DAGs with No Curl: An Efficient DAG Structure Learning Approach

Yue Yu [1] **Tian Gao** [2] **Naiyu Yin** [3] **Qiang Ji** [3]

Abstract

Recently directed acyclic graph (DAG) structure learning is formulated as a constrained continuous optimization problem with continuous acyclicity constraints and was solved iteratively through subproblem optimization. To further improve efficiency, we propose a novel learning framework to model and learn the weighted adjacency matrices in the DAG space directly. Specifically, we first show that the set of weighted adjacency matrices of DAGs are equivalent to the set of weighted gradients of graph potential functions, and one may perform structure learning by searching in this equivalent set of DAGs. To instantiate this idea, we propose a new algorithm, DAG-NoCurl, which solves the optimization problem efficiently with a two-step procedure: 1) first we find an initial cyclic solution to the optimization problem, and 2) then we employ the Hodge decomposition of graphs and learn an acyclic graph by projecting the cyclic graph to the gradient of a potential function. Experimental studies on benchmark datasets demonstrate that our method provides comparable accuracy but better efficiency than baseline DAG structure learning methods on both linear and generalized structural equation models, often by more than one order of magnitude.

1. Introduction

Bayesian Networks (BN) have been widely used in various machine learning applications (Ott et al., 2004; Spirtes et al., 1999). Efficient structure learning of BN remains an active area of research. The structure takes the form of a directed acyclic graph (DAG) and plays a vital part in other machine learning sub-areas such as causal inference (Pearl, 1988). However, DAG learning is proven to be NP-hard (Chickering et al., 2004) and scalability becomes a major issue.

Conventional DAG learning methods usually use independence tests (Spirtes et al., 2000b; Tsamardinos et al., 2006b) or perform score-and-search for discrete variables, with different scoring functions (Huang et al., 2018) or search procedures (Silander and Myllymaki, 2006; Chickering, 2002; Cussens, 2011). Learning DAG structures for continuous variables is often limited to Gaussian models (Yuan and Lin, 2007; Foygel and Drton, 2010; Schmidt et al., 2007; Mohan et al., 2012; Mohammadi et al., 2015). Recently, a fully continuous optimization formulation is proposed (Zheng et al., 2018), which transforms the discrete DAG constraint into a continuous equality constraint. This approach enables a suite of continuous optimization techniques such as gradient descent to be used, and has been extend to a more general parameter class with various neural methods (Yu et al., 2019; Lachapelle et al., 2019; Zhu and Chen, 2019; Ng et al., 2019).

In this work, we take a step further and investigate if we could directly optimize in the DAG space, hence resulting in a structure learning approach *without any explicit* DAG constraint. To this end, we propose a continuous optimization framework for DAG structure learning, which *implicitly* enforces the acyclicity of the learned graph. Varando (2020) and Ng et al. (2020) have studied the usage of empirical correlation and covariance matrices for similar purposes. Different from their works, we propose a new graph-exterior-calculus-based framework of DAG such that one can directly optimize in the DAG space. To solve the resultant unconstrained optimization problem, we further propose an efficient algorithm, DAG-NoCurl, developed based on the graph Hodge theory (Jiang et al., 2011). Hodge theory (Hodge, 1989), along with the related Helmholtz-Hodge Decomposition (Bhatia et al., 2012) on vector fields, describes the decomposition of a vector field into divergence-free and curl-free components. Hodge theory on graphs (Lim, 2015) shows that a DAG is a sum of three components: a curl-free, a divergence-free, and a harmonic component, where the curl-free component is an acyclic graph and hence motivates the naming of our algorithm.

DAG-NoCurl relies on the key step of mapping the adjacency weighted matrix of a directed graph onto its curl-free

[1]Department of Mathematics, Lehigh University, Bethlehem, PA. [2]IBM Research, Yorktown Heights, NY. [3]Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY.. Correspondence to: Yue Yu <yuy214@lehigh.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

component, i.e., an acyclic graph. The main advantages of the proposed method over conventional structure learning algorithms are: 1) the new model provides an equivalent representation of DAGs, which can be readily combined with many existing structural equation models (Zheng et al., 2018; Yu et al., 2019), 2) the new model naturally transfers the DAG structure learning problem as a continuous optimization framework, which avoids the combinatorial search and enables a suite of continuous optimization techniques; 3) comparing with other fully continuous optimization frameworks for DAG learning (Lachapelle et al., 2019; Yu et al., 2019; Zheng et al., 2020), our framework needs no explicit DAG constraints, many iterations, nor expensive post-processing, hence our learning approach achieves a substantially better computational efficiency.

Contributions. We make several major contributions in this work. 1) We propose a new model for DAGs based on the graph combinatorial gradient operator. Specifically, we theoretically show that the weighted adjacency matrix of a DAG can be represented as the Hadamard product of a skew-symmetric matrix and the gradient of a potential function on graph vertices, and vice versa. 2) Based on the new model, we develop a new continuous optimization framework for DAG structure learning without any explicit constraint. 3) To solve for the optimization problem efficiently, we propose a new DAG structure learning algorithm, DAG-NoCurl, that learns a weighted adjacency matrix to the original problem efficiently without constraints or iterations. 4) We demonstrate the effectiveness of the proposed method on synthetic and benchmark datasets. While the optimization problem remains nonconvex, the new formulation can substantially improve the efficiency, often by more than one order of magnitude, while preserving the accuracy.

2. Problem Statement and Motivation

Let V denote a set of d numbers of random variables, $X = (X_1, \cdots, X_d) \in \mathbb{R}^d$ be an observation on V, and $\mathbb{D}$ denotes the space of DAGs $\mathcal{G} = (V, E)$ on V, we aim to learn a DAG $\mathcal{G} \in \mathbb{D}$ given n i.i.d. observations of the random vector $X^i \in \mathbb{R}^d$, $i = 1, \cdots, n$. We assume no latent variables. To model X, we consider a (generalized) structural equation model (SEM) defined by a weighted adjacency matrix $A = [a_1|\cdots|a_d] \in \mathbb{R}^{d\times d}$ such that $\mathbb{E}(X_j|X_{pa(X_j)}) = f(a_j^T X)$, where $pa(X_j)$ denote the parents of random variable j in V. Therefore $[A]_{ij} \neq 0$ indicates a directed edge from vertex i to vertex j in a directed graph $\mathcal{G}_A$ and zero otherwise. With a slight abuse of notation, we will treat the weighted adjacency matrix A as a (weighted) directed graph $\mathcal{G}_A$.

Given the data matrix $\mathbf{X} = (X^1, \cdots, X^n) \in \mathbb{R}^{d\times n}$ and a loss function $F(A, \mathbf{X})$ that measures the goodness of fit of $\mathbf{X}$ for A, in the *structure learning* problem we aim to find the best DAG A^* that minimizes $F(A, \mathbf{X})$. Hence, the overall objective can be written as:

$$\begin{aligned} A^* = \underset{A}{\text{argmin}} \quad & F(A, \mathbf{X}) \\ \text{subject to} \quad & \mathcal{G}_A \in \mathbb{D} \quad \text{or} \quad h(A) = 0, \end{aligned} \tag{1}$$

where $\mathbb{D}$ is the DAG space and $h(A) = 0$ is an alternative continuous DAG constraint (Zheng et al., 2018; Wei et al., 2020). Formally, $h(A) = \text{tr}(\exp(A \circ A)) - d$ (Zheng et al., 2018) or $h(A) = \text{tr}[(I + A \circ A/d)^d] - d$ (Yu et al., 2019).

