On the Convergence of Continuous Constrained Optimization for Structure Learning

Ignavier Ng[1], **Sébastien Lachapelle**[2], **Nan Rosemary Ke**[3], **Simon Lacoste-Julien**[2,4], **Kun Zhang**[1,5]
[1] Carnegie Mellon University
[2] Mila, Université de Montréal
[3] DeepMind
[4] Canada CIFAR AI Chair
[5] Mohamed bin Zayed University of Artificial Intelligence

Abstract

Recently, structure learning of directed acyclic graphs (DAGs) has been formulated as a continuous optimization problem by leveraging an algebraic characterization of acyclicity. The constrained problem is solved using the augmented Lagrangian method (ALM) which is often preferred to the quadratic penalty method (QPM) by virtue of its standard convergence result that does not require the penalty coefficient to go to infinity, hence avoiding ill-conditioning. However, the convergence properties of these methods for structure learning, including whether they are guaranteed to return a DAG solution, remain unclear, which might limit their practical applications. In this work, we examine the convergence of ALM and QPM for structure learning in the linear, nonlinear, and confounded cases. We show that the standard convergence result of ALM does not hold in these settings, and demonstrate empirically that its behavior is akin to that of the QPM which is prone to ill-conditioning. We further establish the convergence guarantee of QPM to a DAG solution, under mild conditions. Lastly, we connect our theoretical results with existing approaches to help resolve the convergence issue, and verify our findings in light of an empirical comparison of them.

Proceedings of the 25[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

1 Introduction

Structure learning of directed acyclic graphs (DAGs) is a fundamental problem in many scientific endeavors, such as biology (Sachs et al., 2005) and economics (Koller and Friedman, 2009). Traditionally, score-based structure learning methods cast the problem into a discrete optimization program using a predefined score function. Most of these methods, such as GES (Chickering, 2002), involve local heuristics owing to the large search space of graphs (Chickering, 1996).

A recent work by Zheng et al. (2018) has reformulated score-based learning of linear DAGs as a continuous constrained optimization problem. At the heart of the method is an algebraic characterization of acyclicity expressed as a nonlinear constraint and used to minimize the least squares loss while enforcing acyclicity. In the context of structure learning, various works have adopted this continuous constrained formulation to support linear non-Gaussian models (Zheng, 2020), nonlinear models (Yu et al., 2019; Ng et al., 2019; Lachapelle et al., 2020; Zheng et al., 2020; Gao et al., 2021b; Ng et al., 2022b; Geffner et al., 2022), time series (Pamfil et al., 2020; Sun et al., 2021; Hsieh et al., 2021), unobserved confounding (Bhattacharya et al., 2021; Bellot and van der Schaar, 2021), interventional data (Brouillard et al., 2020; Faria et al., 2022), multi-domain data (Zeng et al., 2021), mixed data (Zeng et al., 2022), low rank DAGs (Fang et al., 2020), incomplete data (Wang et al., 2020), prior knowledge (Cai et al., 2021), federated learning (Ng et al., 2022a; Gao et al., 2021a), and multi-task learning (Chen et al., 2021). The continuous constrained formulation has also been applied to other domains, e.g., reinforcement learning (Pruthi et al., 2020; Ruan et al., 2022), normalizing flows (Wehenkel and Louppe, 2020; Dai and Chen, 2022), domain adaptation (Yang et al., 2021b), recommendation system (Wang et al., 2022), and computer vision (Cui et al., 2020; Yang et al., 2021a; Zhang et al., 2021, 2022).

Like in the original work, most of these extensions rely on the *augmented Lagrangian method* (ALM) (Bertsekas, 1982, 1999) to solve the continuous constrained optimization problem. This choice of algorithm was originally motivated by the convergence result of ALM, which, unlike the classical *quadratic penalty method* (QPM) (Powell, 1969; Fletcher, 1987), does not require increasing the penalty coefficient to infinity (Zheng et al., 2018, Proposition 3). Despite abundant extensions and applications of the continuous constrained formulation, it remains unclear whether the required conditions for the standard convergence result of ALM, or more specifically, the regularity conditions, are satisfied in these settings, and whether the continuous constrained formulation is guaranteed to converge to a DAG solution, which is a key to structure learning.

Contributions. We examine the convergence properties of ALM and QPM for structure learning in the linear, nonlinear, and confounded cases. We conclude (i) that, unfortunately, the conditions behind standard convergence result of ALM are not satisfied in these settings, (ii) that, furthermore, the empirical behavior of ALM is similar to QPM that requires the penalty coefficient to go to infinity and is prone to ill-conditioning, and (iii) that, interestingly, QPM is guaranteed to converge to a DAG solution, under mild conditions. We then provide the implications of our theoretical results for existing approaches to help resolve the convergence issue, with an empirical comparison of them that verifies our findings and makes them more intuitive.

We note that the problem has received considerable attention. For instance, Wei et al. (2020) studied a related problem, involving the regularity conditions of the continuous constrained formulation. It is worth mentioning that our contributions are different and more complete in terms of the technical development and results. Specifically, Wei et al. (2020) focused on the Karush-Kuhn-Tucker (KKT) conditions, while our study focuses on the convergence of specific constrained optimization methods (i.e., ALM and QPM), which provides practical insight for solving the optimization problem. Furthermore, they focused on the linear case, while our results also apply to the nonlinear and confounded cases, under the same umbrella.

Organization of the paper. We give an overview of score-based learning, continuous constrained formulation of structure learning, and the ALM in Section 2. In Section 3, we examine the convergence of ALM and QPM for structure learning in the linear and nonlinear cases. We extend the results to the confounded case in Section 4, and connect them with different approaches to resolve the convergence issue in Section 5. We provide empirical studies in Section 6 to verify our results, and conclude our work in Section 7.

2 Background

We provide a brief review of score-based structure learning, the NOTEARS method (Zheng et al., 2018) and the standard convergence result of ALM.

2.1 Score-Based Structure Learning

Structure learning refers to the problem of learning a graphical structure (in our case a DAG) from data. Given the random vector $X = (X_1, \dots, X_d)$ consisting of d random variables, we assume that the corresponding design matrix $\mathbf{X} = [\mathbf{X}_1 j \quad j \mathbf{X}_d]$ 2 $\mathsf{R}^{n \ d}$ is generated from a joint distribution $P(X)$ (with a density $p(x)$) that is Markov with respect to the ground truth DAG G and can be factorized as $p(x) = \prod_{i=1}^{d} p_i(x_i j x_{\mathsf{PA}_i^{\mathsf{G}}})$, where $\mathsf{PA}_i^{\mathsf{G}}$ designates the set of parents of X_i in the DAG G. In general, the underlying DAG is only identifiable up to Markov equivalence under the faithfulness (Spirtes et al., 2000) or the sparsest Markov representation assumption (Raskutti and Uhler, 2018). Under certain assumptions on the data distribution, the DAG G is fully identifiable, such as the linear non-Gaussian model (Shimizu et al., 2006), linear Gaussian model with equal noise variances (Peters and Bühlmann, 2013a), nonlinear additive noise model (Hoyer et al., 2009; Peters et al., 2014), and post-nonlinear model (Zhang and Hyvärinen, 2009).

To recover the structure G or its Markov equivalence class, a major class of structure learning methods are the score-based methods that solves an optimization problem over the space of graphs using some goodness-of-fit measure with a sparsity regularization term. Some examples include GES (Chickering, 2002), ℓ_0-regularized likelihood (Van de Geer and Bühlmann, 2013), integer linear programming (Jaakkola et al., 2010; Cussens, 2011), dynamic programming (Koivisto and Sood, 2004; Ott et al., 2004; Singh and Moore, 2005), and A* (Yuan and Malone, 2013). Most of these methods tackle the structure search problem in its natural discrete form.

2.2 Continuous Constrained Optimization for Structure Learning

NOTEARS. Zheng et al. (2018) proposed a continuous constrained formulation for score-based learning of linear DAGs. In particular, the linear DAG model is equivalently represented by the linear structural equation model (SEM)

$$X = W^{\mathsf{T}} X + N, \tag{1}$$

where W is the weighted adjacency matrix of a DAG, and N is a noise vector characterized by the noise covariance matrix Ω. We assume here that the elements of N

are mutually independent, and there is no unobserved confounder. Denoting by $\circ$ the element-wise product, and by e^M the matrix exponential of a square matrix M, the authors have shown that $\mathrm{tr}(e^{W \circ W}) - d = 0$ holds if and only if W represents a DAG. The resulting continuous constrained optimization problem is

$$\min_{W \in \mathbb{R}^{d \times d}} \frac{1}{2n}\|\mathbf{X} - \mathbf{X}W\|_2^2 + \lambda\|W\|_1 \quad \text{subject to} \quad \mathrm{tr}(e^{W \circ W}) - d = 0,$$

where $\|\cdot\|_2$ and $\|\cdot\|_1$ denote the element-wise ℓ_2 and ℓ_1 norms, respectively, and λ is the regularization coefficient. Here, $(1/2n)\|\mathbf{X} - \mathbf{X}W\|_2^2$ is the least squares objective and is equal, up to a constant, to the log-likelihood of linear Gaussian DAGs assuming equal noise variances. The ℓ_1 regularization term $\|W\|_1$ is useful for enforcing sparsity on the matrix W.

