We argued that model unidentifiability/ overparameterization is one of the major obstacles for improving the quality of

[1]When there isn't a specific target variable $\mathbf{x}_\phi$, we may still select the most informative feature by reducing $p_{\boldsymbol{\theta}}(\mathbf{x}_U|\mathbf{x}_O)$.

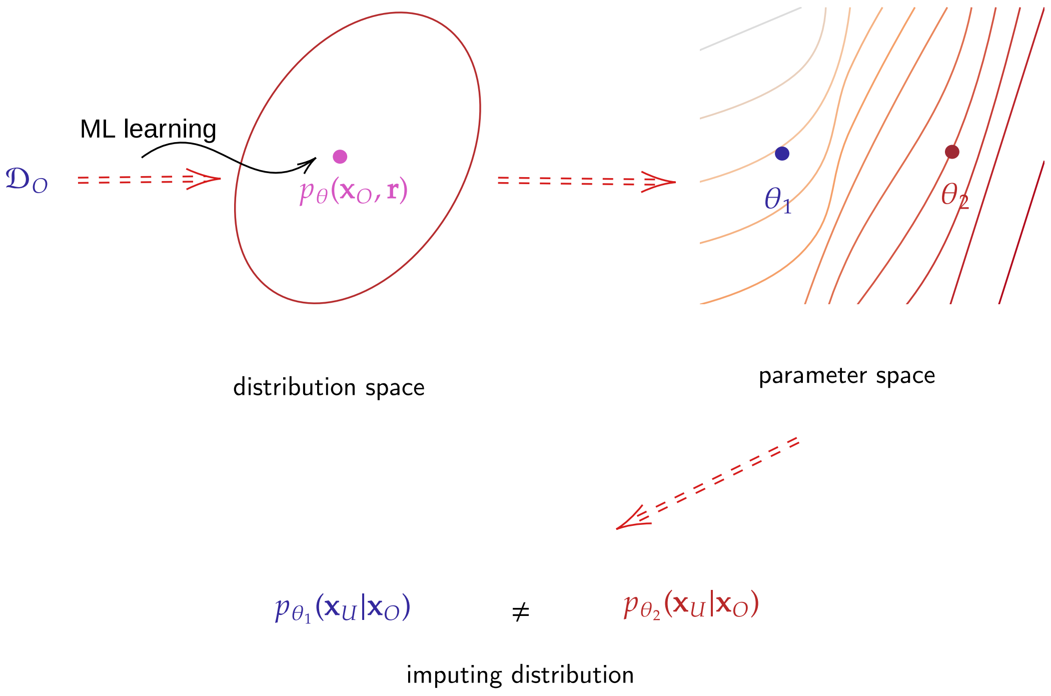

ML learning
$\mathcal{D}_O$
$p_\theta(\mathbf{x}_O, \mathbf{r})$
θ_1
θ_2
distribution space
parameter space
$p_{\theta_1}(\mathbf{x}_U|\mathbf{x}_O)$
$\neq$
$p_{\theta_2}(\mathbf{x}_U|\mathbf{x}_O)$
imputing distribution

Figure 6.1 Model non-identifiability under MNAR will introduce additional biases when performing missing data imputation

approximate inference, hence we propose to perform inference in function space. In Part B, we ask: will model non-identifiability cause similar problems in deep generative models for unsupervised learning?

In many scenarios, when learning a generative model in the form of Equation (6.0.1), we are more interested in quantifying the aleatoric uncertainty modeled by $\mathbf{z}$, and will only obtain a point estimate of $\boldsymbol{\theta}$. Therefore, the pathologies of approximate inference caused by $\boldsymbol{\theta}$-non-identifiability are not as prevalent in generative models. However, when learning under missing data, $\boldsymbol{\theta}$-non-identifiability will cause biases when performing missing data imputation, $p_{\boldsymbol{\theta}}(\mathbf{x}_U|\mathbf{x}_O)$. We will show this as below.

As shown in Figure 6.1, suppose that our generative model is correctly specified. Then given a partially observed dataset $\mathcal{D}_O$ of infinite examples, we can perfectly fit a model $p_{\boldsymbol{\theta}}(\mathbf{x}_O, \mathbf{r})$ to all the observed variables, $\mathbf{x}_O$ and $\mathbf{r}$. However, since the model $p_{\boldsymbol{\theta}}(\mathbf{x})$ is not identifiable w.r.t. $\boldsymbol{\theta}$, the same distribution $p_{\boldsymbol{\theta}}(\mathbf{x}_O, \mathbf{r})$ can be generated from say, two different parameter settings, $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$. Therefore, there might exist two different (complete data) distributions $p_{\boldsymbol{\theta}_1}(\mathbf{x}, \mathbf{r}) \neq p_{\boldsymbol{\theta}_2}(\mathbf{x}, \mathbf{r})$, that give the same marginals $p_{\boldsymbol{\theta}_1}(\mathbf{x}_O, \mathbf{r}) = p_{\boldsymbol{\theta}_2}(\mathbf{x}_O, \mathbf{r})$.

As a result, the imputation based on two parameter settings will be different:

$$p_{\boldsymbol{\theta}_1}(\mathbf{x}_O|\mathbf{x}_U) \neq p_{\boldsymbol{\theta}_2}(\mathbf{x}_O|\mathbf{x}_U). \tag{6.3.1}$$

Apparently, at least one of them must be biased. Thus, when dealing with MNAR missing mechanisms, we must take into account the model identifiability and investigate sufficient conditions for identifiability under missing data. This research question will be studied in Chapter 8.

Remark (Non-identifiability under MCAR). Model non-identifiability will not cause the aforementioned biases under MCAR assumptions. When the data is MCAR, we can recover the distribution of the complete data $p_{\boldsymbol{\theta}}(\mathbf{x})$ by using listwise deletion: we can delete all data points that contain missing values, and only fit the model to complete data instances. This is equivalent to estimating the conditional distribution,

$$p_{\boldsymbol{\theta}}(\mathbf{x}_O|\mathbf{r}=1).$$

Since the missing data is MCAR, we have

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{r}=1)}{p(\mathbf{r}=1)} = p_{\boldsymbol{\theta}}(\mathbf{x}_O|\mathbf{r}=1),$$

which is exactly the same as the listwise deletion estimation, $p_{\boldsymbol{\theta}}(\mathbf{x}_O|\mathbf{r}=1)$. Similarly, any marginal distribution $p_{\boldsymbol{\theta}}(\mathbf{x}_O)$ can be estimated by

$$p_{\boldsymbol{\theta}}(\mathbf{x}_O) = \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_O,\mathbf{r}_O=1,\mathbf{r}_U=0)}{p(\mathbf{r}_O=1,\mathbf{r}_U=0)} = p_{\boldsymbol{\theta}}(\mathbf{x}_O|\mathbf{r}_O=1,\mathbf{r}_U=0).$$

Hence, the imputation distribution can be given by Bayes rule,

$$p_{\boldsymbol{\theta}}(\mathbf{x}_U|\mathbf{x}_O) = \frac{p_{\boldsymbol{\theta}}(\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{x}_O)},$$

and no additional biases are introduced.

The above estimation procedure is not always possible under MNAR assumptions. By definition, we have

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{r}=1)}{p(\mathbf{r}=1|\mathbf{x})}.$$

This time, the numerator, $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{r} = 1)$ can still be estimated by listwise deletion

$$p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{r} = 1) = p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{r} = 1)p(\mathbf{r} = 1) = p_{\boldsymbol{\theta}}(\mathbf{x}_O|\mathbf{r} = 1)p(\mathbf{r} = 1).$$

However, note that the denominator $p(\mathbf{r} = 1|\mathbf{x})$ can not always be estimated from observational data. It requires additional assumptions, on the recoverability of $p(\mathbf{r} = 1|\mathbf{x})$ [233]. Here, recoverability means there exist a functional g, such that $p(\mathbf{r} = 1|\mathbf{x}) = g(p(\mathbf{x}_O, \mathbf{r}))$. Without this guarantee, we can not recover the complete data distribution $p_{\boldsymbol{\theta}}(\mathbf{x})$.

6.4 Part B highlight

In Part B, we also take a progressive approach to address the question of (unsupervised) learning, inference and decision making in the presence of missing data uncertainties.

- Chapter 7: built upon deep generative models and probabilistic modeling, Chapter 7 presents a practical framework for learning, inference, and high-value information acquisition under MAR (missing at random) missing values. This framework is referred as EDDI (Efficient Dynamic Discovery of high-value Information).
- Chapter 8: this chapter leans towards theoretical aspects more and revisit the assumptions of the approach used in Chapter 7. We extends the work of EDDI to more general missing not at random (MNAR) assumptions, and studies the model identifiability of deep generative models under MNAR.

Chapter 7

Efficient Dynamic Discovery of High-Value Information with Partial VAE

Human experts are not only good at evaluating the level of uncertainty in certain decision-making problems, but also actively collecting new information *that is most useful* for reducing those uncertainties. Imagine a person walking into a hospital with a broken arm. The first question from healthcare personnel would likely be "How did you break your arm?" instead of "Do you have a cold?", because the answer reveals the most relevant information for this patient's treatment. However, automating this human expertise of information acquisition is difficult. In applications such as online questionnaires, for example, most existing online questionnaire systems either present exhaustive questions [174, 310] or use extremely time-consuming human labeling work to manually build a decision tree to reduce the number of questions [375]. This wastes the valuable time of experts or users (patients). An automated solution for the personalized dynamic acquisition of information has great potential to save much of this time in many real-life applications.

What are the technical challenges to building an intelligent information acquisition system? If we carefully analyze the previous healthcare example, we can break down the doctor's thinking process into three steps. First, he/she would evaluate the current situation of the patient, with uncertainty in mind; second, given the current situation, he/she will investigate what are the possible scenarios and outcomes; and finally, based on those possible scenarios, he/she will ask questions accordingly, that are the most relevant and impactful. If we translate this thinking process into machine learning terms, we can first identify that missing data uncertainty *is a key issue*: at any point in time we only observe a small subset of the patient's symptoms or medical test results, yet have to reason about possible causes for his symptom. We are thus uncertain regarding the missing parts of the dataset and will need an accurate probabilistic model that can quantify the missing data uncertainty, and perform inference given a variable subset of observed answers. *Another key problem is deciding* what to ask next: can we optimally choose which variable to observe next, so that we can reduce the missing data uncertainty, and improve the decision quality of relevant tasks? this requires assessing the value of each possible question or measurement, the exact computation of which is intractable. However, compared to traditional active learning methods, here we need to actively select individual features, not data instances. Therefore, many existing methods are not applicable. In addition, these traditional methods are often not scalable to the large volume of data available in many practical cases [305, 174].

In this Chapter, we propose the EDDI (Efficient Dynamic Discovery of high-value Information) framework as a scalable unsupervised learning and information acquisition system under missing data. We assume that only a partially observed version of the dataset is available for analysis, and information acquisition is always associated with some cost. Given a specific decision task, such as estimating the customers' experience or assessing population health status, we can utilize the framework to dynamically decide which piece of information to acquire next. The EDDI framework is very general, and the information can be presented in any form such as question-answering tasks, or lab test results in medical diagnosis. Our contributions are:

- We propose a novel efficient information acquisition framework, EDDI (Section 7.2). To enable EDDI, we contribute technically:

    1. *A new partial amortized inference method for generative modeling under partially observed data (Section 7.2.1).* We extend the variational autoencoder (VAE) [150, 272], to account for partial observations. The resulting method, which we call the Partial VAE, is inspired by the set formulation of the data [260, 374]. The Partial VAE, as a probabilistic framework in the presence of missing data, is highly scalable, and serves as the base for the EDDI framework. Note that Partial VAE itself is widely applicable and can be used on its own as a non-linear probabilistic framework for missing-data imputation.
    2. *An information-theoretic acquisition function with a novel efficient approximation, yielding a novel variable-wise active learning method (Section 7.2.2).* Based on the partial VAE, we actively select the unobserved variable which contributes most to the decision task, such as customer surveys and health assessments, evaluated using the mutual information. This acquisition function does not have an analytical solution, and we derive a novel efficient approximation.

- We demonstrate the performance of EDDI in various settings and apply it in real-life health-care scenarios (Section 7.3).

  1. We first show the superior performance of the Partial VAE framework on an image inpainting task (Section 7.3.1).
  2. We then use 6 different datasets from the Machine Learning Repository of University of Irvine (UCI) [59] to demonstrate the behavior of EDDI, comparing with multiple baseline methods (Section 7.3.2).
  3. Finally, we evaluate EDDI on two real-life health-care applications: risk assessment in intensive care (Section 7.3.3) and public health assessment using a national survey (Section 7.3.4), where traditional methods without amortized inference do not scale. EDDI shows clear improvements in both applications.

7.1 Problem formulation

Before introducing our framework, we first formalize the unsupervised learning and active variable selection problem under missing data. Let $\mathbf{x} = (x_1, \ldots, x_{|I|})$ be a set of random variables with probability density $p(\mathbf{x})$. We call $p(\mathbf{x})$ the *complete data distribution*. Furthermore, we assume that the complete data $\mathbf{x}$ can only be *partially observed*. Let a subset of the variables $\mathbf{x}_O$, $O \subset I$, be observed while the variables $\mathbf{x}_U$, $U = I \setminus O$, are unobserved. Under this setting, we are interested in the following problems:

- **(Unsupervised learning under missing data).** Given a class of generative models $\{p_{\boldsymbol{\theta}}(\mathbf{x}) | \boldsymbol{\theta} \in \boldsymbol{\theta}\}$, the goal of unsupervised learning is to approximate the ground truth density $p_{\boldsymbol{\theta}^\star}(\mathbf{x}) \approx p(\mathbf{x})$, using only partially observed data as training sets. This model can be used to perform missing data imputation $p_{\boldsymbol{\theta}}(\mathbf{x}_U | \mathbf{x}_O)$, or help to perform High-value information acquisition, as detailed below.

- **(High-value information acquisition).** Assume that given a decision task, we are interested in certain task-specific quantity of interest $f(\mathbf{x})$, where$f(\cdot)$ can be any (random) function. Assume also that we can query the value of variables x_i for $i \in U$. Then, the goal of active variable selection is to query a sequence of variables in U in order to predict $f(\mathbf{x})$, as accurately as possible while simultaneously performing as few queries as possible. This information acquisition problem, in the simplified myopic setting, can be formalized as that of proposing the next variable x_{i^*} to be queried by maximizing a reward function R at each step:

$$i^* = \arg\max_{i \in U} R_{p_{\boldsymbol{\theta}}}(i \mid \mathbf{x}_O), \tag{7.1.1}$$

where $R_{p_{\boldsymbol{\theta}}}(i \mid \mathbf{x}_O)$ quantifies the merit of our prediction of $f(\cdot)$ given $\mathbf{x}_0$ and x_i. In this thesis, $R_{p_{\boldsymbol{\theta}}}(i \mid \mathbf{x}_O)$ usually depend on a generative model $p_{\boldsymbol{\theta}}(\mathbf{x})$ estimated from partially observed training set. Furthermore, the reward can quantify other properties important to the problem, e.g. the cost of acquiring x_i.

7.2 Methodology

In this section, we present the Partial VAE to model and perform inference on partial observations. Finally, we complete the EDDI framework by presenting our new acquisition function and estimation method.

7.2.1 Partial Amortization of Inference Queries

We first introduce how to establish a generative probabilistic model of random variables $\mathbf{x}$, that is capable of handling unobserved (missing) variables $\mathbf{x}_U$ with variable size. Our approach to this, named the Partial VAE, is based on the variational autoencoder (VAE), which enables amortized inference to scale to large volumes of data.

VAE and amortized inference. A VAE defines a generative model in which the data $\mathbf{x}$ is generated from latent variables $\mathbf{z}$, $p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z}) = \prod_i p_{\boldsymbol{\theta}}(\mathbf{x}_i|\mathbf{z})p(\mathbf{z})$. The data generation, $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$, is realized by a deep neural network. To approximate the posterior of the latent variable $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$, VAEs use *amortized* variational inference. Specifically, it uses an encoder, which is another neural network with the data $\mathbf{x}$ as input to produce a variational approximation of the posterior $q(\mathbf{z}|\mathbf{x};\phi)$. As traditional variational inference, VAE is trained by maximizing an evidence lower bound (ELBO), which is equivalent to minimizing the KL divergence between $q(\mathbf{z}|\mathbf{x};\phi)$ and $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$.

VAEs are not directly applicable when data points have arbitrary subset of data entries missing. Consider the situation that the variables are divided into *observed* variables $\mathbf{x}_O$ and *unobserved* variables $\mathbf{x}_U$. In this setting, we would like to efficiently and accurately infer $p(\mathbf{z}|\mathbf{x}_O)$ and $p(\mathbf{x}_U|\mathbf{x}_O)$. One main challenge is that there are many possible partitions $\{U,O\}$, where the size of observed variables might vary. Therefore, classic approaches to training a VAE with the variational bound and amortized inference networks are not applicable. We propose to extend amortized inference to handle partial observations.

Partial VAE

Partial VAE. In a VAE, $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ is factorized, i.e.

$$p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) = \prod_i p_i(\mathbf{x}_{i,\boldsymbol{\theta}}|\mathbf{z}). \tag{7.2.1}$$

This implies that given $\mathbf{z}$, the observed variables $\mathbf{x}_O$ are conditionally independent of $\mathbf{x}_U$. Therefore,

$$p_{\boldsymbol{\theta}}(\mathbf{x}_U|\mathbf{x}_O, \mathbf{z}) = p_{\boldsymbol{\theta}}(\mathbf{x}_U|\mathbf{z}), \tag{7.2.2}$$

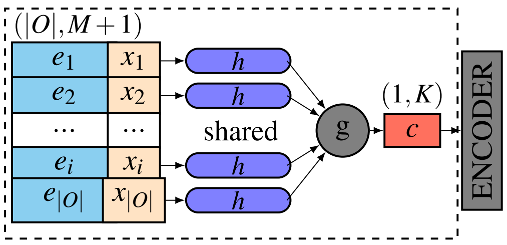
and inferences about $\mathbf{x}_U$ can be reduced to inference about $\mathbf{z}$. Hence, the key object of interest in this setting is $p(\mathbf{z}|\mathbf{x}_O)$, i.e., the posterior over the latent variables $\mathbf{z}$ given the observed variables $\mathbf{x}_O$. Once we obtain $\mathbf{z}$, computing $\mathbf{x}_U$ is straightforward. To approximate $p(\mathbf{z}|\mathbf{x}_O)$, we introduce a variational inference network $q(\mathbf{z}|\mathbf{x}_O)$ and define a partial variational lower bound, or the *partial ELBO*

$$\begin{aligned}\log p(\mathbf{x}_O) &\geq \log p(\mathbf{x}_O) - D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x}_O)\|p(\mathbf{z}|\mathbf{x}_O)) \qquad (7.2.3)\\ &= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x}_O)}[\log p(\mathbf{x}_O|\mathbf{z}) + \log p(\mathbf{z}) - \log q(\mathbf{z}|\mathbf{x}_O)]\\ &\equiv \mathcal{L}_{partial}.\end{aligned}$$

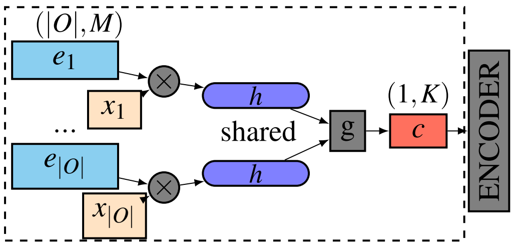
This bound, $\mathcal{L}_{partial}$, depends only on the observed variables $\mathbf{x}_O$, whose dimensionality may vary among different data points. We thus call the the inference net, $q(\mathbf{z}|\mathbf{x}_O)$, the *partial inference net*. Specifying $q(\mathbf{z}|\mathbf{x}_O)$ requires distributions for any partition $\{O, U\}$ of I. Given a set of partially observed data, $\mathcal{D}_O = \{\mathbf{x}_O^{(n)}\}_{1\leq n\leq N}$, we can further write the partial ELBO w.r.t. this particular dataset:

$$\log p(\mathcal{D}_O) \geq \sum_n \mathbb{E}_{\mathbf{z}^{(n)}\sim q(\mathbf{z}^{(n)}|\mathbf{x}_O^{(n)})}[\log p(\mathbf{x}_O^{(n)}|\mathbf{z}^{(n)})] + \sum_n [\log p(\mathbf{z}^{(n)}) - \log q(\mathbf{z}^{(n)}|\mathbf{x}_O^{(n)})]. \tag{7.2.4}$$

Remark (Assumptions on missing mechanism). Here, we have implicitly assumed that the missing mechanism is MCAR or MAR. That is, the missing mask $\mathbf{r}$ is independent of $\mathbf{x}_U$. In the classic paper by Rubin [279], it is argued that when the missing data is MCAR or MAR, we can ignore the missing mechanism and just perform maximum likelihood learning by



(a) Partial VAE PN setting



(b) Partial VAE PNP setting

Figure 7.1 Illustration of Partial VAE encoder architecture.

Figure 7.2 Partial VAE PNP setting

maximizing $\log p(\mathbf{x}_O) = \log \int_{\mathbf{x}_U} p(\mathbf{x}_O, \mathbf{x}_U) d\mathbf{x}_U$. The reasoning behind this is as follows:

$$\begin{aligned}
&\arg\max_{\boldsymbol{\theta}} \sum_n \log p_{\boldsymbol{\theta}}(\mathbf{x}_O^{(n)}) \\
=&\arg\max_{\boldsymbol{\theta}} \sum_n \log p_{\boldsymbol{\theta}}(\mathbf{x}_O^{(n)}) p(\mathbf{r}^{(n)}|\mathbf{x}_O^{(n)}) \\
=&\arg\max_{\boldsymbol{\theta}} \sum_n \log \int_{\mathbf{x}_U^{(n)}} p_{\boldsymbol{\theta}}(\mathbf{x}_O^{(n)}, \mathbf{x}_U^{(n)}) p(\mathbf{r}^{(n)}|\mathbf{x}^{(n)}) d\mathbf{x}_U^{(n)} \\
=&\arg\max_{\boldsymbol{\theta}} \sum_n \log p_{\boldsymbol{\theta}}(\mathbf{x}_O^{(n)}, \mathbf{r}^{(n)}).
\end{aligned}$$

Therefore, we can ignore the missing mask $\mathbf{r}$, as well as the associated missing mechanism, $p(\mathbf{r}|\mathbf{x})$, when performing learning and infernece.

Amortized Inference with partial observations. Inference under partial observations requires the inference net of VAE to be capable to handle arbitrary set of observed data, and sharing parameters across these different sized sets of observations for amortization.

Inspired by the Point Net (PN) approach for point cloud classification [260, 374], we specify the approximate distribution $q(\mathbf{z}|\mathbf{x}_O)$ by a *permutation invariant set function encoding*, given by:

$$\mathbf{c}(\mathbf{x}_O) := g(h(\mathbf{s}_1), h(\mathbf{s}_2), ..., h(\mathbf{s}_{|O|})), \tag{7.2.5}$$

where $\mathbf{s}_d$ carries the information of the input of the d-th observed variable, and $|O|$ is the number of observed variables. In particular, $\mathbf{s}_d$ contains the information about the identity of the input $\mathbf{e}_d$ and the corresponding input value x_d. There are many ways to define the identity variable, $\mathbf{e}_d$. Naively, it could be the coordinates of observed pixels for images, and one-hot embedding of the number of questions in a questionnaire. With different problem settings, it can be beneficial to learn $\mathbf{e}$ as an embedding of the identity of the variable, either with or without an naive encoding as input. In this work, we treat $\mathbf{e}$ as an unknown embedding, to be optimized during training.

There are also different ways to construct $\mathbf{s}_d$. A common choice is concatenation, $\mathbf{s}_d = [\mathbf{e}_d, x_d]$, which is often used in computer vision applications [260]. Such architecture is illustrated in Figure 7.1a. We refer to this setting as the *Pointnet (PN)* specification of Partial VAE. However, the construction of $\mathbf{s}_d$ can be more flexible. We propose to construct $\mathbf{s}_d = \mathbf{e}_d * x_d$ using element-wise multiplication as an alternative, shown in Figure 7.1b. We show that this formulation generalizes naive Zero Imputation (ZI) VAE [240] (cf. Appendix 7.C.1). We refer to the multiplication setting as the *Pointnet Plus (PNP)* specification of Partial VAE.

We can then use a neural network $h(\cdot)$ to map the input $\mathbf{s}_d$ to $\mathbb{R}^K$, where and K is the latent space size. The key to the PNP/PN structure is the permutation invariant aggregation operation $g(\cdot)$, such as max-pooling or summation. In this way, the mapping $\mathbf{c}(\mathbf{x}_O)$ is invariant to the permutations of elements of $\mathbf{x}_O$, and $\mathbf{x}_O$ can have arbitrary length. Finally, the fixed-size code $\mathbf{c}(\mathbf{x}_O)$ is fed into an ordinary neural network, that transforms the code into the statistics of a multivariate Gaussian distribution to approximate $p(\mathbf{z}|\mathbf{x}_O)$. The procedure is illustrated in Figure 7.2. As discussed before, given $p(\mathbf{z}|\mathbf{x}_O)$, we can estimate $p(\mathbf{x}_U|\mathbf{z})$.

7.2.2 Efficient Dynamic Discovery of High-value Information

We now cast the active variable selection problem (7.1.1) as an adaptive Bayesian experimental design problem, utilizing $p(\mathbf{x}_U|\mathbf{x}_O)$ inferred by the Partial VAE. The learning and active information acquisition procedure of the EDDI framework is summarized in algorithm 3.

Information Reward. We designed a variable selection acquisition function in an information theoretic way following Bayesian experimental design [186, 26]. Lindley [186] provides a generic formulation of Bayesian experimental design by maximizing the expected Shannon information. Bernardo [26] generalizes the idea, by considering the context of decision making task.

For a given task, we are interested in statistics of some variables $\mathbf{x}_\phi$, where $\mathbf{x}_\phi \subset \mathbf{x}_U$. Given a new instance (user), assume that we have observed $\mathbf{x}_O$ so far for this instance, and we need to select the next variable x_i (an element of $\mathbf{x}_{U\setminus\phi}$) to observe. Following Bernardo [26], we select x_i by maximizing the following information reward:

$$R(i, \mathbf{x}_O) = \mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_O)} D_{\mathrm{KL}} \left[p(\mathbf{x}_\phi|\mathbf{x}_i, \mathbf{x}_O) \,||\, p(\mathbf{x}_\phi|\mathbf{x}_O) \right]. \tag{7.2.6}$$

Algorithm 3 EDDI: algorithm overview

```
Require: Training dataset $\mathbf{X}$, which is partially observed; Test dataset $\mathbf{X}^*$ with no observations collected yet; Indices $\phi$ of target variables.
1: Train Partial VAE by optimizing partial variational bound with $\mathbf{X}$ (cf. Section 7.2.1)
2: Actively acquire feature value $x_i$ to estimate $\mathbf{x}_\phi^*$ for each test point (cf. Section 7.2.2)
   for each test instance do
      $\mathbf{x}_O \leftarrow \emptyset$ (no variable value has been observed for
      any test point)
      repeat
         Choose variable $x_i$ from $U \setminus \phi$ to maximize the
         information reward (Equation (7.2.9))
         $\mathbf{x}_O \leftarrow x_i \cup \mathbf{x}_O$
      until Stopping criterion reached (e.g. the time budget)
   end for
```

In this chapter, we mainly consider the case that a subset of interesting observations represents the statistics of interest $\mathbf{x}_\phi$. Sampling $\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)$ is approximated by $\mathbf{x}_i \sim \hat{p}(\mathbf{x}_i|\mathbf{x}_o)$, where $\hat{p}(\mathbf{x}_i|\mathbf{x}_o)$ can be obtained by using the Partial VAE. It is implemented by first sampling $\mathbf{z} \sim q(\mathbf{z}|\mathbf{x}_o)$, and then $\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{z})$. The same applies for $p(\mathbf{x}_i, \mathbf{x}_\phi|\mathbf{x}_o)$ which appears in Equation (7.2.8).

Efficient approximation of the Information reward. The Partial VAE allows us to sample $\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)$. However, the KL term in Equation (7.2.6),

$$\begin{aligned} &\mathrm{D_{KL}}\left[p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{x}_\phi|\mathbf{x}_o)\right] \\ &= -\int_{\mathbf{x}_\phi} p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)\log\frac{p(\mathbf{x}_\phi|\mathbf{x}_o)}{p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}d\mathbf{x}_\phi, \end{aligned} \tag{7.2.7}$$

is intractable since both $p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)$ and $p(\mathbf{x}_\phi|\mathbf{x}_o)$ are intractable. For high dimensional $\mathbf{x}_\phi$, entropy estimation could be difficult. The entropy term $\int_{\mathbf{x}_\phi} p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)\log p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)d\mathbf{x}_\phi$ depends on i hence cannot be ignored. In the following, we show how to approximate this expression.

Note that analytic solutions of KL-divergences are available under specific variational distribution families of $q(\mathbf{z}|\mathbf{x}_O)$ (such as the Gaussian distribution commonly used in VAEs). Instead of calculating the information reward in $\mathbf{x}$ space, we have shown that one can compute in the $\mathbf{z}$ space (cf. Appendix 7.A.1):

$$\begin{aligned} R(i,\mathbf{x}_o) =& \mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)} \mathrm{D}_{\mathrm{KL}}\left[p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_o)\right] - \\ & \mathbb{E}_{\mathbf{x}_\phi,\mathbf{x}_i\sim p(\mathbf{x}_\phi,\mathbf{x}_i|\mathbf{x}_o)} \mathrm{D}_{\mathrm{KL}}\left[p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o)\right]. \end{aligned} \tag{7.2.8}$$

Note that Equation (7.2.8) is exact. Additionally, we use the partial VAE approximation $p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o) \approx q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)$, $p(\mathbf{z}|\mathbf{x}_o) \approx q(\mathbf{z}_i|\mathbf{x}_o)$ and $p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o) \approx q(\mathbf{z}_i|\mathbf{x}_i,\mathbf{x}_o)$. This leads to the final approximation of the information reward:

$$\begin{aligned} \hat{R}(i,\mathbf{x}_o) =& \mathbb{E}_{\mathbf{x}_i\sim \hat{p}(\mathbf{x}_i|\mathbf{x}_o)} \mathrm{D}_{\mathrm{KL}}\left[q(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||q(\mathbf{z}|\mathbf{x}_o)\right] - \\ & \mathbb{E}_{\mathbf{x}_\phi,\mathbf{x}_i\sim \hat{p}(\mathbf{x}_\phi,\mathbf{x}_i|\mathbf{x}_o)} \mathrm{D}_{\mathrm{KL}}\left[q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o)\right]. \end{aligned} \tag{7.2.9}$$

With this approximation, the divergence between $q(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)$ and $q(\mathbf{z}|\mathbf{x}_o)$ can often be computed analytically in the Partial VAE setting, for example, under Gaussian parameterization. The only Monte Carlo sampling required is the one set of samples $\mathbf{x}_\phi,\mathbf{x}_i \sim p(\mathbf{x}_\phi,\mathbf{x}_i|\mathbf{x}_o)$ that can be shared across different KL terms in Equation (7.2.9).

Remark (Stop criterion based on missing data uncertainty.) When actively acquiring new variables, it is sometimes useful to apply certain stop criterion that will terminate actions when certain conditions are met. Throughout this thesis, although we do not explicitly apply any stop criterion, we would like to point out the fact that the EDDI framework naturally provide a principled stop criterion. The intuition is that, when there isn't enough information to predict $\mathbf{x}_\phi$, the estimated uncertainty level of missing data uncertainty $p(\mathbf{x}_U|\mathbf{x}_O)$ (or $p(\mathbf{x}_\phi)$) should be quite high. This indicates that we need to continue to acquire more variables to $\mathbf{x}_O$, until we feel significantly more certain.

This uncertainty-based stop criterion is further developed and investigated in our recent preliminary work [106], where we have shown that such stop criterion helps to significantly improve the efficiency of EDDI algorithm in the context of symptom-based self-diagnosis.