As one may know, the DAG space $\mathbb{D}$ for d variables is super-exponential. For structure learning over discrete variables, many exact and approximate algorithms from observational data have been proposed (Spirtes et al., 2000a; de Campos et al., 2009; Shimizu, 2014) with different search strategies, such as dynamic programming (Singh and Moore, 2005; Koivisto and Sood, 2004), A* (Yuan and Malone, 2013), or integer programming (Jaakkola et al., 2010; Cussens et al., 2016). In large-scale problems, approximate methods are often needed, with additional assumptions such as bounded tree-width (Nie et al., 2014). Sampling (Madigan et al., 1995; Grzegorczyk and Husmeier, 2008; Niinimaki et al., 2012; He et al., 2016) and topological order search (Friedman and Koller, 2000; Teyssier and Koller, 2012; Scanagatta et al., 2015) are also popular.

In this work we assume that all variables are continuous, and study the DAG structure learning problem with the focus of SEM and smooth loss function F defined over A. By using an alternative continuous DAG constraint $h(A) = 0$, the constrained optimization problem in Eq (1) becomes fully continuous. An augmented Lagrangian method is then employed in (Zheng et al., 2018; Yu et al., 2019) which solves unconstrained subproblems iteratively to impose the continuous DAG constraint explicitly. We investigate whether the explicit DAG constraint can be eliminated entirely, and therefore no iteration would be required. The key device in accomplishing this is the observation that DAG is associated with curl-free functions on edges E (please see Definition A.7 in the supplementary material for a formal definition of curl-free), which motivates a new representation of DAGs with $A = \gamma(W, p)$, $W \in \mathbb{R}^{d\times d}$ and $p \in \mathbb{R}^d$, as will be elaborated in the next section. The constrained optimization problem (1) can then be replaced by the following objective:

$$(W^*, p^*) = \underset{W,p}{\text{argmin}}\ F(\gamma(W, p), \mathbf{X}), \tag{2}$$

with the optimal DAG $A^* = \gamma(W^*, p^*)$. To the best of our knowledge, both the equivalence representation of DAGs and the application of Hodge theory in DAG structure learning have never been studied.

Main Results We propose a new equivalent model for DAGs, discussed in Theorem 2.1 and in Section 3, and show

that $\gamma(W, p) = W \circ \mathrm{ReLU}(\mathrm{grad}(p))$ can be used in Eq (2), where ReLU is the rectified linear unit and grad is the graph gradient operator.

Theorem 2.1. ***An Equivalent DAG Space.*** *Consider a set of d random variables, given any real vector $p \in \mathbb{R}^d$ and any skew-symmetric weight matrix $W \in \mathbb{R}^{d\times d}$ satisfying $W = -W^T$, the following (weighted) directed graph spaces are equivalent:*

$$\{\mathcal{G}_{W\circ \mathrm{ReLU}(\mathrm{grad}(p))}\} = \mathbb{D}.$$

We defer the proof of Theorem 2.1 to Section 3, given together by Theorem 3.5 and Theorem 3.7. To solve (2) with the new γ function, we further propose a new algorithm in Section 4, based on the combinatorial Hodge theory on graphs (Lim, 2015; Jiang et al., 2011). As we elaborate details below, we first briefly introduce a few useful basic graph exterior calculus operators (Bang-Jensen and Gutin, 2008; Jiang et al., 2011), then formulate DAGs into the equivalent model. Due to the page limit, we provide basic definitions for important concepts that are used in this paper in Appendix A. For a more thorough introduction of graph calculus, we refer the interested readers to (Lim, 2015).

Notation-wise, we use A^0 to denote the ground truth DAG, A^* for the global optimal solution in Eq (1), and $\tilde{A}$ for the approximated solution from numerical algorithms. The ultimate goal of our DAG structural learning problem is to obtain a DAG $\tilde{A}$ which recovers the structure of A^0 and obtains a comparable score $F(\tilde{A}, \mathbf{X})$ as $F(A^*, \mathbf{X})$.

3. An Equivalent Model for DAGs

Let $\widehat{\mathcal{G}} = (V, E)$ be a complete undirected graph where $V := \{1, \cdots, d\}$ is the set of vertices and E is the set of undirected edges. Note here since $\widehat{\mathcal{G}}$ is a complete graph, the set of k-th cliques of $\widehat{\mathcal{G}}$ is equivalent to all (unordered) subsets of V with size k. The ordered and unordered pairs of vertices are delimited by (i, j) and $\{i, j\}$, respectively, where i, j denote the i-th and j-th vertices. Real-valued functions on graphs can be defined on vertices, edges, triangles, and so on. On vertices, a real-valued function $f : V \to \mathbb{R}$ is called a potential function, and we denote the Hilbert space of all potential functions as $L^2(V)$. We may also define real-valued functions on edges $E = \{\{i, j\}, i, j \in V\}$ and triangles $T = \{\{i, j, k\}, i, j, k \in V\}$, with the requirement that these functions are alternating. In particular, an alternating function on edges $Y : V \times V \to \mathbb{R}$ requires $Y(i, j) = -Y(j, i)$; an alternating function on triangles $\Theta : V \times V \times V \to \mathbb{R}$, requires $\Theta(i, j, k) = -\Theta(j, i, k) = -\Theta(i, k, j)$. In the following we use $L^2_\wedge(E)$ and $L^2_\wedge(T)$ to denote the Hilbert spaces of real-valued alternating functions on edges and triangles, respectively. Moreover, we note that $p \in L^2(V)$ corresponds to a real vector $p = [p(1), \cdots, p(d)] \in \mathbb{R}^d$, and an alternating function $Y \in L^2_\wedge(E)$ corresponds to a skew-symmetric real matrix $Y \in \mathbb{R}^{d\times d}$ with $[Y]_{ij} = Y(i, j)$ and $Y = -Y^T$. Here we use the same letter to denote a vector/matrix and the corresponding function on vertices/edges.

Next, we introduce graph calculus operators grad, curl, and their adjoint operators below.

Definition 3.1. *(Lim, 2015)*

The gradient ($\mathrm{grad} : L^2(V) \to L^2_\wedge(E)$*) is an operator on any function p on vertices:*

$$(\mathrm{grad}\ p)(i, j) = p(j) - p(i), \quad \forall \{i, j\} \in E,$$

and $\mathrm{grad}(p)$ *is called a gradient flow. Its adjoint operator (*$\mathrm{grad}^* : L^2_\wedge(E) \to L^2(V)$*) is defined on any alternating function Y on edges:*

$$(\mathrm{grad}^*\ Y)(i) = -\sum_{j=1}^{d} Y(i, j), \quad \forall i \in V.$$

The divergence operator ($\mathrm{div} : L^2_\wedge(E) \to L^2(V)$*) is the negative operator of* grad^**, which is also defined on any alternating function Y on edges:*

$$(\mathrm{div}\ Y)(i) = -(\mathrm{grad}^*\ Y)(i) = \sum_{j=1}^{d} Y(i, j), \quad \forall i \in V.$$

The curl ($\mathrm{curl} : L^2_\wedge(E) \to L^2_\wedge(T)$*) is an operator for any alternating function Y on edges:*

$$(\mathrm{curl}\ Y)(i, j, k) = Y(i, j) + Y(j, k) + Y(k, i),$$
$$\forall \{i, j, k\} \in T,$$

and its adjoint operator $\mathrm{curl}^* : L^2_\wedge(T) \to L^2_\wedge(E)$ *is for any alternating function Θ on triangles:*

$$(\mathrm{curl}^*\ \Theta)(i, j) = \sum_{k=1}^{d} \Theta(i, j, k), \quad \forall \{i, j\} \in E.$$