NOTEARS-MLP. To generalize the above formulation to the nonlinear case, Zheng et al. (2020) used multi-layer perceptrons (MLPs) to model nonlinear relationships. For each variable X_i, let $\mathsf{MLP}(u; A_i)$ be the corresponding MLP with input row vector u, ℓ layers, weights $A_i = (A_i^{(1)}, \ldots, A_i^{(\ell)})$, and element-wise activation function $\sigma(\cdot)$, defined as follows:

$$\mathsf{MLP}(u; A_i) = \sigma(\cdots \sigma(\sigma(uA_i^{(1)})A_i^{(2)})\cdots)A_i^{(\ell)},$$
$$A_i^{(t)} \in \mathbb{R}^{s_{t-1} \times s_t}, \quad s_0 = d, \; s_\ell = 1.$$

Let $A = (A_1, \ldots, A_d)$ denote the weights of MLPs corresponding to all variables. The authors defined an equivalent adjacency matrix $(B(A))_{ji} = \|j\text{th-row}(A_i^{(1)})\|_2$ and proposed to solve the optimization problem

$$\min_A \frac{1}{n}\sum_{i=1}^{d}\|\mathbf{X}_i - \mathsf{MLP}(\mathbf{X}; A_i)\|_2^2 + \lambda\|A_i^{(1)}\|_1 \quad \text{subject to} \quad \mathrm{tr}(e^{B(A) \circ B(A)}) - d = 0.$$

As described in Section 1, there are several other extensions of NOTEARS to the nonlinear case that also adopt the continuous constrained formulation, e.g., DAG-GNN (Yu et al., 2019), GraN-DAG (Lachapelle et al., 2020), and MCSL (Ng et al., 2022b). In this work we focus on NOTEARS-MLP for convergence analysis, since it is conceptually simple compared to the others. We leave the analysis for the others for future work.

Optimization. The above optimization problems involve a hard DAG constraint and are solved via ALM, a general method for continuous constrained optimization, which we review next.

2.3 Augmented Lagrangian Method

Consider the generic constrained optimization problem

$$\min_{\theta \in \mathbb{R}^m} f(\theta) \quad \text{subject to} \quad h(\theta) = 0, \tag{2}$$

Algorithm 1 Augmented Lagrangian Method (Nocedal and Wright, 2006, Framework 17.3)

Require: starting penalty coefficient $\rho_1 > 0$; starting Lagrange multiplier α_1; multiplicative factor $\beta > 1$; reduction factor $\gamma < 1$; nonnegative sequence $\{\tau_k\}$; starting point θ_0

for $k = 1, 2, \ldots$ **do**
  Find an approximate minimizer θ_k of $L(\cdot, \alpha_k; \rho_k)$, starting at point θ_{k-1}, and terminating when $\|\nabla_\theta L(\theta, \alpha_k; \rho_k)\|_2 \le \tau_k$
  if final convergence test satisfied **then**
    stop with approximate solution θ_k
  end if
  Update multiplier $\alpha_{k+1} = \alpha_k + \rho_k h(\theta_k)$
  if $\|h(\theta_k)\|_2 > \gamma\|h(\theta_{k-1})\|_2$ **then**
    Update penalty coefficient $\rho_{k+1} = \beta\rho_k$
  else
    Update penalty coefficient $\rho_{k+1} = \rho_k$
  end if
end for

where the functions $f : \mathbb{R}^m \to \mathbb{R}$ and $h : \mathbb{R}^m \to \mathbb{R}^p$ are both twice continuously differentiable.

The ALM transforms a constrained optimization problem like (2) into a sequence of unconstrained ones with solutions converging to a solution of the original problem. The key idea is to combine the Lagrangian formulation with QPM, yielding an augmented problem

$$\min_{\theta \in \mathbb{R}^m} f(\theta) + \frac{\rho}{2}\|h(\theta)\|_2^2 \quad \text{subject to} \quad h(\theta) = 0,$$

where $\rho > 0$ is the penalty coefficient. The augmented Lagrangian function of the formulation above is

$$L(\theta, \alpha; \rho) = f(\theta) + \alpha^\top h(\theta) + \frac{\rho}{2}\|h(\theta)\|_2^2,$$

where $\alpha \in \mathbb{R}^p$ is an estimate of the Lagrange multiplier. A version of the procedure is described in Algorithm 1, which is essentially based on dual ascent. The minimization problem of $L(\theta, \alpha; \rho)$ can sometimes be solved only to stationarity, if, for example, it is nonconvex, as is the case in the formulation of NOTEARS owing to the nonconvexity of its specific constraint function.

Based on the procedure of ALM outlined above, we review one of its standard convergence results (Bertsekas, 1982, 1999; Nocedal and Wright, 2006). The following definition is required to state this result and is crucial to the contribution of our work.

Definition 1 (Regular point). *We say that a point θ is regular, or that it satisfies the linear independence constraint qualification (LICQ), if the rows of the Jacobian matrix of h evaluated at θ, $\nabla_\theta h(\theta) \in \mathbb{R}^{p \times m}$, are linearly independent.*

Theorem 1 (Nocedal and Wright (2006, Theorem 17.5 & 17.6)). *Let θ be a regular point of (2) that satisfies the second-order su cient conditions (see Appendix A) with vector α . Then there exist positive scalars $\bar{\rho}$ (su ciently large), δ, ϵ, and M such that for all α_k and ρ_k satisfying*

$$k\alpha_k \quad \alpha \; k_2 \quad \rho_k\delta, \quad \rho_k \quad \bar{\rho},$$

the problem

$$\min_{\theta 2 \mathsf{R}^m} \; L(\theta, \alpha_k; \rho_k) \;\; \text{subject to} \;\; k\theta \quad \theta \; k_2 \quad \epsilon$$

has a unique solution θ_k. Moreover, we have

$$k\theta_k \quad \theta \; k_2 \quad Mk\alpha_k \quad \alpha \; k_2/\rho_k \tag{3}$$

and

$$k\alpha_{k+1} \quad \alpha \; k_2 \quad Mk\alpha_k \quad \alpha \; k_2/\rho_k \,, \tag{4}$$

where $\alpha_{k+1} = \alpha_k + \rho_k h(\theta_k)$.

An important consequence of Theorem 1 is that if the penalty coefficient ρ_k is larger than both $\bar{\rho}$ and $k\alpha_k \quad \alpha \; k_2/\delta$, then the inequalities (3) and (4) hold. If, in addition, $\rho_k > M$, then $\alpha_k \; ! \; \alpha$ by (4) and $\theta_k \; ! \; \theta$ by (3), *without increasing the coe cient ρ_k to infinity*. This property often motivates the usage of the ALM over the other approaches, e.g., QPM, for constrained optimization. Specifically, this was the original motivation provided by Zheng et al. (2018, Proposition 3) for using the ALM. The reason is that the QPM requires bringing the penalty coefficient ρ_k to infinity, which may lead to ill-conditioning issue when solving the minimization problem. In the next section, we examine the required conditions and show that Theorem 1 does not apply to the continuous constrained formulation proposed by Zheng et al. (2018, 2020).

3 Convergence of the Continuous Constrained Optimization Methods

In this section, we take a closer look at the convergence of ALM and QPM for learning DAGs in the linear and nonlinear cases. We assume that there is no unobserved confounder and focus on the formulation of NOTEARS and NOTEARS-MLP. We will show in Section 4 that our analysis generalizes to the confounded case. We consider the constrained optimization problem

$$\min_{\theta} \;\; f(\theta) \;\; \text{subject to} \;\; h(B(\theta)) = 0, \tag{5}$$

where $f(\theta)$ is the objective function and $h(B(\theta))$ is a (scalar-valued) constraint function that enforces acyclicity on the (equivalent) weighted adjacency matrix $B(\theta)$. We assume here that the functions f, h, and B are continuously differentiable. Specifically, the constraint term proposed by Zheng et al. (2018) is given by $h_{\mathsf{exp}}(B(\theta)) = \mathrm{tr}(e^{B(\theta) \;\; B(\theta)}) \quad d$. As described in Section 2.2, in the linear case, the parameter θ corresponds to the weighted adjacency matrix of the linear SEM, i.e., we have $\theta = W$ and $B(W) = W$. In the nonlinear case, the parameter θ corresponds to the weights of the MLPs. That is, we have $\theta = A$ and $(B(A))_{ji} = kj\text{th-row}(A_i^{(1)})k_2$. Hereafter we use $f(W)$ and $h_{\mathsf{exp}}(W)$ to refer to the objective and DAG constraint term in the linear case, and $f(A)$ and $h_{\mathsf{exp}}(B(A))$ to refer to those in the nonlinear case.

By assuming that function f is continuously differentiable, our analysis does not consider the ℓ_1 regularization for simplicity. In Section 6, we study empirically the constrained formulation with and without the excluded regularization term, and show that our analysis appears to generalize to the ℓ_1-regularized case.

3.1 Regularity of DAG Constraint Term

To investigate whether Theorem 1 applies to problem (5), one has to first verify if the DAG constraint term $h(B(\theta))$ satisfies the regularity conditions. The following condition is required for our analysis.

Assumption 1. *The function $h(B(\theta)) = 0$ if and only if its gradient $r_\theta h(B(\theta)) = 0$.*

Both DAG constraint terms in the linear and nonlinear cases satisfy the assumption above, with a proof provided in Appendix B.1.

Theorem 2. *The functions $h_{exp}(W)$ and $h_{exp}(B(A))$ satisfy Assumption 1.*

Assumption 1 implies that the Jacobian matrix of function $h(B(\theta))$ (after reshaping) evaluated at any feasible point of problem (5) corresponds to a zero row vector, which is itself is not linearly independent and therefore leads to the following remark.

Remark 1. *If the function $h(B(\theta))$ satisfies Assumption 1, any feasible solution of problem (5) is not regular and Theorem 1 does not apply.*

With Theorem 2, this shows that the advantage of ALM (illustrated by Theorem 1) does not apply to the DAG constraints developed by Zheng et al. (2018, 2020) in the linear and nonlinear cases. Hence, we are left with no guarantee that the penalty coefficient ρ does not have to go to infinity for ALM to converge. In Section 6.1, we show empirically that ρ grows without converging just like it would in QPM that is prone to ill-conditioning, which we describe in the next section.

3.2 Quadratic Penalty Method

Apart from ALM, QPM is another method for solving the constrained optimization problem (5), whose con-

Algorithm 2 Quadratic Penalty Method (Nocedal and Wright, 2006, Framework 17.1)

Require: starting penalty coefficient $\rho_1 > 0$; multiplicative factor $\beta > 1$; nonnegative sequence $\{\tau_k\}$; starting point θ_0

1: **for** $k = 1, 2, \ldots$ **do**
2: Find an approximate minimizer θ_k of $Q(\cdot\,; \rho_k)$, starting at point θ_{k-1}, and terminating when $\|\nabla_\theta Q(\theta; \rho_k)\|_2 \leq \tau_k$
3: **if** final convergence test satisfied **then**
4: **stop** with approximate solution θ_k
5: **end if**
6: Update penalty coefficient $\rho_{k+1} = \beta\rho_k$
7: **end for**

vergence property is studied in this section. We first define the quadratic penalty function

$$Q(\theta; \rho) = f(\theta) + \frac{\rho}{2} h(B(\theta))^2, \tag{6}$$

and describe the procedure of QPM in Algorithm 2. Note that it is essentially the same as ALM but without the Lagrangian part, and thus has a simpler procedure.