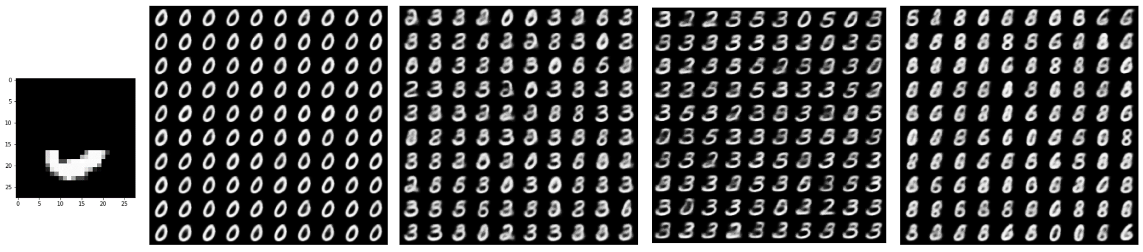
Remark (Numerical example). In order to better demonstrate the information acquisition process of the proposed EDDI framework, we provide a simple numerical example on Boston Housing dataset. In Figure 7.14, we show the information reward estimated during the first 4 active variable selection steps on a randomly chosen Boston Housing test data point. Each row contains two plots regarding the same time step. The bar plots on the left show the information reward estimation of each variable on the y-axis. All unobserved variables start with green bars, and turns purple once selected by the algorithm. The right plot of each row is the corresponding violin plot of the posterior density estimations of remaining unobserved variables. The rightmost variable in each violin plot corresponds to the target variable. At each step, EDDI will estimate the information reward of each unobserved variable (green) given the observed variables. Then, the variable with highest information reward will be acquired (becomes purple), and the posterior density of remaining unobserved variables (especially the target variable) will shrink. At the end of the fourth step, partial VAE is already quite confident on its prediction of the target variable.

7.3 Experiments

Here we evaluate the proposed EDDI framework. We first assess the Partial VAE component of EDDI alone on an image inpainting task both qualitatively and quantitatively (Section 7.3.1). We compare our proposed two PN-based Partial VAE with the zero-imputing (ZI) VAE [240]. Additionally, we modify the ZI VAE to use the mask matrix indicating which variables are currently observed as input. We name this method ZI-m VAE. We then demonstrate the performance of the entire EDDI framework on datasets from the UCI repository (Section 7.3.2), as well as in two real-life application scenarios: Risk assessment in intensive care (Section 7.3.3) and public health assessment with national health survey (Section 7.3.4). We compare the performance of EDDI, using four different Partial VAE settings, with three baseline information acquisition strategies. The first baseline is the *random active feature selection strategy (denoted as RAND)* which randomly picks the next variable to observe. RAND reflects the strategy used in many real-world applications, such as online surveys. The second baseline method is the *single best strategy (denoted as SING)* which finds a single fixed global optimal order of selecting variables. This order is then applied to all data points. SING uses the objective function as in Equation (7.2.9) to find the optimal ordering by averaging over all the test data.

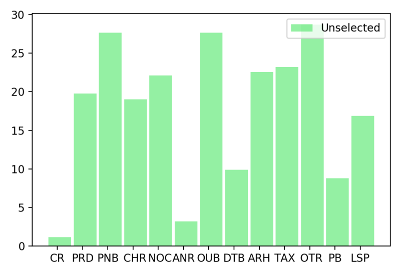
7.3.1 Image inpainting with Partial VAE

We evaluate the performance of Partial VAE with the image inpainting task, which is to fill in the removed pixels. We perform the evaluation in two different settings: in the first setting, pixels are randomly removed, and in the second setting, a continuous patch of pixels are removed.

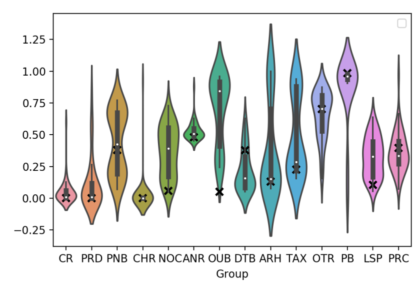


(a) Input (b) ZI (c) ZI-m (d) PN (e) PNP

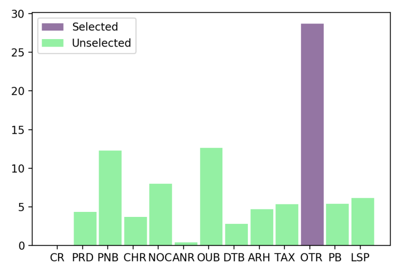
Figure 7.3 Image inpainting example with MNIST dataset using Partial VAE with four settings.



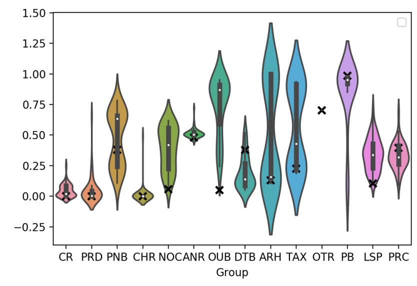
(a) Step 0, information reward



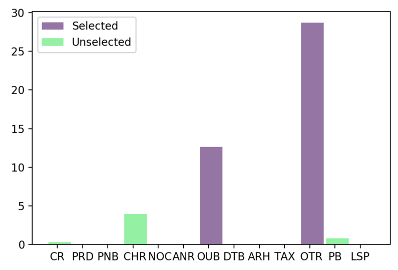
(b) Step 0, violin plot



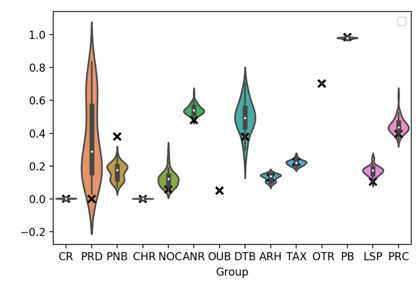
(c) Step 1, information reward



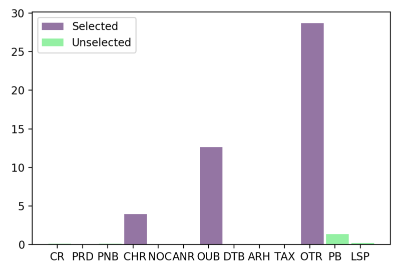
(d) Step 1, violin plot



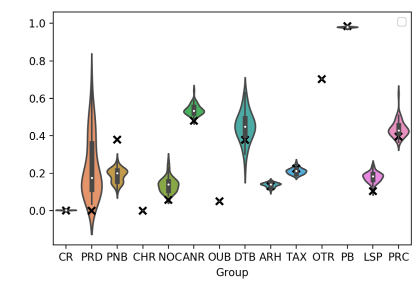
(e) Step 2, information reward



(f) Step 2, violin plot



(g) Step 3, information reward



(h) Step 3, violin plot

Figure 7.4 Information reward estimated during the first 4 active variable selection steps on a randomly chosen Boston Housing test data point. **Model**: PNP, strategy: EDDI. Each row contains two plots regarding the same time step. **Bar plots on the left** show the information reward estimation of each variable on the y-axis. All unobserved variables start with green bars, and turns purple once selected by the algorithm. **Right**: violin plot of the posterior density estimations of remaining unobserved variables.

.

Inpainting Random Missing Pixels. We use MNIST dataset [171] and remove pixels randomly for this task. The same settings are used for all methods (see Appendix 7.B.1 for details). During training, we remove a random portion (uniformly sampled between 0% and 70%) of pixels. We then impute missing pixels on a partially observed test set (constructed by removing 70% of the pixels uniform randomly). The performance of pixel imputation is evaluated by test ELBOs on missing pixels. The first two rows in Table 7.1 show training and test ELBOs for all algorithms using this partially observed dataset. Additionally, we show ordinary VAE (VAE-full) trained on the fully observed dataset as an ideal reference. Among all Partial VAE methods, the PNP approach performs best.

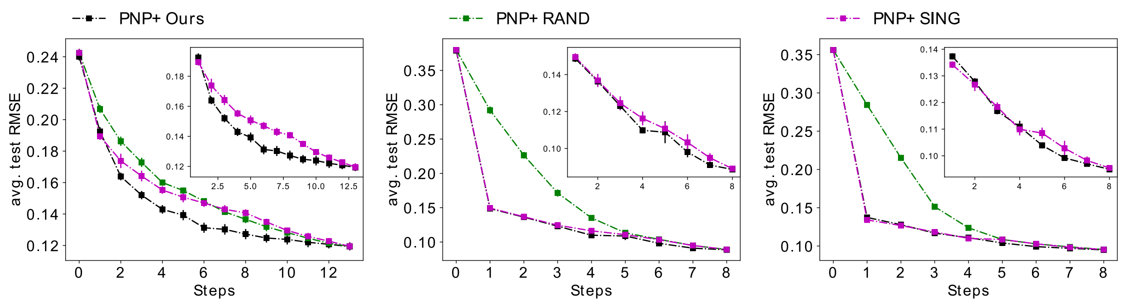
Inpainting Regions. We then consider inpainting large contiguous regions of images. It aims to evaluate the capability of the Partial VAEs to produce all possible outcomes with better uncertainty estimates. With the same trained model as before, we remove the region of the upper 60% pixels of the image in the test set. We then evaluate the average likelihoods of the models. The last row of Table 7.1 shows the results of the test ELBO in this case. PNP based Partial VAE performs better than other settings. Note that given only the lower half of a digit, the number cannot be identified uniquely. ZI (Figure 7.3b) fails to cover the different possible modes due to its limitation in posterior inference. ZI-m (Figure 7.3c) is capable of producing multiple modes. However, some of the generated samples are not consistent with the given part (i.e., some digits of 2 are generated). Our proposed PN (Figure 7.3d) and PNP (Figure 7.3e) are capable of recovering different modes, and are consistent with observations.

Table 7.1 Comparing models trained on partially observed MNIST. VAE-full is an ideal reference.

Method	VAE-full	ZI	ZI-m	PN	PNP
Train ELBO	-95.05	**-113.64**	-117.29	-121.43	**-113.64**
Test ELBO (Rnd.)	-101.46	-116.01	-118.61	-122.20	**-114.01**
Test ELBO (Reg.)	-101.46	-130.61	-123.87	-116.53	**-113.19**

7.3.2 EDDI on UCI datasets

Given the effectiveness of our proposed Partial VAE, we now demonstrate the performance of our proposed EDDI framework in comparison with random selection (RAND) and single optimal ordering (SING). We first apply EDDI on 6 different UCI datasets (cf. Appendix



(a) Boston Housing (b) Energy (c) Wine

Figure 7.5 Information curves of active variable selection, demonstrated on three UCI datasets (based on PNP parameterization of Partial VAE). This displays negative test RMSE (y axis, the lower the better) during the course of active selection (x-axis). Error bars represent standard errors over 10 runs.

.

Table 7.2 Average ranking of AUIC over 6 UCI datasets.

Method	ZI	ZI-m	PNP	PN
EDDI	5.72 (0.03)	5.54 (0.02)	**5.08 (0.02)**	5.25 (0.02)
Random	8.03 (0.03)	8.10 (0.03)	7.77 (0.03)	7.79 (0.03)
Single best	8.68 (0.03)	5.50 (0.02)	5.20 (0.02)	5.28 (0.02)

7.B.2) [59]. We report the results of EDDI with all these four different specifications of Partial VAE (ZI, ZI-m, PN, PNP).

All Partial VAE are first trained on partially observed UCI datasets where a random portion of variables is removed. We actively select variable for each test point starting with empty observation $\mathbf{x}_o = \emptyset$. In all UCI datasets, we randomly sample 10% of the data as the test set. All experiments are repeated for ten times.

Taking PNP based setting as an example, Figure 7.5 shows the test RMSE on $\mathbf{x}_\phi$ for each variable selection step with three different datasets, where $\mathbf{x}_\phi$ is defined by the UCI task. We call this curve the *information curve (IC)*. We see that EDDI can obtain information efficiently. It archives the same test RMSE with less than half of the variables. Single optimal ordering also improves upon random ordering. However, it is less efficient compared with EDDI, since EDDI perform active learning for each data instance which is "personalized". Figure 7.6 shows an example of the decision processes using EDDI and SING. The first step of EDDI overlaps largely with SING. From the second step, EDDI makes "personalized" decisions.

We also present the average performance among all datasets with different settings. The area under the information curve (AUIC), can then be used to compare the performance across models and strategies. Smaller AUIC value indicates better performance. However, due to different datasets have different scales of RMSEs and different numbers of variables (indicated by steps), it is not fair to average the AUIC across datasets to compare overall performances. We thus define average *ranking* of AUIC that compares 12 methods (indexed by i) averaging these datasets as: $r_i = \frac{1}{\sum_j N_j} \sum_{j=1}^{6} \sum_{k=1}^{N_j} r_{ijk}, \; i = 1,..,12$. These 12 methods are cross combinations of four Partial VAE models with three variable selection strategies. r_i is the final ranking of ith combination, r_{ijk} is the ranking of the ith combination (based on AUIC value) regarding the kth test data point in the jth UCI dataset, and N_j is the size of the jth UCI dataset. This gives us $6\sum_j N_j$ different rankings. Finally, we compute the mean and standard error statistics based on these rankings. Table 7.2 summarize the average ranking results. We provide additional statistical significance test (Wilcoxcon signed-rank test for paired data) in Appendix 7.B.2. Based on these experimental results, we see that EDDI outperforms other variable selection order in all different Partial VAE settings. Among different partial VAE settings, PNP/PN-based settings perform better than ZI-based settings.

Comparison with non-amortized method. Additionally, we compare EDDI to DRAL [174] which is the state-of-the-art method for the same problem setting. As discussed in Section 7.4, DRAL is linear and requires high computational cost. The DRAL paper only tested their method on a single test data point due to its limitation on computational efficiency. We compare DRAL with EDDI on Boston Housing dataset with ten randomly selected test points here. Results are shown in Figure 7.7, where EDDI significantly outperforms DARL thanks to more flexible Partial VAE model. Additionally, EDDI is 1000 times more efficient than DARL as shown in Table 7.3.

7.3.3 Risk assessment with MIMIC-III

We now apply EDDI to risk assessment tasks using the Medical Information Mart for Intensive Care (MIMIC III) database [139]. MIMIC III is the most extensive publicly available clinical database, containing real-world records from over 40,000 critical care patients with 60,000 ICU stays. The risk assessment task is to predict the final mortality. We preprocess the data for this task following Harutyunyan et al. [103] [1]. This results in a dataset of 21139 patients. We treat the final mortality of a patient as a Bernoulli variable. For

[1]https://github.com/yerevann/mimic3-benchmarks

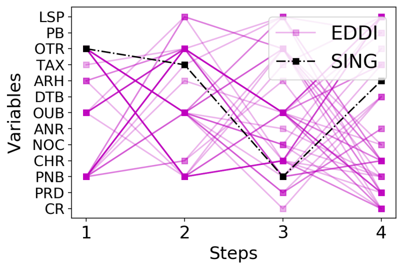


Figure 7.6 First four decision steps on Boston Housing test data. EDDI is “personalized” comparing SING. Full names of the variables are listed in the Appendix 7.B.2.

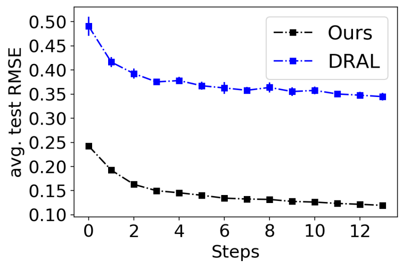


Figure 7.7 Comparison of DRAL [174] and EDDI on Boston Housing dataset. EDDI out performs DRAL significantly regarding test RMSE in every step.

Method	Time
DRAL	2747.16
EDDI	**2.64**

Table 7.3 Test CPU time (in seconds) per test point for active variable selection. EDDI is 10^3 times more efficient than DRAL [174] computationally.

our task, we focus on variable selection, which corresponds to medical instrument selection. We thus further process the time series variables into static variables based on temporal averaging.

Figure 7.8 shows the information curve (based on Bernoulli likelihoods) of different strategies, using PNP based Partial VAE as an example (more results in Appendix 7.B.3). Table 7.4 shows the average ranking of AUIC with different settings. In this application, EDDI significantly outperforms other variable selection strategies in all different settings of Partial VAE, and PNP based setting performs best.

7.3.4 Public Health Assessment with NHANES

Finally, we apply our methods to public health assessment using NHANES 2015-2016 data [1]. NHANES is a program with adaptable components of measurements, to assess the health and nutritional status of adults and children in the United States. Every year, thousands individuals of all ages are interviewed and examined in their homes. This 2015-2016 NHANES data contains three major sections, the questionnaire interview, examinations and lab tests for 9971 subjects in the publicly available version of this cycle. In our setting, we consider the whole set of lab test results (139 variables) as the target variable of interest $\mathbf{x}_\phi$ since they are expensive and reflects the subject’s health status, and we active select the questions from the extensive questionnaire (665 variables).

The questionnaire of NHANES is divided into 73 different groups. In practice, questions in the same group are often examined together. Therefore, we perform active variable

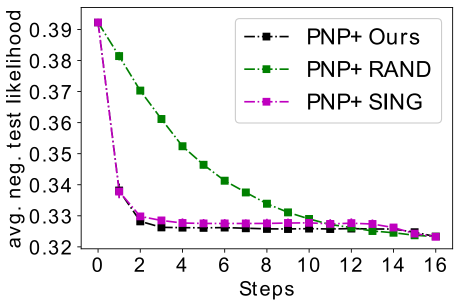


Figure 7.8 Information curves of active variable selection on risk assessment task on MIMIC III, produced with PNP setting.

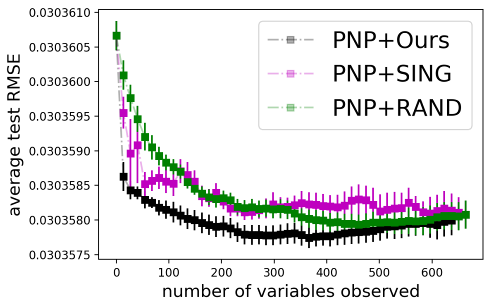


Figure 7.9 Information curves of active (grouped) variable selection on risk assessment task on NHANES, produced with PNP setting.

Method	EDDI	Random	Single best
ZI	8.83 (0.01)	7.97 (0.02)	9.83 (0.01)
ZI-m	4.91 (0.01)	7.00 (0.01)	5.91 (0.01)
PN	4.96 (0.01)	6.62 (0.01)	5.96 (0.01)
PNP	**4.39 (0.01)**	6.18 (0.01)	5.39 (0.01)

Table 7.4 Average ranking on AUIC of MIMIC III

Method	EDDI	Random	Single best
ZI	6.00 (0.10)	8.45 (0.09)	6.51 (0.09)
ZI-m	8.06 (0.09)	8.67 (0.09)	8.68 (0.07)
PN	5.28 (0.10)	5.57 (0.10)	5.46 (0.09)
PNP	**4.80** (0.10)	5.30 (0.10)	5.17 (0.10)

Table 7.5 Average ranking on AUIC of NHANES

selection on the group level: at each step, the algorithm selects one group to observe. This is more challenging than the experiments in previous sections since it requires the generative model to simulate a group of unobserved data in Equation (7.2.9) at the same time. When evaluating test RMSE on the target variable of interest, we treat variables in each group equally. For a fair comparison, the calculation of the area under the information curve (AUIC) is weighted by the size of the group chosen by the algorithms. Specifically, AUIC is calculated after spline interpolation. The information curve plots in Figure 7.9, together with Table 7.5 of AUIC statistics show that our EDDI outperforms other baselines. In addition, this experiment shows that EDDI is capable of performing active selection on a large pool of grouped variables to estimate a high dimensional target.

7.4 Related Work

EDDI is a method that handles partially observed data to enable dynamic variable wise active learning. We thus review related methods for handling partial observation and performing active learning.

7.4.1 Partial Observation

Missing data entries are common in many real-life applications, which has created a long history of research on the topic of dealing with missing data [279, 54]. We describe existing methods below with the focus of probabilistic methods:

Traditional methods without amortization. Prediction based methods have shown advantages for missing value imputation [298]. Efficient matrix factorization based methods have been recently applied [144, 133, 289], where the observations are assumed to be able to decompose as the multiplication of low dimensional matrices. In particular, many probabilistic frameworks with various distribution assumptions [289, 28] have been used for missing value imputation [373, 99] and also recommender systems where unlabeled items are predicted [326, 350, 89].

The probabilistic matrix factorization method has been used in the active variable selection framework called the dimensionality reduction active learning model (DRAL),[174]. These traditional methods suffer from limited model capacity since they are typically linear. Additionally, they do not scale to large volumes of data and thus are usually not applicable in real-world applications. For example, Lewenberg et al. [174] tested the performance of their method with a single user due to the heavy computational cost of traditional inference methods for probabilistic matrix factorization.

Utilizing Amortized Inference. Amortized inference [150, 272, 376] has significantly improved the scalability of deep generative latent variable models. In the case of partially observed data, amortized inference is particularly of interest due to the speed requirement in many real-life applications. Wu et al. [368] use amortized inference during training, where the training dataset is assumed to be fully observed. During test time, the traditional non-scalable inference is used to infer missing data entries from the partially observed dataset using the pre-trained model. This method is restrictive since it is not scalable in the test time and the fully observed training set assumption does not hold for many applications.

Nazabal et al. [241] use zero imputation (ZI) for amortized inference for both training and test sets with missing data entries. ZI is a generic and straightforward method that first fills the missing data with zeros, and then feeds the imputed data as input for the inference network. The drawback of ZI is that it introduces bias when the data are not missing completely at random which leads to a poorly fit model. We also observe artifacts when using it for the image inpainting task. Independent of our work, Garnelo et al. [79]

explore interpreting variational autoencoder (amortized inference) as stochastic processes, which also handles partial observation per se.

Note that after the publication of materials of this chapter [200], similar methods and further extensions have also been advocated in multiple contexts of deep generative models, such as [212, 201, 199, 372, 175, 128, 82, 87]. See Chapter 8.5 for more detailed review of these methods.

7.4.2 Active Learning

Traditional Active Learning. Active learning, also referred to as experimental design, aims to obtain optimal performance with fewer selected data (or experiments) [186, 203, 305]. Traditional active learning aims to select the *next data point* to label. Many information theoretical approaches have shown promising results in various settings with different acquisition functions [203, 216, 121]. These methods commonly assume that the data are fully observed, and the acquisition decision is instance wise. Little work has dealt with missing values within instances. Zheng and Padmanabhan [378] deal with missing data values by imputing with traditional non-probabilistic methods [188] first. It is still an instance-wise active learning framework.

Different from traditional active learning, our proposed framework performs *variable-wise active learning* for *each* instance. In this setting, information theoretical acquisition functions need a new design as well as non-trivial approximations. The most closely related work is the aforementioned DRAL [174], which deals with variable-wise active learning for each instance.

Active Feartue Acquisition (AFA). Active sequential feature selection is of great need, especially in cost-sensitive applications. Thus, many methods have also been applied and resulted in the class of methodologies called Active Feature Acquisition (AFA) [217, 286, 333, 124]. For instance, Melville et al. [217], Saar-Tsechansky et al. [286] have designed objectives to select any feature from any instance to minimize the cost to achieve high accuracy. The proposed framework is very general. However, the problem setting of AFA methods is different from our active variable selection problem.AFA aims to select training set optimally that would result in the best classifier (model), while assume that the test data are fully observed. On the contrary, our framework aims to identify and acquire high value information sequentially for each teat instance.

7.5 Conclusion

In this chapter, we present EDDI, a novel and efficient framework for unsupervised learning and dynamic active variable selection under missing data. Within the EDDI framework, we propose Partial VAE which performs amortized inference to handle missing data. Partial VAE alone can be used as a non-linear computational efficient probabilistic imputation method. Based on it, we design a variable wise acquisition function for EDDI and derive corresponding approximation method. EDDI has demonstrated its effectiveness on active variable selection tasks across multiple real-world applications. In the future, we would extend the EDDI framework to handle more complicated scenarios, such as data missing not at random, time-series, and the cold-start situation.

Appendix for Chapter 7

Appendix 7.A Additional Derivations

7.A.1 Information reward approximation

In our chapter, given the VAE model $p(\mathbf{x}|z)$ and a partial inference network $q(\mathbf{z}|\mathbf{x}_o)$, the experimental design problem is formulated as maximization of the information reward:

$$R(i,\mathbf{x}_o) = \mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)}[\mathrm{D_{KL}}(p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{x}_\phi|\mathbf{x}_o))]$$

Where $p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o) = \int_{\mathbf{z}} p(\mathbf{x}_\phi|\mathbf{z})q(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)$, $p(\mathbf{x}_\phi|\mathbf{x}_o) = \int_{\mathbf{z}} p(\mathbf{x}_\phi|\mathbf{z})q(\mathbf{z}|\mathbf{x}_o)$ and $q(\mathbf{z}|\mathbf{x}_o)$ are approximate condition distributions given by partial VAE models. Now we consider the problem of directly approximating $R(i,\mathbf{x}_o)$.

Applying the chain rule of KL-divergence, we have:

$$\begin{aligned}
&\mathrm{D_{KL}}(p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{x}_\phi|\mathbf{x}_o)) \\
&= \mathrm{D_{KL}}(p(\mathbf{x}_\phi,\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{x}_\phi,\mathbf{z}|\mathbf{x}_o)) \\
&- \mathbb{E}_{\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[\mathrm{D_{KL}}(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right],
\end{aligned}$$

Using again the KL-divergence chain rule on $\mathrm{D_{KL}}(p(\mathbf{x}_\phi,\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{x}_\phi,\mathbf{z}|\mathbf{x}_o))$, we have:

$$\begin{aligned}
&\mathrm{D_{KL}}(p(\mathbf{x}_\phi,\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{x}_\phi,\mathbf{z}|\mathbf{x}_o)) \\
&= \mathrm{D_{KL}}(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_o)) + \mathrm{D_{KL}}(p(\mathbf{x}_\phi|\mathbf{z},\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{x}_\phi|\mathbf{z},\mathbf{x}_o)) \\
&= \mathrm{D_{KL}}(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_o)) + \mathrm{D_{KL}}(p(\mathbf{x}_\phi|\mathbf{z})||p(\mathbf{x}_\phi|\mathbf{z})) \\
&= \mathrm{D_{KL}}(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_o)).
\end{aligned}$$

The KL-divergence term in the reward formula is now rewritten as follows,

$$\begin{aligned}
&\mathrm{D_{KL}}(p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{x}_\phi|\mathbf{x}_o)) \\
&= \mathrm{D_{KL}}(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_o)) \\
&- \mathbb{E}_{\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[\mathrm{D_{KL}}(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right].
\end{aligned}$$

One can then plug in the partial VAE inference approximation:

$$p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o) \approx q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o),$$
$$p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o) \approx q(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o), \;\; p(\mathbf{z}|\mathbf{x}_o) \approx q(\mathbf{z}|\mathbf{x}_o)$$

Finally, the information reward is now approximated as:

$$\begin{aligned}
&R(i,\mathbf{x}_o) \\
&\approx \mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\left[\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||q(\mathbf{z}|\mathbf{x}_o))\right] \\
&-\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right] \\
&= \mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\left[\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||q(\mathbf{z}|\mathbf{x}_o))\right] \\
&-\mathbb{E}_{\mathbf{x}_\phi,\mathbf{x}_i\sim p(\mathbf{x}_\phi,\mathbf{x}_i|\mathbf{x}_o)}\left[\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right] = \hat{R}(i,\mathbf{x}_o).
\end{aligned}$$

This new objective tries to maximize the shift of belief on latent variables $\mathbf{z}$ by introducing $\mathbf{x}_i$, while penalizing the information that cannot be absorbed by $\mathbf{x}_\phi$ (by the penalty term $\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))$). Moreover, it is more computationally efficient since one set of samples $\mathbf{x}_\phi,\mathbf{x}_i \sim p(\mathbf{x}_\phi,\mathbf{x}_i|\mathbf{x}_o)$ can be shared across different terms, and the KL-divergence between common parameterizations of encoder (such as Gaussians and normalizing flows) can be computed exactly without the need for approximate integrals. Note also that under approximation

$$p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o) \approx q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o), \;\; p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o) \approx q(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o), \;\; p(\mathbf{z}|\mathbf{x}_o) \approx q(\mathbf{z}|\mathbf{x}_o)$$

, sampling $\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)$ is approximated by $\mathbf{x}_i \sim \hat{p}(\mathbf{x}_i|\mathbf{x}_o)$, where $\hat{p}(\mathbf{x}_i|\mathbf{x}_o)$ is defined by the following process in Partial VAE. It is implemented by first sampling $\mathbf{z} \sim q(\mathbf{z}|\mathbf{x}_o)$, and then $\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{z})$. The same applies for $p(\mathbf{x}_i,\mathbf{x}_\phi|\mathbf{z})$.

Appendix 7.B Additional Experimental Results

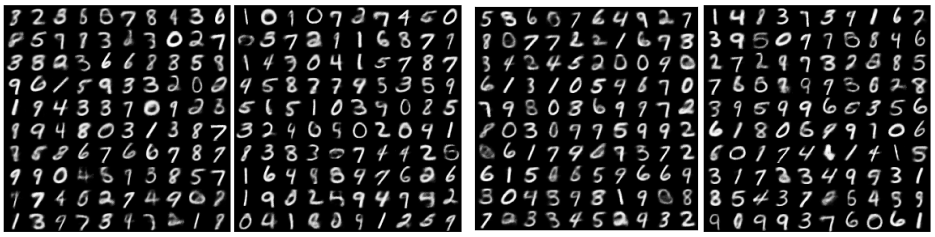
7.B.1 Image inpainting

Preprocessing and model details

For our MNIST experiment, we randomly draw 10% of the whole data to be our test set. Partial VAE models (ZI, ZI-m, PNP and PNs) share the same size of architecture with 20 dimensional diagonal Gaussian latent variables: the generator (decoder) is a 20-200-500-500 fully connected neural network with ReLU activations (where D is the data dimension, $D = 784$). The inference nets (encoder) share the same structure of D-500-500-200-40 that maps the observed data into distributional parameters of the latent space. For the PN-based parameterizations, we use a 500 dimensional feature mapping h parameterized by a single layer neural network, and 20 dimensional ID vectors $\mathbf{e}_i$ (see Section 7.2.1) for each variable. We choose the symmetric operator g to be the basic summation operator.

During training, we apply Adam optimization [148] with default hyperparameter setting, learning rate of 0.001 and a batch size of 100. We generate partially observed MNIST dataset by adding artificially missingness at random in the training dataset during training. We first draw a missing rate parameter from a uniform distribution $\mathcal{U}(0, 0.7)$ and randomly choose variables as unobserved. This step is repeated at each iteration. We train our models for 3K iterations.

Image generation of partial VAEs



(a) (b) (c) (d)

Figure 7.10 Random images generated using **(a)** naive zero imputing, **(b)** zero imputing with mask, **(c)** PN and **(d)** PNP, respectively.

7.B.2 UCI datasets

We applied EDDI on 6 UCI datasets; Boston Housing, Concrete compressive strength, energy efficiency, wine quality, Kin8nm, and Yacht Hydrodynamics. The variables of interest $\mathbf{x}_\phi$ are chosen to be the target variables of each UCI dataset in the experiment.

Preprocessing and model details

All data are normalized and then scaled between 0 and 1. For each of the 10 - in total-repetitions, we randomly draw 10% of the data to be our test set. Partial VAE models (ZI, ZI-m, PNP and PNs) share the same size of architecture with 10 dimensional diagonal

Gaussian latent variables: the generator (decoder) is a 10-50-100-D neural network with ReLU activations (where D is the data dimensions). The inference nets (encoder) share the same structure D-100-50-20 that maps the observed data into distributional parameters of the latent space. For the PN-based parameterizations, we further use a 20 dimensional feature mapping h parameterized by a single layer neural network and 10 dimensional ID vectors $\mathbf{e}_i$ (please refer to section 7.2.1) for each variable. We choose the symmetric operator g to be the basic summation operator.

As in the image inpainting experiment, we apply Adam optimization during training with default hyperparameter setting, and a batch size of 100 and ingest random missingness as before. We trained our models for 3K iterations.