The graph Laplacian ($\triangle_0 : L^2(V) \to L^2(V)$*) is an operator on any function p on vertices:*

$$(\triangle_0 p)(i) = -(\mathrm{div}\ \mathrm{grad}\ p)(i) = d\cdot p(i) - \sum_{j=1}^{d} p(j),\ \forall i \in V.$$

The graph Helmholtzian ($\triangle_1 : L^2_\wedge(E) \to L^2_\wedge(E)$*) is defined on any alternating function Y on edges:*

$$(\triangle_1 Y)(i, j) = (\mathrm{grad}\ \mathrm{grad}^*\ Y + \mathrm{curl}^*\ \mathrm{curl}\ Y)(i, j)$$
$$\forall \{i, j\} \in E.$$

Lemma 3.2. *(Jiang et al., 2011) Let $p \in L^2(V)$ and $\Theta \in L^2_\wedge(E)$, denote $D = \mathrm{grad}(p)$ and $R = \mathrm{curl}^*(\Theta)$, then D and R are curl-free and divergence-free, respectively:*

$$\mathrm{curl}(D)(i, j, k) = 0, \quad \forall \{i, j, k\} \in T;$$
$$\mathrm{div}(R)(i) = -\mathrm{grad}^*(R)(i) = 0, \quad \forall i \in V.$$

Given a complete undirected graph $\widehat{\mathcal{G}}(V,E)$ and a function $Y \in L^2_\wedge(E)$, with ReLU denoting the rectified linear unit function, we can define a weighted adjacency matrix $A = \mathrm{ReLU}(Y) \in \mathbb{R}^{d\times d}$ as:

$$\mathrm{ReLU}(Y)(i,j) := \begin{cases} Y(i,j), & \text{if } Y(i,j) > 0; \\ 0, & \text{else.} \end{cases}$$

and further define a weighted directed graph $\mathcal{G}_{\mathrm{ReLU}(Y)}$ from $\mathrm{ReLU}(Y)$ as the following:

Definition 3.3. *Consider a complete undirected graph $\widehat{\mathcal{G}}(V,E)$ and $Y \in L^2_\wedge(E)$, a directed graph $\mathcal{G}_{\mathrm{ReLU}(Y)}(V, E_{\mathrm{ReLU}(Y)})$ is defined such that there is a directed edge from vertex i to vertex j in $\mathcal{G}_{\mathrm{ReLU}(Y)}$ if and only if $Y(i,j) > 0$, i.e., the set of directed edges $E_{\mathrm{ReLU}(Y)} = \{(i,j)|Y(i,j) > 0\}$. Moreover, note that $\mathrm{ReLU}(Y)$ is a weighted adjacency matrix of $\mathcal{G}_{\mathrm{ReLU}(Y)}$.*

Based on the above definition, we show that curl-free functions are naturally associated with DAGs:

Lemma 3.4. *Consider a complete undirected graph $\widehat{\mathcal{G}}(V,E)$ and a **curl-free** function $Y \in L^2_\wedge(E)$, then $\mathrm{ReLU}(Y) \in \mathbb{R}^{d\times d}$ is the weighted adjacency matrix of a DAG. Moreover, given any skew-symmetric matrix $W \in \mathbb{R}^{d\times d}$, $W \circ \mathrm{ReLU}(Y)$ is also a DAG, where $\circ$ is the Hadamard product.*

All proofs can be found in the supplemental material. Since the gradient of any potential function (gradient flow) is curl-free, with Lemma 3.4 the first part of our main theoretical results is obtained:

Theorem 3.5. *Consider V as a set of d random variables, given any real vector $p \in \mathbb{R}^d$ and any skew-symmetric weight matrix $W \in \mathbb{R}^{d\times d}$ satisfying $W = -W^T$, $W \circ \mathrm{ReLU}(\mathrm{grad}(p))$ is the weighted adjacency matrix of a DAG, i.e., $\{\mathcal{G}_{W\circ\mathrm{ReLU}(\mathrm{grad}(p))}\} \subset \mathbb{D}$.*

Here we note that the proof of Theorem 3.5 is immediately obtain by taking $Y = \mathrm{grad}(p)$ in Lemma 3.4.

Remark 3.6. *The skew-symmetry requirement of W was made based on the fact that at least one or both of $\mathrm{ReLU}(\mathrm{grad}(p))(i,j) = 0$ and $\mathrm{ReLU}(\mathrm{grad}(p))(j,i) = 0$ must hold true for any $i,j \in V$. Here we note that W is set to be a skew-symmetric matrix instead of a full matrix so as to have a smaller degrees of freedom ($d(d-1)/2$) in the optimization problem (11). In fact, one can use a full matrix (with degrees of freedom d^2) or a symmetric matrix (with degrees of freedom $d(d+1)/2$) in place.*

We now show that the other direction also holds true:

Theorem 3.7. *Let $A \in \mathbb{R}^{d\times d}$ be the weighted adjacency matrix of a DAG with d nodes, denote V as the corresponding random variables of these d nodes, then there exists a skew-symmetric matrix $W \in \mathbb{R}^{d\times d}$, $W = -W^T$, and a real vector $p \in \mathbb{R}^d$ such that $A = W \circ \mathrm{ReLU}(\mathrm{grad}(p))$, i.e., $\mathbb{D} \subset \{\mathcal{G}_{W\circ\mathrm{ReLU}(\mathrm{grad}(p))}\}$. Here p is associated with the topological order of the DAG, such that $p(j) > p(i)$ if there is a directed path from vertex i to j.*

Algorithm 1 DAG-NoCurl algorithm

1: $A^{pre} \leftarrow$ solve (4), and threshold A^{pre}.
2: $\tilde{p} \leftarrow$ compute (9) with A^{pre}.
$\tilde{W} \leftarrow$ solve with fixed $\tilde{p}$ (11).
(Full Only) $\tilde{W}, \tilde{p} \leftarrow$ solve (3) with current $\tilde{W}, \tilde{p}$.
Return $\tilde{A} \leftarrow \tilde{W} \circ \mathrm{ReLU}(\mathrm{grad}(\tilde{p}))$, and threshold $\tilde{A}$.

Hence, we have shown that the set of weighted adjacency matrices for DAGs is equivalent to the set of $W \circ \mathrm{ReLU}(\mathrm{grad}(p))$, as stated in Theorem 2.1. Denoting $S := \{W|W \in \mathbb{R}^{d\times d}, W = -W^T\}$ as the space of all $d\times d$ skew-symmetric matrices, $\{W\circ\mathrm{ReLU}(\mathrm{grad}(p))|W \in S, p \in \mathbb{R}^d\}$ provides an admissible solution set for DAG structural learning problems.

While we believe our formulation for DAG is general and can have many applications where DAGs are used, in this paper we apply it to the continuous DAG learning framework. With a given loss function $F(A,\mathbf{X})$, the DAG structural learning problem can then be written as an optimization problem without an explicit constraint:

$$(W^*, p^*) = \underset{W\in S, p\in\mathbb{R}^d}{\mathrm{argmin}} \; F(W \circ \mathrm{ReLU}(\mathrm{grad}(p)), \mathbf{X}), \quad (3)$$

and the optimal solution $A^* = W^* \circ \mathrm{ReLU}(\mathrm{grad}(p^*))$.

The loss function in (3) is generally nonconvex. Therefore, we inherit the difficulties associated with nonconvex optimization. As will be discussed in the ablation study in Section 5.2, numerically solving (3) with a random initialization of W and p may result in a stationary point which is far from the global optimum. Therefore, instead of solving (3) directly, we propose a two-step algorithm, DAG-NoCurl, or NoCurl for short.