This approach adds a quadratic penalty term for the constraint violation to the objective $f(\theta)$. By gradually increasing the penalty coefficient ρ, we penalize the constraint violation with increasing severity. Therefore, it makes intuitive sense to think that the procedure converges to a feasible solution (i.e., a DAG solution) as we bring ρ to infinity. However, this is not necessarily true: in general, Algorithm 2 returns only a stationary point of the quadratic penalty term $h(B(\theta))^2$ (Nocedal and Wright, 2006, Theorem 17.2). Fortunately, if the DAG constraint term $h(B(\theta))$ satisfies Assumption 1, the procedure is guaranteed to converge to a feasible solution, under mild conditions, formally stated in Theorem 3. The proof is provided in Appendix B.2. Note that this theorem and its proof are adapted from Theorem 17.2 in Nocedal and Wright (2006).

Theorem 3. *Suppose in Algorithm 2 that the penalty coefficients satisfy $\rho_k \to \infty$ and the sequence of nonnegative tolerances $\{\tau_k\}$ is bounded.*[1] *Suppose also that the function $h(B(\theta))$ satisfies Assumption 1. Then every limit point θ^* of the sequence $\{\theta_k\}$ is feasible.*

Remark 2. *With the constraint terms $h_{exp}(W)$ and $h_{exp}(B(A))$, Theorems 3 and 2 guarantee that, under mild conditions, Algorithm 2 returns a DAG solution as $\rho_k \to \infty$ based on inexact minimizations of $Q(\cdot\,; \rho_k)$.*

Although the standard convergence result of ALM (i.e., Theorem 1) does not hold as the DAG constraint terms proposed by Zheng et al. (2018, 2020) satisfy Assumption 1, this property ensures that QPM returns a DAG, which is indeed a key to structure learning. The remark above also explains why the implementations of ALM with these two constraints often return DAG solutions in practice (after thresholding). Furthermore, if Assumption 1 is satisfied, Theorem 3 verifies that one can directly use the value of DAG constraint term as an indicator for the final convergence test in Algorithm 2, i.e., $h(B(\theta_k)) \leq \epsilon$ with $\epsilon > 0$ being a small tolerance. It is worth noting that this convergence test has been adopted in the current implementation of NOTEARS (Zheng et al., 2018) and most of its extensions.

Practical issue. In practice, one is, at most, only able to increase the penalty coefficient ρ to a very large value, e.g., of order 10^{16}. Therefore, the final solution can only satisfy $h(B(\theta)) \leq \epsilon$ up to numerical precision with $\epsilon > 0$ being a small tolerance, e.g., of order 10^{-8}. In this case, the solution may contain many entries close to zero and does not correspond exactly to a DAG. Following Zheng et al. (2018), a thresholding step on the estimated entries is needed to convert the solution into a DAG; the experiments in Section 6.1 suggest that a small threshold (e.g., 0.05) suffices. However, a moderately large threshold (e.g., 0.3) can still be useful for reducing the false discoveries.

3.3 Other DAG Constraint Term

Apart from the matrix exponential term proposed by Zheng et al. (2018), Yu et al. (2019) developed a polynomial alternative that may have better numerical stability with a proper choice of $\mu > 0$:

$$h_{\text{poly}}(B(\theta)) = \operatorname{tr}\left((I + \mu B(\theta) \circ B(\theta))^d\right) - d.$$

Our analysis generalizes to the above constraint term.

Corollary 1. *The functions $h_{poly}(W)$ and $h_{poly}(B(A))$ satisfy Assumption 1.*

4 With Unobserved Confounding

We study whether our convergence analysis is applicable to the confounded case. Recently, Bhattacharya et al. (2021) applied the continuous constrained formulation proposed by Zheng et al. (2018) to estimate structures with unobserved confounding, by deriving algebraic characterizations for different classes of acyclic directed mixed graphs (ADMGs), i.e., ancestral, arid, and bow-free graphs. Here we consider the bow-free graphs that are the least restrictive for our further analysis. Note that a bow-free ADMG refers to an ADMG in which the directed and bidirected edges do not both appear for any pair of vertices.

[1] A stricter condition $\tau_k \to 0$ is often used in the analysis of QPM (Nocedal and Wright, 2006, Theorem 17.2) but is not required here.

Denote by W and Ω the weighted adjacency matrix and noise covariance matrix of the linear SEM defined in Eq. (1). In the confounded case, there exist unobserved variables that are parents of more than one observed variable, implying that the noise terms are correlated (Pearl, 2009). Since Ω is symmetric, we have $X_i \$ X_j$, $i \neq j$ in the ADMG if and only if $\Omega_{ji} \neq 0$. In other words, the weighted adjacency matrix W and noise covariance matrix Ω represent the directed and bidirected edges in the ADMG, respectively. Bhattacharya et al. (2021) adopted the ALM to solve the constrained optimization problem (5) with the approximate BIC score (Su et al., 2016), where θ corresponds to the parameters W and Ω. In this case, the algebraic constraint term of bow-free ADMGs is given by

$$h_{\mathsf{bf}}(W,\Omega) = \mathrm{tr}(e^{W \ W}) \quad d + \mathrm{sum}(W \quad W \quad \Omega \quad \Omega).$$

The authors have shown that $h_{\mathsf{bf}}(W,\Omega) = 0$ if and only if the ADMG defined by W and Ω corresponds to a bow-free graph. To study the convergence property of ALM in this case, we have the following result regarding its regularity, with a proof given in Appendix B.3.

Theorem 4. *$h_{bf}(W,\Omega)$ satisfies Assumption 1.*

Similar result also holds for the polynomial constraint term proposed by Yu et al. (2019).

Corollary 2. *Theorems 4 holds if the matrix exponential $e^{W \ W}$ in the function $h_{bf}(W,\Omega)$ is replaced with the matrix polynomial $(I + \mu W \quad W)^d$ for any $\mu > 0$.*

As a consequence, similar to the setting of NOTEARS and NOTEARS-MLP studied in Section 3, Remark 1 indicates there is no guarantee such that the penalty coefficient ρ does not have to go to infinity for ALM to converge. Fortunately, with Theorem 3, using QPM to solve the constrained optimization problem is guaranteed to return a solution that satisfies $h_{\mathsf{bf}}(W,\Omega) \quad \epsilon$ up to numerical precision, under mild conditions.

5 Resolving the Convergence Issue

The analysis in Sections 3 and 4 implies that the advantage of ALM illustrated by Theorem 1 does not hold in the continuous constrained formulation for structure learning. Specifically, it is not guaranteed that the penalty coefficient does not have to be increased indefinitely for the convergence of ALM. The experiments in Section 6.1 verify this study and show empirically that ALM requires increasing the coefficient to a very large value to converge to a DAG solution, similar to QPM. This is known to cause numerical difficulties and ill-conditioning issues on the objective landscape (Bertsekas, 1999; Nocedal and Wright, 2006). The reason is that when the penalty term is large, the Hessian matrix is ill-conditioned and has a high condition number. In this case, the function contour is stretched out, and the gradients may not be the best direction to descend to the minimum, leading to a zigzag path. This is illustrated by a bivariate example in Appendix C.

In light of our theoretical results, we give a brief overview on different approaches that help resolve the convergence issue, and provide an empirical comparison of them in Sections 6.2 and 6.3 to illustrate our findings. In particular, the first approach uses a second-order method that is less susceptible to ill-conditioning, while the second approach devises an alternative algebraic DAG constraint with a local search procedure. The last two approaches adopt a different unconstrained formulation, thus avoiding the convergence issue caused by the hard DAG constraint in problem (5).

It is worth noting that the effectiveness of some of the approaches below have been studied separately in the papers that proposed them. To the best of our knowledge, some of these approaches have not been connected with the convergence issue of NOTEARS, which is the focus of this section as well as Sections 6.2 and 6.3 . Doing so allows one to understand which approach better resolves the convergence issue.

Second-order method. As pointed out by Bertsekas (1999); Antoniou and Lu (2007); Bottou et al. (2018), one may consider using a second-order method such as the quasi-Newton method (Nocedal and Wright, 2006) that handles ill-conditioning better by incorporating curvature information through approximations of the Hessian matrix. This is consistent with the recent works (Zheng et al., 2018, 2020; Pamfil et al., 2020) that adopt L-BFGS (Byrd et al., 2003) to solve the optimization subproblems. However, the original motivation of using quasi-Newton method was mainly about efficiency consideration, or, specifically, to reduce the number of evaluations of the matrix exponential that takes $O(d^3)$ cost (Al-Mohy and Higham, 2009; Zheng et al., 2018). We note here that another key advantage of using L-BFGS is, interestingly, to help resolve ill-conditioning issue, as verified by the experiments in Section 6.2 and a bivariate example in Appendix C.

Absolute value adjacency matrix and KKT-informed local search. In contrast to the quadratic adjacency matrix $B(\theta) \quad B(\theta)$ in the DAG constraint term $h_{\mathsf{exp}}(B(\theta))$, the Abs-KKTS method proposed by Wei et al. (2020) adopts an absolute value adjacency matrix given by $h^{\emptyset}_{\mathsf{exp}}(B(\theta)) = \mathrm{tr}(e^{jB(\theta)j}) \quad d$, where $j \ j$ denotes the element-wise absolute value of a matrix, together with a local search procedure informed by the KKT conditions as a post-processing step. The resulting procedure returns a solution that satisfies the KKT conditions, and leads to improvement in the structure learning performance of NOTEARS.