During active learning, we draw 50 samples in order to estimate the expectation under $\mathbf{x}_\phi, \mathbf{x}_i \sim p(\mathbf{x}_\phi, \mathbf{x}_i|\mathbf{x}_o)$ in Equation (7.2.8). Other than information curves based on test RMSEs, we will also provide information curves based on test negative log likelihoods. This will be provided in Appendix 7.B.2. Note that this test nllh of the target variable is also estimated using 50 samples of $\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_o)$. Then, we approximately compute the (expected) log predictive likelihood through $\log p(\mathbf{x}_\phi|\mathbf{x}_o) \approx \log \frac{1}{M}\sum_{m=1}^{M} p(\mathbf{x}_\phi|\mathbf{z}_m)$, where $\mathbf{z}_m \sim q(\mathbf{z}|\mathbf{x}_o)$.

Statistical signifcant test results

In this section, we perform Wilcoxcon signed-rank significance test on the AUIC (RMSE-based) performance of different methods, to support our result in Table 7.2. Since Table 7.2 suggests that EDDI-PNP-Partial VAE is the best algorithm overall, we set EDDI-PNP-Partial VAE as default and perform Wilcoxcon test between EDDI-PNP-Partial VAE and all other 15 different settings, to see whether the improvement is significant. Table 7.6 displays the corresponding p-value for each test. It is obvious that in all 15 tests, the EDDI-PNP-Partial VAE results are significant (compared with the standard $\alpha = 0.05$ cutoff). This provides strong evidence that confirms our results in Table 7.2.

Table 7.6 p- values of Wilcoxon signed-rank test of EDDI-PNP vs. 11 other settings, on 6 UCI datasets, using AUIC (RMSE-based) as evaluation metric.

Method	ZI	ZI-m	PNP	PN
EDDI	$< 10^{-48}$	$< 10^{-23}$	N/A	$< 10^{-2}$
Random	0	0	0	0
Single best	0	$< 10^{-13}$	$< 10^{-2}$	$< 10^{-4}$

Additional plots of PN, ZI and ZI-m on UCI datasets

Here we present additional plots of the RMSE information curves during active learning. Figure 7.11 presents the results for the Boston Housing, the Energy and the Wine datasets and for the three approaches, i.e. PN, ZI and masked ZI.

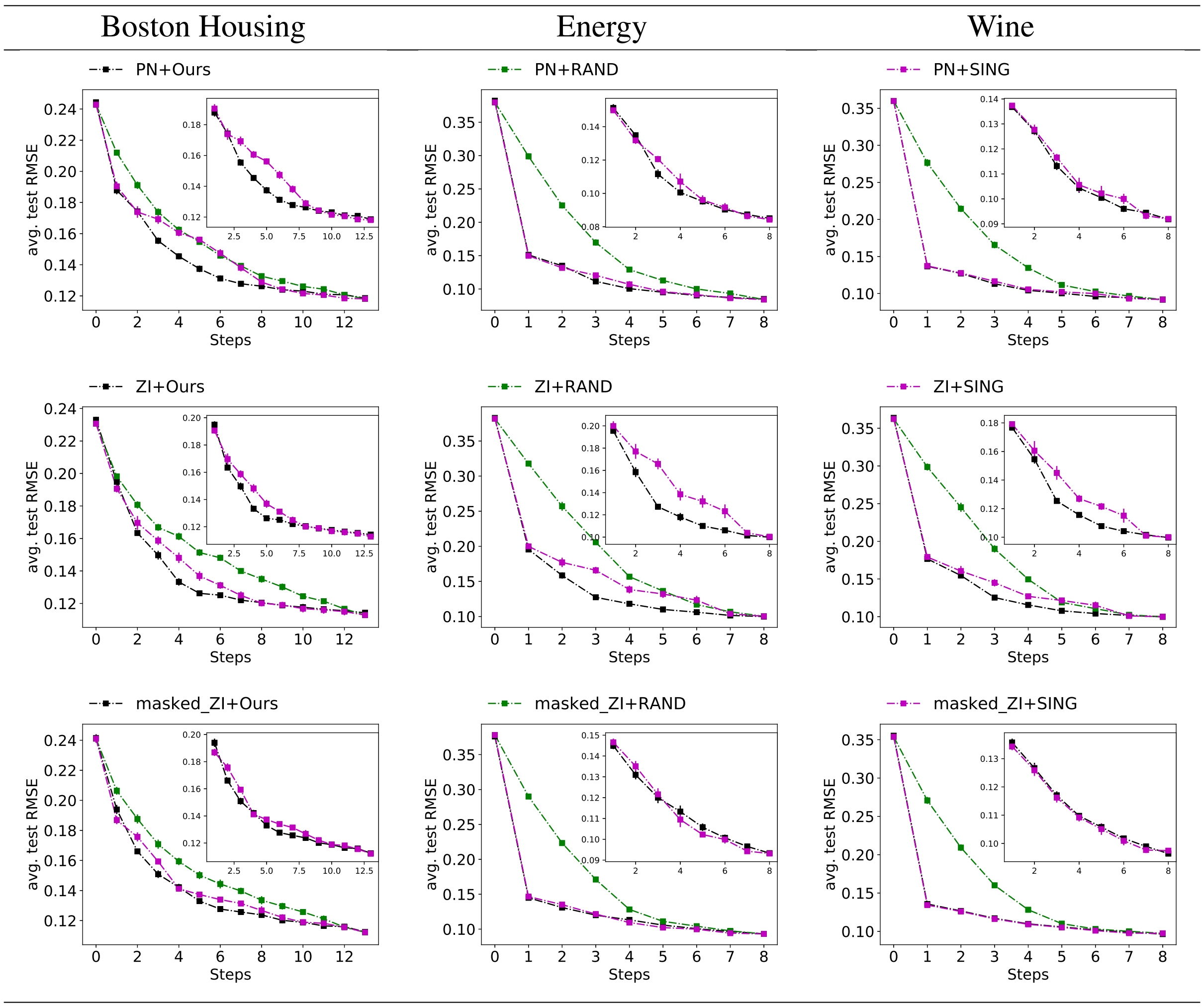


Figure 7.11 information curves (based on RMSE) of active variable selection for the three UCI datasets and the three approaches, i.e. **(First row)** PointNet (PN), **(Second row)** Zero Imputing (ZI), and **(Third row)** Zero Imputing with mask (ZI-m). **Green**: random strategy; **Black**: EDDI; **Pink**: Single best ordering. This displays RMSE (y axis, the lower the better) during the course of active selection (x-axis).

Negative test log Likelihood plots of PN, ZI and ZI-m on UCI datasets

Here we present additional plots of the negative test log likelihood curves during active variable selection. Figure 7.12 presents the results for the Boston Housing, the Energy and the Wine datasets and for the three approaches, i.e. PN, ZI and masked ZI.

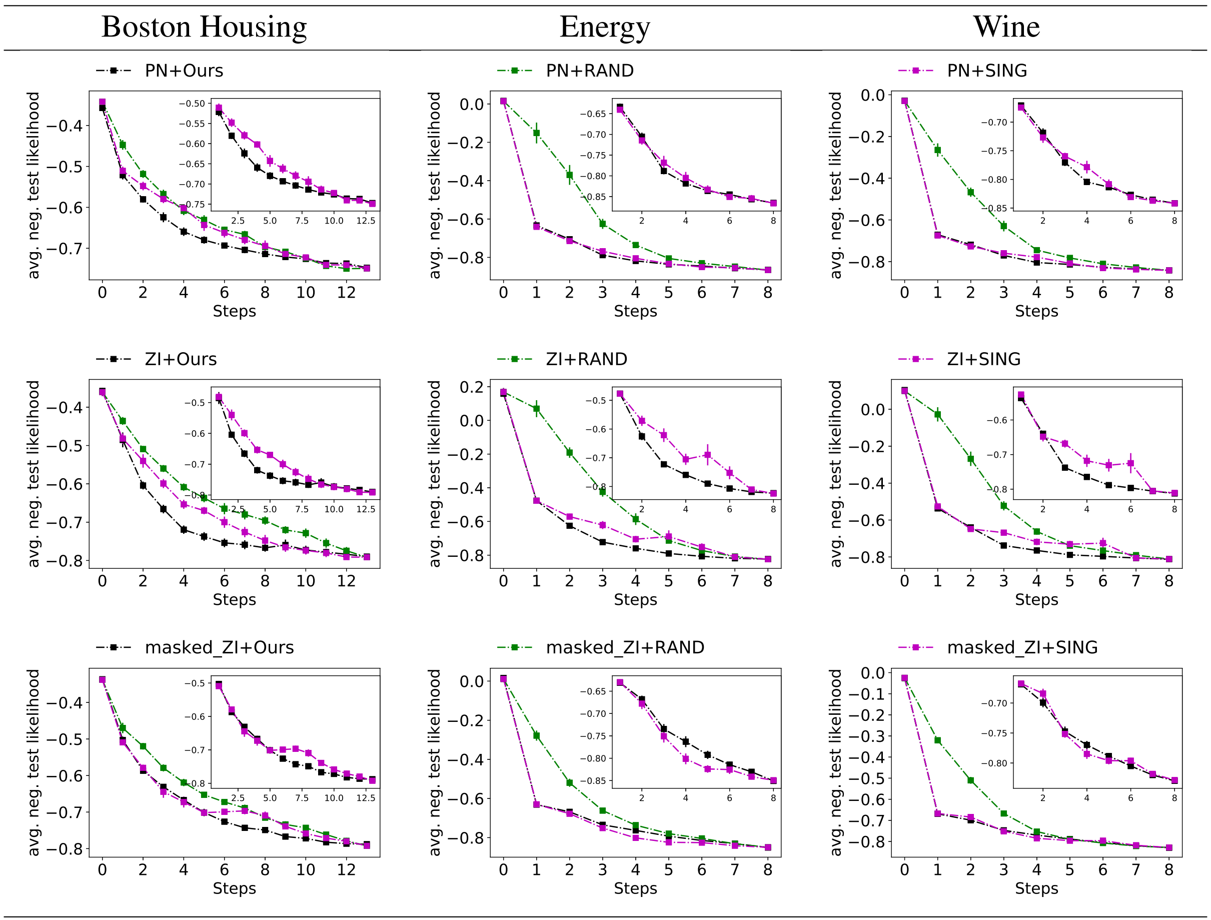


Figure 7.12 Information curves (based on test negative log-likelihood) of active variable selection for the three UCI datasets and the three approaches, i.e. **(First row)** PointNet (PN), **(Second row)** Zero Imputing (ZI), and **(Third row)** Zero Imputing with mask (ZI-m). **Green**: random strategy; **Black**: EDDI; **Pink**: Single best ordering. This displays negative test log likelihood (y axis, the lower the better) during the course of active selection (x-axis).

Comparisons between EDDI and LASSO-based method

Here we present additional results of a new baseline, the LASSO-based feature selection. This is not presented in the main text since LASSO is designed for a different problem setting. It requires fully observed data, and only works in regression problems with one dimensional outputs. Both MIMIC III and NHANES tasks do not fulfill these requirements. Additionally, LASSO aims to select a global set of features to obtain the best performance instead of select the most informative feature given partially observed information, thus cannot be used in a sequential setting. We thus construct the LASSO feature selection baseline as follows for comparison: we first apply LASSO regression on training dataset which is fully observed in these UCI datasets, and select the features (denoted by $\mathcal{A}$) that correspond to non-zero coefficients. Then, during test time, LASSO strategy will observe the features one by one from $\mathcal{A}$ randomly. When all variables selected by LASSO are already picked, we stop the feature selection progress. Once LASSO has completed feature selection, we use we use the corresponding partial-VAE (ZI,ZI-m,PNP,PN) to make predictions for fairness.

Figure 7.13 presents the results for the Boston Housing, the Energy and the Wine datasets as examples. Full results of all UCI datasets are presented in Table 7.7. Note that in Table 7.7, Wilcoxon signed-rank test is performed between EDDI and LASSO strategies for each Partial VAE models, respectively. The results indicates that EDDI significantly outperforms LASSO in all circumstances. This is despite the fact that EDDI is a greedy sequential variable selection method that built upon partially observed data, while LASSO-baseline makes use of the information from *fully observed data*, and selects the set of variables in a *non-greedy, global manner*, which is often unrealistic in many pratical application settings.

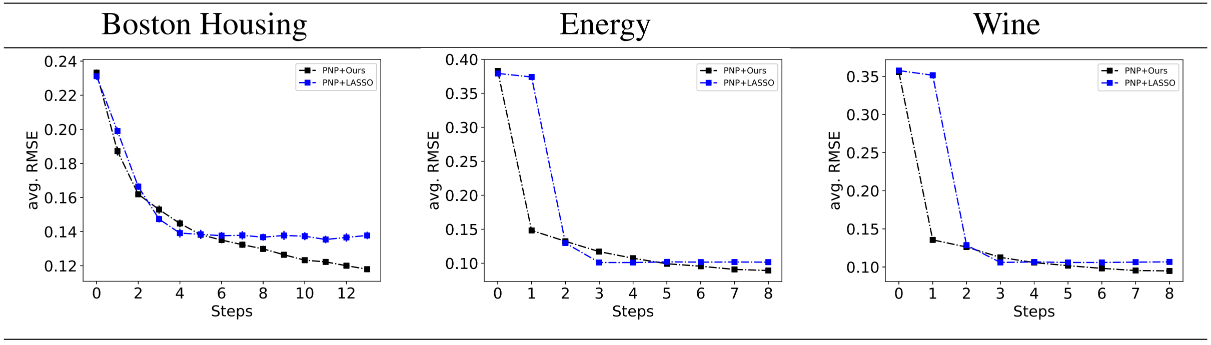


Figure 7.13 Information curves of active variable selection for the three UCI datasets and PNP-Partial VAE. **Black**: EDDI; **Blue**: Single best ordering. This displays test RMSE (y axis, the lower the better) during the course of active selection (x-axis).

Table 7.7 Avg. rankings of AUIC (RMSE-based), and p- values of Wilcoxon signed-rank test that EDDI outperforms LASSO (on 6 UCI datasets).

Method	ZI	ZI-m	PNP	PN
EDDI	4.66 (0.02)	4.53(0.02)	**4.14**(0.02)	4.24(0.02)
LASSO	4.86(0.02)	4.63(0.02)	4.41(0.02)	4.48(0.02)
p-value	$< 10^{-4}$	$< 10^{-6}$	$< 10^{-24}$	$< 10^{-19}$

Illustration of decision process of EDDI (Boston Housing as example)

The decision process facilitated by the active selection of the variables (for the EDDI framework) is efficiently illustrated in Figure 7.14 and Figure 7.15 for the Boston Housing dataset and for the PNP and PNP with single best ordering approaches, respectively.

For completeness, we provide details regarding the abbreviations of the variables used in the Boston dataset and appear both figures.

CR - per capita crime rate by town

PRD - proportion of residential land zoned for lots over 25,000 sq.ft.

PNB - proportion of non-retail business acres per town.

CHR - Charles River dummy variable (1 if tract bounds river; 0 otherwise)

NOC - nitric oxides concentration (parts per 10 million)

ANR - average number of rooms per dwelling

AOUB - proportion of owner-occupied units built prior to 1940

DTB - weighted distances to five Boston employment centres

ARH - index of accessibility to radial highways

TAX - full-value property-tax rate per $10,000

OTR - pupil-teacher ratio by town

PB - proportion of blacks by town

LSP - % lower status of the population

7.B.3 MIMIC-III

Here we provide additional results of our approach on the MIMIC-III dataset.

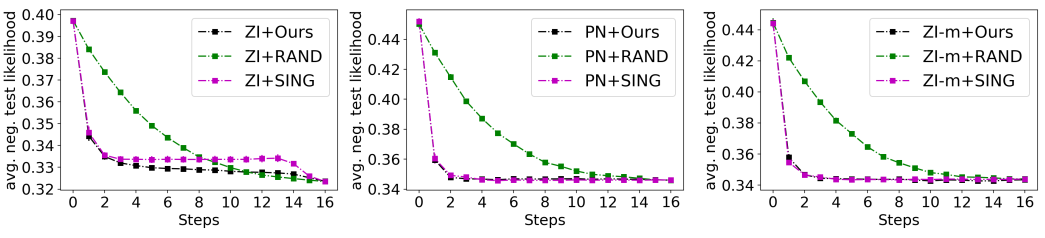
Preprocessing and model details

For our active learning experiments on MIMIC III datasets, we chose the variable of interest $\mathbf{x}_\phi$ to be the binary mortality indicator of the dataset. All data (except the binary mortality indicator) are normalized and then scaled between 0 and 1. We transformed the categorical variables into real-valued using the dictionary deduced from [139] that makes use of the actual medical implications of each possible values. The binary mortality indicator are treated as Bernoulli variables and Bernoulli likelihood function is applied. For each repetition (of the 5 in total), we randomly draw 10% of the whole data to be our test set. Partial VAE models (ZI, ZI-m, PNP and PNs) share the same size of architecture with 10 dimensional diagonal Gaussian latent variables: the generator (decoder) is a 10-50-100-D neural network with ReLU activations (where D is the data dimensions). The inference nets (encoder) share the same structure of D-100-50-20 that maps the observed data into distributional parameters of the latent space. Additionally, for PN-based parameterizations, we further use a 20 dimensional feature mapping h parameterized by a single layer neural network, and 10 dimensional ID vectors $\mathbf{e}_i$ (please refer to section 7.2.1) for each variable. We choose the symmetric operator g to be the basic summation operator.

Adam optimization and random missingness is applied as in the previous experiments. We trained our models for 3K iterations. During active learning, we draw 50 samples in order to estimate the expectation under $\mathbf{x}_\phi, \mathbf{x}_i \sim p(\mathbf{x}_\phi, \mathbf{x}_i|\mathbf{x}_o)$ in Equation (7.2.8). Loss functions (RMSEs and negative log likelihoods) of the target variable is also estimated using samples of $\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_o)$ through $p(\mathbf{x}_\phi|\mathbf{x}_o) \approx \frac{1}{M}\sum_{m=1}^{M} p(\mathbf{x}_\phi|\mathbf{z}_m)$, where $\mathbf{z}_m \sim q(\mathbf{z}|\mathbf{x}_o)$.

Additional plots of ZI, PN and ZI-m on MIMIC III

Figure 7.16 shows the information curves (Bernoulli negative test likelihood-based) of active variable selection on the risk assessment task for MIMIC-III as produced by the three approaches, i.e. ZI, PN and masked ZI.



(a) (b) (c)

Figure 7.16 Information curves of active variable selection on risk assessment task on MIMIC III, produced from: **(a)** Zero Imputing (ZI), **(b)** PointNet (PN) and **(c)** Zero Imputing with mask (ZI-m). **Green**: random strategy; **Black**: EDDI; **Pink**: Single best ordering. This displays negative test log likelihood (y axis, the lower the better) during the course of active selection (x-axis)

.

7.B.4 NHANES

Preprocessing and model details

For our active learning experiments on NHANES datasets, we chose the variable of interest $\mathbf{x}_\phi$ to be the lab test result section of the dataset. All data are normalized and scaled between 0 and 1. For categorical variables, these are transformed into real-valued variables using the code that comes with the dataset, which makes use of the actual ordering of variables in questionnaire. Then, for each repetition (of the 5 repetitions in total), we randomly draw 8000 data as training set and 100 data to be test set. All partial VAE models (ZI, ZI-m, PNP and PNs) uses gaussian likelihoods, with an diagonal Gaussian inference model (encoder). Partial VAE models share the same size of architecture with 20 dimensional diagonal Gaussian latent variables: the generator (decoder) is a 20-50-100-D neural network. The inference nets (encoder) share the same structure of D-100-50-20 that maps the observed data into distributional parameters of the latent space. Additionally, for PN-based parameterizations, we further use a 20 dimensional feature mapping h parameterized by a single layer neural network, and 100 dimensional ID vectors $\mathbf{e}_i$ (please refer to section 7.2.1) for each variable. We choose the symmetric operator g to be the basic summation operator.

Adam optimization and random missingness is applied as in the previous experiments. We trained all models 1K iterations. During active learning, 10 samples were drawn to estimate the expectation in Equation (7.2.9). Losses (RMSEs) of the target variable is also estimated using 10 samples.

Appendix 7.C Additional Theoretical Contributions

7.C.1 Zero imputing as a Point Net

Here we present how the zero imputing (ZI) and PointNet (PN) approaches relate.

Zero imputation with inference net In ZI, the natural parameter of λ (e.g., Gaussian parameters in variational autoencoders) is approximated using the following neural network:

$$f(\mathbf{x}) := \sum_{l=1}^{L} w_l^{(1)} \sigma(\mathbf{w}_l^{(0)} \mathbf{x}^T)$$

,

where L is the number of hidden units, $\mathbf{x}$ is the input image with x_i be the value of the i^{th} pixel. To deal with partially observed data $\mathbf{x} = \mathbf{x}_o \cup \mathbf{x}_u$, ZI simply sets all $\mathbf{x}_u$ to zero, and use the full inference model $f(\mathbf{x})$ to perform approximate inference.

PointNet parameterization The PN approach approximates the natural parameter λ by a permutation invariant set function

$$g(h(\mathbf{s}_1), h(\mathbf{s}_2), ..., h(\mathbf{s}_O)),$$

where $\mathbf{s}_i = [x_i, \mathbf{e}_i]$, $\mathbf{e}_i$ is the I dimensional embedding/ID/location vector of the i^{th} pixel, $g(\cdot)$ is a symmetric operation such as max-pooling and summation, and $h(\cdot)$ is a nonlinear feature mapping from $\mathbb{R}^{I+1}$ to $\mathbb{R}^K$ (we will always refer h as *feature maps*). In the current version of the partial-VAE implementation, where Gaussian approximation is used, we set $K = 2H$ with H being the dimension of latent variables. We set g to be the element-wise summation operator, i.e. a mapping from $\mathbb{R}^{KO}$ to $\mathbb{R}^K$ defined by:

$$g(h(\mathbf{s}_1), h(\mathbf{s}_2), ..., h(\mathbf{s}_O)) = \sum_{i \in O} h(\mathbf{s}_i).$$

This parameterization corresponds to products of multiple Exp-Fam factors $\prod_{i \in O} \exp\{-\langle h(\mathbf{s}_i), \Phi \rangle\}$.

From PN to ZI To derive the PN correspondence of the above ZI network we define the following PN functions:

$$h(\mathbf{s}_i) := \mathbf{e}_i * x_i$$

$$g(h(\mathbf{s}_1), h(\mathbf{s}_2), ..., h(\mathbf{s}_O)) := \sum_{k=1}^{I} \theta_k \sigma(\sum_{i \in O} h_k(\mathbf{s}_i)),$$

where $h_k(\cdot)$ is the k^{th} output feature of $h(\cdot)$. The above PN parameterization is also permutation invariant; setting $L = I$, $\theta_l = w_l^{(1)}, (\mathbf{w}_l^{(0)})_i = (\mathbf{e}_i)_l$ the resulting PN model is equivalent to the ZI neural network.

Generalizing ZI from PN perspective In the ZI approach, the missing values are replaced with zeros. However, this ad-hoc approach does not distinguish missing values from actual observed zero values. In practice, being able to distinguish between these two is crucial for improving uncertainty estimation during partial inference. One the other hand, we have found that PN-based partial VAE experiences difficulties in training. To alleviate both issues, we proposed a generalization of the ZI approach that follows a PN perspective. One of the advantages of PN is setting the *feature maps* of the unobserved variables to zero instead of the related weights. As discussed before, these two approaches are equivalent to each other only if the factors are linear. More generally, we can parameterize the PN by:

$$h^{(1)}(\mathbf{s}_i) := \mathbf{e}_i * x_i$$

$$h^{(2)}(h_i^{(1)}) := NN_1(h_i^{(1)})$$

$$g(h(\mathbf{s}_1), h(\mathbf{s}_2), ..., h(\mathbf{s}_O)) := NN_2(\sigma(\sum_{i \in O} h_k^{(2)}(h_i^{(1)}))),$$

where NN_1 is a mapping from $\mathbb{R}^I$ to $\mathbb{R}^K$ defined by a neural network, and NN_2 is a mapping from $\mathbb{R}^K$ to $\mathbb{R}^{2H}$ defined by another neural network.

7.C.2 Approximation Difficulty of the Acquisition Function

Traditional variational approximation approaches provide wrong approximation direction when applied in this case (resulting in an upper bound of the objective $R_\phi(i, \mathbf{x}_O)$ which we maximize). Justification issues aside, (black box) variational approximation requires sampling from approximate posterior $q(\mathbf{z}|\mathbf{x}_O)$, which leads to extra uncertainties and computations. For common proposals of approximation:

- Directly estimate entropy via sampling $\Rightarrow$ problematic for high dimensional target variables
- Using reversed information reward $\mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)}[\mathrm{D}_{\mathrm{KL}}(p(\mathbf{x}_\phi|\mathbf{x}_o)||p(\mathbf{x}_\phi|\mathbf{x}_o, \mathbf{x}_i))]$, and then apply ELBO (KL-divergence) $\Rightarrow$ This does not make sense mathematically, since this

will result in upper bound approximation of the (reversed) information objective, this is in the wrong direction.

- Ranganath's bound [268] on estimating entropy$\Rightarrow$ gives upper bound of the objective, wrong direction.
- All the above methods also needs samples from latent space (therefore second level approximation needed).

7.C.3 Connection of EDDI information reward with BALD

We briefly discuss connection of EDDI information reward with BALD [121] and. MacKay's work [203]. Assuming the model is correct, i.e. $q = p$, we have

$$\begin{aligned}R(i,\mathbf{x}_o) = &\,\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\left[\mathrm{D_{KL}}(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_o))\right]\\ &-\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[\mathrm{D_{KL}}(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right].\end{aligned}$$

Note that based on McKay's relationship between entropy and KL-divergence reduction, we have:

$$\begin{aligned}&\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\left[\mathrm{D_{KL}}(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_o))\right]\\ =&\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o))-H(p(\mathbf{z}|\mathbf{x}_o))\right].\end{aligned}$$

Similarly, we have

$$\begin{aligned}&\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[\mathrm{D_{KL}}(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right]\\ =&\mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_\phi,\mathbf{x}_o)}\left[\mathrm{D_{KL}}(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right]\\ =&\mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_\phi,\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o))-H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right]\\ =&\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o))\right]-\mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_\phi,\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right]\\ =&\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o))\right]-\mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right],\end{aligned}$$

where MacKay's result is applied to $\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_\phi,\mathbf{x}_o)}\left[\mathrm{D_{KL}}(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right]$. Putting everything together, we have

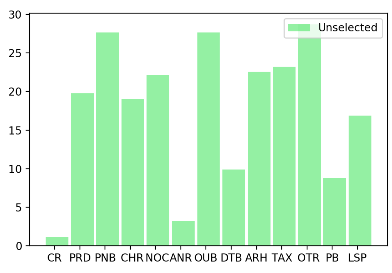
$$\begin{aligned}
R(i,\mathbf{x}_o) &= \mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)) - H(p(\mathbf{z}|\mathbf{x}_o))\right]] \\
&\quad - \mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o))\right] + \mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right] \\
&= \left\{\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o))\right] - \mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o))\right]\right\} \\
&\quad - \left\{\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_o))\right] - \mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right]\right\}.
\end{aligned}$$

We can show that

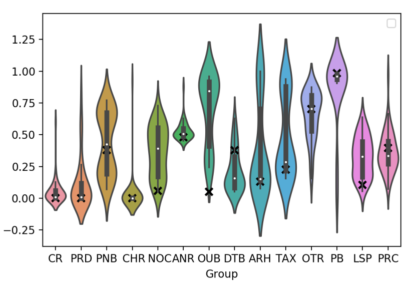
$$\begin{aligned}
&H(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)) - \mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o))\right] \\
=& -\int_{\mathbf{z}} p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)\log p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)d\mathbf{z} + \int_{\mathbf{z},\mathbf{x}_\phi} p(\mathbf{z},\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)\log p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o) \\
=& \int_{\mathbf{z},\mathbf{x}_\phi} p(\mathbf{z},\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)\log\frac{p(\mathbf{z},\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}{p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)} \\
=& \mathcal{I}\left[\mathbf{z},\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o\right],
\end{aligned}$$

which is exactly the conditional mutual information $\mathcal{I}\left[\mathbf{z},\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o\right]$ used in BALD. Therefore, our chain rule representation of reward function leads us to

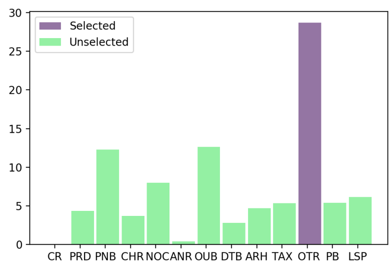
$$R(i,\mathbf{x}_o) = \mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathcal{I}\left[\mathbf{z},\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o\right] - \mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathcal{I}\left[\mathbf{z},\mathbf{x}_\phi|\mathbf{x}_o\right].$$



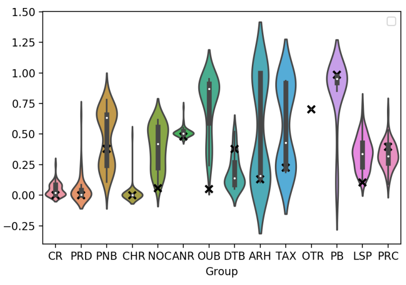
(a)



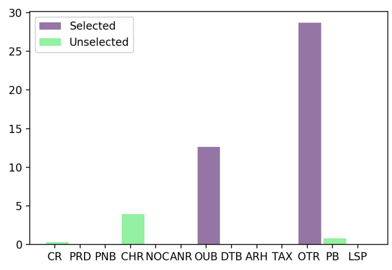
(b)



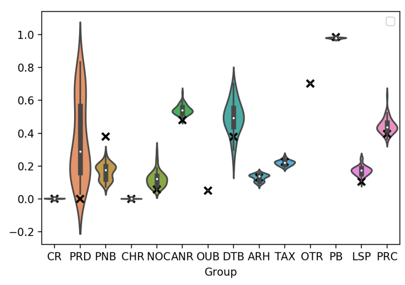
(c)



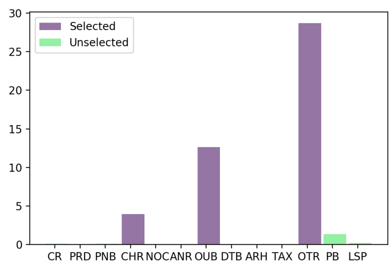
(d)



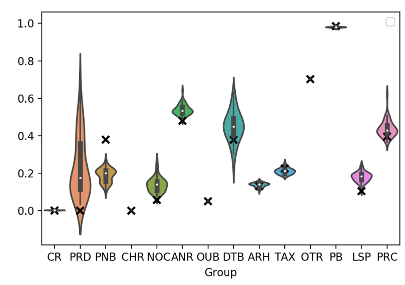
(e)



(f)



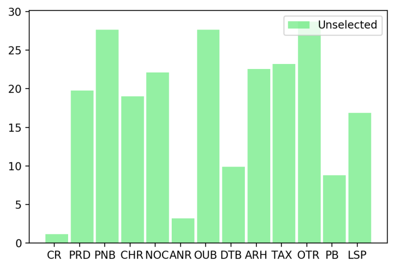
(g)



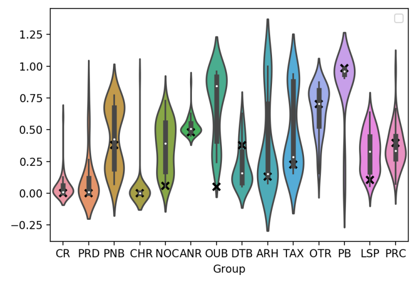
(h)

Figure 7.14 Information reward estimated during the first 4 active variable selection steps on a randomly chosen Boston Housing test data point. **Model**: PNP, strategy: EDDI. Each row contains two plots regarding the same time step. **Bar plots on the left** show the information reward estimation of each variable on the y-axis. All unobserved variables start with green bars, and turns purple once selected by the algorithm. **Right**: violin plot of the posterior density estimations of remaining unobserved variables.