4. Algorithm: DAG with No Curl

The proposed DAG-NoCurl has two main steps. In Step 1, we compute an initial estimate of A^*, $A^{pre} \in \mathbb{R}^{d\times d}$, whose associated graph $\mathcal{G}_{A^{pre}}$ is not necessarily acyclic. In Step 2, we refine the predicted solution by projecting A^{pre} into the set of DAGs and obtaining the final solution $\tilde{A} = \tilde{W} \circ \mathrm{ReLU}(\mathrm{grad}(\tilde{p}))$. The full algorithm is outlined in Algorithm 1, and we introduce each step of the algorithm in more details as follows.

Step 1: In order to compute an initial solution A^{pre}, we solve for the following unconstrained optimization problem

$$A^{pre} = \underset{A}{\mathrm{argmin}} \; F(A,\mathbf{X}) + \lambda h(A) \quad (4)$$

where $\lambda > 0$ is a constant penalty coefficient, $F(A, \mathbf{X})$ is the loss function and $h(A)$ is a proper continuous constraint for acyclicity. For example, in NOTEARS (Zheng et al., 2018) the authors proposed $h(A) = \mathrm{tr}[\exp(A \circ A)] - d$ and in DAG-GNN (Yu et al., 2019) $h(A) = \mathrm{tr}[(I+\alpha A \circ A)^d] - d$, $\alpha > 0$, was employed. Note that when increasing the penalty parameter λ to infinity and solving (4) iteratively, an acyclic graph A will be obtained and the procedure will be equivalent to the classical penalty method for constrained optimization problems. However, here we solve for (4) with fixed λ, $h(A^{pre})$ is generally nonzero and A^{pre} is likely not a DAG. Therefore further computation is required to obtain a DAG. In practice, we find that solving for (4) at most twice, once with a fixed λ (denoted by the NoCurl-1 cases) or twice with two fixed λs (denoted by the NoCurl-2 cases), gives satisfactory initializations. In the NoCurl-2 cases, we first solve for (4) with a $\lambda = \lambda_1$, and then solve for (4) with a larger fixed penalty coefficient $\lambda = \lambda_2$. To achieve a good balance between the computational time and solution accuracy in structural discovery, in Section 5.2 we perform a hyperparameter study for both one λ and two λ cases. Note that the DAG constraint in the first step does not need to be strongly enforced; we use the constraint coefficient up to $\lambda = 10^3$, which is much smaller than up to $\lambda = 10^{16}$ used in NOTEARS.

Step 2: From Step 1, the learnt A^{pre} is likely to be cyclic. To obtain an acyclic graph solution $\tilde{A}$, we use the prediction A^{pre} as an initialization and search for its curl-free component by projecting A^{pre} into the admissible solution set $\{W \circ \mathrm{ReLU}(\mathrm{grad}(p))\}$. In particular, we design a projection procedure based on the following theorem:

Theorem 4.1 (Hodge Decomposition Theorem (Lim, 2015; Bhatia et al., 2012; Jiang et al., 2011))**.** *Consider an undirected graph $\widehat{\mathcal{G}}(V, E)$, any function $Y \in L^2_\wedge(E)$ can be decomposed into three orthogonal components:*

$$Y = \mathrm{grad}(\phi) + \mathrm{curl}^*(\Psi) + H \tag{5}$$

where $\phi \in L^2(V)$, $\Psi \in L^2_\wedge(T)$, and $H \in L^2_\wedge(E)$. Moreover, H is a harmonic function satisfying $\triangle_1 H = 0$, $\mathrm{curl}\, H = 0$ and $\mathrm{div}\, H = 0$. Here $\triangle_1$ is the graph Helmholtzian operator as defined in Definition 3.1.

The Hodge decomposition shows that every alternating function on edges $Y \in L^2_\wedge(E)$ can be decomposed into two orthogonal components: a gradient flow $\mathrm{grad}(\phi)$, which represents the L^2-optimal ordering of the vertices, and a divergence-free component, $\mathrm{curl}^*(\Psi) + H$, which measures the inconsistency of the vertices ordering. Hence we define a L^2 projection operator $\mathrm{Proj} : L^2_\wedge(E) \to \mathrm{grad}(L^2(V))$ as:

$$\mathrm{Proj}(Y) = \mathrm{grad}(\phi), \tag{6}$$

such that $\langle \mathrm{Proj}(Y), Y - \mathrm{Proj}(Y) \rangle = 0$, where $\langle \cdot, \cdot \rangle$ is the standard L^2 inner product on $L^2_\wedge(E)$: $\langle Y, Z \rangle := \sum_{i,j} Y(i,j)Z(i,j)$.

We note that ϕ is only unique up to a constant potential function, i.e., if $\mathrm{Proj}(Y) = \mathrm{grad}(\phi)$, then $\mathrm{Proj}(Y) = \mathrm{grad}(\phi + C)$ also holds. Adding a constant to ϕ will yield the same ordering of vertices and the same gradient flow $\mathrm{grad}(\phi)$. For the sake of well-posedness, we determine a unique solution ϕ by fixing its value on the last vertex such that $\phi(d) = 0$. Taking the divergence of (5) yields:

$$\mathrm{div}(Y) = \mathrm{div}\ \mathrm{grad}(\phi) = -\triangle_0 \phi, \tag{7}$$

where $\triangle_0$ is the graph Laplacian operator as defined in Definition 3.1.

We obtain the following theorem for the solution of ϕ:

Theorem 4.2. *Consider an undirected complete graph $\widehat{\mathcal{G}}(V, E)$ and $Y \in L^2_\wedge(E)$, a solution of (6) is given by*

$$\phi = -\triangle_0^\dagger \mathrm{div}(Y) = -\triangle_0^\dagger \sum_j Y(i,j) \tag{8}$$

where $\dagger$ indicates the Moore-Penrose pseudo-inverse. Fixing $\phi(d) = 0$, the matrix representing the graph Laplacian $\triangle_0$ is given by

$$[\triangle_0]_{ij} = \begin{cases} d-1, & \text{if } i = j \text{ and } i, j \neq d, \\ -1, & \text{if } i \neq j \text{ and } i, j \neq d, \\ 0, & \text{otherwise}. \end{cases}$$

Note that a key requirement in the above projection operator is that Y has to be an alternating function on edges, which corresponds to a $d \times d$ skew-symmetric matrix. However, A^{pre} is generally not skew-symmetric. Fortunately, any square matrix M can be written uniquely as a symmetric and a skew-symmetric components: $M = M_{sym} + M_{skew}$ with $M_{sym} = \frac{1}{2}(M + M^T)$ and $M_{skew} = \frac{1}{2}(M - M^T)$. Let $C(M)$ denote the connectivity matrix (Nievergelt and Hinrichs, 1993) of a directed graph M such that $[C(M)]_{ij} = 1$ only if a directed path exists from vertex i to vertex j. We apply the projection to the skew-symmetric component of the connectivity matrix of A^{pre}, and we will show this operation preserves the topological order of vertices in a DAG.