Soft constraints. Ng et al. (2020) showed that soft sparsity and DAG constraints suffice to asymptotically estimate a DAG equivalent to the true DAG under mild conditions when using the likelihood of linear Gaussian directed graphical models (possibly cyclic) as the objective function instead of the least squares loss that corresponds to the likelihood of linear Gaussian DAGs. This gives rise to the following *unconstrained* optimization problem in the case of equal noise variances, denoted as GOLEM-EV:

$$\min_{W \in \mathbb{R}^{d \times d}} \frac{d}{2} \log \|\mathbf{X} - \mathbf{X}W\|_2^2 - \log|\det(I - W)| + \lambda_1 \|W\|_1 + \lambda_2 h_{\text{exp}}(W),$$

where λ_1 and λ_2 are the regularization coefficients. With this unconstrained formulation, the ill-conditioning issue caused by the hard DAG constraint in problem (5) can be completely avoided, since one is able to directly solve the above problem using continuous optimization, without the need of any constrained optimization method like ALM or QPM.

Direct optimization in DAG space. Yu et al. (2021) developed an algebraic representation of DAGs based on graph Hodge theory (Jiang et al., 2011; Bang-Jensen and Gutin, 2009), and showed that one can directly perform continuous optimization in the space of all possible DAGs without relying on a hard DAG constraint. Similar to GOLEM-EV, the constrained optimization problem (5) can be reformulated in the linear case as an *unconstrained* one

$$(U, p) = \arg\min_{U \in S, p \in \mathbb{R}^d} f(U - \text{ReLU}(\text{grad}(p))), \quad (7)$$

where S refers to the space of all $d \times d$ skew-symmetric matrices, $(\text{grad}(p))_{ji} = p_i - p_j$ denotes the gradient flow defined on the nodes of a graph (Lim, 2015), and $(\text{ReLU}(M))_{ji} = \max(0, M_{ji})$ denotes the rectified linear unit function (Nair and Hinton, 2010) of a square matrix M. The final solution is given by $W = U - \text{ReLU}(\text{grad}(p))$, whose nonzero entries are guaranteed to represent a DAG (Yu et al., 2021, Theorem 3.5). Since the objective function is highly nonconvex, randomly initializing U and p may lead to a stationary point far from the global optimum. The authors thus proposed a two-step procedure, denoted as NoCurl, that first obtains a rough estimate of the solution by solving the subproblem of NOTEARS either once or twice with a slightly large penalty coefficient (e.g., $\rho = 10^3$), and uses that estimate to compute the initialization of U and p for problem (7). Similar to second-order method, this method was originally motivated by efficiency consideration. Since it avoids the hard DAG constraint and accordingly does not require a large penalty coefficient, we note that this method can also help avoid the ill-conditioning issue.

6 Experiments

We conduct experiments on the structure learning tasks and take a closer look at the optimization processes to verify our study. In Section 6.1, we demonstrate that ALM behaves similarly to QPM, both of which converge to an approximately DAG solution when the penalty coefficients are very large. We compare the ability of different optimization algorithms to handle ill-conditioning in Section 6.2, and the other approaches to help resolve the convergence issue in Section 6.3.

Methods. We experiment with both NOTEARS and NOTEARS-MLP. We also consider their variants with the ℓ_1 regularization term, denoted as NOTEARS-L1 and NOTEARS-MLP-L1, respectively.

Implementations. Our implementations are based on the code[2] released by Zheng et al. (2018, 2020) with the DAG constraint term $h_{\text{exp}}(B)$. We also use the least squares objective and default hyperparameters in our experiments. Unless otherwise stated, we employ the L-BFGS algorithm (Byrd et al., 2003) to solve each subproblem and a threshold of 0.3 for post-processing. In the linear case, we use a pre-processing step to center the data by subtracting the mean of each variable from the samples $\mathbf{X}$. The code is available at `https://github.com/ignavierng/notears-convergence`.

Simulations. We simulate the ground truth DAGs using the Erdös–Rényi (Erdös and Rényi, 1959) or scale-free (Barabási and Albert, 1999) model with kd edges on average, denoted as ERk or SFk, respectively. Unless otherwise stated, based on the graph sizes $d \in \{10, 20, 50, 100\}$ and different data generating procedures, we generate 1000 samples with standard Gaussian noises. For NOTEARS and NOTEARS-L1, we simulate the linear DAG model with edge weights sampled uniformly from $[-2, -0.5] \cup [0.5, 2]$, similar to (Zheng et al., 2018). For the nonlinear variants NOTEARS-MLP and NOTEARS-MLP-L1, we consider the data generating procedure used by Zheng et al. (2020), where each function is sampled from a Gaussian process with RBF kernel of bandwidth one. Both data models are known to be fully identifiable (Peters and Bühlmann, 2013a; Peters et al., 2014).

Metrics. We report the structural Hamming distance (SHD), structural intervention distance (SID) (Peters and Bühlmann, 2013b) and true positive rate (TPR), averaged over 30 random trials.

6.1 ALM Behaves Similarly to QPM

We conduct experiments to show that ALM behaves similarly to QPM in the context of structure learning,

[2] `https://github.com/xunzheng/notears`

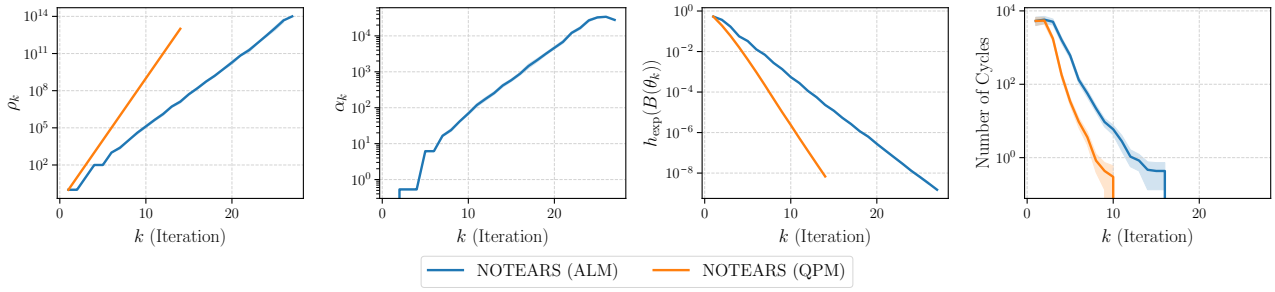


Figure 1: Optimization processes of NOTEARS using ALM and QPM on synthetic data. The ground truths are 10-node ER1 graphs and the sample size is $n = 1000$. Each data point corresponds to the k-th iteration.

and that both of them converge to a DAG solution. Our goal here is not to show that QPM performs better than ALM, but rather to study their empirical behavior.

We first take a closer look at the optimization processes of ALM and QPM on the 10-node ER1 graphs. Figure 1 depicts the penalty coefficient ρ_k, estimate of Lagrange multiplier α_k (only for ALM), value of DAG constraint term $h_{\mathsf{exp}}(B(\theta_k))$, and number of cycles in the k-th iteration of the optimization for NOTEARS, while those for NOTEARS-L1, NOTEARS-MLP, and NOTEARS-MLP-L1 are visualized in Figure 5 in Appendix E. Note that we use a small threshold 0.05 when computing the number of cycles. Complementing our study in Section 3.1, ALM requires a very large coefficient ρ_k to converge, similar to QPM, which suggests that they both behave similarly and that the standard convergence result of ALM appears to not hold here. On the other hand, when the penalty coefficient ρ_k is very large, one observes that both ALM and QPM converge to a solution whose value of $h_{\mathsf{exp}}(B(\theta_k))$ is very close to zero, i.e., smaller than 10^{-8}, which yields a DAG solution after thresholding at 0.05. Since one is not able to increase the penalty coefficient ρ_k to infinity in practice, this serves as an empirical validation of Theorem 3. Interestingly, Figures 1 and 5 suggest that one may consider using QPM instead of ALM in practice as it converges in fewer number of iterations.

We further investigate whether ALM and QPM yield similar structure learning performance. The results with ER1 graphs are reported in Figure 6 in Appendix E, showing that ALM performs similarly to QPM across all metrics. All these observations appear to generalize to the case with ℓ_1 regularization term that is not covered by our analysis in Section 3.

Real data. We conduct empirical studies to verify whether our observations hold on the protein signaling dataset by Sachs et al. (2005). Due to the limited space, the optimization processes of ALM and QPM on this dataset are shown in Figure 7, which yield consistent observations with those on synthetic data, i.e., ALM behaves similarly to QPM that requires the penalty coefficient to be very large (e.g., 10^{12}), both of which converge to a solution whose value of $h_{\mathsf{exp}}(B(\theta_k))$ is very close to zero, yielding a DAG after thresholding.

6.2 Different Optimization Algorithms for Handling Ill-Conditioning

The previous experiments demonstrate that ALM behaves similarly to QPM that requires bringing the penalty coefficient to infinity in order to converge to a DAG solution, which is known to cause numerical difficulties and ill-conditioning issues on the objective landscape (Bertsekas, 1999; Nocedal and Wright, 2006). Here we experiment with different optimization algorithms for solving the QPM subproblems of NOTEARS and NOTEARS-MLP-L1, to investigate which of them handle ill-conditioning better and produce a better solution in practice. The optimization algorithms include gradient descent with momentum (Qian, 1999), Nesterov accelerated gradient (NAG) (Nesterov, 1983), Adam (Kingma and Ba, 2014) and L-BFGS (Byrd et al., 2003); see Appendix D for the implementation details. Note that gradient descent with momentum and NAG often terminate earlier because of numerical difficulties, so we report its results right before termination.