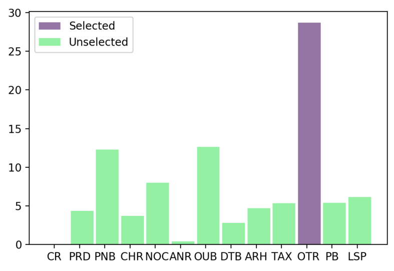
.



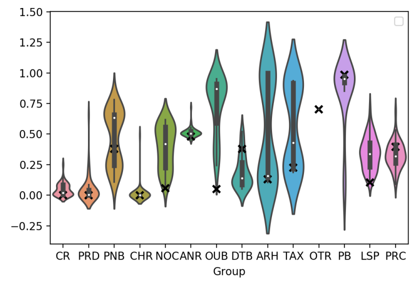
(a)



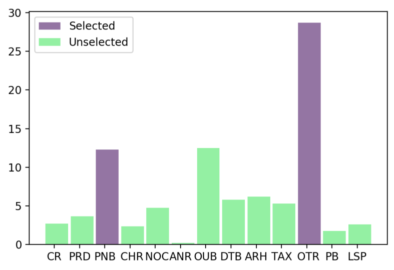
(b)



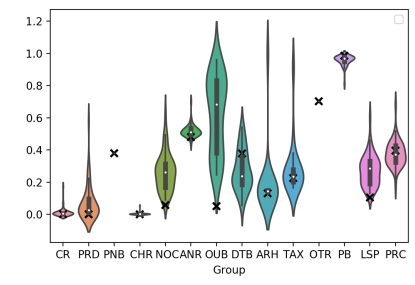
(c)



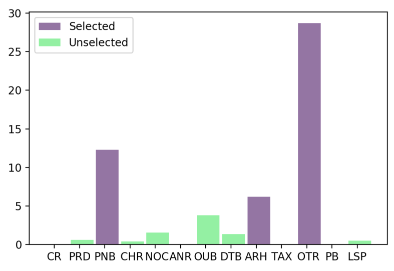
(d)



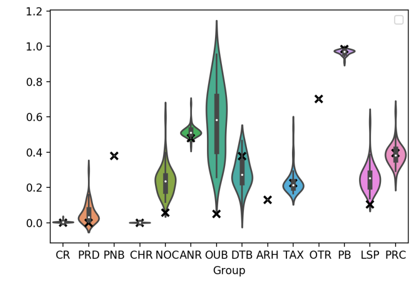
(e)



(f)



(g)



(h)

Figure 7.15 Information reward estimated during the first 4 active variable selection steps on a randomly chosen Boston Housing test data point. **Models**: PNP, strategy: single ordering. Each row contains two plots regarding the same time step. **Bar plots on the left** show the information reward estimation of each variable on the y-axis. All unobserved variables start with green bars, and turns purple once selected by the algorithm. **Right**: violin plot of the posterior density estimations of remaining unobserved variables.

.

Chapter 8

Identifiable Generative Models Under Missing Not at Random Data

So far, our discussions have been based on the assumption that the missing data follows a MAR (missing at random) mechanism. However, real-world datasets often have missing values associated with complex generative processes, where the cause of the missingness may not be fully observed. This is known as missing not at random (MNAR) data. Although there are many relevant methods in the literature that have considered the MNAR scenario, their model's identifiability under MNAR is generally not guaranteed. That is, model parameters can not be uniquely determined even with infinite data samples. Therefore, as discussed in Chapter 6 Section 6.3, lack of model identifiability might introduce additional biases in missing data imputation. This issue is especially overlooked by many modern deep generative models. In this Chapter, we fill in this gap by systematically analyzing the identifiability of generative models under MNAR. Furthermore, we propose a practical deep generative model which can provide identifiability guarantees under mild assumptions, for a wide range of MNAR mechanisms.

8.1 Introduction

Missing data is an obstacle in many data analysis problems, which may seriously compromise the performance of machine learning models, as well as downstream tasks based on these models. Being able to successfully recover/impute missing data in an unbiased way is the key to understanding the structure of real-world data. This requires us to identify the underlying data-generating process, as well as the probabilistic mechanism that decides which data is missing.

In general, there are three types of missing mechanisms [279]. The first type is missing completely at random (MCAR), where the probability of a data entry being missing is independent of both the observed and unobserved data (Figure 8.1 (a)). In this case, no statistical bias is introduced by MCAR.The second type is missing at random (MAR), which assumes that the missing data mechanism is independent of the value of unobserved data (Figure 8.1 (b)). Under this assumption, maximum likelihood learning methods without explicit modeling of the missingness mechanism can be applied by marginalizing out the missing variables [55, 197, 200]. However, both MCAR and MAR do not hold in many real-world applications, such as recommender systems [112, 136], healthcare [134], and surveys [312]. For example, in a survey, participants with financial difficulties are more likely to refuse to complete the survey about financial incomes. This is an example of missing not at random (MNAR), where the cause of the missingness (financial income) can be unobserved. In this case, ignoring the missingness mechanism will result in biased imputation, which will jeopardize down-stream tasks.

There are few works considering the MNAR setting in scalable missing value imputation. On the one hand, many practical methods for MNAR does not have identifiability guarantees [128, 112, 190]. That is, the parameters can not be uniquely determined, even with access to infinite samples [219, 278]. As a result, missing value imputation based on such parameter estimation could be biased (Chapter 6, Section 6.3). On the other hand, there are theoretical analyses on the identifiability in certain scenarios [219–221, 233, 321, 332, 352], but without associated practical algorithms for flexible and scalable settings (such as deep generative models). Moreover, MNAR data have many possible cases (Figure 8.1) based on different independence assumptions [233], making the discussion of identifiability difficult. This motivates us to fill this gap by extending identifiability results of deep generative models to different missing mechanisms, and provide a scalable practical solution. Our contribution are threefold:

- We provide a theoretical analysis of identifiability for generative models under different MNAR scenarios (Section 8.3). More specifically, we provide sufficient conditions, under which the ground truth parameters can be uniquely identified via maximum likelihood (ML) learning using observed information [190]. We also demonstrate how the assumptions can be relaxed in the face of real-world datasets. This provides foundation for practical solutions using deep generative models.

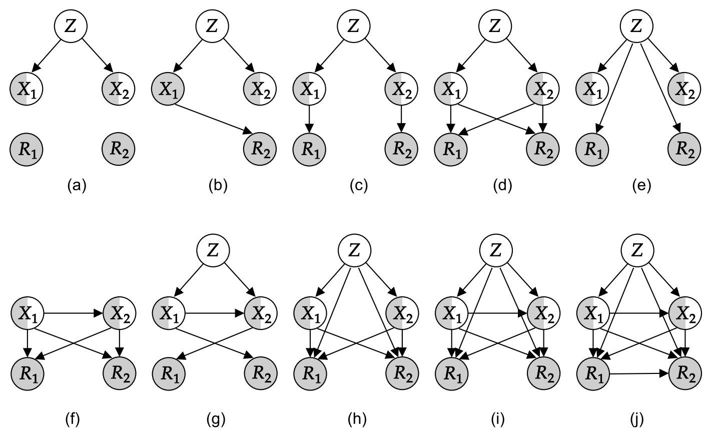


Figure 8.1 Exemplar missing data situations. **(a)**: MCAR; **(b)**: MAR; **(c)-(i)**: MNAR.

- Based on our analysis, we propose a practical algorithm model based on VAEs (Section 8.4), named GINA (deep generative imputation model for missing not at random). This enables us to apply flexible deep generative models in a principled way, even in the presence of MNAR data.
- We demonstrate the effectiveness and validity of our approach by experimental evaluations (Section 8.6) on both synthetic data modeling, missing data imputation in real-world datasets, as well as downstream tasks such as active feature selection under missing data.

8.2 Backgrounds

8.2.1 Problem Setting

A critical component to develop model to impute MNAR data is the model identifiablity [154, 278]. We give the definition below:

Definition 8.2.1 (Model identifiability)**.** *Assume $p_{\boldsymbol{\theta}}(X)$ is a distribution of some random variable X, $\boldsymbol{\theta}$ is its parameter that takes values in some parameter space $\boldsymbol{\Omega_\theta}$. Then, if $p_{\boldsymbol{\theta}}(X)$ satisfies $p_{\boldsymbol{\theta}_1}(X) \neq p_{\boldsymbol{\theta}_2}(X) \iff \boldsymbol{\theta}_1 \neq \boldsymbol{\theta}_2, \forall \boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \boldsymbol{\Omega_\theta}$, we say that $p_{\boldsymbol{\theta}}$ is identifiable w.r.t. $\boldsymbol{\theta}$ on $\boldsymbol{\Omega_\theta}$.*

In other words, a model $p_{\boldsymbol{\theta}}(X)$ is identifiable, if different parameter configurations implies a different probabilistic distributions over the observed variables. With identifiability guarantee, if the model assumption is correct, the true generation process can be recovered.

Next, we introduce necessary notations and of missing data, and set up a concrete problem setting.

Basic Notation In this Chapter, we will use a notation system that is slightly different from Chapter 7, which will allow us to denote different quantities more accurately. Similar to the notations introduced by [128, 279], let X be the complete set of variables in the system of interest. We call it *observable variables*. Let $\mathcal{I} = \{1, ..., D\}$ be the *index set* of all observable variables, i.e., $X = \{X_i | i \in \mathcal{I}\}$. Let X_O denote the set of actually *observed variables*, here $O \in \mathcal{I}$ is a index set such that $X_O \subset X$. We call O the *observable pattern*. Similarly, X_U denotes the set of *missing/unobserved variables*, and $X = X_O \bigcup X_U$. Additionally, we use R to denote the missing mask indicator variable, such that $R_i = 1$ indicates X_i is observed, and $R_i = 0$ indicates otherwise. We call a probabilistic distribution $p(X)$ on X the *reference distribution*, that is, the distribution that we would have observed if no missing mechanism is present; and we call the conditional distribution $p(R|X)$ the *missing mechanism*, which decides the probability of each X_i being missing. Then, we can define the marginal distribution of *partially observed variables*, which is given by $\log p(X_O, R) = \log \int_{X_U} p(X_O, X_U, R) dX_U$. Finally, we will use lowercase vectors to denote the *realized values* of the corresponding random variable. For example, $(\mathbf{x}_O, \mathbf{r}) \sim p(X_O, R)$ is the realization/samples of X_O and R, and the dimensionality of $\mathbf{x}_O$ may vary for each realizations.

Problem setting Suppose that we have a ground truth data generating process, denoted by $p_{\mathcal{D}}(X_O, R)$, from which we can obtain (partially observed) samples $(\mathbf{x}_O, r) \sim p_{\mathcal{D}}(X_O, R)$. We also have a model to be optimized, denoted by $p_{(\boldsymbol{\theta}, \boldsymbol{\psi})}(X_O, X_U, R)$,where $\boldsymbol{\theta}$ is the parameter of reference distribution $p_{\boldsymbol{\theta}}(X)$, and $\boldsymbol{\psi}$ the parameter of missing mechanism $p_{\boldsymbol{\psi}}(R|X)$. Our goal can then be described as follows:

- To establish the identifiability of the model $p_{(\boldsymbol{\theta}, \boldsymbol{\psi})}(X_O, R)$. That is, we wish to uniquely and correctly identify $\hat{\boldsymbol{\theta}}$, such that $p_{\hat{\boldsymbol{\theta}}}(X) = p_{\mathcal{D}}(X)$, given infinite amount of partially observed data samples from ground truth, $(\mathbf{x}_O, r) \sim p_{\mathcal{D}}(X_O, R)$.

- Then, given the identified parameter, we will be able to perform missing data imputation, using $p_{\hat{\boldsymbol{\theta}}}(X_U | X_O)$. If our parameter estimate is unbiased, then our imputation is also unbiased, that is, $p_{\hat{\boldsymbol{\theta}}}(X_U | X_O) = p_{\mathcal{D}}(X_U | X_O)$ for all possible configurations of X_O.

8.2.2 Dealing with MNAR: Rubin's approach, and challenges

Recall the three types of missing mechanisms: if data is MCAR, $p(R|X) = p(R)$; if data is MAR, $p(R|X) = p(R|X_O)$; otherwise, we call it MNAR. When missing data is MCAR or MAR, missing mechanism can be ignored when performing maximum likelihood (ML) inference based only on the observed data [279], as:

$$\arg\max_{\boldsymbol{\theta}} \mathbb{E}_{(\mathbf{x}_O,\mathbf{r})\sim p_{\mathcal{D}}(X,R)} \log p_{\boldsymbol{\theta}}(X_O = \mathbf{x}_O) = \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{(\mathbf{x}_O,\mathbf{r})\sim p_{\mathcal{D}}(X,R)} \log p_{\boldsymbol{\theta}}(X_O = \mathbf{x}_O, R = \mathbf{r})$$

where $\log p(X_O) = \log \int_{X_U} p(X_O, X_U) dX_U$.In practice, ML learning on X_O can done by EM algorithm [55, 190]. However, when missing data is MNAR, the above argument does not hold, and the missing data mechanism cannot be ignored during learning. Consider the representative graphical model example in Figure 8.1 (d), which has appeared in many context of machine learning. In this graphical model, X is the cause of R, and the connections between X and R are fully connected, i.e., each single node in R are caused by the entire set X. All nodes in R are conditionally independent from each other given X.

Clearly, this is an example of a data generating process with MNAR mechanism. In this case, Rubin proposed to jointly optimize both the reference distribution $p_{\boldsymbol{\theta}}(X)$ and the missing data mechanism $p_{\boldsymbol{\psi}}(R|X)$, by maximizing:

$$\arg\max_{\boldsymbol{\theta},\boldsymbol{\psi}} \mathbb{E}_{(\mathbf{x}_O,\mathbf{r})\sim p_{\mathcal{D}}(X,R)} \log p_{(\boldsymbol{\theta},\boldsymbol{\psi})}(X_O = \mathbf{x}_O, R = \mathbf{r}) \tag{8.2.1}$$

This factorization is referred as *selection modeling* [128, 190]. There are multiple challenges if we want to Eq. 8.2.1 to obtain a practical model that provide unbiased imputation. First, we need model assumption to be consistent with the real-world data generation process, $p_{\mathcal{D}}(X_O, R)$. Given a wide range of possible MNAR scenarios, it is a challenge to design a general model. Secondly, the model need to be identifiable to enable the possibility to learn the underlying process which leads to unbiased imputation.

8.2.3 Variational Autoencoders and its identifiability

As introduced in Chapter 2, recall that Variational auto-encdoers [149, 273] are flexible deep generative models that are commonly used for estimating densities of $p_{\mathcal{D}}(X)$. It takes the following form:

$$\log p_{\boldsymbol{\theta}}(X) = \log \int_Z dZ p_{\boldsymbol{\theta}}(X|Z) p(Z), \tag{8.2.2}$$

where Z is some latent variable model with prior $p(Z)$, and $p_{\boldsymbol{\theta}}(X|Z)$ is given by $p_{\boldsymbol{\theta}}(X|Z) = \mathcal{N}(f_\theta(Z), \sigma)$, with $f_\theta(\cdot)$ being a neural network parameterized by θ. Generally, VAEs do not have identifiability guarantees w.r.t. θ [146]. Nevertheless, inspired by the identifiablity of nonlinear ICA, [146] shows that the identifiability of VAE can be established up to equivalence permutation under mild assumptions, if the unconditional prior $p(Z)$ of VAE is replaced by the following the conditionally factorial exponentially family prior,

$$p_{T,\zeta}(Z|U) \propto \prod_{i=1} Q(Z_i) \exp[\sum_{j=1}^{K} T_{i,j}(Z_i)\zeta_{i,j}(U)], \tag{8.2.3}$$

where U is some additional observations (called auxiliary variables), $Q(Z_i)$ is some base measure, $\mathbf{T}_i(U) = (T_{i,1}, ..., T_{i,K})$ the sufficient statistics, and $\boldsymbol{\zeta}_i(U) = (\zeta_{i,1}, ..., \zeta_{i,K})$ the corresponding natural parameters. Then, the new VAE model given by

$$\log p_{\boldsymbol{\theta}}(X|U) = \log \int_Z dZ p_{\boldsymbol{\theta}}(X|Z) p_{T,\zeta}(Z|U) \tag{8.2.4}$$

is identifiable (Theorem 1 and 2 of [146], see Appendix 8.G).We call the model (8.2.4) the identifiable VAE. Unfortunately, this identifiability results for VAE only hold when all variables are fully observed; thus, it cannot be immediately applied to address the challenges of dealing with MNAR data stated in Section 8.2.2. Next, we will analyze the identifiablity of generative models under general MNAR settings (Section 8.3), and propose a practical method that can be used in MNAR (Section 8.4).

8.3 Establishing model identifiability under MNAR

One key issue of training probabilistic models under MNAR missing data is its identifiability. Recall that (Definition 8.2.1) model identifiability characterize the property that the mapping from parameter $\boldsymbol{\theta}$ to the distribution $p_{\boldsymbol{\theta}}(X)$ is one-to-one. This is often closely related to maximum likelihood learning. In fact, it is not hard to show that Definition 8.2.1 is equivalent to the following Definition 8.3.1:

Definition 8.3.1 (Equivalent definition of identifiability)**.** *We say a model $p_{\boldsymbol{\theta}}(X)$ is identifiable, if:*

$$\arg\max_{\boldsymbol{\theta} \in \boldsymbol{\Omega}_{\boldsymbol{\theta}}} \mathbb{E}_{x \sim p_{\boldsymbol{\theta}^*}(X)} \log p_{\boldsymbol{\theta}}(X = x) = \boldsymbol{\theta}^*, \quad \forall \boldsymbol{\theta}^* \in \boldsymbol{\Omega}_{\boldsymbol{\theta}} \tag{8.3.1}$$

In other words, the "correct" model parameter $\boldsymbol{\theta}^*$ can be identified via maximum likelihood learning (under complete data), and the ML solution is unbiased. Similarly, when

MNAR missing mechanism is present, we perform maximum likelihood learning on both X_O and R using Eq. 8.2.1. Thus, we need $\log p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,R)$ to be identifiable under MNAR, so that we can correctly identify the ground truth data generating process, and achieve unbiased imputation. The identifiability of $\log p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,R)$ under MNAR is usually not guaranteed, even in some simplistic settings [219]. In this section, we will give the sufficient conditions for model identifiability under MNAR, and study how these can be relaxed for real-world applications

8.3.1 Sufficient conditions for identifiability under MNAR

In this section, we give sufficient conditions where the model parameters $\boldsymbol{\theta}$ can be uniquely identified by Rubin's objective, Eq. 8.2.1. Our aim is to i), find a set of model assumptions, so that it can cover many common scenarios and be flexible for practical interests; and ii), under those conditions, we want to show that its parameters can be uniquely determined by the partial ML solution Eq. 8.2.1. As shown in Figure 8.1, MNAR have many possible difference cases depending on its graphical structures. We want our results to cover every situation.

Instead of doing case by case analysis, we will start our identifiability anaylsis with one fairly general case as the example shown in Figure 8.1 (h) where the missingness can be caused by other partially observed variable, by itself (self-masking) or by latent variables. Then, we will discuss how these analysis can be applied to other MNAR scenarios in Section 8.3.2.

Data setting D1 Suppose the ground truth data generation process satisfies the following conditions: all variables X are generated from a shared latent confounder Z, and there are no connections among X; and the missingness indicator R variable can not be the parent of other variables. A typical example of such distribution is depicted in Figure 8.1 (h). We further assume that $p_{\mathcal{D}}(X_O,X_U,R)$ has the following parametric form: $p_{\mathcal{D}}(X_O,X_U,R) = \int_Z \prod_d p_{\boldsymbol{\theta}_d^*}(X_d|Z)p(Z)p_{\boldsymbol{\psi}^*}(R|X,Z)dZ$, where $p_{\boldsymbol{\psi}^*}(R|X,Z) = \prod_d p_{\boldsymbol{\psi}_d^*}(R_d|X,Z)$, for some $\boldsymbol{\theta}^*, \boldsymbol{\psi}^*$.

Then, consider the following model:

Model assumption A1. We assume that our model has the same graphical representation, as well as parametric form as **data setting D1**, that is, our model can be written as:

$$p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,R) = \int_{X_U,Z} dX_U dZ \prod_d p_{\boldsymbol{\theta}_d}(X_d|Z) \prod_d p_{\boldsymbol{\psi}_d}(R_d|X,Z)p(Z) \tag{8.3.2}$$

Here, $(\boldsymbol{\theta}, \boldsymbol{\psi}) \in \boldsymbol{\Omega}$ are learnable parameters that belong to some parameter space $\boldsymbol{\Omega} = \boldsymbol{\Omega}_{\boldsymbol{\theta}} \times \boldsymbol{\Omega}_{\boldsymbol{\psi}}$. Each $\boldsymbol{\theta}$ is the parameter that parameterizes the conditional distribution that connects X_d and Z, $p_{\boldsymbol{\theta}_d}(X_d|Z)$. Assume that the ground truth parameter of $p_{\mathcal{D}}$ belongs to the model parameter space, $(\boldsymbol{\theta}^*, \boldsymbol{\psi}^*) \in \boldsymbol{\Omega}$.

Given such a model, our goal is to correctly identify the ground truth parameter settings given partially observed samples from $p_{\mathcal{D}}(X_O, X_U, R)$. That is, let

$$(\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\psi}}) = \arg \max_{(\boldsymbol{\theta}, \boldsymbol{\psi}) \in \boldsymbol{\Omega}} \mathbb{E}_{(\mathbf{x}_O, \mathbf{r}) \sim p_{\mathcal{D}}(X,R)} \log p_{(\boldsymbol{\theta}, \boldsymbol{\psi})}(X_O = \mathbf{x}_O, R = \mathbf{r}),$$

we would like to achieve $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}^*$. In order to achieve this, we must make additional assumptions.

Assumption A2. Subset identifiability: There exist a partition[1] of $\mathcal{I}$, denoted by $\mathcal{A}_{\mathcal{I}} = \{C_s\}_{1 \leq s \leq S}$, such that: for all $C_s \in \mathcal{A}_{\mathcal{I}}$, $p_{\boldsymbol{\theta}}(X_{o_s})$ is identifiable on a subset of parameters $\{\boldsymbol{\theta}_d | d \in C_s\}$.

This assumption basically formalizes the idea of divide and conquer: we partition the whole index set into several smaller subsets $\{C_s\}_{1 \leq s \leq S}$, on which each reference distribution $p_{\boldsymbol{\theta}}(X_{C_s})$ is only responsible for the identifiability on a subset of parameters.

Assumption A3. There exists a collection of observable patterns, denote by $\bar{\mathcal{A}}_{\mathcal{I}} := \{C'_l\}_{1 \leq l \leq L}$, such that: 1), $\bar{\mathcal{A}}_{\mathcal{I}}$ is a cover [1] of $\mathcal{I}$; 2), $p_{\mathcal{D}}(X, R_{C'_l} = 1, R_{\mathcal{I} \setminus C'_l})$ (hence any of its marginal distributions) is positive for all X and $1 \leq l \leq L$; and 3), for all index $c \in C'_l$, there exists $C_s \in \mathcal{A}_{\mathcal{I}}$ defined in **A2**, such that $c \in C_s \subset C'_l$.

This assumption is about the strict positivity of the ground truth data generating process, $p_{\mathcal{D}}(X_O, X_U, R)$. Instead of assuming that complete case data are available as in [233], here we assume we should at least have some observations, $p_{\mathcal{D}}(X, R_O = 1, R_U = 0) > 0$ for $O \in \hat{\mathcal{A}}_{\mathcal{I}}$, on which $p_{\boldsymbol{\theta}}(X_O)$ is identifiable.

To summarize, **A1** ensures that our model has the same graphical representation/parametric forms as the ground truth; **A2** $p_{\boldsymbol{\theta}}(X_O) = \int_{X_U} p_{\boldsymbol{\theta}}(X_O, X_U) dX_U$ should be at least identifiable for a collection of observable patterns that forms a partition of $\mathcal{I}$; and **Assumption A3** ensures that $p_{\mathcal{D}}(X_O, X_U, R)$ should be positive for certain *important* patterns (i.e., those on which $p_{\boldsymbol{\theta}}(X_O)$ is identifiable). Given these assumptions, we have the following proposition (See Appendix 8.C for proof.):

Proposition 8.1 (Sufficient conditions for identifiability under MNAR)**.** *Let $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O, X_U, R)$ be a model on the observable variables X, and missing pattern R, and $p_{\mathcal{D}}(X_O, X_U, R)$ be the*

[1]It can be arbitrary partition in the set theory sense.

ground truth distribution. Assume that they satisfies ***Data setting D1, Assumptions A1, A2*** *and* ***A3****.*

Let $\boldsymbol{\theta} = \arg\max_{(\boldsymbol{\theta},\boldsymbol{\psi})\in\boldsymbol{\Omega}} \mathbb{E}_{(\mathbf{x}_O,\mathbf{r})\sim p_{\mathcal{D}}(X,R)} \log p_{(\boldsymbol{\theta},\boldsymbol{\psi})}(X_O = \mathbf{x}_O, R = \mathbf{r})$ *be the set of ML solutions of Equation 8.2.1. Then, we have* $\boldsymbol{\theta} = \{\boldsymbol{\theta}^*\} \times \boldsymbol{\theta}_{\boldsymbol{\psi}}$*. That is, the ground truth model parameter* $\boldsymbol{\theta}^*$ *can be uniquely identified via (partial) maximum likelihood learning.*

Missing value imputation as inference Given a model $p_{(\boldsymbol{\theta})}(X_O, X_U)$, the missing data imputation problem can be then formularized by the Bayesian inference problem $p_{\boldsymbol{\theta}}(X_U|X_O) \propto p_{\boldsymbol{\theta}}(X_U, X_O)$. If the assumptions of Proposition 8.2 are satisfied, it enables us to correctly identify the ground truth reference model parameter, $\boldsymbol{\theta}^*$. Therefore, the imputed values sampled from the posterior $p_{\boldsymbol{\theta}^*}(X_U|X_O)$ will be unbiased, and can be used for down stream decision making tasks.

Remark: Note that Proposition 8.1 can be extended to the case where model identifiability is defined by equivalence classes [146, 321]. See Appendix 8.F for details.

8.3.2 Relaxing "correctness of parametric model" assumption (A1)

In this section, we further extend our previous results to the general MNAR cases including all different examples in Figure 8.1. In particular, we would like to see the if the same model setting in Section 8.3.1 can be applied to scenarios where $p_{\mathcal{D}}(X_O, X_U, R)$ and $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O, X_U, R)$ might have different parametric forms, or even different graphical representations.

To start with, we would like to point out that the mismatch between $p_{\mathcal{D}}(X_O, X_U, R)$ and the model $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O, X_U, R)$ can be, to a certain extend, modeled by the *mappings between spaces of parameters*. Let $\boldsymbol{\Omega} \subset \mathbb{R}^I$ denote the parameter domain of our model, $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O, X_U, R)$. Suppose we have a mapping $\Phi : \underline{\boldsymbol{\Omega}} \subset \mathbb{R}^I \mapsto \mathbb{R}^J$, such that $(\boldsymbol{\theta}, \boldsymbol{\psi}) \in \underline{\boldsymbol{\Omega}} \subset \boldsymbol{\Omega}$ is mapped to another parameter space $(\tau, \gamma) = \Phi(\boldsymbol{\theta}, \boldsymbol{\psi}) \in \Xi \subset \mathbb{R}^J$ via $\Phi(\cdot)$. Here, $\underline{\boldsymbol{\Omega}}$ is a subset of $\boldsymbol{\Omega}$ on which Φ is defined. Then, the *re-parameterized* $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O, X_U, R)$ on parameter space Ξ can be rewritten as:

$$\tilde{p}_{\tau,\gamma}(X_O, X_U, R) := p_{\Phi^{-1}(\tau,\gamma)}(X_O, X_U, R)$$

Assuming that the inverse mapping Φ^{-1} exists. Then trivially, if $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O, R)$ is identifiable with respect to $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$, then $\tilde{p}_{\tau,\gamma}(X_O, R)$ should be also identifiable with respect to τ and γ:

Proposition 8.2. *Let* $\boldsymbol{\Omega} \subset \mathbb{R}^I$ *be the parameter domain of the model* $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O, X_U, R)$*. Assume that the mapping* $\Phi : (\boldsymbol{\theta}, \boldsymbol{\psi}) \in \underline{\boldsymbol{\Omega}} \subset \mathbb{R}^I \mapsto (\tau, \gamma) \in \Xi \subset \mathbb{R}^J$ *is one-to-one on* $\underline{\boldsymbol{\Omega}}$ *(equivalently, the inverse mapping* $\Phi^{-1} : \Xi \mapsto \underline{\boldsymbol{\Omega}}$ *is injective, and* $\underline{\boldsymbol{\Omega}}$ *is its image set). Consider* *the induced distribution with parameter space Ξ, defined as $\tilde{p}_{\tau,\gamma}(X_O,R) := p_{\Phi^{-1}(\tau,\gamma)}(X_O,R)$. Then, $\tilde{p}$ is identifiable w.r.t. (τ,γ), if $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,R)$ is identifiable w.r.t. $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$.*

Proposition 8.2 basically shows that if two distributions $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,R)$ and $\tilde{p}_{\tau,\gamma}(X_O,R)$ are related by a mapping Φ with nice properties, than the identifiability will translate between them. This already covers many scenarios of the data-model mismatch. For example, consider the case where ground truth data generation process satisfies the following assumption:

Data setting D2 Suppose the ground truth $p_{\mathcal{D}}(X_O,X_U,R)$ satisfies: X are all generated by shared latent confounders Z (as in **D1**), and R cannot be the cause of any other variables as in [233, 342]. Typical examples are given by any of the cases in Fig 8.1(excluding (j) where R_1 is the cause of R_2). Furthermore, the ground truth data generating process is given by the parametric form $p_{\mathcal{D}}(X_O,X_U,R) = \tilde{p}_{\tau^*,\gamma^*}(X_O,X_U,R)$, where $\Xi = \Xi_\tau \times \Xi_\gamma$ denotes its parameter space.

Then, for such ground truth data generating process, we can show that we can always find a model in the form of Equation 8.3.2, such that there exists some mapping Φ, that can model their relationship:

Lemma 8.1. *Suppose the ground truth data generating process $\tilde{p}_{\tau^*,\gamma^*}(X_O,X_U,R)$ satisfies* ***setting D2****. Then, there exists a model $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,X_U,R)$, such that: 1), $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,X_U,R)$ can be written in the form of Equation 8.3.2 (i.e.,* ***Assumption A1****; and 2), there exists a mapping Φ as described in Proposition 8.2, such that $\tilde{p}_{\tau,\gamma}(X_O,R) = p_{\Phi^{-1}(\tau,\gamma)}(X_O,R)$, for all $(\tau,\gamma) \in \Xi$.*

Model identification under data-model mismatch. Since we showed the identifiability can be preserved under the parameter space mapping (Proposition 8.2), next we will prove that if the model $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,X_U,R)$ is trained on partially observed data points sampled from $\tilde{p}_{\tau,\boldsymbol{\psi}}(X_O,X_U,R)$ that satisfies **data setting D2**, then the ML solution is still unbiased. For this purpose, inspired by Lemma 8.1, we work with the following additional assumption:

Model Assumption A4 Let $\tilde{p}_{\tau^*,\gamma^*}(X_O,X_U,R)$ denote our ground truth data generating process that satisfies **data setting D2**. Then, we assume our model $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,X_U,R)$ is the one that satisfies the description given by Lemma 8.1. That is, its parametric form is given by Equation 8.3.2, and there exists a mapping Φ as described in Proposition 8.2, such that $\tilde{p}_{\tau,\gamma}(X_O,R) = p_{\Phi^{-1}(\tau,\gamma)}(X_O,R)$.