Theorem 4.3. *Let $A \in \mathbb{R}^{d \times d}$ be the weighted adjacency matrix of a DAG with d nodes, then*

$$p = -\triangle_0^\dagger \mathrm{div}\left(\frac{1}{2}(C(A) - C(A)^T)\right), \tag{9}$$

preserves the topological order in A such that $p(j) > p(i)$ if there is a directed path from vertex i to j. Moreover, we have $A = W \circ \mathrm{ReLU}(\mathrm{grad}(p))$ with the skew-symmetric matrix W defined as

$$[W]_{ij} = \begin{cases} 0, & \text{if } p(i) = p(j) \text{ or} \\ & \quad A(i,j) = A(j,i) = 0; \\ \frac{A(i,j)}{p(j)-p(i)}, & \text{if } A(i,j) \neq 0 \text{ and } A(j,i) = 0; \\ \frac{A(j,i)}{p(j)-p(i)}, & \text{if } A(i,j) = 0 \text{ and } A(j,i) \neq 0. \end{cases} \tag{10}$$

A detailed proof is provided in Appendix B. To help the readers better understand the formulations in Theorems 4.2 and 4.3, we also provide concrete examples of the projection procedure for several sample A^{pre}, including both acyclic and non-acyclic ones, in Appendix D.

Generally, from Theorem 4.3 we can see that $\tilde{p} = -\triangle_0^{\dagger} \operatorname{div}\left(\frac{1}{2}(C(A^{pre}) - C(A^{pre})^T)\right)$ provides the topological order for an acyclic approximation of A^{pre}. Given an adjacency matrix A of a DAG, p can be directly computed from A following formulation (9). Therefore, when the learnt A^{pre} from Step 1 is a DAG with the correct topological ordering (even though A^{pre} might be different from the ground truth A^0), Theorem 4.3 ensures the learning of an accurate p in Step 2. On the other hand, when A^{pre} is not a DAG but contains some correct ordering information, one can still apply formulation (9) to learn p which encodes an approximated topological ordering of the vertices. Hence, learning A^{pre} in Step 1 ensures the learning of p.

To further refine the solution of W, instead of employing (10) directly we optimize $\tilde{W}$ via:

$$\tilde{W} = \underset{W \in S}{\operatorname{argmin}}\ F(W \circ \operatorname{ReLU}(\operatorname{grad}(\tilde{p})), \mathbf{X}). \tag{11}$$

To enforce the skew-symmetric property on W we only optimize elements in the upper triangular matrix of W, while setting diagonal elements zero and each lower triangular element as negative of the corresponding upper triangular element, i.e., $W_{ij} = -W_{ji}$.

In the full version of DAG-NoCurl, given the separately optimized $\tilde{W}$ and $\tilde{p}$, we then jointly optimize both together using Equation 3. As shown in Algorithm 1, our two-step approach does not require any iterative procedure to increase the penalty coefficient λ. The full version of DAG-NoCurl consists of only 3 (if using only one λ) or 4 (if using two λs) unconstrained optimization problems.

Return a DAG Solution: we get a DAG solution by $\tilde{A} = \tilde{W} \circ \operatorname{ReLU}(\operatorname{grad}(\tilde{p}))$. By the standard practice of thresholding in DAG learning problems (see, e.g., Zheng et al. (2018)), we employ the same procedure to post-process both A^{pre} after Step 1 and $\tilde{A}$ after Step 2. The thresholding step in NOTEARS is motivated by rounding numerical solutions to obtain an exact DAG and remove false discoveries, while our threshold step aims to remove false discoveries only since our solution is a DAG already. Specifically, we threshold the edge weights of a learned A as follows: given a fixed threshold $\epsilon > 0$, set $A(i,j) = 0$ if $|A(i,j)| < \epsilon$. In all the numerical tests we use a fixed threshold value $\epsilon = 0.3$ as suggested by NOTEARS (Zheng et al., 2018).

Consistency. While our DAG-NoCurl algorithm can use any loss function as long as DAGs can be represented by an weighted adjacency matrix, we use the same loss as NOTEARS. With the same loss and the equivalence of the DAG space per Theorem 2.1, the global minimizer of the full version of DAG-NoCurl is the same as that of NOTEARS in Eq 1. In practice, the solution is only guaranteed to be a stationary point due to the nonconvexity associated with the DAG space, which is shared by all other algorithms within the framework (Zheng et al., 2018).

Further Efficiency Improvement. We observe that jointly optimizing W and p in Eq (3) in Step 2 of Algorithm 1 is often not needed in practice, as it may not improve upon the solutions from Eq (11). Intuitively, since p indicates the order of all nodes, it means that the topological order $\tilde{p}$ from Step 1 and $\tilde{W}$ from Eq (11) given $\tilde{p}$ are similar solution as the ones from Eq (3). We show the numerical evidence in the ablation study and other experiments of Section 5.2. Hence we use this more efficient version without Eq (3) as the DAG-NoCurl algorithm for experiments.

5. Empirical Evaluations

We present empirical evaluations to demonstrate the effectiveness of the proposed DAG continuous representation and learning algorithm. Specifically, we conduct experiments on both linear and nonlinear benchmark synthetic and real-world datasets, and compare the proposed method DAG-NoCurl against competitive baselines. Here we outline the empirical set-up, with more details including all parameter choices provided in the supplementary material. The code will be publicly released at `https://github.com/fishmoon1234/DAG-NoCurl`.

Datasets. We first test our algorithm in linear synthetic datasets. We employ similar experimental setups as existing works (Zheng et al., 2018). In each experiment, a random graph $\mathcal{G}$ is generated by using the Erdős–Rényi (ER) model or the scale-free (SF) model with k expected edges (denoted as ERk or SFk cases) and uniformly random edge weights to obtain a ground-truth weighted matrix A^0. Given A^0, we take $n = 1000$ i.i.d. samples of X from the linear SEM $X = (A^0)^T X + Z$, where the noise $Z \in \mathbb{R}^d$ is generated from two different noise models: Gaussian and Gumbel. To apply the NoCurl algorithm in the linear SEM, we use the least-square loss $F(A, \mathbf{X}) = \frac{1}{2n}||\mathbf{X} - A^T\mathbf{X}||_F^2$ and numerically solve unconstrained optimization problems with L-BFGS (Liu and Nocedal, 1989), although we note that other standard smooth optimization schemes can also be employed. We have also tested loss functions with a smooth L_1 regularization in implementation. However results show little improvements. We suspect that using the DAG regularization and a standard threshold procedure may have similar sparsity effect (Zheng et al., 2018) – a similar observation can be found in regression problems (Jain et al., 2014).

We also test nonlinear SEM and real datasets in Section 5.5 and 5.6. The graphs in the synthetic datasets can be iden-

Table 1. Ablation Study: results (mean ± standard error over 100 trials) for $d = 30$ ER6-Gaussian Cases from DAG-NoCurl, where bold numbers highlight the best method for each case. Lower is better in both time and SHD. Note the usage of color in the table.

Method	Time (Sec)	SHD
rand init	6.04 ± 0.26	84.88 ± 2.65
rand p	7.74 ± 0.38	130.37 ± 1.66
NoCurl-1s	**1.58 ± 0.10**	37.36 ± 1.34
NoCurl-2s	4.69 ± 0.23	29.88 ± 1.55
NoCurl-1-	1.69 ± 0.11	32.82 ± 1.07
NoCurl-2-	4.67 ± 0.23	26.08 ± 1.07
NoCurl-1+	5.32 ± 0.35	21.44 ± 1.56
NoCurl-2+	10.38 ± 0.23	17.81 ± 1.29
NoCurl-1	3.34 ± 0.21	21.44 ± 1.56
NoCurl-2	7.68 ± 0.39	**17.37 ± 1.18**

tified exactly in these SEM settings with additive noises (Peters et al., 2014) and hence there is no Markov equivalence.