Due to the space limit, the optimization processes of NOTEARS and NOTEARS-MLP-L1 on the 100-node ER1 graphs are visualized in Figure 8 in Appendix E. One first observes that momentum and NAG terminate when the coefficient ρ_k reaches 10^7 and 10^6, respectively, in the linear case, and reaches 10^9 and 10^8, respectively, in the nonlinear case, indicating that they fail to handle ill-conditioning in these settings. In the linear case, L-BFGS is more stable than Adam for large ρ_k, thus returning a solution with a lower objective value $f(\theta_k)$ and SHD. The opposite is observed in the nonlinear case where Adam has consistently lower $f(\theta_k)$ and SHD than L-BFGS. Similar observations are also made for the overall structure learning performance on ER1 and SF4 graphs, as depicted in Figures 2 and 9 in Appendix

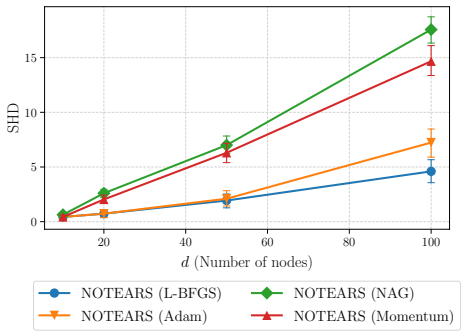


Figure 2: Empirical results of different optimization algorithms for solving the QPM subproblems of NOTEARS on synthetic data. The ground truths are ER1 graphs and the sample size is $n = 1000$.

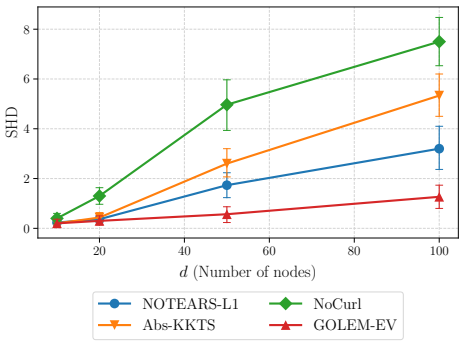


Figure 3: Empirical results of different methods on synthetic data with ER1 graphs and $n = 1000$ samples.

E, with graph sizes $d \in \{10, 20, 50, 100\}$. In the linear case, L-BFGS performs the best across most settings, while the performance of Adam is slightly better than L-BFGS in the nonlinear case. Gradient descent with momentum and NAG give rise to much higher SHD and SID, especially on large graphs.

As compared to first-order method, the observations above suggest that second-order method such as L-BFGS handles ill-conditioning better by incorporating curvature information through approximations of the Hessian matrix, which verifies our findings in Section 5. This is also consistent with the optimization literature (Bertsekas, 1999) and the bivariate example in Appendix C. The Adam algorithm, on the other hand, lies in the middle as it employs diagonal rescaling on the parameter space by maintaining running averages of past gradients (Bottou et al., 2018). In the nonlinear case, it may be surprising that Adam performs slightly better than L-BFGS, as its estimate of the Hessian matrix is not as accurate as that of L-BFGS. Nevertheless, this demonstrates its effectiveness for training MLPs, and may be understandable given its popularity in various deep learning tasks (Schmidt et al., 2021).

6.3 Further Resolving the Convergence Issue

In this section, we provide empirical comparisons of different methods for resolving the convergence issue of the NOTEARS formulation, as described in Section 5. In particular, we compare NOTEARS-L1 (with L-BFGS) to Abs-KKTS[3], NoCurl, and GOLEM-EV. The implementation details of these methods are described in Appendix D. Here we consider ER1 and SF4 graphs with 1000 and $3d$ samples.

[3]We consider only Abs-KKTS instead of NOTEARS-KKTS because otherwise we also have to apply the KKT-informed local search for NoCurl and GOLEM-EV to ensure a fair comparison, which is not the focus of our work.

Figure 3 shows the SHD on ER1 graphs with 1000 samples, while the complete results on ER1 and SF4 graphs with 1000 and $3d$ samples can be found in Figures 10 and 11 in Appendix E. Overall, GOLEM-EV has the best performance across nearly all settings, especially in terms of SID. NOTEARS-L1 and Abs-KKTS perform similarly on SF4 graphs, while the former has lower SHD and SID on ER1 graphs. NoCurl has a high SHD especially for the case with $3d$ samples, which indicates that it requires a larger sample size to perform well. This experiment suggests that despite being susceptible to ill-conditioning, NOTEARS-L1 with L-BFGS is still very competitive in practice and performs even better than Abs-KKTS and NoCurl that are less susceptible to the convergence issue, possibly because L-BFGS can help remedy the ill-conditioning issue, as demonstrated in Section 6.2.

7 Conclusion

We examined the convergence of ALM and QPM for structure learning in the linear, nonlinear, and confounded cases. In particular, we dug into the standard convergence result of ALM and showed that the required regularity conditions are not satisfied in this setting. Further experiments demonstrate that ALM behaves similarly to QPM that requires bringing the penalty coefficient to infinity and is prone to ill-conditioning. We then showed theoretically and empirically that QPM guarantees convergence to a DAG solution, under mild conditions. The empirical studies also suggest that our analysis generalizes to the cases with ℓ_1 regularization. Lastly, we connected our theoretical results with different approaches to help resolve the convergence issue, and provided an empirical comparison of them to further illustrate our findings. In particular, it is worth noting that second-order method can help remedy the ill-conditioning issue of the continuous constrained formulation and therefore leads to competitive structure learning results in practice.

Acknowledgments

The authors would like to thank the anonymous reviewers for their useful comments. This work was supported in part by the National Institutes of Health (NIH) under Contract R01HL159805, by the NSF-Convergence Accelerator Track-D award #2134901, by the United States Air Force under Contract No. FA8650-17-C7715, by a grant from Apple, by the Canada CIFAR AI Chair Program, by an IVADO excellence PhD scholarship, and by a Google Focused Research award. The NIH or NSF is not responsible for the views reported in this article. Simon Lacoste-Julien is a CIFAR Associate Fellow in the Learning in Machines & Brains program.

References

A. Al-Mohy and N. Higham. A new scaling and squaring algorithm for the matrix exponential. *SIAM Journal on Matrix Analysis and Applications*, 31, 2009.

A. Antoniou and W.-S. Lu. *Practical Optimization: Algorithms and Engineering Applications*. 01 2007.

J. Bang-Jensen and G. Gutin. *Digraphs. Theory, Algorithms and Applications*. Springer Monographs in Mathematics, 2009.

A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

A. Bellot and M. van der Schaar. Deconfounded score method: Scoring DAGs with dense unobserved confounding. *arXiv preprint arXiv:2103.15106*, 2021.

D. P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Academic Press, 1982.

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 2nd edition, 1999.

R. Bhattacharya, T. Nagarajan, D. Malinsky, and I. Shpitser. Differentiable causal discovery under unmeasured confounding. In *International Conference on Artificial Intelligence and Statistics*, 2021.

L. Bottou, F. E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *SIAM Rev.*, 60:223–311, 2018.

P. Brouillard, S. Lachapelle, A. Lacoste, S. Lacoste-Julien, and A. Drouin. Differentiable causal discovery from interventional data, 2020.

R. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16, 2003.

H. Cai, R. Song, and W. Lu. ANOCE: Analysis of causal effects with multiple mediators via constrained structural learning. In *International Conference on Learning Representations*, 2021.

X. Chen, H. Sun, C. Ellington, E. Xing, and L. Song. Multi-task learning of order-consistent causal graphs. In *Advances in Neural Information Processing Systems*, 2021.

D. M. Chickering. Learning Bayesian networks is NP-complete. In *Learning from Data: Artificial Intelligence and Statistics V*. Springer, 1996.

D. M. Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3(Nov):507–554, 2002.

Z. Cui, T. Song, Y. Wang, and Q. Ji. Knowledge augmented deep neural networks for joint facial expression and action unit recognition. In *Advances in Neural Information Processing Systems*, 2020.

J. Cussens. Bayesian network learning with cutting planes. In *Conference on Uncertainty in Artificial Intelligence*, 2011.

E. Dai and J. Chen. Graph-augmented normalizing flows for anomaly detection of multiple time series. In *International Conference on Learning Representations*, 2022.

P. Erdös and A. Rényi. On random graphs I. *Publicationes Mathematicae*, 6:290–297, 1959.

Z. Fang, S. Zhu, J. Zhang, Y. Liu, Z. Chen, and Y. He. Low rank directed acyclic graphs and causal structure learning. *arXiv preprint arXiv:2006.05691*, 2020.

G. R. A. Faria, A. Martins, and M. A. T. Figueiredo. Differentiable causal discovery under latent interventions. In *Conference on Causal Learning and Reasoning*, 2022.

R. Fletcher. *Practical Methods of Optimization*. Wiley-Interscience, 1987.

E. Gao, J. Chen, L. Shen, T. Liu, M. Gong, and H. Bondell. Federated causal discovery. *arXiv preprint arXiv:2112.03555*, 2021a.

Y. Gao, L. Shen, and S.-T. Xia. DAG-GAN: Causal structure learning with generative adversarial nets. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021b.

T. Geffner, E. Kiciman, A. Lamb, M. Kukla, M. Allamanis, and C. Zhang. FCause: Flow-based causal discovery, 2022. URL https://openreview.net/forum?id=HO_LL-oqBzW.

P. Hoyer, D. Janzing, J. M. Mooij, J. Peters, and B. Schölkopf. Nonlinear causal discovery with additive noise models. In *Advances in Neural Information Processing Systems*, 2009.

T.-Y. Hsieh, Y. Sun, X. Tang, S. Wang, and V. G. Honavar. SrVARM: State regularized vector autoregressive model for joint learning of hidden state transitions and state-dependent inter-variable dependen-

cies from multi-variate time series. In *Proceedings of the Web Conference*, 2021.

T. Jaakkola, D. Sontag, A. Globerson, and M. Meila. Learning Bayesian network structure using LP relaxations. In *International Conference on Artificial Intelligence and Statistics*, 2010.

X. Jiang, L.-H. Lim, Y. Yao, and Y. Ye. Statistical ranking and combinatorial Hodge theory. *Mathematical Programming*, 127, 11 2011.

D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2014.

M. Koivisto and K. Sood. Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research*, 5(Dec):549–573, 2004.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, Cambridge, MA, 2009.

S. Lachapelle, P. Brouillard, T. Deleu, and S. Lacoste-Julien. Gradient-based neural DAG learning. In *International Conference on Learning Representations*, 2020.

L.-H. Lim. Hodge laplacians on graphs. *SIAM Review*, 62, 07 2015.

V. Nair and G. Hinton. Rectified linear units improve restricted Boltzmann machines. In *International Conference on Machine Learning*, 2010.