Then, we have the following proposition:

Proposition 8.3 (Sufficient conditions for identifiability under MNAR and data-model mismatch)**.** *Let $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,X_U,R)$ be a model on the observable variables X and missing pattern R,* *and $p_{\mathcal{D}}(X_O, X_U, R)$ be the ground truth distribution. Assume that they satisfies* ***Data setting D2, Assumption A2, A3,*** *and* ***A4****. Let $\boldsymbol{\theta} = \arg\max_{(\boldsymbol{\theta},\boldsymbol{\psi})\in\underline{\boldsymbol{\Omega}}} \mathbb{E}_{(\mathbf{x}_O,\mathbf{r})\sim p_{\mathcal{D}}(X,R)} \log p_{(\boldsymbol{\theta},\boldsymbol{\psi})}(X_O = \mathbf{x}_O, R = \mathbf{r})$ be the set of ML solutions of Equation 8.2.1. Then, we have $\boldsymbol{\theta} = \{\Phi_{\tau}^{-1}(\tau^*)\} \times \boldsymbol{\theta}_{\boldsymbol{\psi}}$. Namely, the ground truth model parameter τ^* of $p_{\mathcal{D}}$ can be uniquely identified (as $\Phi(\boldsymbol{\theta}^*)$) via ML learning.*

Remark: practical implications Proposition 8.3 allows us to deal with the cases where the parameterization of ground truth data generating process and model distribution are related through a set of mappings, $\{\Phi_O\}$. In general, the graphical structure of $p_{\mathcal{D}}(X_O, X_U, R)$ can be any cases in Figure 8.1 excluding (j). Then, in those cases, we are still able to use a model that corresponds to Equation 8.3.2 (Fig 8.1 (h)) to perform ML learning, provided that our model is flexible enough (**Assumption A4**). This greatly improves the applicability of our identifiability results, and we can build a practical algorithm based on Equation 8.3.2 to handle many practical MNAR cases.

8.4 GINA: A Practical Imputation Algorithm for MNAR

In the previous section, we have established the identifiability conditions for models in the form of Equation (8.3.2). However, in order to derive a practically useful algorithm, we still need to specify a parametric form of the model, that is both flexible and compatible with our assumptions. In this section, by utilizing the results in Section 8.3, we propose GINA, a deep generative imputation model for MNAR data (Figure 2). GINA fulfill identifiability assumptions above, and can handle general MNAR case as discussed in section 8.3.2.

The parametric form of GINA We use utilize the flexibility of deep generative models to model the data generating process. We assume that the reference model $p_{\boldsymbol{\theta}}(X)$ is parameterized by an identifiable VAE [146] to satisfy Assumption A2. That is, $p_{\boldsymbol{\theta}}(X|V) = \int_Z dZ p_{\varepsilon}(X - f(Z))p(Z|V)$, where V is some fully observed auxiliary inputs. The decoder $p_{\varepsilon}(X - f_{\boldsymbol{\theta}}(Z))$ is parameterized by a neural network, $f : \mathbb{R}^H \mapsto \mathbb{R}^D$. For convenience, we will drop the input V to $p_{\boldsymbol{\theta}}(X|V)$, and simply use $p_{\boldsymbol{\theta}}(X)$ to denote $p_{\boldsymbol{\theta}}(X|V)$. Finally, for the missing model $p_{\boldsymbol{\psi}}(R|X,Z)$, we use a Bernoulli likelihood model, $p_{\boldsymbol{\psi}}(R|X,Z) := \prod_d \pi_d(X,Z)^{R_d}(1-\pi_d(X,Z))^{1-R_d}$, where $\pi_d(X,Z)$ is parameterized by a neural network.

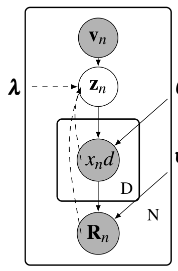


Figure 8.2 Graphical representations of our GINA.

In Appendix 8.G, we show that GINA fulfill the required assumptions of Proposition 8.1 and 8.3. Thus, we can use GINA to identify the ground truth data generating process, and perform missing value imputation under MNAR.

Learning and imputation In practice, the joint likelihood in Equation 8.2.1 is intractable. Similar to the approach proposed in [128], we introduce a variational inference network, $q_{\boldsymbol{\lambda}}(Z|X_O)$, which enable us to derive a importance weighted lower bound of $\log p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,R)$:

$$\log p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,R) \geq \mathcal{L}_K(\boldsymbol{\theta},\boldsymbol{\psi},\boldsymbol{\lambda},X_O,R) := \mathbf{E}_{z^1,\ldots,z^K,x_U^1,\ldots,x_U^K \sim p_{\boldsymbol{\theta}}(X_U|Z)q_{\boldsymbol{\lambda}}(Z|X_O)} \log \frac{1}{K}\sum_k w_k$$

where $w_k = \frac{p_{\boldsymbol{\psi}}(R|X_O,X_U=x_U^k,Z=z^k)p_{\boldsymbol{\theta}}(X_O,Z=z^k)}{q_{\boldsymbol{\lambda}}(Z=z^k|X_O)}$ is the importance weights. Note that we did not notate the missing pattern R as additional input to $q_{\boldsymbol{\lambda}}$, as this information is already contained in X_O. Then, we can optimize the parameters $\boldsymbol{\theta},\boldsymbol{\psi},\boldsymbol{\lambda}$ by solving the following optimization problem

$$\boldsymbol{\theta}^*,\boldsymbol{\psi}^*,\boldsymbol{\lambda}^* = \arg\max_{\boldsymbol{\theta},\boldsymbol{\psi},\boldsymbol{\lambda}} \mathbb{E}_{(\mathbf{x}_O,\mathbf{r})\sim p_{\mathcal{D}}(X,R)}\mathcal{L}_K(\boldsymbol{\theta},\boldsymbol{\psi},\boldsymbol{\lambda},X_O=x_O,R=r)$$

Given $\boldsymbol{\theta}^*,\boldsymbol{\psi}^*,\boldsymbol{\lambda}^*$, we can impute missing data by solving the approximate inference problem:

$$p_{\boldsymbol{\theta}}(X_U|X_O) = \int_Z p_{\boldsymbol{\theta}}(X_U|Z)p_{\boldsymbol{\theta}}(Z|X_O)dZ \approx \int_Z p_{\boldsymbol{\theta}}(X_U|Z)q_{\boldsymbol{\lambda}}(Z|X_O)dZ.$$

8.5 Related works

We mainly review recent works for handling MNAR data. In Appendix 8.A, we provide a brief review of traditional methods that deal with MCAR and MAR.

When the missing data is MNAR, a general framework is to learn a joint model on both observable variables and missing patterns [190], in which a model of missing data is usually assumed [320, 126]. This approach is also widely adopted in imputation tasks. For example, in the field of recommender systems, different probabilistic models are used within such a framework [112, 208, 353, 187, 184]. A similar approach has also been taken in the context of causal approach to imputation [355, 354, Liang et al.]. Similar to the use of the missing model, they have used an explicit model of exposure and adopted a causal view, where MNAR is treated as a confounding bias. Apart from these, inverse probability weighting methods are also used to debias the effect of MNAR [299, 353, 202] for imputation.

One issue that is often ignored by many MNAR methods is the model identifiability. Identifiability under MNAR has been discussed for certain cases ([220, 219, 221, 352, 332, 321]). For example, [352] proposed the instrumental variable approach to help the identification of MNAR data. [219] investigated the identifiability of normal and normal mixture models, and showed that identifiability for parametric models is highly non-trivial under MNAR. [220] studied conditions for nonparametric identification using shadow variable technique. Despite the resemblance to the auxiliary variable in our approach, [219, 220] mainly considers the supervised learning (multivariate regression) scenario. [233, 232, 311] also discussed a similar topic based on a graphical and causal approach in a non-parametric setting. Although the notion of recoverability has been extensively discussed, their methods do not directly lead to practical imputation algorithms in a scalable setting. On the contrary, our work takes a different approach, in which we handle MNAR with a parametric setting, by dealing with learning and inference in latent variable models. We step aside from the computational burden with the help of recent advances in deep generative models for scalable imputation.

There has been a growing interest in applying deep generative models to missing data imputation. In [200, 197, 241], scalable methods for training VAEs under MAR have been proposed. Similar methods have also been advocated in the context of importance weighted VAEs, multiple imputation [212], and heterogeneous tabular data imputation [241, 201, 199]. Generative adversarial networks (GANs) have also been applied to MCAR data [372, 175]. More recently, deep generative models under MNAR have been studied [128, 82, 87], where different approaches such as selection models [279, 107] and pattern-set mixture models [189] has been combined with partial variational inference for training VAEs. However, without additional assumptions, the model identifiability remains unclear in these approaches, and the posterior distribution of missing data conditioned on observed data might be biased.

8.6 Experiments

We study the empirical performance of the proposed algorithm of Section 8.4 with both synthetic data (Section 8.6.1) and two real-world datasets with music recommendation (Section 8.6.2) and personalized education (Section 8.6.3) . The experimental setting details can be found in Appendix 8.B.

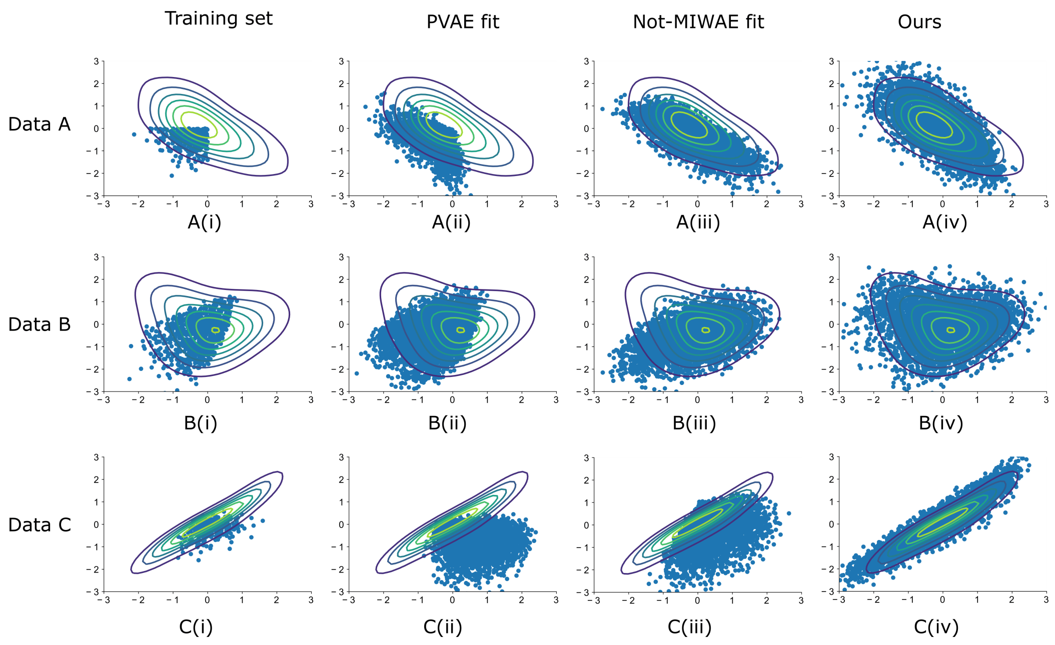


Figure 8.3 Visualization of generated X_2 and X_3 from synthetic experiment. **Row-wise (A-C)** plots for dataset A, B, and C, respectively; **Column-wise (i-iv):** training set (only displays fully observed samples), PVAE samples, Not-MIWAE samples, and GINA samples, respectively. **Contour plot**: kernel density estimate of ground truth density of complete data;

8.6.1 Synthetic MNAR dataset

We first consider 3D synthetic MNAR datasets. We generate three synthetic datasets with nonlinear data generation process (shown in Appendix 8.B.1). For all datasets, X_1, X_2, X_3 are generated via the latent variables, Z_1, Z_2, Z_3 ,where X_1 are fully observed and X_2 and X_3 are MNAR. For dataset A, we apply *self-masking*(similar to Figure 8.1(c)): X_i will be missing if $X_i > 0$. For datasets B and C, we apply *latent-dependent self-masking*: X_i will be missing, if $g(X_i, Z_1, Z_2, Z_3) > 0$, where g is a linear mapping whose coefficients are randomly chosen.

We train GINA and baseline models with partially observed data. Then, we use the trained models to generate random samples. By comparing the generated samples with the ground truth data density, we can evaluate whether $p_{\mathcal{D}}(X)$ is correctly identified. Results are visualized in Figure 8.3. In addition, we show the imputation results in Appendix 8.I. Across three datasets, PVAE performs poorly, as it does not account for the MNAR mechanism. Not-MIWAE performs better than PVAE, as it is able to generate samples that are closer to the mode. However, it is still biased more towards the observed values. On the other hand, GINA is much more aligned to ground truth, and is able to recover the ground truth from partially observed data. This experiment showed the clear advantage of our method under different MNAR situations.

8.6.2 Recommender dataset imputation with random test set

Method	Test MSE
Matrix Factorization Methods	
PMF	1.401
IPW-PMF	1.375
Deconfounded-PMF	1.329
PMF-MNAR	1.483
PMF-MAR	1.480
VAE-based models	
PVAE	1.259±0.003
PVAE w/o IW	1.261±0.004
Not-MIWAE	1.078±0.000
GINA	**1.052±0.002**
Others	
CPTv-MNAR	1.056
Logitvd-MNAR	1.141
AutoRec	1.199
Oracle-test	1.057

Table 8.1 Test MSE on Yahoo! R3

Next, we apply our models to recommendation systems on Yahoo! R3 dataset [208, 355] for user-song ratings which is designed to evaluate MNAR imputation. It contains an MNAR training set of more than 300K self-selected ratings from 15,400 users on 1,000 songs, and an MCAR test set of randomly selected ratings from 5,400 users on 10 random songs. We train all models on the MNAR training set, and evaluate on MCAR test set. This is repeated 10 times with different random seeds. Both the missing model for GINA ($p(R|X,Z)$) and Not-MIWAE ($p(R|X)$) are parameterized by linear neural nets with Bernoulli likelihood functions. The decoders for GINA, PVAE, and Not-MIWAE uses Gaussian likelihoods with the same network structure. See Appendix 8.B for implementation details and network structures.

We compare to the following baselines: 1), probabilistic matrix factorization (PMF) [228]; 2), inverse probability weighted PMF [299]; 3), Deconfounded PMF [355]; 4), PMF with MNAR/MAR data [112]; 5), CPTv and Logitv models for MNAR rating [208]; 6), Oracle [112]: predicts ratings based on their marginal distribution in the test set; and 7) AutoRec [303]: Autoencoders that ignores missing data.

Results are shown in Table 8.1. Our method (GINA) gives the best performance among all methods. Also, VAE-based methods are consistently better than PMF-based methods, and MNAR-based models consistently outperform their MAR versions. More importantly, among VAE-based models, our GINA outperforms its non-identifiable counterpart (Not-MIWAE), and MAR counterpart (PVAE), where both models can not generate unbiased imputation.

8.6.3 Missing data imputation and active question selection on Eedi education dataset

Finally, we apply our methods to the Eedi education dataset [356], one of the largest real-world education response datasets. We consider the Eedi competition task 3 dataset, which contains over 1 million responses from 4918 students to 948 multiple-choice diagnostic questions. Each diagnostic question is a multiple-choice question. We consider predicting whether a student answers a question correctly or not. Over 70% of the entries are missing. The dataset also contains student metadata which we use as the auxiliary variables. In this experiment, we randomly split the data in a 90% train/ 10% test/ 10% validation ratio, and train our models on the response outcome data.

We evaluate our model on two tasks. Firstly, we perform missing data imputation, where different methods perform imputation over the test set. As opposed to Yahoo! R3 dataset, now the test set is MNAR, thus we use the evaluation method suggested by [355], where we evaluate the average per-question MSE For each question, over all students with non-empty response. Then, the MSEs of all questions averaged. We call this metric the debiased MSE. While regular MSE might be biased toward questions with more responses, the debiased MSE treats all questions equally, and can avoid selection bias to a certain degree. We report results for 10 repeats in the first column in Table 8.2. We can see that our proposed GINA achieves significantly improved results comparing to the baselines.

Secondly, we evaluate personalized education through active question selection [200] on the test set which is task 4 from this competition dataset. The procedure is as follows: for each student in the test set, at each step, the trained generative models are used to decide which is the most informative missing question to collect next. This is done by maximizing the information reward as in [200] (see Appendix 8.H for details).Since at each step, different students might collect different questions, there isn't a simple way to debias the predictive MSE as in the imputation task. Alternatively, we evaluate each method with the help of *question meta data* (difficulty level, which is a scalar).Intuitively, when the student response to the previously collected question is correct, we expect the next diagnostic question which has higher difficulty levels, and vice versa.Thus, we can evaluate the mean level change after correct/incorrect responses, and expect them to have significant differences. We also perform t-test between the level changes after incorrect/correct responses and report the p-value. We can see in Table 8.2, GINA is the only method that reports a significant p-value (<0.05) between the level changes of next collected questions after incorrect/correct responses which

Table 8.2 Performance on Eedi education dataset (with standard errors)

Method	Debiased MSE	Level change (correct)	Level change (incorrect)	p-value
PVAE	0.194±0.001	0.131±0.138	-0.101±0.160	0.514
Not-MIWAE	0.192±0.000	0.062±0.142	-0.073±0.179	0.561
GINA	**0.188±0.001**	**0.945±0.151**	**-0.353±0.189**	$\mathbf{1.01\times10^{-7}}$

are desired. This further indicates that our proposed GINA predicts the unobserved answer with the desired behavior.

8.7 Conclusion

In this chapter, we provide a analysis of identifiability for generative models under MNAR, and studies sufficient conditions of identifiability under different scenarios. We provide sufficient conditions under which the model parameters can be uniquely identified, via joint maximum likelihood learning on X_O and R. Therefore, the learned model can be used to perform unbiased missing data imputation. We proposed a practical algorithm based on VAEs, which enables us to apply flexible generative models that is able to handle missing data in a principled way. The main limitation of our proposed pracitical algorithm is the need for auxiliary variables (meta feature) which is inherited from identifiable VAE models [146]. In practice, they may not be always available. For future work, we will investigate how to address such limitation, and how to extend to more complicated scenarios.

Appendix for Chapter 8

Appendix 8.A Traditional methods for handling missing data

Methods for handling missing data has been extensively studied in the past few decades. Those methods can be roughly classified into two categories: complete case analysis (CCA) based, and imputation based methods. CCA based methods, such as listwise deletion [5] and pairwise deletion [209] focuses on deleting data instances that contains missing entries, and keeping those that are complete. Listwise/pairwise deletion methods are known to be unbiased under MCAR, and will be biased under MAR/MNAR. On the contrary, imputation based methods tries to replace missing values by imputed/predicted values. One popular imputation technique is called single imputation, where only produce one single set of imputed values for each data instance. Standard techniques of single imputation include mean/zero imputation, regression-based imputation [5], no- parametric methods [143, 325]. As opposed to single imputation, the multiple imputation (MI) methods such as MICE [362], was first proposed by Rubin [280, 281, 119, 237] is essentially a simulation-based methods that returns multiple imputation values for subsequent statistical analysis. Unlike single imputation, the standard errors of estimated parameters produced with MI is known to be unbiased [282]. Apart from MI, there exists other methods such as full information maximum likelihood [9, 67] and inverse probability weighting [277, 120], which can be directly applied to MAR without introducing additional bias. However, these methods assumes a MAR missing data mechanism, and cannot be directly applied to MNAR without introducing bias.

Appendix 8.B Implementation details

We first introduce the general settings of GINA and other baselines. Our model (GINA) is based on the practical algorithm in Section 8.4. By default, we will set the auxiliary variable U to be some fully observed meta feature (if there's any) or the missing mask pattern (if the dataset does not have a fully observed meta feature). The most important baselines are as follows: 1), Partial VAE (PVAE) [200]: a VAE model with slightly modified ELBO objective, specifically designed for MAR data; and 2), Not-MIWAE [128], a VAE model for MNAR data trained by jointly maximizing the likelihood on both the partially observed data and the missing pattern. As opposed to our model, the latent priors $p(Z)$ for both PVAE and Not-MIWAE are parameterized by a standard normal distribution, hence no auxiliary variables are used. Also, note that the graphical model of Not-MIWAE is described by Fig 8.1 (a), and does not handle the scenarios where the ground truth data distribution follows other graphs like Fig 8.1 (g). Finally, the inference model $q(Z|X)$ for the underlying VAEs is set to be diagonal Gaussian distributions whose mean and variance are parameterized by neural nets as in standard VAEs [149] (with missing values replaced by zeros[241, 128, 212]), or a permutation invariant set function proposed in [200]. See Appendix 8.B for more implementation details for each tasks.

8.B.1 Synthetic dataset implementation details

Data generation The ground truth data generating process is given by $Z_1, Z_2, Z_3 \sim \mathcal{N}(0,1), X_1 = h_w(Z_1, Z_2, Z_3) + \varepsilon_1, X_2 = f_{\boldsymbol{\theta}_1}(X_1, Z_1, Z_2, Z_3) + \varepsilon_2, X_3 = f_{\boldsymbol{\theta}_2}(X_1, X_2, Z_1, Z_2, Z_3) + \varepsilon_3$ where h_w is a linear mapping with coefficients w, f is some non-linear mapping whose functional form is given by Appendix 8.B, $\boldsymbol{\theta}_1$ & $\boldsymbol{\theta}_2$ are two different sets of parameters for f, and ε_1, ε_2, ε_3 are observational noise variables with mean 0, variance 0.01. We randomly sample three different sets of parameters, and generate the corresponding datasets (Figure 8.3), namely dataset A, B, and C. Each dataset consists of 2000 samples. Then, we apply different missing mechanisms for each dataset. For all datasets, we assume that X_1 are fully observed and X_2 and X_3 are MNAR. , and missing mechanism will be only applied to X_2 and X_3. Finally, all observable variables are standardized.

Network structure and training We use 5 dimensional latent space with fully factorized standard normal priors. The decoder part $p_{\boldsymbol{\theta}}(X|Z)$ uses a 5-10-D structure, where $D = 3$ in our case. For inference net, we use a zero imputing [200] with structure 2D-10-10-5, that maps the concatenation of observed data (with missing data filled with zero) and mask variable R into distributional parameters of the latent space. For the factorized prior $p(Z|V)$ of the i-VAE component of GINA, we used a linear network with one auxiliary input (which is set to be fully observed dimension, X_1). The missing model $p_{\boldsymbol{\psi}}(R|X)$ for GINA and i-NotMIWAE is a single layer neural network with 10 hidden units. All neural networks use Tanh activations (except for output layer, where no activation function is used). All baselines uses importance weighted VAE objective with 5 importance samples. The observational noise for continuous variables are fixed to $\log \sigma = -2$. All methods are trained with Adam optimizer with batchsize with 100, and learning rate 0.001 for 20k epochs.

8.B.2 Yahoo!R3 experiment implementation details

Before training, all user ratings are scaled to be between 0 and 1 (such scaling will be reverted during evaluation). For all baselines, we use Gaussian likelihood with variance of 0.02. We use 20 dimensional latent space, and the decoder $p_{\boldsymbol{\theta}}(X|Z)$ uses a 20-10-D structure. We use Tanh activation function for the decoder (except for output layer, where no activation function is used). For inference net, we uses the point net structure proposed in [200], we use 20 dimensional feature mapping h parameterized by a single layer neural network and 20 dimensional ID vectors for each variable. The symmetric operator is set to be the summation operator. The missing model $p_{\boldsymbol{\psi}}(R=1|X)$ for GINA and i-NotMIWAE is parameterized by linear neural network. All methods are trained with 400 epochs with batchsize 100.

8.B.3 Eedi dataset experiment implementation details

Since Eedi dataset is a binary matrix with 1/0 indicating that the student response is correct/incorrect, we use Bernoulli likelihood for decoder $p_{\boldsymbol{\theta}}(X|Z)$. For We use 50 dimensional latent space, and the decoder $p_{\boldsymbol{\theta}}(X|Z)$ uses a 50-20-50-D structure. Such structure is chosen using the validation set using grid search. We use ReLU activation function for the decoder (except for output layer, where no activation function is used). For inference net, we uses the point net structure that were used in Yahoo!R3 dataset. Here, the difference is that we we use 50 dimensional feature mapping h parameterized by a single layer neural network and 10 dimensional ID vectors for each variable. All methods are trained with 1k epochs with batchsize 100. A trick that we used for both not-MIWAE and GINA to improve the imputation performance, is to turn down the weight of the likelihood term for $p_{\boldsymbol{\psi}}(R|X)$, by multiplying a factor of $\beta = 0.5$. This is due to that majority of the student response matrix is missing, the $p_{\boldsymbol{\psi}}(R|X)$ will most likely dominate the training, hence the learning algorithm will prefer more about learning the models that explains the missing mechanism better, over the models that explains the observable variables X better.

Appendix 8.C Proof for Proposition 8.1

Proof : First, we show that $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_{C_l'}, R)$ is *partially identifiable* (i.e., identifiable on subset of parameters) on $\{\boldsymbol{\theta}_d\}_{d\in C_l'}$ for $\forall C_l' \in \bar{\mathcal{A}}_{\mathcal{I}}$. We prove this by contradiction. Suppose there exists two different set of parameters $(\boldsymbol{\theta}^1, \boldsymbol{\psi}^1)$ $(\boldsymbol{\theta}^2, \boldsymbol{\psi}^2)$, such that there exits at least one index $c \in C_l'$ for some l, such that $\boldsymbol{\theta}_c^1 \neq \boldsymbol{\theta}_c^2$, and $p_{(\boldsymbol{\theta}^1,\boldsymbol{\psi}^1)}(X_{C_l'},R) = p_{(\boldsymbol{\theta}^2,\boldsymbol{\psi}^2)}(X_{C_l'},R)$. That is, $p(X_{C_l'},R)$ is not identifiable on $\{\boldsymbol{\theta}_d\}_{d\in C_l'}$.

According to **Assumption A3**, there exists $C_s \in \mathcal{A}_{\mathcal{I}}$, such that $c \in C_s \subset C_l'$. Then, consider the marginal

$$p_{\boldsymbol{\theta}}(X_{C_s}) = \int_{Z,R,X_{\setminus C_s}} dZ \prod_{d\in C_s} p_{\boldsymbol{\theta}_d}(X_d|Z)p_{\boldsymbol{\psi}}(R|X,Z)p(Z) = p_{\boldsymbol{\theta}_{d\in C_s}}(X_{C_s})$$

. Since $p_{(\boldsymbol{\theta}^1,\boldsymbol{\psi}^1)}(X_{C_l'},R) = p_{(\boldsymbol{\theta}^2,\boldsymbol{\psi}^2)}(X_{C_l'},R)$, we have $p_{(\boldsymbol{\theta}^1_{C_s})}(X_{C_s}) = p_{(\boldsymbol{\theta}^2_{C_s})}(X_{C_s})$ (the joint uniquely determines marginals). However, this contradicts with our **Assumption A2** that $p_{\boldsymbol{\theta}_{C_s}}(X_{C_s})$ is identifiable: this identifiability assumption implies that we should have $p_{(\boldsymbol{\theta}^1_{C_s})}(X_{C_s}) \neq p_{(\boldsymbol{\theta}^2_{C_s})}(X_{C_s})$. Therefore, by contradiction, we have $p(X_{C_l'},R)$ is partially identifiable on $\{\boldsymbol{\theta}_d\}_{d\in C_l'}$ for $\forall C_l' \subset \bar{\mathcal{A}}_{\mathcal{I}}$.

Then, we proceed to prove that the ground truth parameter $\boldsymbol{\theta}^*$ can be uniquely identified via ML learning. Based on our **Assumption A1**, upon optimal ML solution,

$$\boldsymbol{\theta}_{ML} = \arg \max_{(\boldsymbol{\theta},\boldsymbol{\psi})\in\Omega} \mathbb{E}_{(\mathbf{x}_O,\mathbf{r})\sim p_{\mathcal{D}}(X,R)} \log p_{(\boldsymbol{\theta},\boldsymbol{\psi})}(X_O = \mathbf{x}_O, R = \mathbf{r})$$

, we have the following identity:

$$p_{(\boldsymbol{\theta}_{ML},\boldsymbol{\psi}_{ML})}(X_O,R) = p_{(\boldsymbol{\theta}^*,\boldsymbol{\psi}^*)}(X_O,R)$$

holds for all $(\boldsymbol{\theta}_{ML},\boldsymbol{\psi}_{ML}) \in \boldsymbol{\theta}_{ML}$, and all $\forall O \subset \mathcal{I}$ that satisfies $p(X_O,R) > 0$.

Note also that:

$$p_{(\boldsymbol{\theta}_{ML},\boldsymbol{\psi}_{ML})}(X_O,R) = \int_{Z,X_{\mathcal{I}\setminus O}} dZ \prod_d p_{\boldsymbol{\theta}_d^{ML}}(X_d|Z)p_{\boldsymbol{\psi}^{ML}}(R|X)p(Z)$$

, which depends on both $\boldsymbol{\theta}_O$ and $\boldsymbol{\psi}$. Since we have already shown that $p_{(\boldsymbol{\theta},\boldsymbol{\psi})}(X_{C_l'},R)$ are partially identifiable on $\{\boldsymbol{\theta}_d\}_{d\in C_l'}$ for $\forall C_l' \subset \bar{\mathcal{A}}_{\mathcal{I}}$, according to **Assumption A3**, upon optimal solution , we have that

$$\{\boldsymbol{\theta}_d = \boldsymbol{\theta}_d^*\}_{d\in C_l'}$$

holds for all $\forall C_l' \subset \bar{\mathcal{A}}_{\mathcal{I}}$. Since we have assumed that $\bigcup_{C_l'\in\bar{\mathcal{A}}_{\mathcal{I}}} X_{C_l'} = \mathcal{I}$ in **Assumption 3** (i.e.,$\bar{\mathcal{A}}_{\mathcal{I}}$ is a cover of $\mathcal{I}$), this guarantees that

$$\boldsymbol{\theta}_d^{ML} = \boldsymbol{\theta}_d^*$$

for all d. In other words, we are able to uniquely identify $\boldsymbol{\theta}^*$ from observed data, therefore

$$\boldsymbol{\theta} = \{\boldsymbol{\theta}^*\} \times \boldsymbol{\theta}_{\boldsymbol{\psi}}$$

. □

Appendix 8.D Proof of Proposition 8.2

Proof Let (τ_1, γ_1) and (τ_2, γ_2) be two different parameters in Ξ. Then, we have

$$\begin{aligned}
&\tilde{p}_{\tau_1,\gamma_1}(X_O,R)\\
=&p_{\Phi^{-1}(\tau_1,\gamma_1)}(X_O,R)\\
\neq&p_{\Phi^{-1}(\tau_2,\gamma_2)}(X_O,R)\\
=&\tilde{p}_{\tau_2,\gamma_2}(X_O,R)
\end{aligned}$$

where the third line is due to the fact that Φ^{-1} is injective and $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,R)$ is identifiable with respect to $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$. □

Appendix 8.E Relaxing Assumption A1

8.E.1 Proof of Lemma 8.1

Lemma 1. *Suppose the ground truth data generating process $\tilde{p}_{\tau^*,\gamma^*}(X_O,X_U,R)$ satisfies **setting D2**. Then, there exists a model $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,X_U,R)$, such that: 1), $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,X_U,R)$ can be written in the form of Equation 8.3.2 (i.e., **Assumption A1**; and 2), there exists a mapping Φ as described in Proposition 8.2, such that $\tilde{p}_{\tau,\gamma}(X_O,R) = p_{\Phi^{-1}(\tau,\gamma)}(X_O,R)$, for all $(\tau,\gamma) \in \Xi$. Additionally, such Φ is decoupled, i.e., $\Phi(\boldsymbol{\theta},\boldsymbol{\psi}) = (\Phi_{\boldsymbol{\theta}}(\boldsymbol{\theta}),\Phi_{\boldsymbol{\psi}}(\boldsymbol{\psi}))$.*

Proof: [2]

Case 1 (connections among X): Suppose the ground truth data generating process $p_{\mathcal{D}}(X,R) = \tilde{p}_{\tau^*,\gamma^*}(X_O,X_U,R)$ is given by Figure 8.1 (i). That is, $p_{\mathcal{D}}(X,R) =$

[2]We mainly consider the case where all variables are continuous. Discrete variables will complicate the discussion, but will not change the conclusion.