Metrics and Baselines. We use Structure Hamming Distance (SHD) (Zheng et al., 2018) to show the structure learning accuracy. To assess the ability of methods in solving the original optimization problem in Eq (1), we also report its score difference from the ground truth: $\Delta F = F(\tilde{A}, \mathbf{X}) - F(A^0, \mathbf{X})$. For each graph-noise type combination, 100 trials are performed. The exact numerical values for mean accuracy, mean CPU time (in seconds), and mean score difference along with their perspective standard errors of the mean are reported in full in the supplementary materials. We compare our method with fast greedy equivalent search (FGS) (Ramsey et al., 2017), Causal Additive Models (Bühlmann et al., 2014), MMPC (Tsamardinos et al., 2006a), and NOTEARS (Zheng et al., 2018). In the supplemental materials, we also compare our method with Equal Variance DAG variants (Chen et al., 2019) , which is designed to handle data with equal variances. For nonlinear datasets, we also compare with several neural methods and generalized score GES (GSGES) (Huang et al., 2018).

5.1. Hyperparameter Study

We first perform a quantitative study on the hyperparameter λ choices. We use the ER3-Gaussian and ER6-Gaussian cases to select hyperparameters. We test many sets of hyperparameters and due to the page limit the full numerical results are listed in Table 4 for ER3-Gaussian and Table 5 for ER6-Gaussian cases in the supplement. It is observed that as long as $\lambda \geq 10$, the accuracy results are all satisfactory. Among which, $\lambda = 10^2$ and $\lambda = (10, 10^3)$ are generally the best values in term of both accuracy and computational efficiency. Hence, we select them as the default values and refer them as **NoCurl-1** and **NoCurl-2**, respectively. For more discussion, please refer to the supplement.

5.2. Ablation Study

We conduct an ablation study on our proposed algorithm, listed in Algorithm 1, testing how each step would affect the final result, with five settings: 1) solving the optimization problem Eq (3) directly with random initialization of (W, p) (denoted as "**rand init**"); 2) NoCurl without Step 1, by solving for W with a random p then performing one additional step to jointly optimize (W, p) with Eq (3) (denoted as "**rand** p"); 3) NoCurl without Step 2, by repeatedly increasing the threshold of the structure until a DAG is obtained (denoted by **an additional "s"** in the end). We use the thresholds starting from 0.3 (anything below produces much worse results) and with increments of 0.05 until $h(A) < 10^{-8}$; 4) NoCurl with Eq (10) instead of Eq (11) to find $\tilde{W}$ (denoted as **an additional "-"** in the end); 5) the full version of NoCurl with the step to jointly optimize (W, p) by solving Eq (3) after Step 2 (denoted as **an additional "+"** in the end).

We list the results of $d = 30$ in ER6-Gaussian in Table 1, and full numerical results are shown in Table 4 to 9 in the supplement. As one can see from Table 1, NoCurl with random initializations ("rand init" and "rand p") performs subpar, indicating the importance of Step 1 in Algorithm 1. Results from NoCurl without Step 2 (NoCurl-1s and NoCurl-2s) are poorer than the full algorithm (NoCurl-1 and NoCurl-2), indicating that optimizing Eq (11) after an initialization is important to refine the solution for better accuracy. Moreover, results from NoCurl-1- and NoCurl-2- also show the important effects of Step 2 of our algorithm: using Eq (11) to learn $\tilde{W}$ instead of Eq (10) further improves the accuracy. As can be seen from *#Missing Edge* in Table 6 to 9 in the supplement, Step 2 of our algorithm generally reduces the number of missing edges in comparison to NoCurl-·s and NoCurl-·- alternatives. Lastly, adding extra optimization steps (NoCurl-1+ and NoCurl-2+), which are the full version of DAG-NoCurl, does not result much improvements on accuracy or ΔF. This result indicates that the efficient version of NoCurl reaches similar solutions in practice. All discussions above are general and consistent for all dataset tested, as shown in Table 4 to 9 in the supplement.

5.3. Optimization Objective Results

We now study the performance of NoCurl in approximately solving the optimization problem given by (1), by comparing the scores from NoCurl solution $\tilde{A}$ with those from NOTEARS and the exact global minimizer from the GOBNILP algorithm (Cussens et al., 2016). Since GOBNILP involves enumerating all possible parent sets, its experiments are limited to small DAGs with $d = 10$ cases. In Table 2, we show the relative score ΔF with respect to the ground truth graph A^0. Surprisingly, although NoCurl provides an approximate solution, we can see that the score from NoCurl-2

Table 2. Comparison on score differences (lower is better) from the ground truth, $\triangle F = F(\tilde{A}, \mathbf{X}) - F(A^0, \mathbf{X})$.

Method	d=10 ER3,Gauss	d=50 ER3,Gauss	d=100 ER3,Gauss	d=10 ER6,Gauss	d=50 ER6,Gauss	d=100 ER6,Gauss
GOBNILP	-0.03 ± 0.00	-	-	-0.03 ± 0.00	-	-
NOTEARS	0.03 ± 0.01	-0.25 ± 0.04	-1.65 ± 0.08	0.22 ± 0.40	1.97 ± 0.26	2.49 ± 0.37
NoCurl-1	0.09 ± 0.02	0.05 ± 0.05	-0.82 ± 0.16	0.54 ± 0.22	2.31 ± 0.41	4.30 ± 0.99
NoCurl-2	0.06 ± 0.02	-0.10 ± 0.05	-1.44 ± 0.09	0.36 ± 0.07	1.77 ± 0.38	2.61 ± 0.93

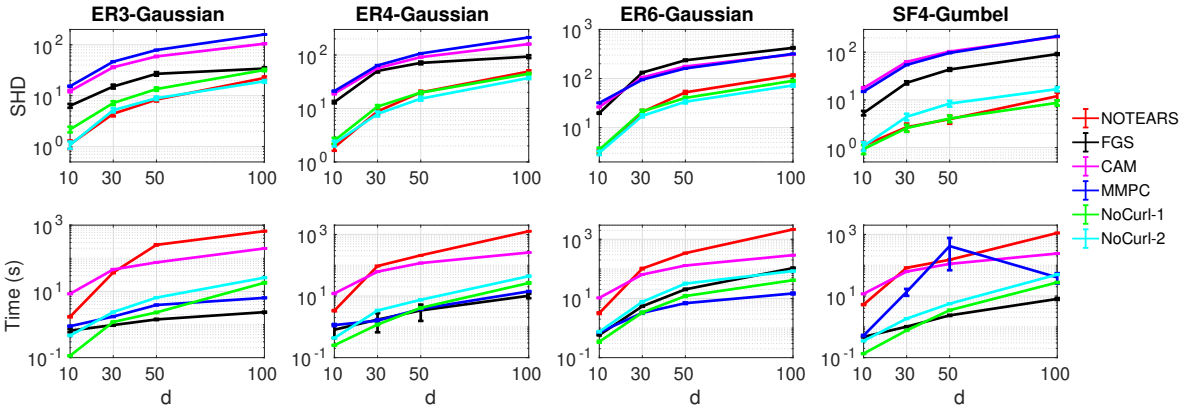

Figure 1. Structural discovery results in terms of SHD (lower is better) and computational time in seconds on linear SEM datasets (log-scale). Error bars represent standard errors over 100 simulations.

case is very close to the objective values of NOTEARS and GOBNILP, and even outperforms NOTEARS in the ER6 $d = 50$ case. When d increases and the optimization problem becomes more difficult, NoCurl remains competitive and does not deteriorate.

5.4. Structure Recovery Results

We now present the comparison with other baselines methods by comparing structure recovery accuracy and computational efficiency of NoCurl with NOTEARS, FGS, CAM, and MMPC. In Figure 1, the top row shows the SHD of different methods while the buttom row shows the CPU time, in seconds, of different methods, both in log scales. For the detailed results in all graph types, please refer to Table 6 to 9 in the supplement.