Y. E. Nesterov. A method for solving the convex programming problem with convergence rate $O(1/k^2)$. *Doklady ANSSSR*, 269:543–547, 1983.

I. Ng, S. Zhu, Z. Chen, and Z. Fang. A graph autoencoder approach to causal structure learning. *arXiv preprint arXiv:1911.07420*, 2019.

I. Ng, A. Ghassami, and K. Zhang. On the role of sparsity and DAG constraints for learning linear DAGs. In *Advances in Neural Information Processing Systems*, 2020.

I. Ng, , and K. Zhang. Towards federated Bayesian network structure learning with continuous optimization. In *International Conference on Artificial Intelligence and Statistics*, 2022a.

I. Ng, S. Zhu, Z. Fang, H. Li, Z. Chen, and J. Wang. Masked gradient-based causal structure learning. In *SIAM International Conference on Data Mining*, 2022b.

J. Nocedal and S. J. Wright. *Numerical optimization*. Springer series in operations research and financial engineering. Springer, 2nd edition, 2006.

S. Ott, S. Imoto, and S. Miyano. Finding optimal models for small gene networks. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, 9:557–67, 2004.

R. Pamfil, N. Sriwattanaworachai, S. Desai, P. Pilgerstorfer, P. Beaumont, K. Georgatzis, and B. Aragam. DYNOTEARS: Structure learning from time-series data. In *International Conference on Artificial Intelligence and Statistics*, 2020.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, 2019.

J. Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2009.

J. Peters and P. Bühlmann. Identifiability of Gaussian structural equation models with equal error variances. *Biometrika*, 101(1):219–228, 2013a.

J. Peters and P. Bühlmann. Structural intervention distance (SID) for evaluating causal graphs. *Neural Computation*, 27, 2013b.

J. Peters, J. M. Mooij, D. Janzing, and B. Schölkopf. Causal discovery with continuous additive noise models. *Journal of Machine Learning Research*, 15(1): 2009–2053, 2014.

M. J. D. Powell. Nonlinear programming—sequential unconstrained minimization techniques. *The Computer Journal*, 12(3), 1969.

P. Pruthi, J. González, X. Lu, and M. Fiterau. Structure mapping for transferability of causal models. *arXiv preprint arXiv:2007.09445*, 2020.

N. Qian. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12(1):145–151, 1999.

G. Raskutti and C. Uhler. Learning directed acyclic graph models based on sparsest permutations. *Stat*, 7(1):e183, 2018.

J. Ruan, Y. Du, X. Xiong, D. Xing, X. Li, L. Meng, H. Zhang, J. Wang, and B. Xu. GCS: Graph-based coordination strategy for multi-agent reinforcement learning. In *International Conference on Autonomous Agents and MultiAgent Systems*, 2022.

K. Sachs, O. Perez, D. Pe'er, D. A. Lauffenburger, and G. P. Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721):523–529, 2005.

R. M. Schmidt, F. Schneider, and P. Hennig. Descending through a crowded valley - benchmarking deep learning optimizers. In *International Conference on Machine Learning*, 2021.

S. Shimizu, P. O. Hoyer, A. Hyvärinen, and A. Kerminen. A linear non-Gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7 (Oct):2003–2030, 2006.

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

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. MIT press, 2nd edition, 2000.

X. Su, C. S. Wijayasinghe, J. Fan, and Y. Zhang. Sparse estimation of Cox proportional hazards models via approximated information criteria. *Biometrics*, 72 (3):751–759, 09 2016.

X. Sun, G. Liu, P. Poupart, and O. Schulte. NTS-NOTEARS: Learning nonparametric temporal DAGs with time-series data and prior knowledge. *arXiv preprint arXiv:2109.04286*, 2021.

S. Van de Geer and P. Bühlmann. ℓ_0-penalized maximum likelihood for sparse directed acyclic graphs. *The Annals of Statistics*, 41(2):536–567, 2013.

Y. Wang, V. Menkovski, H. Wang, X. Du, and M. Pechenizkiy. Causal discovery from incomplete data: A deep learning approach. *arXiv preprint arXiv:2001.05343*, 2020.

Z. Wang, X. Chen, Z. Dong, Q. Dai, and J.-R. Wen. Sequential recommendation with causal behavior discovery. *arXiv preprint arXiv:2204.00216*, 2022.

A. Wehenkel and G. Louppe. Graphical normalizing flows. *arXiv preprint arXiv:2006.02548*, 2020.

D. Wei, T. Gao, and Y. Yu. DAGs with no fears: A closer look at continuous optimization for learning Bayesian networks. In *Advances in Neural Information Processing Systems*, 2020.

M. Yang, F. Liu, Z. Chen, X. Shen, J. Hao, and J. Wang. CausalVAE: Disentangled representation learning via neural structural causal models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021a.

S. Yang, K. Yu, F. Cao, L. Liu, H. Wang, and J. Li. Learning causal representations for robust domain adaptation. *IEEE Transactions on Knowledge and Data Engineering*, 2021b.

Y. Yu, J. Chen, T. Gao, and M. Yu. DAG-GNN: DAG structure learning with graph neural networks. In *International Conference on Machine Learning*, 2019.

Y. Yu, T. Gao, N. Yin, and Q. Ji. DAGs with no curl: An efficient DAG structure learning approach. In *International Conference on Machine Learning*, 2021.

C. Yuan and B. Malone. Learning optimal Bayesian networks: A shortest path perspective. *Journal of Artificial Intelligence Research*, 48(1):23–65, 2013.

Y. Zeng, S. Shimizu, R. Cai, F. Xie, M. Yamamoto, and Z. Hao. Causal discovery with multi-domain LiNGAM for latent factors. In *International Joint Conference on Artificial Intelligence*, 2021.

Y. Zeng, S. Shimizu, H. Matsui, and F. Sun. Causal discovery for linear mixed data. In *Conference on Causal Learning and Reasoning*, 2022.

C. Zhang, B. Jia, M. Edmonds, S.-C. Zhu, and Y. Zhu. ACRE: Abstract causal reasoning beyond covariation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

K. Zhang and A. Hyvärinen. On the identifiability of the post-nonlinear causal model. In *Conference on Uncertainty in Artificial Intelligence*, 2009.

W. Zhang, J. Liao, Y. Zhang, and L. Liu. CMGAN: A generative adversarial network embedded with causal matrix. *Applied Intelligence*, 2022.

X. Zheng. *Learning DAGs with Continuous Optimization*. PhD thesis, Carnegie Mellon University, 2020.

X. Zheng, B. Aragam, P. Ravikumar, and E. P. Xing. DAGs with NO TEARS: Continuous optimization for structure learning. In *Advances in Neural Information Processing Systems*, 2018.

X. Zheng, C. Dan, B. Aragam, P. Ravikumar, and E. P. Xing. Learning sparse nonparametric DAGs. In *International Conference on Artificial Intelligence and Statistics*, 2020.

Supplementary Material: On the Convergence of Continuous Constrained Optimization for Structure Learning

A Optimality Conditions for Equality Constrained Problems

We review the optimality conditions for equality constrained optimization problems, which are required for the study in Sections 2.3 and 3.1. Note that the following conditions are adopted from Theorems 12.1 and 12.6 in Nocedal and Wright (2006), respectively.

Definition 2 (First-order necessary conditions). *Suppose that θ^* is a local solution of (2), that the functions f and h in (2) are continuously differentiable, and that the LICQ holds at θ^*. Define the Lagrangian function of (2) as*

$$L(\theta, \alpha) = f(\theta) + \alpha^\top h(\theta),$$

where $\alpha \in \mathbb{R}^p$. Then there is a Lagrange multiplier vector α^* such that the following conditions are satisfied at (θ^*, α^*):

$$\nabla_\theta L(\theta^*, \alpha^*) = 0, \tag{8a}$$
$$h(\theta^*) = 0. \tag{8b}$$

Definition 3 (Second-order sufficient conditions). *Let $L(\theta, \alpha)$ be the Lagrangian function of (2) as in Definition 2. Suppose that for some feasible point θ^* there is a Lagrange multiplier vector α^* such that the conditions (8) are satisfied. Suppose also that*

$$y^\top \nabla^2_{\theta\theta} L(\theta^*, \alpha^*) y > 0, \quad \text{for all } y \neq 0 \text{ with } \nabla_\theta h(\theta^*) y = 0.$$

Then θ^* is a strict local solution for (2).

B Proofs

B.1 Proof of Theorem 2

Proof for function $h_{\exp}(W)$.

We first provide the proof for function $h_{\exp}(W)$. Its gradient is given by

$$\nabla_W h_{\exp}(W) = (e^{W \circ W})^\top \circ 2W.$$

If part:
We first consider the diagonal entries W_{ii} for $i \in [d]$[4]. Since $\left((e^{W \circ W})^\top\right)_{ii} \geq 1$, $(\nabla_W h_{\exp}(W))_{ii} = 0$ implies that $W_{ii} = 0$, or equivalently, the structure defined by W does not have any self-loop. Now we consider the (j, i)-th entry of W with $j \neq i$. $(\nabla_W h_{\exp}(W))_{ji} = 0$ indicates that at least one of W_{ji} and $\left((e^{W \circ W})^\top\right)_{ji}$ is zero. Therefore, the edge from node j to node i, if exists, must not belong to any cycle. Combining the above cases, we conclude that all edges must not be part of any self-loop or cycle, and that W represents a DAG, i.e., $h_{\exp}(W) = 0$.

Only if part:
Notice that $h_{\exp}(W) = 0$ implies that W represents a DAG. Since there is not any self-loop, the diagonal entries of W are zero, and so are the diagonal entries of $\nabla_W h_{\exp}(W)$. It remains to consider the (j, i)-th entry of $\nabla_W h_{\exp}(W)$ with $j \neq i$:

[4] We denote by $[d]$ the set $\{1, 2, \ldots, d\}$, likewise for the others.