$\tilde{p}_\gamma(X|R)\int_Z \prod_i \tilde{p}_{\tau_i}(X_i|Z, pa(X_i)\bigcap X)p(Z)dZ$. Without loss of generality, assume that probabilistic distributions $\tilde{p}_{\tau_i}(X_i|Z, pa(X_i)\bigcap X)$ takes the form as $\tilde{p}_{\tau_i}(X_i|Z, pa(X_i)\bigcap X) = \int_{\varepsilon_i}\delta(X_i - f_i^{\psi_i}(\varepsilon_i, pa(X_i)\bigcap X, Z))p(\varepsilon_i)d\varepsilon_i$. Therefore, we have

$$\begin{aligned}
&\tilde{p}_\tau(X)\\
&=\int_Z \prod_i \tilde{p}_{\tau_i}(X_i|Z, pa(X_i)\bigcap X)p(Z)dZ\\
&=\int_z \left[\prod_{\{i|N(X_i)\bigcap X\neq\emptyset\}} \int_{\varepsilon_i} d\varepsilon_i \delta(X_i - f_i^{\psi_i}(\varepsilon_i, pa(X_i)\bigcap X, Z))p(\varepsilon_i)\right]\\
&\left[\prod_{\{j|N(X_j)\bigcap X=\emptyset\}} p(X_j|Z)\right] p(Z)dZ\\
&=\int_{z,\{i|N(X_i)\bigcap X\neq\emptyset\}} \left[\prod_{\{i|N(X_i)\bigcap X\neq\emptyset\}} \delta(X_i - f_i^{\psi_i}(\varepsilon_i, pa(X_i)\bigcap X, Z))p(\varepsilon_i)\right]\\
&\left[\prod_{\{j|N(X_j)\bigcap X=\emptyset\}} p(X_j|Z)\right] p(Z)dZ
\end{aligned}$$

Apparently, there exists a set of function $\{g_i(\cdot)|N(X_i)\bigcap X \neq \emptyset\}$, such that:

$$\begin{aligned}
&\int_{z,\{i|N(X_i)\bigcap X\neq\emptyset\}} \left[\prod_{\{i|N(X_i)\bigcap X\neq\emptyset\}} \delta(X_i - f_i^{\psi_i}(\varepsilon_i, pa(X_i)\bigcap X, Z))p(\varepsilon_i)\right]\\
&\left[\prod_{\{j|N(X_j)\bigcap X=\emptyset\}} p(X_j|Z)\right] p(Z)dZ\\
&=\int_{z,\{i|N(X_i)\bigcap X\neq\emptyset\}} \left[\prod_{\{i|N(X_i)\bigcap X\neq\emptyset\}} \delta(X_i - g_i(\varepsilon_i, anc_\varepsilon(i), Z))p(\varepsilon_i)\right]\\
&\left[\prod_{\{j|N(X_j)\bigcap X=\emptyset\}} p(X_j|Z)\right] p(Z)dZ
\end{aligned}$$

Where $anc_\varepsilon(i)$ is the shorthand for

$$\{\varepsilon_k | X_k \in ancX_i \bigcap Z, 1 \leq k \leq D\}$$

Note that, the graphical model of the new parameterization,

$$p(X) = \int_{Z,\{i|N(X_i)\bigcap X\neq\emptyset\}} \left[\prod_{\{i|N(X_i)\bigcap X\neq\emptyset\}} \delta(X_i - g_i(\varepsilon_i, anc_\varepsilon(i), Z))p(\varepsilon_i)\right] \left[\prod_{\{j|N(X_j)\bigcap X=\emptyset\}} p(X_j|Z)\right] p(Z)dZ$$

has a new aggregated latent space, $\{Z, \{\varepsilon_i | 1 \leq i \leq D\}\}$. That is, for each X_i that has non empty neighbour in X, a new latent variable will be created. With this new latent space, the connections among X can be decoupled, and the new graphical structure of $p(X,R)$ corresponds to Figure 8.1 (h).

The mapping Φ that connects $\tilde{p}_{\tau_i}(X,R)$ and $p(X,R)$ can now be defined as identity mapping, since no new parameters are introduced/removed when reparameterizing $\tilde{p}_{\tau_i}(X,R)$ into $p(X,R)$. Hence, the two requirements of Lemma 8.1 are fulfilled.

Case 2(subgraph): Next, consider the case that the ground truth data generating process $p_\mathcal{D}(X,R) = \tilde{p}_{\tau^*,\gamma^*}(X_O,X_U,R)$ is given by one of the Figure 8.1 (a)-(g). That is, it is a subgraph of Figure 8.1 (h). Without loss of generality, assume that $\tilde{p}_{\gamma_i}(R_i = 1|pa(R_i)) = \mathrm{logit}^{-1}(\mathrm{f}_{\gamma_i}(\mathrm{pa(R_i)}))$, and $pa(R_i) \subsetneq \{X,Z\}$; in other words, certain connections from $\{X,Z\}$ to R_i is missing. Consider the model distribution parameterized by $p(R_i = 1|X,Z) = \mathrm{logit}^{-1}(\mathrm{f}_{\gamma_i}(\mathrm{pa(R_i)}) + \mathrm{g}_{\boldsymbol{\theta}_i}(\{\mathrm{X,Z}\} \setminus \mathrm{pa(R_i)}))$, satisfying $g_{\boldsymbol{\theta}_i=0}(\cdot) \equiv 0$. Therefore, the mapping Φ^{-1} is given as $\Phi^{-1}(\gamma_i) := (\gamma_i, \boldsymbol{\theta}_i = 0)$. Apparently, Φ^{-1} is injective, hence satisfying the requirement of Proposition 8.2.

□

8.E.2 Proof for Proposition 8.3

Proposition 3 (Sufficient conditions for identifiability under MNAR and data-model mismatch). *Let $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_O,X_U,R)$ be a model on the observable variables X and missing pattern R, and $p_\mathcal{D}(X_O,X_U,R)$ be the ground truth distribution. Assume that they satisfies* ***Data setting D2, Assumption A2, A3,*** *and* ***A4****. Let $\boldsymbol{\theta} = \arg\max_{(\boldsymbol{\theta},\boldsymbol{\psi})\in\Omega} \mathbb{E}_{(\mathbf{x}_O,\mathbf{r})\sim p_\mathcal{D}(X,R)} \log p_{(\boldsymbol{\theta},\boldsymbol{\psi})}(X_O = \mathbf{x}_O, R = \mathbf{r})$ be the set of ML solutions of Equation 8.2.1. Then, we have $\boldsymbol{\theta} = \{\Phi_\tau^{-1}(\tau^*)\} \times \boldsymbol{\theta}_{\boldsymbol{\psi}}$. Namely, the ground truth model parameter τ^* of $p_\mathcal{D}$ can be uniquely identified (as $\Phi(\boldsymbol{\theta}^*)$) via ML learning.*

Proof : First, it s not hard to show that $p_{\boldsymbol{\theta},\boldsymbol{\psi}}(X_{C'_l},R)$ is partially identifiable on $\{\boldsymbol{\theta}_d\}_{d\in C'_l}$ for $\forall C'_l \in \bar{\mathcal{A}}_{\mathcal{I}}$. This has been shown in the proof of Proposition 8.1, and we will not repeat this proof again.

Next, given data setting **D2** and **Assumption A4**, define

$$\boldsymbol{\theta}_{ML} = \arg\max_{(\boldsymbol{\theta},\boldsymbol{\psi})\in\Omega} \mathbb{E}_{(\mathbf{x}_O,\mathbf{r})\sim p_{\mathcal{D}}(X,R)} \log p_{(\boldsymbol{\theta},\boldsymbol{\psi})}(X_O=\mathbf{x}_O, R=\mathbf{r})$$

, then we have:

$$p_{(\boldsymbol{\theta}_{ML},\boldsymbol{\psi}_{ML})}(X_O,R) = p_{\Phi^{-1}(\tau^*,\gamma^*)}(X_O,R)$$

holds for all $(\boldsymbol{\theta}_{ML},\boldsymbol{\psi}_{ML}) \in \boldsymbol{\theta}_{ML}$, and all $\forall O \subset \mathcal{I}$ that satisfies $p(X_O,X_U,R_O=1,R_U=0) > 0$.

Since $p_{(\boldsymbol{\theta},\boldsymbol{\psi})}(X_{C'_l},R)$ are partially identifiable on $\{\boldsymbol{\theta}_d\}_{d\in C'_l}$ for $\forall C'_l \subset \bar{\mathcal{A}}_{\mathcal{I}}$ and according to **Assumption A3**, $p_{\mathcal{D}}(X_O,X_U,R_{C'_l}=1,R_{\mathcal{I}\setminus C'_l}=0) > 0$. Therefore,

$$\{\boldsymbol{\theta}_d = \Phi_{\boldsymbol{\theta}}^{-1}(\tau^*,\gamma^*)_d\}_{d\in C'_l}$$

must be true for all $\forall C'_l \subset \bar{\mathcal{A}}_{\mathcal{I}}$, where $\Phi_{\boldsymbol{\theta}}^{-1}(\tau^*,\gamma^*)$ denotes the components of $\Phi^{-1}(\tau^*,\gamma^*)$ that corresponds to the entries of $\boldsymbol{\theta}$. Since we have assumed that $\bigcup_{C'_l\in\bar{\mathcal{A}}_{\mathcal{I}}} X_{C'_l} = \mathcal{I}$ in **Assumption 3** (i.e.,$\bar{\mathcal{A}}_{\mathcal{I}}$ is a cover of $\mathcal{I}$), this guarantees that

$$\boldsymbol{\theta}_d^{ML} = \Phi_{\boldsymbol{\theta}}^{-1}(\tau^*,\gamma^*)_d$$

for all d. In other words, we are able to uniquely identify $\boldsymbol{\theta}^*$ from observed data, therefore

$$\boldsymbol{\theta} = \{\Phi_{\boldsymbol{\theta}}^{-1}(\tau^*,\gamma^*)\} \times \boldsymbol{\theta}_{\boldsymbol{\psi}}$$

.

Finally, according to **Assumption 4** and the proof of Lemma 8.1, Φ is decoupled as $(\Phi_{\boldsymbol{\theta}}(\boldsymbol{\theta}),\Phi_{\boldsymbol{\psi}}(\boldsymbol{\psi}))$. Therefore, we can write $\boldsymbol{\theta} = \{\Phi^{-1}(\tau^*)\} \times \boldsymbol{\theta}_{\boldsymbol{\psi}}$. That is, the ground truth model parameter τ^* of $p_{\mathcal{D}}$ can be uniquely identified (as $\Phi(\boldsymbol{\theta}^*)$).

□

Appendix 8.F Identifiability based on equivalence classes

In this section, we introduce the notion of identifiability based on equivalence classes. Let $\sim$ be a equivalence relation on a parameter space Ω. That is, it satisfies reflexivity ($\boldsymbol{\theta}_1 \sim \boldsymbol{\theta}_1$), symmetry ($\boldsymbol{\theta}_1 \sim \boldsymbol{\theta}_2$ if and only if $\boldsymbol{\theta}_2 \sim \boldsymbol{\theta}_1$), and transitivity (if $\boldsymbol{\theta}_1 \sim \boldsymbol{\theta}_2$ and $\boldsymbol{\theta}_2 \sim \boldsymbol{\theta}_3$, then $\boldsymbol{\theta}_1 \sim \boldsymbol{\theta}_3$). Then, a equivalence class of $\boldsymbol{\theta}_1 \in \Omega$ is defined as $\{\boldsymbol{\theta} | \boldsymbol{\theta} \in \Omega, \boldsymbol{\theta} \sim \boldsymbol{\theta}\}$. We denote this by $[\boldsymbol{\theta}_1]$. Then, we are able to give the definition of model identifiability based on equivalence classes:

Definition 8.F.1 (Model identifiability)**.** *Assume $p_{\boldsymbol{\theta}}(X)$ is a distribution of some random variable X, $\boldsymbol{\theta}$ is its parameter that takes values in some parameter space $\Omega_{\boldsymbol{\theta}}$, and sim some equivalence relation on Ω Then, if $p_{\boldsymbol{\theta}}(X)$ satisfies $p_{\boldsymbol{\theta}_1}(X) = p_{\boldsymbol{\theta}_2}(X) \iff \boldsymbol{\theta}_1 \sim \boldsymbol{\theta}_2 \iff [\boldsymbol{\theta}_1] = [\boldsymbol{\theta}_2], \forall \boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \Omega_{\boldsymbol{\theta}}$, we say that $p_{\boldsymbol{\theta}}$ is $\sim$* identifiable *w.r.t. $\boldsymbol{\theta}$ on $\Omega_{\boldsymbol{\theta}}$.*

Apparently, definition 8.2.1 is a special case of definition 8.F.1, where $\sim$ is given by the equality operator, $=$. When the discussion is based on the identifiability under equivalence relation, then it is obvious that all the arguments of Proposition 8.1, 8.2, and 8.3 still holds. Also, the statement of the results needs to adjusted accordingly. For example, in Proposition 8.1, instead of “the ground truth model parameter $\boldsymbol{\theta}^*$ can be uniquely identified", we now have “the ground truth model parameter $\boldsymbol{\theta}^*$ can be uniquely identified *up to a equivalence relation, $\sim$*".

Appendix 8.G Subset identifiability (A2) for identifiable VAEs

The GINA model needs satisfy the requirement on model of Proposition 8.1 or 8.3, if we wish to use it to fit to the partially observed data and then perform (unbiased) missing data imputation. In order to show that the identifiability result of Proposition 8.1/8.3 can be applied to GINA, the key assumption that we need to verify is the local identifiability (**Assumption A2**).

To begin with, in [146], the following theorem on VAE identifiability has been proved:

Theorem 8.1. *Assume we sample data from the model given by $p(X,Z|V) = p_\varepsilon(X - f(Z))p_{T,\zeta}(Z|V)$, where f is a multivariate function $f : \mathbb{R}^H \mapsto \mathbb{R}^D$. $p_{T,\zeta}(Z|V)$ is parameterized by exponential family of the form $p_{T,\zeta}(Z|V) \propto \prod_{i=1^M} Q(Z_i) \exp[\sum_{j=1^K} T_{i,j}(Z_i)\zeta_{i,j}(V)]$, where $Q(Z_i)$ is some base measure, M is the dimensionality of the latent variable Z, $\mathbf{T}_i(V) = (T_{i,1}, ..., T_{i,K})$ are the sufficient statistics, and $\boldsymbol{\zeta}_i(V) = (\zeta_{i,1}, ..., \zeta_{i,K})$ are the corresponding parameters, depending on V. Assume the following holds:*

1. *The set $\{X \in \mathcal{X} | \phi_\varepsilon(x) = 0\}$ has zero measure, where ϕ is the characteristic function of p_ε;*

2. *The multivariate function f is injective;*

3. *$T_{i,j}$ are differentiable a.e., and $(T_{i,j})_{1 \leq j \leq k}$ are linearly independent on any subset of $\mathcal{X}$ of measure greater than zero;*

4. *There exists $nk+1$ distinct points $V^0, ..., V^{nk}$, such that the matrix $L = (\boldsymbol{\zeta}(V^1 - U^0), ..., \boldsymbol{\zeta}(V^{nk} - V^0))$ of size nk by nk is invertible.*

Then, the parameters (f, T, ζ) are $\sim_A$-identifiable, where $\sim_A$ is the equivalence class defined as (see also Appendix 8.F):

$$(f, T, \zeta) \sim (\tilde{f}, \tilde{T}, \tilde{\zeta}) \iff \exists \mathbf{A}, \mathbf{c} | \mathbf{T}(f^{-1}(X)) = \mathbf{A}\mathbf{T}(\tilde{f}^{-1}(X)) + \mathbf{c}$$

. Here, $\mathbf{A}$ is a nk by nk matrix, and $\mathbf{c}$ is a vector.

Note that under additional mild assumptions, the $\mathbf{A}$ in the $\sim_A$ equivalence relation can be further reduced to a permutation matrix. That is, the model parameters can be identified, such that the latent variables differs up to a permutation. This is not inconsequential in many applications. We refer to [146] for more discussions on permutation equivalence.

So far, Theorem 8.1 only discussed the identifiability of $p(X)$ on the *full* variables, $X = X_O \bigcup X_U$. However, in **Assumption A2**, we need the reference model to be (partially) identifiable on a partition $C_s \in \mathcal{A}_{\mathcal{I}}$, $p_{\boldsymbol{\theta}}(X_{o_s})$. Naturally, we need additional assumptions on the the injective function f, as stated below:

Assumption A5 There exists an integer D_O, such that $f_O : \mathbb{R}^H \mapsto \mathbb{R}^{|O|}$ is injective for all O that $|O| \geq D_0$. Here, f_O is the entries from the output of f, that corresponds to the index set O.

Remark Note that, under assumption **A5**, the Assumption **A3** in Section 8.3 becomes more intuitive: it means that in order to uniquely recover the ground truth parameters, our training data must contain training examples that have more than D_0 observed features. This is different from some previous works ([233] for example), where complete case data must be available.

Finally, given these new assumptions, it is easy to show that:

Corollary 8.1 (Local identifiability)**.** *Assume that $p(X, Z|V) = p_\varepsilon(X - f(Z))p_{T,\zeta}(Z|V)$ is the model parameterized according to Theorem 8.1. Assume that the assumptions in Theorem 8.1 holds for $p(X|V)$. Additionally, assume that f satisfies assumption* **A5***.*

Then, consider the subset of variables, X_O. Then, $p(X_O|V)$ is $\sim_A$-identifiable on (f_O, T, ζ) for all O that satisfies $|O| \geq D_0$, where f_O is the entries from the output of f, that corresponds to the index set O.

Proof : it is trivial to see that the assumptions 1, 3, and 4 in Theorem 8.1 automatically holds regarding $p(X_O|V)$. f_O is injective according to Assumption **A5**. Hence, $p(X_O|V)$ satisfies all the assumptions in Theorem 8.1, and $p(X_O|V)$ is $\sim_A$-identifiable on (f_O, T, ζ) for all O that satisfies $|O| \geq D_0$. □

Remark In practice, **Assumption A5** is often satisfied. For example, consider the f that is parameterized by the following MLP composite function:

$$f(Z) = h(W \circ g(Z)) \tag{8.G.1}$$

, where g is a D_0 dimensional, injective multivariate function $g : \mathbb{R}^H \mapsto \mathbb{R}^{D_0}$, h is some activation function $h : \mathbb{R} \mapsto \mathbb{R}$, and W is a injective linear mapping $W : \mathbb{R}^{D_0} \mapsto \mathbb{R}^D$ represented by the matrix $W_{D_0 \times D}$, whose submatrices that consists of $|O| \geq D_0$ arbitrary selected columns are also injective. Note that this assumption for W is not hard to fulfill: a randomly generated matrix (e.g., with element-wise i.i.d. Gaussian prior) satisfies this condition with probability 1. To verify f_O is injective for all $|O| \geq D_0$, notice that $f_O(Z) = h(W_O \circ g(Z))$, where W_O is the output dimensions of W that corresponds to the index set O. Since W is injective and $|O| \geq D_0$, we have that W_O is also injective, hence f_O is also injective.

Appendix 8.H Active question selection

Suppose X_O be the set of observed variables, that represents the correctness of student's response to questions that are presented to them. Then, in the problem of active question selection, we start with $O = \emptyset$, and we would like to decide which variable X_i from X_U to observe/query next, so that it will most likely provide the most valuable information for some target variable of interest, X_ϕ; Meanwhile, we should while make as few queries as possible. Once we have decided which X_i to observed next, we will make query and add i to O. This process is done by maximizing the information reward proposed by [200]:

$$i^* = \arg\max_{i \in U} R(i \mid X_O) := \mathbb{E}_{X_i \sim p(X_i|X_O)} \mathbb{KL}\left[p(X_\phi|X_i, X_O) \,\|\, p(X_\phi|X_O)\right].$$

In the Eedi dataset, as we do not have a specific target variable of interest, it is defined as $X_\phi = X_U$. In this case, X_ϕ could be ver high-dimensional, and direct estimation of $\mathbb{KL}\left[p(X_\phi|X_i, X_O) \,\|\, p(X_\phi|X_O)\right]$. could be inefficient. In [200], a fast approximation has been proposed:

$$
\begin{aligned}
R(i \mid X_O) =& \mathbb{E}_{X_i \sim p(X_i|X_O)} D_{KL}\left[p(Z|X_i, X_O)||p(Z|X_O)\right] - \\
& \mathbb{E}_{X_\phi, X_i \sim p(X_\phi, X_i|X_O)} D_{KL}\left[p(Z|X_\phi, X_i, X_O)||p(Z|X_\phi, X_O)\right]. \\
\approx& \mathbb{E}_{X_i \sim \hat{p}(X_i|X_O)} D_{KL}\left[q(Z|X_i, X_O)||q(Z|X_O)\right] - \\
& \mathbb{E}_{X_\phi, X_i \sim \hat{p}(X_\phi, X_i|X_O)} D_{KL}\left[q(Z|X_\phi, X_i, X_O)||q(Z|X_\phi, X_O)\right].
\end{aligned}
$$

In this approximation, all calculation happens in the latent space of the model, hence we can make use of the learned inference net to efficeintly estimate $R(i \mid X_O)$.

Appendix 8.I Additional results

8.I.1 Imputation results for synthetic datasets

In addition to the data generation samples visualized in Figure 8.3, we present the imputation results for synthetic datasets in Figure 8.4. The procedure of generating the imputed samples are as follows. First, each model are trained on the randomly generated, partially observed synthetic dataset described in Section 8.6.1. Once the models are trained, they are used to impute the missing data in the training set. For each training data, we draw exactly one sample from the (approximate) conditional distribution $p_theta(X_U|X_O)$. Thus, we have "complete" version of the training set, one for each different model. Finally, we draw the scatter plot for each imputed training set, per dataset and per model. If the model is doing a good job recovering the ground truth distribution $p_{\mathcal{D}}(X)$ from training set, then its scatter plot should be close to the KDE estimate of the ground truth density of complete data. According to Figure 8.4, the imputed distribution is similar to the generated distribution in Figure 8.3.

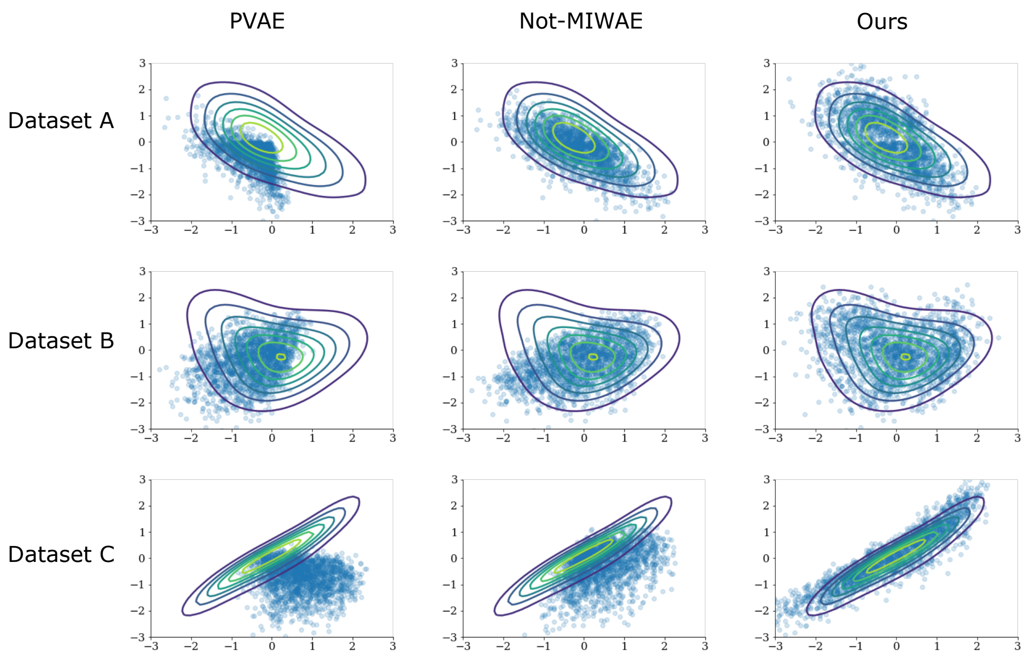


Figure 8.4 Visualization of imputed X_2 and X_3 from synthetic experiment. **Row-wise (A-C)** plots for dataset A, B, and C, respectively; **Column-wise:** PVAE imputed samples, Not-MIWAE imputed samples, and GINA imputed samples, respectively. **Contour plot**: kernel density estimate of ground truth density of complete data;

Chapter 9

Conclusion and Future Work

9.1 Conclusion: the hidden story line

The contribution of the thesis is centered around two closed connected themes: function space inference for quantifying model uncertainty in supervised learning (Theme A), and generative models for missing data uncertainty and decision making (Theme B). These are summarized in Table 9.1 below.

Table 9.1 Two themes in the thesis

	Theme A	*Theme B*
Type of ML problems	supervised learning	unsupervised learning
Type of uncertainties	model/parameter uncertainty (epistemic)	missing data uncertainty (aleatoric)
Contribution	function space inference	deep generative models for missing data uncertainty
Type of inference algorithms	function space VI	amortized VI
Consequences of model non-identifiability	symmetries, multi-modality, and posterior inconsistency	imputation biases
Main applications	Bayesian regression	multiple imputation, decision making

Recall that in Chapter 1, both themes are motivated from the perspective of Bayesian approaches to uncertainties. Here, we provide another hidden driving force that connects all the chapters in this thesis, from the perspective of (Bayesian) deep learning. We all know that machine learning, especially deep learning, has been the driving force behind modern AI research, due to its unmatched flexibility and scalability. Despite its empirical success, it has been argued that deep learning methods often fail to produce reliable uncertainty estimates for their predictions, which might jeopardize their performance in critical real-life decision-making tasks. Following the recent development of (approximate) Bayesian inference techniques, there has been a resurgence of interest in combining Bayesian techniques with deep learning methods. This results in Bayesian deep learning (BDL) tools that can tell their users when the algorithms are "making a random guess".

Both research themes (Table 9.1) presented in this thesis were largely inspired by many works in Bayesian deep learning, especially by the work of two of my lab alumnus, Yarin Gal [74] and Yingzhen Li [176]. Both of their works have been widely recognized to be quite essential to the field of modern Bayesian deep learning, as well as approximate inference. In some sense, the works presented in this thesis can also be treated as the "reverse" of typical Bayesian deep learning paradigms. In Bayesian deep learning, scalable approximate inference algorithms are usually developed and applied for specific deep learning models, such as deep neural networks for regression and classification. On the contrary, we utilized the ideas from (Bayesian) deep learning to help propose new directions and develop new approaches of Bayesian approximate inference. Examples of this paradigm can be found in many places of the thesis, for instance:

- In Part A, the research question of performing inference in the function space is largely motivated by analyzing the pathologies of model non-identifiability in neural networks. The fact that the deep learning literature cares more about prediction functions than the specific neural network weights motivates us to perform inference in the space of minimal sufficient parameters, i.e., the function space.

- In the work of variational implicit processes (Chapter 4), the concept of implicit distributions developed in deep learning (GANs) was applied to create new stochastic process priors. The flexibility of neural networks helps us extend the existing Bayesian non-parametric priors (GPs) to some class of more general and flexible priors, namely the implicit processes. Furthermore, we developed a wake-sleep approximate inference procedure for implicit processes, which is a method that is generalized from the Helmholtz machines in deep unsupervised learning literature.

- In the work of functional variational inference (Chapter 5), we further generalize the idea proposed in Chapter 4, to a more general method that performs non-Gaussian approximations under the framework of variational inference. One of the defining features of this work is that it allows efficient and scalable estimation of the (grid-) functional KL divergence, which is due to the VAE parameterization of the non-Gaussian coefficients, $\{a_s\}_{1\leq s\leq S}$. The introduction of such deep learning structures not only allows us to specify rich posterior approximations in function space, but also to avoid using computational costly gradient estimators such as SSGEs.

- Part B is solely based on the idea of using deep generative models to quantify missing data uncertainty, perform multiple imputations, and acquire new information. The empirical success in the work of EDDI (Chapter 7) also benefits a lot from many innovations from the deep learning literature. First, the neural network decoder used by the partial VAE allows us to represent expressive distributions for accurate density estimation. Second, the proposed partial amortization method is based on the point-net structures in point clouds modeling, which enables us to handle inference queries of 2^D different possible combinations of missing patterns. Last but not the least, the efficient information reward estimation method is only possible due to the latent representations provided by the encoder-decoder structures of VAEs.
- Finally, the introduction of deep neural networks to missing data imputation does introduce certain pathologies due to model non-identification. Therefore, in Chapter 8, this problem is further studied by analyzing the sufficient conditions of identifiability for generative models under MNAR assumption.

In this thesis, we have proposed a number of new directions and new approaches in Bayesian inference, by introducing ideas from the deep learning literature to approximate inference and Bayesian machine learning. All these new advances would not be possible without the recent developments of deep learning methods in the past decades. Therefore, the new directions discussed in this thesis also open leads to future works, as detailed in the next section.

9.2 Future research questions for function space inference

9.2.1 Function space inference beyond VI

Our contributions in function space inference are mainly based on variational inference, i.e., constructing algorithms that minimize (grid-) functional KL divergences between stochastic processes. Nevertheless, the way that we define the grid-functional KL divergence in Chapter

5 is constructive: we take the KL divergence on finite measure points, and marginalize out the measure points w.r.t. n and $\mathbf{X}_n$. This gives us a valid divergence in function space. The same approach can be used to define function space variants of other divergences, such as α divergences [113], f-divergences [351], χ-divergences [60], etc. It would be interesting to see how different divergence measures will impact the behavior of inference algorithms.

9.2.2 End-to-end functional divergence approximations

To some degree, the grid-functional divergence estimation method proposed in Chapter 5 is a two-stage method: it first approximates the functional prior with stochastic process generators (SPGs) and then uses the resulting SPG as a surrogate before estimating the functional divergence. The drawback of this approach is that the quality of the divergence estimation is largely based on the quality of the surrogate SPG, which is not always guaranteed since this SPG is learned via a finite number of samples drawn from the prior. Therefore, it will be desirable to develop an end-to-end divergence estimation method that can be directly computed using samples from the prior. One possibility is to develop the function space variant of Stein discrepancy [191–193], which is a special case of the integral probability metric, which is commonly used for quantifying differences between a probability measure and a set of samples.

9.2.3 Function space VI for BNNs

The two function space inference methods presented in this thesis are developed for general purposes and can be applied to functional priors that are not limited to BNNs. However, it would be interesting to develop FVI algorithms that are specifically designed for BNNs. This would allow us to take advantage of the structures of BNNs, such that further algorithmic simplifications and accelerations can be established. Moreover, the dimension reduction methods for BNNs discussed in in Chapter 2 Section 2.3.2 [51, 131, 64] are naturally compatible with our functional variational inference (Chapter 5). The reason for this is that their low dimensional representations can be used to efficiently estimate the grid-functional KL divergences using latent space approximations similar to Proposition 5.5.

9.2.4 Theoretical properties of FVI

The functional divergence estimation method proposed in Chapter 5 is a combination of different approximation techniques, such as surrogate approximation of the prior, MC sampling of the measure points, MC sampling from the posterior process, as well as debiasing techniques based on Russian roulette estimators. It is, therefore, important to carry further theoretical analysis regarding the variance and asymptotic convergence properties of such estimators, as well as related variance reduction techniques for function space inference. This is a topic that is ignored by many function space inference literature and is worth further investigation. Finally, it is also helpful to investigate the posterior consistency and contraction rates of function space inference methods (assuming all approximations are accurate), and compare with the corresponding contraction rates of weight-space VI [27].

9.3 Future directions for generative modeling under missing data uncertainty

9.3.1 Fast approximations and stop criterion for large-scale information acquisition

In the EDDI framework introduced in Chapter 9, the computational time complexity for active information acquisition is $\mathcal{O}(D^2 \log D)$, where D is the dimension of features in each data point. This is too demanding for problems with high dimensional features, such as time-series clinical data, recommender systems data, and image data, etc. Most recently, our preliminary work of [106] proposed a new acceleration strategy designed specifically for symptom-based self-diagnosis problems, which allows us to apply the EDDI framework to large healthcare datasets. The acceleration strategy is based on irrelevant symptom identification, fast information reward approximation, and an early stopping criterion by monitoring the posterior (missing data) uncertainty levels. When combined together, we are able to outperform state-of-the-art reinforcement learning baselines for self-diagnosis. The empirical success of such acceleration strategies motivates us to further develop large-scale information acquisition methods for general tasks beyond self-diagnosis.