Consistent with existing observations (Zheng et al., 2018; Bühlmann et al., 2014), FGS, MMPC, and CAM's performances suffer when the number of edges gets larger. While NOTEARS is significantly more accurate than other baselines, NoCurl achieves a similar accuracy as NOTEARS, and can sometimes beat NOTEARS, especially on dense and large graphs. For instance, when $d = 100$, in all ERk-Gaussian cases the NoCurl-2 achieves the lowest SHD, while in SF4-Gumbel cases the NoCurl-1 achieves the lowest SHD among all methods. When comparing the efficiency, NoCurl requires a similar runtime as FGS and MMPC, which is faster than NOTEARS by more than one or two orders of magnitude. Overall, NoCurl substantially improves the efficiency comparing with NOTEARS, while still sustaining the comparable structure discovery accuracy.

In addition, we also test our method with Equal Variance DAG learning algorithm and its variants (EqV-TD and EqV-BU) (Chen et al., 2019), with their results shown in Table 6 to 9 in the supplement. Our method outperforms both EqV variants by a significant margin.

5.5. Nonlinear Synthetic Datasets

We further test the capability of NoCurl with more general models and datasets by sampling X from nonlinear SEM $X_j = f(A, X_{pa(j)}) + Z_j$ for $j = 1, \cdots, d$ with nonlinear functions f, following the same setting as Yu et al. (2019). For the nonlinear SEM, we combine NoCurl with DAG-GNN, where nonlinear SEM is learnt using neural networks and the standard machinery of augmented Lagrangian was applied to enforce the continuous constraint. We generated 3 Datasets (denoted as Nonlinear Case 1 to 3), and for the exact modeling and implementation details, please refer to the supplement.

For nonlinear SEM datasets, we compare NoCurl with DAG-GNN with CAM, MMPC, GSGES, and recent neural-based methods DAG-GNN (Yu et al., 2019), GraN-DAG (Lachapelle et al., 2019) and NOTEARS-MLP (Zheng et al.,

2020). The results are shown in Table 10 in the supplement. Generally neural-based models outperform heuristic-based methods. Comparing with DAG-GNN, NoCurl has similar accuracy performance across different variable sizes d. NoCurl (along with DAG-GNN) is better than NOTEAR-MLP in Nonlinear Case 1 but does not perform as well as GraN-DAG. NoCurl achieves the best overall performance in Nonlinear Case 2. In Nonlinear Case 3, NoCurl is worse than NOTEARS-MLP but much better than GraN-DAG. Regarding the computational time, NoCurl achieves about $3 \sim 4$ times computational efficiency gain over its base model DAG-GNN on average. NoCurl is also more than one order of magnitude faster than NOTEARS-MLP and GraN-DAG. Note that NoCurl's performance is limited by the base model, and we choose DAG-GNN as the base model to combine with NoCurl because other neural methods (such as NOTEARS-MLP and GraN-DAG) use a gradient-based adjacency matrix representation, which is different from the weighted A formulation in NoCurl. It would be an interesting future work to extend NoCurl to these frameworks.

5.6. Real-World Dataset

We now apply NoCurl+DAG-GNN to a real-world bioinformatics dataset (Sachs et al., 2005) for the discovery of a protein signaling network based on expression levels of proteins and phospholipids. This is a widely used dataset for research on graphical models, with experimental annotations accepted by the biological research community. Based on $n = 7466$ samples of $d = 11$ cell types, 20 edges were estimated in the ground truth graph (Sachs et al., 2005). In Table 11 of the supplement, we compare our results and baselines against the ground truth offered (Sachs et al., 2005). The proposed NoCurl+DAG-GNN obtains an SHD of 16 with 18 estimated edges. The learnt graph is also plotted in supplementary materials. Comparing with the other methods reporting a similar number of nonzero edges, NoCurl has a better performance in terms of SHD.

6. Conclusion

We proposed a novel theoretically-justified continuous representation of DAG structures based on graph exterior calculus operators, and proved that this new formulation can represent the adjacency matrices of DAGs without explicit acyclicity constraints, which is often the most intricate part of the optimization. We proposed a new algorithm, which we coin DAG-NoCurl, to approximately solve for the unconstrained optimization problem efficiently. The key step in this approach is based on the Hodge decomposition theorem, which projects a cyclic graph to the gradient of a potential function and obtains a DAG approximation of the graph. Empirically the proposed DAG-NoCurl achieves comparable accuracy but with substantially better computational efficiency than their counter parts with constraints in all the datasets tested. We believe it is a promising new framework for DAG structure learning where both continuous and discrete optimization approaches can be applied.

Assumptions made in the paper (e.g., SEMs and smooth loss functions) are widely used and studied in the literature, including the NOTEARS and many related methods. We assume smoothness so we could use gradient-based optimization methods (such as LBFGS) in our proposed algorithm (for ease of comparison with baselines). In fact, the smoothness assumption of loss functions can be further relaxed. For instance, the BFGS method is proved to converge for continuously differentiable loss functions (Li and Fukushima, 2001). Moreover, we note that the proposed framework and algorithm is general, which enables the employment of other optimization methods. For instance, our framework could handle L_0 penalties if one were to use optimization methods such as dynamic programming or equivalent search as suggested by Van de Geer and Bühlmann (2013). Investigation of the new DAG space in these DAG learning frameworks would be an interesting future work.

Acknowledgements

We thank Dr. Jie Chen and Dr. Changhe Yuan for helpful discussions. We thank to anonymous reviewers who have provide helpful comments. Y. Yu's research is supported in part by the National Science Foundation under award DMS 1753031. Part of this work is also supported by DARPA grant FA8750-17-2-0132.

References

Jørgen Bang-Jensen and Gregory Z Gutin. *Digraphs: theory, algorithms and applications*. Springer Science & Business Media, 2008.

Harsh Bhatia, Gregory Norgard, Valerio Pascucci, and Peer-Timo Bremer. The helmholtz-hodge decomposition—a survey. *IEEE Transactions on visualization and computer graphics*, 19(8):1386–1404, 2012.

Peter Bühlmann, Jonas Peters, Jan Ernest, et al. Cam: Causal additive models, high-dimensional order search and penalized regression. *The Annals of Statistics*, 42(6): 2526–2556, 2014.

Wenyu Chen, Mathias Drton, and Y Samuel Wang. On causal discovery with an equal-variance assumption. *Biometrika*, 106(4):973–980, 2019.

David Maxwell Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 2002.

David Maxwell Chickering, David Heckerman, and Christopher Meek. Large-sample learning of Bayesian networks is NP-hard. *Journal of Machine Learning Research*, 5: 1287–1330, 2004.

James Cussens. Bayesian network learning with cutting planes. In *UAI*, 2011.

James Cussens, David Haws, and Milan Studenỳ. Polyhedral aspects of score equivalence in Bayesian network structure learning. *Mathematical Programming*, pages 1–40, 2016.

Cassio P. de Campos, Zhi Zeng, and Qiang Ji. Structure learning of Bayesian networks using constraints. In *ICML*, pages 113–120, New York, NY, USA, 2009. ACM.

Rina Foygel and Mathias Drton. Extended bayesian information criteria for gaussian graphical models. In *Advances in neural information processing systems*, pages 604–612, 2010.

Nir Friedman and Daphne Koller. Being bayesian about network structure. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 201–210. Morgan Kaufmann Publishers Inc., 2000.