- If $W_{ji} = 0$, then it is clear that $(\nabla_W h_{\exp}(W))_{ji} = 0$.
- If $W_{ji} \neq 0$, then there is an edge from node j to node i with weight W_{ji}. The other term $\left((e^{W \circ W})^\intercal\right)_{ji}$ indicates the total number of weighted walks from node i to node j. If both W_{ji} and $\left((e^{W \circ W})^\intercal\right)_{ji}$ are nonzero, then there is at least a weighted closed walk passing through nodes i and j, contradicting the statement that W represents a DAG.

Therefore, at least one of W_{ji} and $\left((e^{W \circ W})^\intercal\right)_{ji}$ must be zero, and we have $(\nabla_W h_{\exp}(W))_{ji} = 0$.

Proof for function $h_{\exp}(B(A))$.

We now provide the proof for function $h_{\exp}(B(A))$, which is an extension to the proof for function $h_{\exp}(W)$. Let $A^{(t)} = (A_1^{(t)}, \ldots, A_d^{(t)}) \in \mathbb{R}^{d \times s_t \times 1 \times s_t}$ denote the weights in the t-th layer of the MLPs corresponding to all variables. Since $B(A)$ depends only on $A^{(1)}$ by definition, we have

$$\nabla_{A^{(t)}} h_{\exp}(B(A)) = 0, \quad t = 2, \ldots, \ell.$$

Therefore, it suffices to consider the case of $t = 1$, i.e., the weights in the first layer of the MLPs , and show that $h(B(A)) = 0$ if and only if $\nabla_{A^{(1)}} h_{\exp}(B(A)) = 0$. Note that we have $A^{(1)} \in \mathbb{R}^{d \times d \times s_1}$, and the (j, i, s)-entry of the gradient is given by

$$(\nabla_{A^{(1)}} h_{\exp}(B(A)))_{jis} = ((e^{B(A) \circ B(A)})^\intercal)_{ji} \quad 2(A^{(1)})_{jis}, \quad j, i \in [d],\ s \in [s_1].$$

For clarity, we restate the definition of $B(A)$ in terms of $A^{(1)}$:

$$\begin{aligned}(B(A))_{ji} &= \|j\text{th-row}(A_i^{(1)})\|_2 \\ &= \left(\sum_{s=1}^{s_1} ((A^{(1)})_{jis})^2\right)^{\frac{1}{2}}, \quad j, i \in [d]. \end{aligned} \tag{9}$$

If part:
We first consider the diagonal entries $(B(A))_{ii}$ for $i \in [d]$. Since $\left((e^{B(A) \circ B(A)})^\intercal\right)_{ii} \quad 1$, $(\nabla_{A^{(1)}} h_{\exp}(B(A)))_{iis} = 0$ implies that $(A^{(1)})_{iis} = 0$ for $s \in [s_1]$ and therefore $(B(A))_{ii} = 0$, indicating that the structure defined by $B(A)$ does not have any self-loop. Now we consider the (j, i)-th entry of $B(A)$ with $j \neq i$. Suppose $(B(A))_{ji} \neq 0$, i.e., there is an edge from node j to node i in the structure defined by $B(A)$. By Eq. (9), there exists $s \in [s_1]$ such that $(A^{(1)})_{jis} > 0$, which, with the assumption of $(\nabla_{A^{(1)}} h_{\exp}(B(A)))_{jis} = 0$, implies that $\left((e^{B(A) \circ B(A)})^\intercal\right)_{ji} = 0$. This indicates that there is not any weighted walk from node i to node j. Therefore, the edge from node j to node i, if exists, must not belong to any cycle. Combining the above cases, we conclude that all edges must not be part of any self-loop or cycle, and that $B(A)$ represents a DAG, i.e., $h_{\exp}(B(A)) = 0$.

Only if part:
Notice that $h_{\exp}(B(A)) = 0$ implies that $B(A)$ represents a DAG. Since there is not any self-loop, $(B(A))_{ii} = 0$ indicates that $(A^{(1)})_{iis} = 0$ for $s \in [s_1]$, and therefore $(\nabla_{A^{(1)}} h_{\exp}(B(A)))_{iis} = 0$. It remains to consider the (j, i, s)-th entry of $\nabla_{A^{(1)}} h_{\exp}(B(A))$ with $j \neq i$ and $s \in [s_1]$:

- If $(A^{(1)})_{jis} = 0$, then it is clear that $(\nabla_{A^{(1)}} h_{\exp}(B(A)))_{jis} = 0$.
- If $(A^{(1)})_{jis} \neq 0$, then we must have $(B(A))_{ji} \neq 0$ by Eq. (9) and there is an edge from node j to node i in the structure defined by $B(A)$. The other term $\left((e^{B(A) \circ B(A)})^\intercal\right)_{ji}$ indicates the total number of weighted walks from node i to node j. If both $(A^{(1)})_{jis}$ and $\left((e^{B(A) \circ B(A)})^\intercal\right)_{ji}$ are nonzero, there is at least a weighted closed walk passing through nodes i and j, contradicting the statement that $B(A)$ represents a DAG.

Therefore, at least one of $(A^{(1)})_{jis}$ and $\left((e^{B(A) \circ B(A)})^\intercal\right)_{ji}$ must be zero, and we have $(\nabla_{A^{(1)}} h_{\exp}(B(A)))_{jis} = 0$.

B.2 Proof of Theorem 3

By differentiating $Q(\theta; \rho_k)$ in Eq. (6), we obtain

$$\nabla_\theta Q(\theta_k; \rho_k) = \nabla_\theta f(\theta_k) + \rho_k h(B(\theta_k)) \nabla_\theta h(B(\theta_k)).$$

From the termination criterion of the subproblems in Algorithm 2, we have

$$\|\nabla_\theta f(\theta_k) + \rho_k h(B(\theta_k)) \nabla_\theta h(B(\theta_k))\|_2 \leq \tau_k \leq \tau,$$

where τ is an upper bound on $\{\tau_k\}$. By rearranging this expression (and in particular using the inequality $\|a\|_2 \leq \|b\|_2 \leq \|a + b\|_2$), we obtain

$$\|h(B(\theta_k)) \nabla_\theta h(B(\theta_k))\|_2 \leq \frac{1}{\rho_k} (\tau + \|\nabla_\theta f(\theta_k)\|_2).$$

Let θ^* be a limit point of the sequence of iterates. Then there is a subsequence $\mathcal{K}$ such that $\lim_{k \in \mathcal{K}} \theta_k = \theta^*$. By the continuity of $\nabla_\theta f$, when we take limits as $k \to \infty$ for $k \in \mathcal{K}$, the bracketed term on the right-hand-side approaches $\tau + \|\nabla_\theta f(\theta^*)\|_2$, so because $\rho_k \to \infty$, the right-hand-side approaches zero. This implies that $\lim_{k \in \mathcal{K}} h(B(\theta_k)) \nabla_\theta h(B(\theta_k))\|_2 = 0$. By the continuity of h, $\nabla_\theta h$, and B, we conclude that

$$h(B(\theta^*)) \nabla_\theta h(B(\theta^*)) = 0,$$

which, by Assumption 1, yields

$$h(B(\theta^*)) = 0.$$

B.3 Proof of Theorem 4

The gradient of $h_{\mathsf{bf}}(W, \Omega)$ is given by

$$\begin{aligned} \nabla_W h_{\mathsf{bf}}(W, \Omega) &= (e^{W \circ W})^\intercal \circ 2W \circ \Omega, \\ \nabla_\Omega h_{\mathsf{bf}}(W, \Omega) &= 2W \circ W. \end{aligned}$$

If part:
The gradient term $\nabla_\Omega h_{\mathsf{bf}}(W, \Omega) = 2W \circ W \circ \Omega = 0$ indicates that $W_{ji}\Omega_{ji} = 0$ for $j, i \in [d]$, and thus there is not any bow in the structure defined by W and Ω. Therefore, we also have $2W \circ \Omega \circ \Omega = 0$, which, with $\nabla_W h_{\mathsf{bf}}(W, \Omega) = 0$, implies that $(e^{W \circ W})^\intercal \circ 2W = 0$. By Theorem 2, this indicates that there is not any directed cycle. Therefore, the structure defined by W and Ω is a bow-free ADMG and we have $h_{\mathsf{bf}}(W, \Omega) = 0$.

Only if part:
Notice that $h_{\mathsf{bf}}(W, \Omega) = 0$ implies that the structure defined by W and Ω is a bow-free ADMG. Since there is not any directed cycle in an ADMG, we have $(e^{W \circ W})^\intercal \circ 2W = 0$ by Theorem 2. Furthermore, since there is not any bow in the structure, we must have $W_{ji}\Omega_{ji} = 0$ for $j, i \in [d]$, which implies $2W \circ \Omega \circ \Omega = 0$ and $2W \circ W \circ \Omega = 0$. Therefore, we have $\nabla_W h_{\mathsf{bf}}(W, \Omega) = 0$ and $\nabla_\Omega h_{\mathsf{bf}}(W, \Omega) = 0$.

C Bivariate Example: Ill-Conditioning and Optimization Algorithms

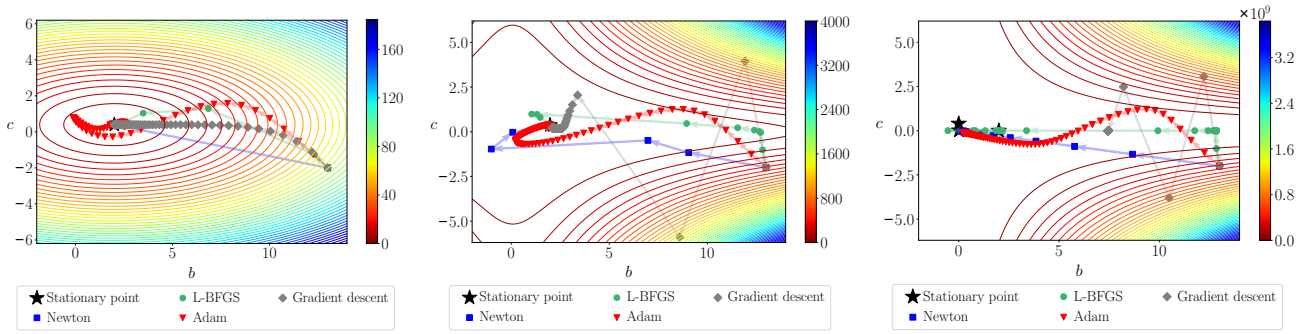
We provide a bivariate example to illustrate the ill-conditioning issue and the effectiveness of different optimization algorithms. Here we focus on the asymptotic case in which the true covariance matrix is known.