9.3.2 Learning hierarchical Bayesian deep generative models under missing data

Deep neural networks are the key ingredient of deep generative models, which enables us to process a massive amount of unlabelled raw data. Therefore, similar to the supervised learning case, it is also equivalently important to quantify the parameter uncertainties of these neural networks. Also, it has been argued [86] that such epistemic uncertainty in generative models is crucial to handling the ice-start problem, where there is little or no training data from the beginning. Therefore, it is desirable to also assume a prior on the generative model parameters $\boldsymbol{\theta}$, and perform approximate inference over both model parameters $\boldsymbol{\theta}$ and the latent variables, $\mathbf{z}$. Unfortunately, this would add heavy computational burdens to VAE-like amortized inference methods: imagine up to millions of distributions over deep neural net weights, each having complex interactions with latent variables $\{\mathbf{z}\}$. When applying amortized inference to this case, such correlations between $\boldsymbol{\theta}$ and $\mathbf{z}$ can be difficult to model using inference networks. In the preliminary work of [86], such interactions are ignored by applying a mean-field approximation. However, such approximations will be quite limited, especially when applied to more complex models (such as hierarchical generative models). Therefore, one possible future direction is to investigate how to perform learning and inference for such hierarchical Bayesian generative models under missing data. A potentially promising direction is to combine (Hamiltonian) MCMC methods and variational inference for specific generative models, to demystify the correlations between $\boldsymbol{\theta}$ and $\mathbf{z}$.

9.3.3 Deep generative models for partially observed, mixed type tabular data.

In the deep learning literature, deep generative models such as VAEs are typically applied to standard homogeneous datasets in which each data dimension has a similar type and similar statistical properties (e.g., consider spatial or temporal correlations found in images and videos). However, many real-world datasets are tabular datasets, which are *heterogeneous* and contain variables with different types. For instance, in healthcare applications, a patient record may contain demographic information such as nationality (which is of categorical type), age (which is ordinal), and height (which is continuous). In our work in Part B, we have demonstrated that deep generative models can also be successfully applied to partially observed tabular datasets, which opens up new possibilities that broaden the range of applications where deep generative models can be deployed. However, our treatment of mixed-type data is quite ad-hoc: we simply convert all variables into continuous type variables and apply Partial VAEs with gaussian likelihoods to the processed data. A more principled way of learning from tabular data is to treat each different variable type correctly and apply the different likelihood functions for each type (e.g. Gaussian likelihoods for real-valued variables and Bernoulli likelihoods for binary variables). In our preliminary work in [201], we argue that naively applying VAEs to such mixed-type heterogeneous data can lead to unsatisfying results. The reason for this is that the contribution that each likelihood makes to the training objective can be very different, leading to challenging optimization problems in which some data dimensions may be poorly-modeled in favor of others. In the same paper, we proposed a preliminary two-stage solution to this problem, by first learning a homogeneous representation of each heterogeneous variable, and then learning a partial VAE over those heterogeneous representations. Nevertheless, it is still an open question how to develop extensions of deep generative models to properly handle partially observed, mixed typed tabular datasets.

9.3.4 Extending identifiability results for deep generative models under missing data.

In Chapter 8, we have studied certain sets of sufficient conditions for the identifiability of generative models under MNAR. To some degree, our proposed conditions are "meta" conditions, in the sense that these sufficient conditions rely on the subset-identifiability of the base model, $p_{\boldsymbol{\theta}}(\mathbf{x})$. While such setting is convenient to work with in practice, our results are naturally restricted by the limitations of the specific base model, $p_{\boldsymbol{\theta}}(\mathbf{x})$. For example, we mainly used identifiable VAE models [146] as the base model, which requires a set of fully observed auxiliary variables (V), which is often quite hard to obtain, especially in applications such as recommender systems and surveys. Therefore, a future direction would be to lift such constraints.

9.4 Interplay between the techniques developed in Theme A and Theme B

As summarized in Table 9.1, each of Theme A and Theme B covers quite a different set of topics, methodologies, and applications. Due to this reason, these materials are organized and presented separately in this thesis. Nevertheless, we do believe that the techniques developed under each individual theme will also benefit the other one. Therefore, one possible direction for future work is to further explore such interplay between these two themes. For example, similar to the approach developed in Gaussian latent variable models (GPLVMs) [165], the function space inference method in Theme A can also be used to quantify epistemic uncertainty in Bayesian deep generative models. This may subsequently help improve the missing data imputation and decision making of Theme B. Also, it would also be interesting to investigate how function space inference methods would impact the model identifiability and asymptotic convergence properties of the deep generative models under MNAR from Theme B. Finally, models such as implicit processes, or Bayesian regression models trained by FVI can also be trivially combined with Partial VAE to perform active feature acquisition. This can be done via the factorization $p(\mathbf{x}_\phi, \mathbf{x} \setminus \mathbf{x}_\phi, \mathbf{z}) = p(\mathbf{x} \setminus \mathbf{x}_\phi, \mathbf{z}) p(\mathbf{x}_\phi | \mathbf{z}, \mathbf{x} \setminus \mathbf{x}_\phi)$, where the term $p(\mathbf{x}_\phi | \mathbf{z}, \mathbf{x} \setminus \mathbf{x}_\phi)$ can be modelled by a Bayesian regression model. By monitoring the uncertainty level on both $\mathbf{z}$ and $\mathbf{x}_\phi$, we can also establish an optimal stop criterion for active information acquisition.

Bibliography

[1] (2005). National health and nutrition examination survey.

[2] Agakov, F. V. and Barber, D. (2004). An auxiliary variational method. In *International Conference on Neural Information Processing*, pages 561–566. Springer.

[3] Aitchison, L. (2020). A statistical theory of cold posteriors in deep neural networks. In *International Conference on Learning Representations*.

[4] Al-Shedivat, M., Wilson, A. G., Saatchi, Y., Hu, Z., and Xing, E. P. (2017). Learning scalable deep kernels with recurrent structure. *The Journal of Machine Learning Research*, 18(1):2850–2886.

[5] Allison, P. D. (2001). *Missing data*. Sage publications.

[6] Allman, E. S., Matias, C., and Rhodes, J. A. (2009). Identifiability of parameters in latent structure models with many observed variables. *The Annals of Statistics*, 37(6A):3099–3132.

[7] Andrieu, C., De Freitas, N., Doucet, A., and Jordan, M. I. (2003). An introduction to MCMC for machine learning. *Machine learning*, 50(1):5–43.

[8] Anscombe, F. J., Aumann, R. J., et al. (1963). A definition of subjective probability. *Annals of mathematical statistics*, 34(1):199–205.

[9] Arbuckle, J. L., Marcoulides, G. A., and Schumacker, R. E. (1996). Full information estimation in the presence of incomplete data. *Advanced structural equation modeling: Issues and techniques*, 243:277.

[10] Arenz, O., Neumann, G., and Zhong, M. (2018). Efficient gradient-free variational inference using policy search. In *International conference on machine learning*, pages 234–243. PMLR.

[11] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR.

[12] Asuncion, A. and Newman, D. (2007). UCI machine learning repository.

[13] Bahadur, R. R. (1954). Sufficiency and statistical decision functions. *The annals of mathematical Statistics*, pages 423–462.

[14] Ball, K. (1992). Eigenvalues of euclidean distance matrices. *Journal of Approximation Theory*, 68(1):74–82.

[15] Balog, M., Lakshminarayanan, B., Ghahramani, Z., Roy, D. M., and Teh, Y. W. (2016). The mondrian kernel. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pages 32–41.

[16] Barankin, E. (1961). Sufficient parameters: solution of the minimal dimensionality problem. *The Annals of Mathematical Statistics*, pages 91–118.

[17] Barber, D. and Bishop, C. M. (1998). Ensemble learning in Bayesian neural networks. *Nato ASI Series F Computer and Systems Scieneces*, 168:215–238.

[18] Baxter, B. (1994). Norm estimates for inverses of Toeplitz distance matrices. *Journal of Approximation Theory*, 79(2):222–242.

[19] Beal, M. J. (2003). *Variational algorithms for approximate Bayesian inference*. University of London, University College London (United Kingdom).

[20] Beaumont, M. A., Cornuet, J.-M., Marin, J.-M., and Robert, C. P. (2009). Adaptive approximate Bayesian computation. *Biometrika*, 96(4):983–990.

[21] Beck, D. and Cohn, T. (2017). Learning kernels over strings using Gaussian processes. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 67–73.

[22] Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and trends® in Machine Learning*, 2(1):1–127.

[23] Bengio, Y., Delalleau, O., and Le Roux, N. (2005). The curse of dimensionality for local kernel machines. *Techn. Rep*, 1258.

[24] Berger, J. O. (2013). *Statistical decision theory and Bayesian analysis*. Springer Science & Business Media.

[25] Berk, R. H. (1970). Consistency a posteriori. *The Annals of Mathematical Statistics*, pages 894–906.

[26] Bernardo, J. M. (1979). Expected information as expected utility. *The Annals of Statistics*, pages 686–690.

[27] Bhattacharya, S., Liu, Z., and Maiti, T. (2020). Variational Bayes neural network: Posterior consistency, classification accuracy and computational challenges. *arXiv preprint arXiv:2011.09592*.

[28] Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

[29] Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015). Weight uncertainty in neural network. In *International conference on machine learning*, pages 1613–1622. PMLR.

[30] Bonassi, F. V., West, M., et al. (2015). Sequential Monte Carlo with adaptive weights for approximate Bayesian computation. *Bayesian Analysis*, 10(1):171–187.

[31] Bornschein, J. and Bengio, Y. (2014). Reweighted wake-sleep. *arXiv:1406.2751.*

[32] Botev, A., Ritter, H., and Barber, D. (2017). Practical Gauss-Newton optimisation for deep learning. In *International Conference on Machine Learning*, pages 557–565. PMLR.

[33] Bottou, L. et al. (1998). Online learning and stochastic approximations. *On-line learning in neural networks*, 17(9):142.

[34] Bradshaw, J., Matthews, A. G. d. G., and Ghahramani, Z. (2017). Adversarial examples, uncertainty, and transfer testing robustness in Gaussian process hybrid deep networks. *arXiv:1707.02476.*

[35] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

[36] Bui, T., Hernández-Lobato, D., Hernandez-Lobato, J., Li, Y., and Turner, R. (2016a). Deep Gaussian processes for regression using approximate expectation propagation. In *International Conference on Machine Learning*, pages 1472–1481.

[37] Bui, T. D. and Turner, R. E. (2014). Tree-structured Gaussian process approximations. In *Advances in Neural Information Processing Systems*, pages 2213–2221.

[38] Bui, T. D., Yan, J., and Turner, R. E. (2016b). A unifying framework for sparse Gaussian process approximation using power expectation propagation. *arXiv:1605.07066.*

[39] Burt, D. R., Ober, S. W., Garriga-Alonso, A., and van der Wilk, M. (2020). Understanding variational inference in function-space. *arXiv preprint arXiv:2011.09421.*

[40] Casert, C., Mills, K., Vieijra, T., Ryckebusch, J., and Tamblyn, I. (2020). Optical lattice experiments at unobserved conditions and scales through generative adversarial deep learning. *arXiv preprint arXiv:2002.07055.*

[41] Chen, R. T., Behrmann, J., Duvenaud, D. K., and Jacobsen, J.-H. (2019). Residual flows for invertible generative modeling. In *Advances in Neural Information Processing Systems*, pages 9916–9926.

[42] Cho, Y. and Saul, L. K. (2009). Kernel methods for deep learning. In *Advances in Neural Information Processing Systems*, pages 342–350.

[43] Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 160–167. ACM.

[44] Cover, T. M. and Thomas, J. A. (2012). *Elements of information theory.* John Wiley & Sons.

[45] Cox, R. T. (1946). Probability, frequency and reasonable expectation. *American journal of physics*, 14(1):1–13.

[46] Cunningham, J. P., Shenoy, K. V., and Sahani, M. (2008). Fast Gaussian process methods for point process intensity estimation. In *Proceedings of the 25th International Conference on Machine Learning*, pages 192–199. ACM.

[47] Cutajar, K., Bonilla, E. V., Michiardi, P., and Filippone, M. (2017). Random feature expansions for deep Gaussian processes. In *International Conference on Machine Learning*, pages 884–893. PMLR.

[48] Dai, B. and Wipf, D. (2019). Diagnosing and enhancing VAE models. In *International Conference on Learning Representations*.

[49] Damianou, A. and Lawrence, N. (2013). Deep Gaussian processes. In *Artificial Intelligence and Statistics*, pages 207–215.

[50] Daniely, A., Frostig, R., and Singer, Y. (2016). Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances in Neural Information Processing Systems*, pages 2253–2261.

[51] Daxberger, E., Nalisnick, E., Allingham, J. U., Antorán, J., and Hernández-Lobato, J. M. (2021). Bayesian deep learning via subnetwork inference. In *International Conference on Machine Learning*, pages 2510–2521. PMLR.

[52] Dayan, P., Hinton, G. E., Neal, R. M., and Zemel, R. S. (1995). The helmholtz machine. *Neural computation*, 7(5):889–904.

[53] De Finetti, B. (1937). La prévision: ses lois logiques, ses sources subjectives. In *Annales de l'institut Henri Poincaré*, volume 7, pages 1–68.

[54] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977a). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38.

[55] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977b). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

[56] Denker, J. and Lecun, Y. (1991). Transforming neural-net output levels to probability distributions. In *Advances in Neural Information Processing Systems 3*. Citeseer.

[Depeweg et al.] Depeweg, S., Hernández-Lobato, J. M., Doshi-Velez, F., and Udluft, S. Learning and policy search in stochastic dynamical systems with Bayesian neural networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

[58] Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of*

the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers), pages 4171–4186. Association for Computational Linguistics.

[59] Dheeru, D. and Karra Taniskidou, E. (2017). UCI machine learning repository.

[60] Dieng, A. B., Tran, D., Ranganath, R., Paisley, J., and Blei, D. (2017). Variational inference via χ-upper bound minimization. *Advances in Neural Information Processing Systems*, 30.

[61] Diggle, P. J. and Gratton, R. J. (1984). Monte Carlo methods of inference for implicit statistical models. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 193–227.

[62] Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, pages 647–655.

[63] Doob, J. L. (1949). Applications of the theory of martingales. In *Le Calcul des Probabilite ´s et ses Applications*, 13, Paris. Colloques Internationaux du Centre National de la Recherche Scientifique.

[64] Dusenberry, M., Jerfel, G., Wen, Y., Ma, Y., Snoek, J., Heller, K., Lakshminarayanan, B., and Tran, D. (2020). Efficient and scalable Bayesian neural nets with rank-1 factors. In *International conference on machine learning*, pages 2782–2792. PMLR.

[65] Duvenaud, D., Lloyd, J., Grosse, R., Tenenbaum, J., and Zoubin, G. (2013). Structure discovery in nonparametric regression through compositional kernel search. In *International Conference on Machine Learning*, pages 1166–1174. PMLR.

[66] Duvenaud, D. K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A., and Adams, R. P. (2015). Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems*, pages 2224–2232.

[67] Enders, C. K. and Bandalos, D. L. (2001). The relative performance of full information maximum likelihood estimation for missing data in structural equation models. *Structural equation modeling*, 8(3):430–457.

[68] Farquhar, S., Smith, L., and Gal, Y. (2020). Liberty or depth: Deep Bayesian neural nets do not need complex weight posterior approximations. *Advances in Neural Information Processing Systems*, 33:4346–4357.

[69] Flam-Shepherd, D., Requeima, J., and Duvenaud, D. (2017). Mapping Gaussian process priors to Bayesian neural networks. *NIPS Bayesian deep learning workshop*.

[70] Foong, A., Burt, D., Li, Y., and Turner, R. (2020). On the expressiveness of approximate inference in Bayesian neural networks. *Advances in Neural Information Processing Systems*, 33:15897–15908.

[71] Foong, A. Y., Li, Y., Hernández-Lobato, J. M., and Turner, R. E. (2019). 'in-between'uncertainty in Bayesian neural networks. *arXiv preprint arXiv:1906.11537*.

[72] Fortuin, V., Garriga-Alonso, A., Wenzel, F., Rätsch, G., Turner, R., van der Wilk, M., and Aitchison, L. (2021). Bayesian neural network priors revisited. *arXiv preprint arXiv:2102.06571*.

[73] Fu, M. C. (2006). Gradient estimation. *Handbooks in operations research and management science*, 13:575–616.

[74] Gal, Y. (2016). *Uncertainty in deep learning*. PhD thesis, PhD thesis, University of Cambridge.

[75] Gal, Y. and Ghahramani, Z. (2016a). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059.

[76] Gal, Y. and Ghahramani, Z. (2016b). A theoretically grounded application of dropout in recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1019–1027.

[77] Gal, Y., Hron, J., and Kendall, A. (2017). Concrete dropout. *Advances in neural information processing systems*, 30.

[78] Gal, Y. and Turner, R. (2015). Improving the Gaussian process sparse spectrum approximation by representing uncertainty in frequency inputs. In *International Conference on Machine Learning*, pages 655–664.

[79] Garnelo, M., Rosenbaum, D., Maddison, C., Ramalho, T., Saxton, D., Shanahan, M., Teh, Y. W., Rezende, D., and Eslami, S. A. (2018a). Conditional neural processes. In *International Conference on Machine Learning*, pages 1704–1713. PMLR.

[80] Garnelo, M., Schwarz, J., Rosenbaum, D., Viola, F., Rezende, D. J., Eslami, S., and Teh, Y. W. (2018b). Neural processes. *arXiv preprint arXiv:1807.01622*.

[81] Gershman, S. J., Hoffman, M. D., and Blei, D. M. (2012). Nonparametric variational inference. In *Proceedings of the 29th International Coference on Machine Learning*, pages 235–242.

[82] Ghalebikesabi, S., Cornish, R., Holmes, C., and Kelly, L. (2021). Deep generative missingness pattern-set mixture models. In *International Conference on Artificial Intelligence and Statistics*, pages 3727–3735. PMLR.

[83] Gilboa, I. (2009). *Theory of decision under uncertainty*, volume 45. Cambridge university press.

[84] Globerson, A. and Livni, R. (2016). Learning infinite-layer networks: beyond the kernel trick. arxiv preprint. *arXiv preprint arXiv:1606.05316*.

[85] Gong, W., Li, Y., and Hernández-Lobato, J. M. (2021a). Sliced kernelized Stein discrepancy. In *International Conference on Learning Representations*.

[86] Gong, W., Tschiatschek, S., Nowozin, S., Turner, R. E., Hernández-Lobato, J. M., and Zhang, C. (2019). Icebreaker: Element-wise efficient information acquisition with a Bayesian deep latent Gaussian model.

[87] Gong, Y., Hajimirsadeghi, H., He, J., Durand, T., and Mori, G. (2021b). Variational selective autoencoder: Learning from partially-observed heterogeneous data. In *International Conference on Artificial Intelligence and Statistics*, pages 2377–2385. PMLR.

[88] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680.

[89] Gopalan, P. K., Charlin, L., and Blei, D. (2014). Content-based recommendations with poisson factorization. In *Advances in Neural Information Processing Systems*, pages 3176–3184.

[90] Gordon, J., Bruinsma, W. P., Foong, A. Y. K., Requeima, J., Dubois, Y., and Turner, R. E. (2020). Convolutional conditional neural processes. In *International Conference on Learning Representations*.

[91] Graves, A. (2011). Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems*, pages 2348–2356.

[92] Gray, R. M. (2011). *Entropy and information theory*. Springer Science & Business Media.

[93] Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. (2012). A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773.

[94] Grünwald, P. and Van Ommen, T. (2017). Inconsistency of Bayesian inference for misspecified linear models, and a proposal for repairing it. *Bayesian Analysis*, 12(4):1069–1103.

[95] Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR.

[96] Guo, F., Wang, X., Fan, K., Broderick, T., and Dunson, D. B. (2016). Boosting variational inference. *arXiv preprint arXiv:1611.05559*.

[97] Guss, W. H. (2016). Deep function machines: Generalized neural networks for topological layer expression. *arXiv preprint arXiv:1612.04799*.

[98] Hachmann, J., Olivares-Amaya, R., Jinich, A., Appleton, A. L., Blood-Forsythe, M. A., Seress, L. R., Roman-Salgado, C., Trepte, K., Atahan-Evrenk, S., Er, S., et al. (2014). Lead candidates for high-performance organic photovoltaics from high-throughput quantum chemistry–the harvard clean energy project. *Energy & Environmental Science*, 7(2):698–704.

[99] Hamesse, C., Ackermann, P., Kjellström, H., and Zhang, C. (2018). Simultaneous measurement imputation and outcome prediction for achilles tendon rupture rehabilitation. In *ICML/IJCAI Joint Workshop on Artificial Intelligence in Health*.

[100] Han, S., Liao, X., Dunson, D., and Carin, L. (2016). Variational Gaussian copula inference. In *Artificial Intelligence and Statistics*, pages 829–838. PMLR.

[101] Harsanyi, J. C. (1978). Bayesian decision theory and utilitarian ethics. *The American Economic Review*, 68(2):223–228.

[102] Harsanyi, J. C. (1979). Bayesian decision theory, rule utilitarianism, and arrow's impossibility theorem. *Theory and Decision*, 11(3):289–317.

[103] Harutyunyan, H., Khachatrian, H., Kale, D. C., Ver Steeg, G., and Galstyan, A. (2019). Multitask learning and benchmarking with clinical time series data. *Scientific data*, 6(1):1–18.

[104] Hazan, T. and Jaakkola, T. (2015). Steps toward deep kernel methods from infinite neural networks. *arXiv:1508.05133*.

[105] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

[106] He, W., Mao, X., Ma, C., Huang, Y., Hernàndez-Lobato, J. M., and Chen, T. (2022). BSODA: A bipartite scalable framework for online disease diagnosis. In *Proceedings of the ACM Web Conference 2022*, pages 2511–2521.

[107] Heckman, J. J. (1979). Sample selection bias as a specification error. *Econometrica: Journal of the econometric society*, pages 153–161.

[108] Heinemann, U., Livni, R., Eban, E., Elidan, G., and Globerson, A. (2016). Improper deep kernels. In *Artificial Intelligence and Statistics*, pages 1159–1167.

[109] Hensman, J., Fusi, N., and Lawrence, N. D. (2013). Gaussian processes for big data. *arXiv:1309.6835*.

[110] Hernández-Lobato, J. M. (2010). Balancing flexibility and robustness in machine learning: semi-parametric methods and sparse linear models.

[111] Hernández-Lobato, J. M. and Adams, R. (2015). Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869.

[112] Hernández-Lobato, J. M., Houlsby, N., and Ghahramani, Z. (2014). Probabilistic matrix factorization with non-random missing data. In *International Conference on Machine Learning*, pages 1512–1520. PMLR.

[113] Hernández-Lobato, J. M., Li, Y., Rowland, M., Hernández-Lobato, D., Bui, T., and Turner, R. E. (2016). Black-box α-divergence minimization.

[114] Hinton, G. E., Dayan, P., Frey, B. J., and Neal, R. M. (1995). The" wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158.

[115] Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554.

[116] Hinton, G. E. and Salakhutdinov, R. R. (2008). Using deep belief nets to learn covariance kernels for Gaussian processes. In *Advances in Neural Information Processing Systems*, pages 1249–1256.

[117] Hinton, G. E. and Van Camp, D. (1993). Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, pages 5–13. ACM.

[118] Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347.

[119] Horton, N. J. and Lipsitz, S. R. (2001). Multiple imputation in practice: comparison of software packages for regression models with missing variables. *The American Statistician*, 55(3):244–254.

[120] Horvitz, D. G. and Thompson, D. J. (1952). A generalization of sampling without replacement from a finite universe. *Journal of the American statistical Association*, 47(260):663–685.

[121] Houlsby, N., Huszár, F., Ghahramani, Z., and Lengyel, M. (2011). Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*.

[122] Hron, J., Matthews, A., and Ghahramani, Z. (2018). Variational Bayesian dropout: pitfalls and fixes. In *International Conference on Machine Learning*, pages 2019–2028. PMLR.

[123] Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269. IEEE.

[124] Huang, S.-J., Xu, M., Xie, M.-K., Sugiyama, M., Niu, G., and Chen, S. (2018). Active feature acquisition with supervised matrix completion. *arXiv preprint arXiv:1802.05380*.

[125] Huszár, F. (2017). Variational inference using implicit distributions. *arXiv preprint arXiv:1702.08235*.

[126] Ibrahim, J. G., Lipsitz, S. R., and Chen, M.-H. (1999). Missing covariates in generalized linear models when the missing data mechanism is non-ignorable. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(1):173–190.

[127] Immer, A., Korzepa, M., and Bauer, M. (2021). Improving predictions of Bayesian neural nets via local linearization. In *International Conference on Artificial Intelligence and Statistics*, pages 703–711. PMLR.

[128] Ipsen, N. B., Mattei, P.-A., and Frellsen, J. (2021). not-{miwae}: Deep generative modelling with missing not at random data. In *International Conference on Learning Representations*.

[129] Itô, K. (1984). *An Introduction to Probability Theory*. Cambridge University Press.

[130] Iwata, T. and Ghahramani, Z. (2017). Improving output uncertainty estimation and generalization in deep learning via neural network Gaussian processes. *arXiv:1707.05922*.

[131] Izmailov, P., Maddox, W. J., Kirichenko, P., Garipov, T., Vetrov, D., and Wilson, A. G. (2020). Subspace inference for Bayesian deep learning. In *Uncertainty in Artificial Intelligence*, pages 1169–1179. PMLR.

[132] Izmailov, P., Vikram, S., Hoffman, M. D., and Wilson, A. G. G. (2021). What are Bayesian neural network posteriors really like? In *International conference on machine learning*, pages 4629–4640. PMLR.

[133] Jain, P., Meka, R., and Dhillon, I. S. (2010). Guaranteed rank minimization via singular value projection. In *Advances in Neural Information Processing Systems*.

[134] Jakobsen, J. C., Gluud, C., Wetterslev, J., and Winkel, P. (2017). When and how should multiple imputation be used for handling missing data in randomised clinical trials–a practical guide with flowcharts. *BMC medical research methodology*, 17(1):1–10.

[135] Jang, E., Gu, S., and Poole, B. (2017). Categorical reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*.

[136] Jannach, D., Zanker, M., Felfernig, A., and Friedrich, G. (2010). *Recommender systems: an introduction*. Cambridge University Press.

[137] Jaynes, E. T. (2003). *Probability theory: The logic of science*. Cambridge university press.

[138] Jeffreys, H. (1998). *The theory of probability*. OUP Oxford.

[139] Johnson, A. E., Pollard, T. J., Shen, L., Li-wei, H. L., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Celi, L. A., and Mark, R. G. (2016). Mimic-iii, a freely accessible critical care database. *Scientific Data*, 3:160035.

[140] 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.

[141] Kahn, H. (1955). Use of different Monte Carlo sampling techniques.

[142] Kalchbrenner, N. and Blunsom, P. (2013). Recurrent continuous translation models. In *EMNLP*, volume 3, page 413.

[143] Keerin, P., Kurutach, W., and Boongoen, T. (2012). Cluster-based KNN missing value imputation for dna microarray data. In *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 445–450. IEEE.

[144] Keshavan, R. H., Montanari, A., and Oh, S. (2010). Matrix completion from noisy entries. *Journal of Machine Learning Research*.

[145] Khan, M. E. E., Immer, A., Abedi, E., and Korzepa, M. (2019). Approximate inference turns deep networks into Gaussian processes. *Advances in neural information processing systems*, 32.

[146] Khemakhem, I., Kingma, D., Monti, R., and Hyvarinen, A. (2020). Variational autoencoders and nonlinear ICA: A unifying framework. In *International Conference on Artificial Intelligence and Statistics*, pages 2207–2217. PMLR.

[147] Kim, H., Mnih, A., Schwarz, J., Garnelo, M., Eslami, A., Rosenbaum, D., Vinyals, O., and Teh, Y. W. (2019). Attentive neural processes. In *International Conference on Learning Representations*.

[148] Kingma, D. P. and Ba, J. L. (2015). Adam: a method for stochastic optimization. In *International Conference on Learning Representations*, pages 1–13.

[149] Kingma, D. P. and Welling, M. (2013). Auto-encoding variational Bayes. *arXiv:1312.6114*.

[150] Kingma, D. P. and Welling, M. (2014). Auto-encoding variational Bayes. In *International Conference on Learning Representation*.

[151] Kleijn, B. J. and van der Vaart, A. W. (2012). The Bernstein-von-Mises theorem under misspecification. *Electronic Journal of Statistics*, 6:354–381.

[152] Kleijn, B. J. K. and van der Vaart, A. W. (2006). Misspecification in infinite-dimensional Bayesian statistics. *The Annals of Statistics*, 34(2).

[153] Knapik, B. (2013). *Bayesian Asymptotics: Inverse Problems and Irregular Models*. PhD thesis, Vrije Universiteit Amsterdam. Naam instelling promotie: VU Vrije Universiteit Naam instelling onderzoek: VU Vrije Universiteit.

[154] Koller, D. and Friedman, N. (2009). *Probabilistic graphical models: principles and techniques*. MIT press.

[155] Kolmogorov, A. (1950). *Foundations of the theory of probability*. Chelsea Publishing Company.

[156] Koopmans, T. C. and Reiersol, O. (1950). The identification of structural characteristics. *The Annals of Mathematical Statistics*, 21(2):165–181.

[157] Körding, K. P. and Wolpert, D. M. (2004). Bayesian integration in sensorimotor learning. *Nature*, 427(6971):244–247.

[158] Körding, K. P. and Wolpert, D. M. (2006). Bayesian decision theory in sensorimotor control. *Trends in cognitive sciences*, 10(7):319–326.

[159] Krauth, K., Bonilla, E. V., Cutajar, K., and Filippone, M. (2016). AutoGP: Exploring the capabilities and limitations of Gaussian process models. *arXiv:1610.05392*.

[160] Kristiadi, A., Hein, M., and Hennig, P. (2020). Being Bayesian, even just a bit, fixes overconfidence in Relu networks. In *International Conference on Machine Learning*, pages 5436–5446. PMLR.

[161] Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

[162] 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.

[163] Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

[164] Laplace, P. S. (1820). *Théorie analytique des probabilités*. Courcier.

[165] Lawrence, N. D. (2004). Gaussian process latent variable models for visualisation of high dimensional data. In *Advances in Neural Information Processing Systems*, pages 329–336.

[166] Lázaro-Gredilla, M., Quiñonero-Candela, J., Rasmussen, C. E., and Figueiras-Vidal, A. R. (2010). Sparse spectrum Gaussian process regression. *Journal of Machine Learning Research*, 11(Jun):1865–1881.

[167] Le, Q. V. (2013). Building high-level features using large scale unsupervised learning. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8595–8598. IEEE.

[168] Le Cam, L. (2012). *Asymptotic methods in statistical decision theory*. Springer Science & Business Media.

[169] Le Roux, N. and Bengio, Y. (2007). Continuous neural networks. In *Artificial Intelligence and Statistics*, pages 404–411.

[170] Lean, J., Beer, J., and Bradley, R. (1995). Reconstruction of solar irradiance since 1610: Implications for climate change. *Geophysical Research Letters*, 22(23):3195–3198.

[171] LeCun, Y. (1998). The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.

[172] Lee, H. K. (2000). Consistency of posterior distributions for neural networks. *Neural Networks*, 13(6):629–642.

[173] Lee, J., Sohl-dickstein, J., Pennington, J., Novak, R., Schoenholz, S., and Bahri, Y. (2018). Deep neural networks as Gaussian processes. In *International Conference on Learning Representations*.

[174] Lewenberg, Y., Bachrach, Y., Paquet, U., and Rosenschein, J. S. (2017). Knowing what to ask: A Bayesian active learning approach to the surveying problem. In *AAAI*, pages 1396–1402.