Marco Grzegorczyk and Dirk Husmeier. Improving the structure MCMC sampler for Bayesian networks by introducing a new edge reversal move. *Machine Learning*, 71(2-3):265, 2008.

Ru He, Jin Tian, and Huaiqing Wu. Structure learning in Bayesian networks of a moderate size by efficient sampling. *Journal of Machine Learning Research*, 17(1): 3483–3536, 2016.

William Vallance Douglas Hodge. *The theory and applications of harmonic integrals*. CUP Archive, 1989.

Biwei Huang, Kun Zhang, Yizhu Lin, Bernhard Schölkopf, and Clark Glymour. Generalized score functions for causal discovery. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1551–1560, 2018.

Tommi Jaakkola, David Sontag, Amir Globerson, and Marina Meila. Learning Bayesian network structure using LP relaxations. 2010.

Prateek Jain, Ambuj Tewari, and Purushottam Kar. On iterative hard thresholding methods for high-dimensional m-estimation. In *Proceedings of the 27th Neural Information Processing Systems - Volume 1*, page 685–693, 2014.

Xiaoye Jiang, Lek-Heng Lim, Yuan Yao, and Yinyu Ye. Statistical ranking and combinatorial hodge theory. *Mathematical Programming*, 127(1):203–244, 2011.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Mikko Koivisto and Kismat Sood. Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research*, 5:549–573, 2004.

Sébastien Lachapelle, Philippe Brouillard, Tristan Deleu, and Simon Lacoste-Julien. Gradient-based neural dag learning. *arXiv preprint arXiv:1906.02226*, 2019.

Dong-Hui Li and Masao Fukushima. On the global convergence of the BFGS method for nonconvex unconstrained optimization problems. *SIAM Journal on Optimization*, 11(4):1054–1064, 2001.

Lek-Heng Lim. Hodge laplacians on graphs. *arXiv preprint arXiv:1507.05379*, 2015.

Dong C Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

David Madigan, Jeremy York, and Denis Allard. Bayesian graphical models for discrete data. *International Statistical Review/Revue Internationale de Statistique*, pages 215–232, 1995.

Abdolreza Mohammadi, Ernst C Wit, et al. Bayesian structure learning in sparse Gaussian graphical models. *Bayesian Analysis*, 10(1):109–138, 2015.

Karthik Mohan, Mike Chung, Seungyeop Han, Daniela Witten, Su-In Lee, and Maryam Fazel. Structured learning of Gaussian graphical models. In *NIPS*, 2012.

Ignavier Ng, Zhuangyan Fang, Shengyu Zhu, Zhitang Chen, and Jun Wang. Masked gradient-based causal structure learning. *arXiv preprint arXiv:1910.08527*, 2019.

Ignavier Ng, AmirEmad Ghassami, and Kun Zhang. On the Role of Sparsity and DAG Constraints for Learning Linear DAGs. In *Advances in Neural Information Processing Systems*, 2020.

Siqi Nie, Denis D Mauá, Cassio P De Campos, and Qiang Ji. Advances in learning Bayesian networks of bounded treewidth. In *NIPS*, 2014.

Jurg Nievergelt and Klaus H. Hinrichs. *Algorithms and Data Structures: With Applications to Graphics and Geometry*. Prentice-Hall, Inc., USA, 1993. ISBN 0134894286.

Teppo Niinimaki, Pekka Parviainen, and Mikko Koivisto. Partial order MCMC for structure discovery in Bayesian networks. *arXiv preprint arXiv:1202.3753*, 2012.

Sascha Ott, Seiya Imoto, and Satoru Miyano. Finding optimal models for small gene networks. In *Pacific symposium on biocomputing*, 2004.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. 2017.

Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers, Inc., 2 edition, 1988.

Jonas Peters, Joris M Mooij, Dominik Janzing, and Bernhard Schölkopf. Causal discovery with continuous additive noise models. *The Journal of Machine Learning Research*, 15(1):2009–2053, 2014.

Joseph Ramsey, Madelyn Glymour, Ruben Sanchez-Romero, and Clark Glymour. A million variables and more: the fast greedy equivalence search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images. *International Journal of Data Science and Analytics*, 3 (2):121–129, 2017.

Karen Sachs, Omar Perez, Dana Peer, Douglas A Lauffenburger, and Garry P Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721):523–529, 2005.

Mauro Scanagatta, Cassio P de Campos, Giorgio Corani, and Marco Zaffalon. Learning bayesian networks with thousands of variables. In *Advances in neural information processing systems*, pages 1864–1872, 2015.

Mark Schmidt, Alexandru Niculescu-Mizil, Kevin Murphy, et al. Learning graphical model structure using l1-regularization paths. In *AAAI*, volume 7, pages 1278–1283, 2007.

Shohei Shimizu. LiNGAM: Non-Gaussian methods for estimating causal structures. *Behaviormetrika*, 41(1):65–98, 2014.

Tomi Silander and Petri Myllymaki. A simple approach for finding the globally optimal Bayesian network structure. In *UAI*, 2006.

Ajit P. Singh and Andrew W. Moore. Finding optimal Bayesian networks by dynamic programming. Technical report, Carnegie Mellon University, 2005.

Peter Spirtes, Clark N Glymour, and Richard Scheines. *Computation, Causation, and Discovery*. AAAI Press, 1999.

Peter Spirtes, Clark Glymour, Richard Scheines, Stuart Kauffman, Valerio Aimale, and Frank Wimberly. Constructing Bayesian network models of gene expression networks from microarray data, 2000a.

Peter Spirtes, Clark N Glymour, Richard Scheines, David Heckerman, Christopher Meek, Gregory Cooper, and Thomas Richardson. *Causation, prediction, and search*. MIT press, 2000b.

Marc Teyssier and Daphne Koller. Ordering-based search: A simple and effective algorithm for learning bayesian networks. *arXiv preprint arXiv:1207.1429*, 2012.

Ioannis Tsamardinos, Laura E Brown, and Constantin F Aliferis. The max-min hill-climbing bayesian network structure learning algorithm. *Machine learning*, 65(1): 31–78, 2006a.

Ioannis Tsamardinos, LauraE. Brown, and ConstantinF. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78, 2006b.

Sara Van de Geer and Peter Bühlmann. L0-penalized maximum likelihood for sparse directed acyclic graphs. *Annals of Statistics*, 41(2):536–567, 2013.

Gherardo Varando. Learning dags without imposing acyclicity. *arXiv preprint arXiv:2006.03005*, 2020.

Dennis Wei, Tian Gao, and Yue Yu. Dags with no fears: A closer look at continuous optimization for learning bayesian networks. *arXiv preprint arXiv:2010.09133*, 2020.

Yue Yu, Jie Chen, Tian Gao, and Mo Yu. Dag-gnn: Dag structure learning with graph neural networks. *arXiv preprint arXiv:1904.10098*, 2019.

Changhe Yuan and Brandon Malone. Learning optimal Bayesian networks: A shortest path perspective. 48:23–65, 2013.

Ming Yuan and Yi Lin. Model selection and estimation in the gaussian graphical model. *Biometrika*, 94(1):19–35, 2007.

Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. Dags with no tears: Continuous optimization for structure learning. In *Advances in Neural Information Processing Systems*, pages 9472–9483, 2018.

Xun Zheng, Chen Dan, Bryon Aragam, Pradeep Ravikumar, and Eric P. Xing. Learning sparse nonparametric DAGs. In *International Conference on Artificial Intelligence and Statistics*, 2020.

Shengyu Zhu and Zhitang Chen. Causal discovery with reinforcement learning. *arXiv preprint arXiv:1906.04477*, 2019.