[175] Li, S. C.-X., Jiang, B., and Marlin, B. (2019). MisGAN: Learning from incomplete data with generative adversarial networks. In *International Conference on Learning Representations*.

[176] Li, Y. (2018). *Approximate inference: New visions*. PhD thesis, University of Cambridge.

[177] Li, Y. and Gal, Y. (2017). Dropout inference in Bayesian neural networks with alpha-divergences. In *International conference on machine learning*, pages 2052–2061. PMLR.

[178] Li, Y., Hernández-Lobato, J. M., and Turner, R. E. (2015). Stochastic expectation propagation. In *Advances in Neural Information Processing Systems*, pages 2323–2331.

[179] Li, Y. and Liu, Q. (2016). Wild variational approximations.

[180] Li, Y. and Turner, R. E. (2016). Rényi divergence variational inference. In *Advances in Neural Information Processing Systems*, pages 1073–1081.

[181] Li, Y. and Turner, R. E. (2018). Gradient estimators for implicit models. In *International Conference on Learning Representations*.

[182] Li, Y., Turner, R. E., and Liu, Q. (2017). Approximate inference with amortised MCMC. *arXiv:1702.08343*.

[Liang et al.] Liang, D., Charlin, L., and Blei, D. M. Causal inference for recommendation.

[184] Liang, D., Charlin, L., McInerney, J., and Blei, D. M. (2016). Modeling user exposure in recommendation. In *Proceedings of the 25th international conference on World Wide Web*, pages 951–961.

[185] Lichman, M. et al. (2013). UCI machine learning repository.

[186] Lindley, D. V. (1956). On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, pages 986–1005.

[187] Ling, G., Yang, H., Lyu, M. R., and King, I. (2012). Response aware model-based collaborative filtering. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, pages 501–510.

[188] Little, R. and Rubin, D. (1987). Statistical analysis with missing data. Technical report.

[189] Little, R. J. (1993). Pattern-mixture models for multivariate incomplete data. *Journal of the American Statistical Association*, 88(421):125–134.

[190] Little, R. J. and Rubin, D. B. (2019). *Statistical analysis with missing data*, volume 793. John Wiley & Sons.

[191] Liu, Q. and Feng, Y. (2016). Two methods for wild variational inference. *arXiv:1612.00081*.

[192] Liu, Q., Lee, J. D., and Jordan, M. I. (2016). A kernelized Stein discrepancy for goodness-of-fit tests. In *Proceedings of the International Conference on Machine Learning (ICML)*.

[193] Liu, Q. and Wang, D. (2016). Stein variational gradient descent: A general purpose Bayesian inference algorithm. In *Advances in Neural Information Processing Systems*, pages 2370–2378.

[194] Loeve, M. (1977). In *Probability Theory I-II*. Springer.

[195] Louizos, C. and Welling, M. (2016). Structured and efficient variational deep learning with matrix Gaussian posteriors. In *International Conference on Machine Learning*, pages 1708–1716. PMLR.

[196] Louizos, C. and Welling, M. (2017). Multiplicative normalizing flows for variational Bayesian neural networks. In *International Conference on Machine Learning*, pages 2218–2227. PMLR.

[197] Ma, C., Gong, W., Hernández-Lobato, J. M., Koenigstein, N., Nowozin, S., and Zhang, C. (2018). Partial VAE for hybrid recommender system.

[198] Ma, C., Li, Y., and Hernández-Lobato, J. M. (2019a). Variational implicit processes. In *International Conference on Machine Learning*, pages 4222–4233. PMLR.

[199] Ma, C., Tschiatschek, S., Li, Y., Turner, R., Hernandez-Lobato, J. M., and Zhang, C. (2020a). Hm-vaes: a deep generative model for real-valued data with heterogeneous marginals. In *Symposium on Advances in Approximate Bayesian Inference*, pages 1–8. PMLR.

[200] Ma, C., Tschiatschek, S., Palla, K., Hernandez-Lobato, J. M., Nowozin, S., and Zhang, C. (2019b). EDDI: Efficient dynamic discovery of high-value information with partial VAE. In *International Conference on Machine Learning*, pages 4234–4243. PMLR.

[201] Ma, C., Tschiatschek, S., Turner, R., Hernández-Lobato, J. M., and Zhang, C. (2020b). VAEM: a deep generative model for heterogeneous mixed type data. *Advances in Neural Information Processing Systems*, 33:11237–11247.

[202] Ma, W. and Chen, G. H. (2019). Missing not at random in matrix completion: The effectiveness of estimating missingness probabilities under a low nuclear norm assumption. *Advances in Neural Information Processing Systems*, 32.

[203] MacKay, D. J. (1992a). Information-based objective functions for active data selection. *Neural computation*, 4(4):590–604.

[204] MacKay, D. J. (1992b). A practical Bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472.

[205] Mackay, D. J. C. (1992). *Bayesian methods for adaptive models*. PhD thesis, California Institute of Technology.

[206] Maddox, W. J., Benton, G., and Wilson, A. G. (2020). Rethinking parameter counting in deep models: Effective dimensionality revisited. *arXiv preprint arXiv:2003.02139*.

[207] Marjoram, P., Molitor, J., Plagnol, V., and Tavaré, S. (2003). Markov Chain Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 100(26):15324–15328.

[208] Marlin, B. M. and Zemel, R. S. (2009). Collaborative prediction and ranking with non-random missing data. In *Proceedings of the third ACM conference on Recommender systems*, pages 5–12.

[209] Marsh, H. W. (1998). Pairwise deletion for missing data in structural equation models: Nonpositive definite matrices, parameter estimates, goodness of fit, and adjusted sample sizes. *Structural Equation Modeling: A Multidisciplinary Journal*, 5(1):22–36.

[210] Martens, J. (2020). New insights and perspectives on the natural gradient method. *The Journal of Machine Learning Research*, 21(1):5776–5851.

[211] Martens, J. and Grosse, R. (2015). Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pages 2408–2417. PMLR.

[212] Mattei, P.-A. and Frellsen, J. (2019). MIWAE: Deep generative modelling and imputation of incomplete data sets. In *International Conference on Machine Learning*, pages 4413–4423. PMLR.

[213] Matthews, A. G. d. G., Hensman, J., Turner, R., and Ghahramani, Z. (2016). On sparse variational methods and the Kullback-Leibler divergence between stochastic processes. *Journal of Machine Learning Research*, 51:231–239.

[214] Matthews, A. G. d. G., Rowland, M., Hron, J., Turner, R. E., and Ghahramani, Z. (2018). Gaussian process behaviour in wide deep neural networks. *arXiv:1804.11271*.

[215] Matthews, A. G. d. G., Van Der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., León-Villagrá, P., Ghahramani, Z., and Hensman, J. (2017). GPflow: A Gaussian process library using tensorflow. *The Journal of Machine Learning Research*, 18(1):1299–1304.

[216] McCallumzy, A. K. and Nigamy, K. (1998). Employing EM and pool-based active learning for text classification. In *International Conference on Machine Learning*, pages 359–367. Citeseer.

[217] Melville, P., Saar-Tsechansky, M., Provost, F., and Mooney, R. (2004). Active feature-value acquisition for classifier induction. In *International Conference on Data Mining*, pages 483–486. IEEE.

[218] Mescheder, L., Nowozin, S., and Geiger, A. (2017). Adversarial variational Bayes: Unifying variational autoencoders and generative adversarial networks. In *International Conference on Machine Learning*, pages 2391–2400. PMLR.

[219] Miao, W., Ding, P., and Geng, Z. (2016). Identifiability of normal and normal mixture models with nonignorable missing data. *Journal of the American Statistical Association*, 111(516):1673–1683.

[220] Miao, W., Liu, L., Tchetgen, E. T., and Geng, Z. (2015). Identification, doubly robust estimation, and semiparametric efficiency theory of nonignorable missing data with a shadow variable. *arXiv preprint arXiv:1509.02556*.

[221] Miao, W. and Tchetgen, E. T. (2018). Identification and inference with nonignorable missing covariate data. *Statistica Sinica*, 28(4):2049.

[222] Miller, A. C., Foti, N. J., and Adams, R. P. (2017). Variational boosting: Iteratively refining posterior approximations. In *International Conference on Machine Learning*, pages 2420–2429. PMLR.

[223] Minka, T. P. (2001). Expectation propagation for approximate Bayesian inference. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, pages 362–369. Morgan Kaufmann Publishers Inc.

[224] Minka, T. P. (2004). Power EP. Technical report.

[225] Minka, T. P. (2005). Divergence measures and message passing. Technical report, Technical report, Microsoft Research, Cambridge.

[226] Mnih, A. and Gregor, K. (2014). Neural variational inference and learning in belief networks. In *International Conference on Machine Learning*, pages 1791–1799. PMLR.

[227] Mnih, A. and Rezende, D. (2016). Variational inference for Monte Carlo objectives. In *International Conference on Machine Learning*, pages 2188–2196. PMLR.

[228] Mnih, A. and Salakhutdinov, R. R. (2007). Probabilistic matrix factorization. *Advances in neural information processing systems*, 20:1257–1264.

[229] Moens, V., Ren, H., Maraval, A., Tutunov, R., Wang, J., and Ammar, H. (2021). Efficient semi-implicit variational inference. *arXiv preprint arXiv:2101.06070*.

[230] Mohamed, A.-r., Dahl, G. E., and Hinton, G. (2012). Acoustic modeling using deep belief networks. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):14–22.

[231] Mohamed, S., Rosca, M., Figurnov, M., and Mnih, A. (2020). Monte Carlo gradient estimation in machine learning. *J. Mach. Learn. Res.*, 21(132):1–62.

[232] Mohan, K. and Pearl, J. (2014). Graphical models for recovering probabilistic and causal queries from missing data. Technical report.

[233] Mohan, K., Pearl, J., and Tian, J. (2013). Graphical models for inference with missing data. *Advances in neural information processing systems*, 26:1277–1285.

[234] Molchanov, D., Kharitonov, V., Sobolev, A., and Vetrov, D. (2019). Doubly semi-implicit variational inference. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2593–2602. PMLR.

[235] Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press.

[236] Murray, I. A. (2007). *Advances in Markov Chain Monte Carlo methods*. University of London.

[237] Murray, J. S. et al. (2018). Multiple imputation: a review of practical and theoretical findings. *Statistical Science*, 33(2):142–159.

[238] Mustafa, M., Bard, D., Bhimji, W., Lukić, Z., Al-Rfou, R., and Kratochvil, J. M. (2019). Cosmogan: creating high-fidelity weak lensing convergence maps using generative adversarial networks. *Computational Astrophysics and Cosmology*, 6(1):1.

[239] Narcowich, F. J. and Ward, J. D. (1992). Norm estimates for the inverses of a general class of scattered-data radial-function interpolation matrices. *Journal of Approximation Theory*, 69(1):84–109.

[240] Nazabal, A., Olmos, P. M., Ghahramani, Z., and Valera, I. (2018). Handling incomplete heterogeneous data using VAEs. *arXiv preprint arXiv:1807.03653*.

[241] Nazabal, A., Olmos, P. M., Ghahramani, Z., and Valera, I. (2020). Handling incomplete heterogeneous data using VAEs. *Pattern Recognition*, 107:107501.

[242] Neal, R. M. (1996). Priors for infinite networks. In *Bayesian Learning for Neural Networks*, pages 29–53. Springer.

[243] Neal, R. M. (2012). *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media.

[244] Neal, R. M. et al. (2011). MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2:113–162.

[245] Nelson, J. D. (2008). Towards a rational theory of human information acquisition. *The probabilistic mind: Prospects for rational models of cognition*, pages 143–163.

[246] Nelson, J. D., McKenzie, C. R., Cottrell, G. W., and Sejnowski, T. J. (2010). Experience matters: Information acquisition optimizes probability gain. *Psychological science*, 21(7):960–969.

[247] Nyström, E. J. (1930). Über Die Praktische Auflösung von Integralgleichungen mit Anwendungen auf Randwertaufgaben. *Acta Mathematica*, 54(none):185 – 204.

[248] Ober, S. W. and Rasmussen, C. E. (2019). Benchmarking the neural linear model for regression. *arXiv preprint arXiv:1912.08416*.

[249] O'Neill, B. et al. (2005). Consistency and identifiability in Bayesian analysis.

[250] Osawa, K., Swaroop, S., Khan, M. E. E., Jain, A., Eschenhagen, R., Turner, R. E., and Yokota, R. (2019). Practical deep learning with Bayesian principles. *Advances in neural information processing systems*, 32.

[251] Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J., Lakshminarayanan, B., and Snoek, J. (2019). Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in neural information processing systems*, 32.

[252] Paisley, J., Blei, D. M., and Jordan, M. I. (2012). Variational Bayesian inference with stochastic search. In *Proceedings of the 29th International Coference on Machine Learning*, pages 1363–1370.

[253] Papamakarios, G. and Murray, I. (2016). Fast ε-free inference of simulation models with Bayesian conditional density estimation. In *Advances in Neural Information Processing Systems*, pages 1028–1036.

[254] Parmigiani, G. and Inoue, L. (2009). *Decision theory: principles and approaches*, volume 812. John Wiley & Sons.

[255] Paulino, C. D. M. and de Bragança Pereira, C. A. (1994). On identifiability of parametric statistical models. *Journal of the Italian Statistical Society*, 3(1):125–151.

[256] Pawlowski, J. M. and Urban, J. M. (2020). Reducing autocorrelation times in lattice simulations with generative adversarial networks. *Machine Learning: Science and Technology*, 1(4):045011.

[257] Poole, B., Lahiri, S., Raghu, M., Sohl-Dickstein, J., and Ganguli, S. (2016). Exponential expressivity in deep neural networks through transient chaos. In *Advances in Neural Information Processing Systems*, pages 3360–3368.

[258] Pourzanjani, A. A., Jiang, R. M., and Petzold, L. R. (2017). Improving the identifiability of neural networks for Bayesian inference. In *NIPS Workshop on Bayesian Deep Learning*, volume 4, page 29.

[259] Pyzer-Knapp, E. O., Li, K., and Aspuru-Guzik, A. (2015). Learning from the harvard clean energy project: The use of neural networks to accelerate materials discovery. *Advanced Functional Materials*, 25(41):6495–6502.

[260] Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017). PointNet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660.

[261] Quiñonero-Candela, J. and Rasmussen, C. E. (2005). A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6(Dec):1939–1959.

[262] Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv:1511.06434*.

[263] Rahimi, A. and Recht, B. (2008). Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184.

[264] Ramesh, A. and LeCun, Y. (2018). Backpropagation for implicit spectral densities. *arXiv preprint arXiv:1806.00499*.

[265] Ramsey, F. P. (2016). Truth and probability. In *Readings in formal epistemology*, pages 21–45. Springer.

[266] Ranganath, R., Gerrish, S., and Blei, D. M. (2014). Black box variational inference. In *AISTATS*, pages 814–822.

[267] Ranganath, R., Tran, D., Altosaar, J., and Blei, D. (2016a). Operator variational inference. *Advances in Neural Information Processing Systems*, 29:496–504.

[268] Ranganath, R., Tran, D., and Blei, D. (2016b). Hierarchical variational models. In *International Conference on Machine Learning*, pages 324–333. PMLR.

[269] Rasmussen, C. E. and Williams, C. K. (2006). *Gaussian processes for machine learning*, volume 1. MIT press Cambridge.

[270] Repecka, D., Jauniskis, V., Karpus, L., Rembeza, E., Rokaitis, I., Zrimec, J., Poviloniene, S., Laurynenas, A., Viknander, S., Abuajwa, W., et al. (2021). Expanding functional protein sequence spaces using generative adversarial networks. *Nature Machine Intelligence*, 3(4):324–333.

[271] Rezende, D. and Mohamed, S. (2015). Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR.

[272] Rezende, D. J., Mohamed, S., and Wierstra, D. (2014a). Stochastic backpropagation and approximate inference in deep generative models. In *Interantional Conference on Machine Learning*.

[273] Rezende, D. J., Mohamed, S., and Wierstra, D. (2014b). Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR.

[274] Riquelme, C., Tucker, G., and Snoek, J. (2018). Deep Bayesian bandits showdown: An empirical comparison of Bayesian deep networks for Thompson sampling. In *International Conference on Learning Representations*.

[275] Ritter, H., Botev, A., and Barber, D. (2018). A scalable laplace approximation for neural networks. In *6th International Conference on Learning Representations, ICLR 2018-Conference Track Proceedings*, volume 6. International Conference on Representation Learning.

[276] Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.

[277] Robins, J. M., Rotnitzky, A., and Zhao, L. P. (1994). Estimation of regression coefficients when some regressors are not always observed. *Journal of the American statistical Association*, 89(427):846–866.

[278] Rothenberg, T. J. (1971). Identification in parametric models. *Econometrica: Journal of the Econometric Society*, pages 577–591.

[279] Rubin, D. B. (1976). Inference and missing data. *Biometrika*, 63(3):581–592.

[280] Rubin, D. B. (1977). Formalizing subjective notions about the effect of nonrespondents in sample surveys. *Journal of the American Statistical Association*, 72(359):538–543.

[281] Rubin, D. B. (1988). An overview of multiple imputation. In *Proceedings of the survey research methods section of the American statistical association*, pages 79–84. Citeseer.

[282] Rubin, D. B. (2004). *Multiple imputation for nonresponse in surveys*, volume 81. John Wiley & Sons.

[283] Rudin, W. (1987). *Real and Complex Analysis, 3rd Ed.* McGraw-Hill, Inc., USA.

[284] Rudner, T. G., Chen, Z., and Gal, Y. (2020). Rethinking function-space variational inference in Bayesian neural networks. In *Third Symposium on Advances in Approximate Bayesian Inference*.

[285] Rudner, T. G. J., Chen, Z., Teh, Y. W., and Gal, Y. (2021). Rethinking Function-Space Variational Inference in Bayesian Neural Networks. In *Third Symposium on Advances in Approximate Bayesian Inference*.

[286] Saar-Tsechansky, M., Melville, P., and Provost, F. (2009). Active feature-value acquisition. *Management Science*, 55(4):664–684.

[287] Saatçi, Y. (2012). *Scalable inference for structured Gaussian process models*. PhD thesis, University of Cambridge.

[288] Salakhutdinov, R. and Hinton, G. E. (2009). Deep boltzmann machines. In *AISTATS*, volume 1, page 3.

[289] Salakhutdinov, R. and Mnih, A. (2008). Bayesian probabilistic matrix factorization using Markov Chain Monte Carlo. In *International conference on Machine learning*, pages 880–887. ACM.

[290] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242.

[291] Salimans, T., Kingma, D., and Welling, M. (2015). Markov Chain Monte Carlo and variational inference: Bridging the gap. In *International Conference on Machine Learning*, pages 1218–1226. PMLR.

[292] Salimbeni, H. and Deisenroth, M. P. (2017). Doubly stochastic variational inference for deep Gaussian processes. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 4591–4602.

[293] Samo, Y.-L. K. and Roberts, S. (2015). String Gaussian process kernels. *arXiv preprint arXiv:1506.02239*.

[294] Saul, L. K., Jaakkola, T., and Jordan, M. I. (1996). Mean field theory for sigmoid belief networks. *Journal of artificial intelligence research*, 4:61–76.

[295] Saul, L. K. and Jordan, M. I. (1996). Exploiting tractable substructures in intractable networks. *Advances in neural information processing systems*, pages 486–492.

[296] Savage, L. (1954). *The Foundations of Statistics*. Wiley, New York.

[297] Schaback, R. (1994). Lower bounds for norms of inverses of interpolation matrices for radial basis functions. *Journal of Approximation Theory*, 79(2):287–306.

[298] Scheffer, J. (2002). Dealing with missing data.

[299] Schnabel, T., Swaminathan, A., Singh, A., Chandak, N., and Joachims, T. (2016). Recommendations as treatments: Debiasing learning and evaluation. In *international conference on machine learning*, pages 1670–1679. PMLR.

[300] Schneider, F., Balles, L., and Hennig, P. (2019). DeepOBS: A deep learning optimizer benchmark suite. *arXiv preprint arXiv:1903.05499*.

[301] Schraudolph, N. N. (2002). Fast curvature matrix-vector products for second-order gradient descent. *Neural computation*, 14(7):1723–1738.

[302] Schwartz, L. (1965). On Bayesprocedures. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 4(1):10–26.

[303] Sedhain, S., Menon, A. K., Sanner, S., and Xie, L. (2015). Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th International Conference on World Wide Web*, pages 111–112. ACM.

[304] Seeger, M., Williams, C., and Lawrence, N. (2003). Fast forward selection to speed up sparse Gaussian process regression. In *Artificial Intelligence and Statistics 9*, number EPFL-CONF-161318.

[305] Settles, B. (2012). Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114.

[306] Shah, A., Wilson, A., and Ghahramani, Z. (2014). Student-t processes as alternatives to Gaussian processes. In *Artificial Intelligence and Statistics*, pages 877–885.

[307] Shi, J., Chen, J., Zhu, J., Sun, S., Luo, Y., Gu, Y., and Zhou, Y. (2017). ZhuSuan: A library for Bayesian deep learning. *arXiv:1709.05870*.

[308] Shi, J., Sun, S., and Zhu, J. (2018a). Kernel implicit variational inference. In *International Conference on Learning Representations*.

[309] Shi, J., Sun, S., and Zhu, J. (2018b). A spectral approach to gradient estimation for implicit distributions. In *International Conference on Machine Learning*, pages 4644–4653. PMLR.

[310] Shim, H., Hwang, S. J., and Yang, E. (2018). Joint active feature acquisition and classification with variable-size set encoding. In *Advances in Neural Information Processing Systems*.

[311] Shpitser, I., Mohan, K., and Pearl, J. (2015). Missing data as a causal and probabilistic problem. Technical report.

[312] Shrive, F. M., Stuart, H., Quan, H., and Ghali, W. A. (2006). Dealing with missing data in a multi-question depression scale: a comparison of imputation methods. *BMC medical research methodology*, 6(1):1–10.

[313] Silvestro, D. and Andermann, T. (2020). Prior choice affects ability of Bayesian neural networks to identify unknowns. *arXiv preprint arXiv:2005.04987*.

[314] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*.

[315] Snelson, E. and Ghahramani, Z. (2006). Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, pages 1257–1264.

[316] Snelson, E., Ghahramani, Z., and Rasmussen, C. E. (2004). Warped Gaussian processes. In *Advances in neural information processing systems*, pages 337–344.

[317] Sobolev, A. and Vetrov, D. P. (2019). Importance weighted hierarchical variational inference. *Advances in Neural Information Processing Systems*, 32.

[318] Sønderby, C. K., Caballero, J., Theis, L., Shi, W., and Huszár, F. (2017). Amortised MAP inference for image super-resolution. In *International Conference on Learning Representations*.

[319] Song, J., Zhao, S., and Ermon, S. (2017). A-nice-mc: Adversarial training for MCMC. *Advances in Neural Information Processing Systems*, 30.

[320] Sportisse, A., Boyer, C., and Josse, J. (2020a). Imputation and low-rank estimation with missing not at random data. *Statistics and Computing*, 30(6):1629–1643.

[321] Sportisse, A., Boyer, C., and Josses, J. (2020b). Estimation and imputation in probabilistic principal component analysis with missing not at random data. *Advances in Neural Information Processing Systems*, 33.

[322] Sriperumbudur, B. K., Gretton, A., Fukumizu, K., Schölkopf, B., and Lanckriet, G. R. (2010). Hilbert space embeddings and metrics on probability measures. *The Journal of Machine Learning Research*, 11:1517–1561.

[323] Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

[324] Steinwart, I. (2001). On the influence of the kernel on the consistency of support vector machines. *Journal of machine learning research*, 2(Nov):67–93.

[325] Stekhoven, D. J. and Bühlmann, P. (2012). Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118.

[326] Stern, D., Herbrich, R., and Graepel, T. (2009). Matchbox: Large scale Bayesian recommendations. In *International World Wide Web Conference*.

[327] Stinchcombe, M. B. (1999). Neural network approximation of continuous functionals and continuous functions on compactifications. *Neural Networks*, 12(3):467–477.

[328] Sugiyama, M., Suzuki, T., and Kanamori, T. (2012). *Density ratio estimation in machine learning*. Cambridge University Press.

[329] Sun, S., Chen, C., and Carin, L. (2017). Learning structured weight uncertainty in Bayesian neural networks. In *Artificial Intelligence and Statistics*, pages 1283–1292. PMLR.

[330] Sun, S., Zhang, G., Shi, J., and Grosse, R. (2018). Functional variational Bayesian neural networks.

[331] Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112.

[332] Tang, G., Little, R. J., and Raghunathan, T. E. (2003). Analysis of multivariate missing data with nonignorable nonresponse. *Biometrika*, 90(4):747–764.

[333] Thahir, M., Sharma, T., and Ganapathiraju, M. K. (2012). An efficient heuristic method for active feature acquisition and its application to protein-protein interaction prediction. In *BMC proceedings*, volume 6, page S2. BioMed Central.

[334] Thompson, W. R. (1933). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294.

[335] Titsias, M. K. (2009). Variational learning of inducing variables in sparse Gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, pages 567–574.

[336] Titsias, M. K. and Ruiz, F. (2019). Unbiased implicit variational inference. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 167–176. PMLR.

[337] Tobar, F., Bui, T. D., and Turner, R. E. (2015). Learning stationary time series using Gaussian processes with nonparametric kernels. In *Advances in Neural Information Processing Systems*, pages 3501–3509.

[338] Tran, D., Blei, D., and Airoldi, E. M. (2015). Copula variational inference. In *Advances in Neural Information Processing Systems*, pages 3564–3572.

[339] Tran, D., Ranganath, R., and Blei, D. (2017a). Hierarchical implicit models and likelihood-free variational inference. *Advances in Neural Information Processing Systems*, 30.

[340] Tran, D., Ranganath, R., and Blei, D. M. (2017b). Deep and hierarchical implicit models. *arXiv:1702.08896*.

[341] Trippe, B. and Turner, R. (2018). Overpruning in variational Bayesian neural networks. *arXiv preprint arXiv:1801.06230*.

[342] Tu, R., Zhang, C., Ackermann, P., Mohan, K., Kjellström, H., and Zhang, K. (2019). Causal discovery in the presence of missing data. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1762–1770. PMLR.

[343] Turner, R. E. and Sahani, M. (2010). Statistical inference for single-and multi-band probabilistic amplitude demodulation. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 5466–5469. IEEE.

[344] Turner, R. E. and Sahani, M. (2011). Two problems with variational expectation maximisation for time-series models.

[345] Van der Vaart, A. W. (2000). *Asymptotic statistics*, volume 3. Cambridge university press.

[346] van der Wilk, M., Rasmussen, C. E., and Hensman, J. (2017). Convolutional Gaussian processes. In *Advances in Neural Information Processing Systems*, pages 2845–2854.

[347] Von Neumann, J. and Morgenstern, O. (1944). *Theory of games and economic behavior*. Princeton university press.

[348] Wainwright, M. J. and Jordan, M. I. (2008). *Graphical models, exponential families, and variational inference*. Now Publishers Inc.

[349] Wald, A. (1950). Statistical decision functions.

[350] Wang, C. and Blei, D. M. (2011). Collaborative topic modeling for recommending scientific articles. In *International Conference on Knowledge Discovery and Data Mining*, pages 448–456. ACM.

[351] Wang, D., Liu, H., and Liu, Q. (2018a). Variational inference with tail-adaptive f-divergence. *Advances in Neural Information Processing Systems*, 31.

[352] Wang, S., Shao, J., and Kim, J. K. (2014). An instrumental variable approach for identification and estimation with nonignorable nonresponse. *Statistica Sinica*, pages 1097–1116.

[353] Wang, X., Zhang, R., Sun, Y., and Qi, J. (2019a). Doubly robust joint learning for recommendation on data missing not at random. In *International Conference on Machine Learning*, pages 6638–6647. PMLR.

[354] Wang, Y. and Blei, D. M. (2019). The blessings of multiple causes.

[355] Wang, Y., Liang, D., Charlin, L., and Blei, D. M. (2018b). The deconfounded recommender: A causal inference approach to recommendation. *arXiv preprint arXiv:1808.06581*.

[356] Wang, Z., Lamb, A., Saveliev, E., Cameron, P., Zaykov, Y., Hernández-Lobato, J. M., Turner, R. E., Baraniuk, R. G., Barton, C., Jones, S. P., Woodhead, S., and Zhang, C. (2020). Diagnostic questions: The neurips 2020 education challenge. *arXiv preprint arXiv:2007.12061.*

[357] Wang, Z., Ren, T., Zhu, J., and Zhang, B. (2019b). Function space particle optimization for Bayesian neural networks. In *International Conference on Learning Representations.*

[358] Wathen, A. J. and Zhu, S. (2015). On spectral distribution of kernel matrices related to radial basis functions. *Numerical Algorithms*, 70(4):709–726.

[359] Welling, M. and Teh, Y. W. (2011). Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 681–688.

[360] Wendland, H. (2004). *Scattered data approximation*, volume 17. Cambridge university press.

[361] Wenzel, F., Roth, K., Veeling, B., Swiatkowski, J., Tran, L., Mandt, S., Snoek, J., Salimans, T., Jenatton, R., and Nowozin, S. (2020). How good is the Bayes posterior in deep neural networks really? In *International Conference on Machine Learning*, pages 10248–10259. PMLR.

[362] White, I. R., Royston, P., and Wood, A. M. (2011). Multiple imputation using chained equations: issues and guidance for practice. *Statistics in medicine*, 30(4):377–399.

[363] Wiegerinck, D. B. W. (1999). Tractable variational structures for approximating graphical models. In *Advances in Neural Information Processing Systems 11: Proceedings of the 1998 Conference*, volume 11, page 183. MIT Press.

[364] Wiegerinck, W. (2013). Variational approximations between mean field theory and the junction tree algorithm. *arXiv preprint arXiv:1301.3901.*

[365] Williams, C. K. (1997). Computing with infinite networks. In *Advances in neural information processing systems*, pages 295–301.

[366] Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256.

[367] Wilson, A. G., Hu, Z., Salakhutdinov, R. R., and Xing, E. P. (2016). Stochastic variational deep kernel learning. In *Advances in Neural Information Processing Systems*, pages 2586–2594.

[368] Wu, G., Domke, J., and Sanner, S. (2018). Conditional inference in pre-trained variational autoencoders via cross-coding. *arXiv preprint arXiv:1805.07785.*

[369] Wu, M. and Goodman, N. (2018). Multimodal generative models for scalable weakly-supervised learning. In *Advances in Neural Information Processing Systems*, pages 5575–5585.

[370] Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

[371] Yin, M. and Zhou, M. (2018). Semi-implicit variational inference. In *International Conference on Machine Learning*, pages 5660–5669. PMLR.

[372] Yoon, J., Jordon, J., and Schaar, M. (2018). GAIN: Missing data imputation using generative adversarial nets. In *International Conference on Machine Learning*, pages 5689–5698. PMLR.

[373] Yu, H.-F., Rao, N., and Dhillon, I. S. (2016). Temporal regularized matrix factorization for high-dimensional time series prediction. In *Advances in Neural Information Processing Systems*, pages 847–855.

[374] Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. (2017). Deep sets. In *Advances in Neural Information Processing Systems*, pages 3394–3404.

[375] Zakim, D., Braun, N., Fritz, P., and Alscher, M. D. (2008). Underutilization of information and knowledge in everyday medical practice: Evaluation of a computer-based solution. *BMC Medical Informatics and Decision Making*, 8(1):50.

[376] Zhang, C., Bütepage, J., Kjellström, H., and Mandt, S. (2018). Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026.

[377] Zhang, R., Li, Y., De Sa, C., Devlin, S., and Zhang, C. (2021). Meta-learning divergences for variational inference. In *International Conference on Artificial Intelligence and Statistics*, pages 4024–4032. PMLR.

[378] Zheng, Z. and Padmanabhan, B. (2002). On active learning for data acquisition. In *International Conference on Data Mining*, pages 562–569. IEEE.

[379] Zhou, Y., Shi, J., and Zhu, J. (2020). Nonparametric score estimators. In *International Conference on Machine Learning*, pages 11513–11522. PMLR.

[380] Zhu, H. and Rohwer, R. (1995). Information geometric measurements of generalisation.