Setup. Consider the bivariate linear Gaussian model defined by the weighted adjacency matrix

$$W_0 = \begin{bmatrix} 0 & 2 \\ 0 & 0 \end{bmatrix}$$

and noise covariance matrix

$$\Omega_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$



(a) $\rho = 0$. (b) $\rho = 1$. (c) $\rho = 10^6$.

Figure 4: Contour plots of the quadratic penalty function $Q(W; \rho)$ with different penalty coefficients. The optimization trajectories of different algorithms for minimizing $Q(W; \rho)$ are visualized.

In other words, the additive noises are assumed to be standard Gaussians. The true covariance matrix of X is then given by

$$\begin{aligned}\Sigma_0 &= (I \quad W_0)^{-\intercal} \Omega_0 (I \quad W_0)^{-1} \\ &= \begin{bmatrix} 1 & 2 \\ 2 & 5 \end{bmatrix}.\end{aligned}$$

We define the matrix W with variables b and c as

$$W = \begin{bmatrix} 0 & b \\ c & 0 \end{bmatrix},$$

which yields the corresponding least squares objective

$$\begin{aligned}f(W) &= \frac{1}{2} \operatorname{tr}\left((I \quad W)^{\intercal} \Sigma_0 (I \quad W)\right) \\ &= \frac{1}{2}\left((b \quad 2)^2 + (2c \quad 1)^2 + c^2 + 1\right).\end{aligned}$$

We consider a simplified DAG constraint $bc = 0$, which leads to the quadratic penalty function

$$Q(W; \rho) = f(W) + \frac{\rho}{2} b^2 c^2.$$

Clearly, W represents a DAG if and only if $bc = 0$.

Contour. The contour plots of $Q(W; \rho)$ w.r.t. variables b and c with different penalty coefficients, i.e., $\rho = 0, 1, 10^6$, are visualized in Figure 4. When $\rho = 0$, the contour is elliptical as the least squares objective corresponds to a quadratic function. With a small penalty coefficient $\rho = 1$, the contour is slightly stretched out and deviates from being elliptical. When a large coefficient $\rho = 10^6$ is used, the function contour is highly stretched out along both x and y axes. In this case, when the values of b and c are moderately large, a small change in either of them can lead to a large change in the penalty function $Q(W; \rho)$.

Optimization algorithms. We further study the behavior of different optimization algorithms for solving the minimization problem of $Q(W; \rho)$. In particular, we visualize the trajectories of Newton's method, L-BFGS (Byrd et al., 2003), Adam (Kingma and Ba, 2014), and gradient descent starting from the point $(b, c) = (13, \quad 2)$. We pick an initial point that is relatively far from the origin for better illustration of the ill-conditioning issue.

The optimization trajectories and the stationary points for different penalty coefficients are visualized in Figure 4. When $\rho = 0, 1$, all four algorithms converge to the unique stationary point. With a small penalty coefficient $\rho = 1$, one observes that gradient descent moves along a zigzag path during the first few iterations. A possible reason is that the gradients may not be the best direction to descend to the stationary point. This phenomenon becomes more severe when $\rho = 10^6$ is used; specifically, gradient descent follows a zigzag path and eventually

reaches a point where it no longer makes progress, owing to the ill-posed optimization landscape. In this case, Newton's method and Adam converge to the same stationary point, while L-BFGS converges to the other one. It is also observed that Newton's method converges in the fewest number of iterations in all cases, and L-BFGS is on par with it as compared to Adam and gradient descent.

The observations above demonstrate that ill-conditioning is a huge concern especially for first-order method like gradient descent, which may produce a bad solution or not converge well. They also corroborate our findings in Section 6.2, i.e, second-order method like Newton's method and L-BFGS helps resolve the ill-conditioning issue by incorporating the curvature information. In particular, Newton's method computes the direction of descent using both first-order (Jacobian) and second-order (Hessian) information, while, instead of explicitly evaluating the Hessian matrix, L-BFGS relies on the approximations of Hessian. Therefore, Newton's method is computationally less efficient in practice, which thus is not included in the experiments in Section 6.2. On the other hand, the Adam algorithm, loosely speaking, lies in the middle between first-order and second-order methods as it employs diagonal rescaling on the parameter space that can be interpreted as second-order-type information (Bottou et al., 2018). Therefore, it can be less susceptible to ill-conditioning as compared to gradient descent, but may require more iterations to converge than Newton's method and L-BFGS.

In the empirical study above, we find that gradient descent is sensitive to the choice of learning rate (or referred to as the step size). Therefore, we manually pick it such that the algorithm does not diverge or converge too slowly. For the other algorithms, they are relatively robust to the choice of learning rate.

D Supplementary Experiment Details

This section provides further experiment details of different optimization algorithms and structure learning methods for Section 6.

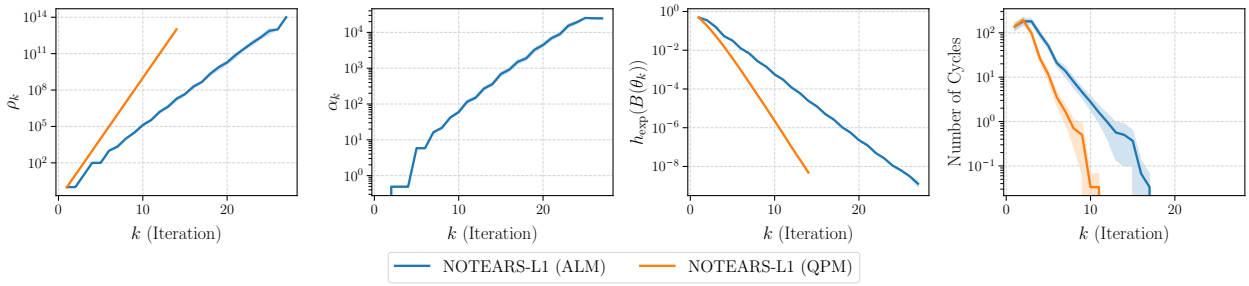
Optimization algorithms. We implement the optimization algorithms including gradient descent with momentum, NAG, and Adam with PyTorch (Paszke et al., 2019). For Adam, we use a learning rate of 10^{-3}. For gradient descent with momentum and NAG, we set the learning rate to 10^{-4} and the momentum factor to 0.9. We set the number of optimization iterations to 10^4 for all these algorithms. We use the implementation and default hyperparameters of L-BFGS released by Zheng et al. (2018, 2020).

Structure learning methods. The implementations of Abs-KKTS[5], NoCurl[6], and GOLEM-EV[7] are available on the authors' GitHub repositories. For all these methods, we use the default hyperparameters and the DAG constraint term proposed by Zheng et al. (2018), i.e., $h_{\mathsf{exp}}(W)$, although some of the authors' original implementations adopt the polynomial alternative proposed by Yu et al. (2019), i.e., $h_{\mathsf{poly}}(W)$. Similar to NOTEARS, we use a pre-processing step to center the data by subtracting the mean of each variable from the samples $\mathbf{X}$.

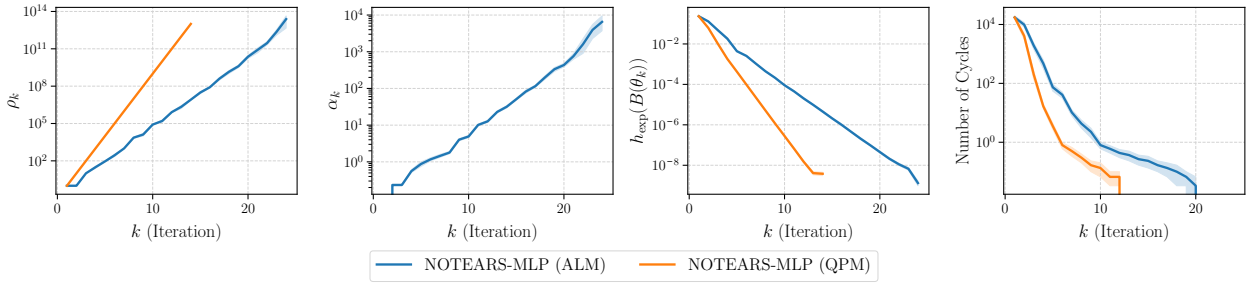
E Supplementary Experiment Results

This section provides further experiment results for Section 6; see Figures 5, 6, 7, 8, 9, 10, and 11.

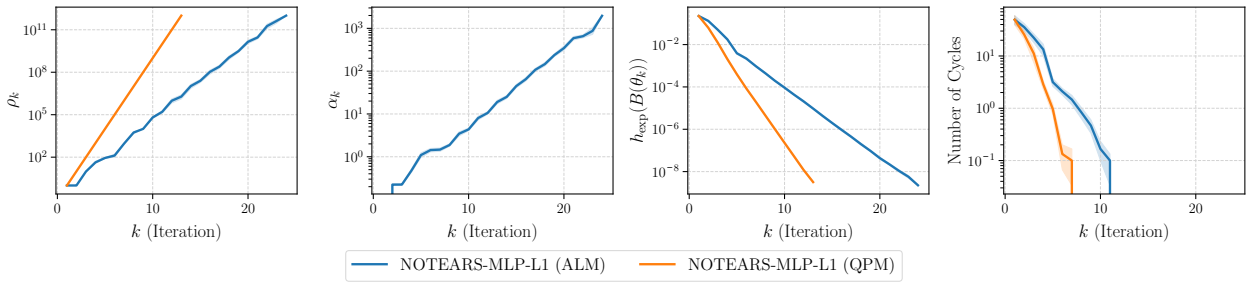
[5] https://github.com/skypea/DAG_No_Fear
[6] https://github.com/fishmoon1234/DAG-NoCurl
[7] https://github.com/ignavierng/golem



(a) NOTEARS-L1.

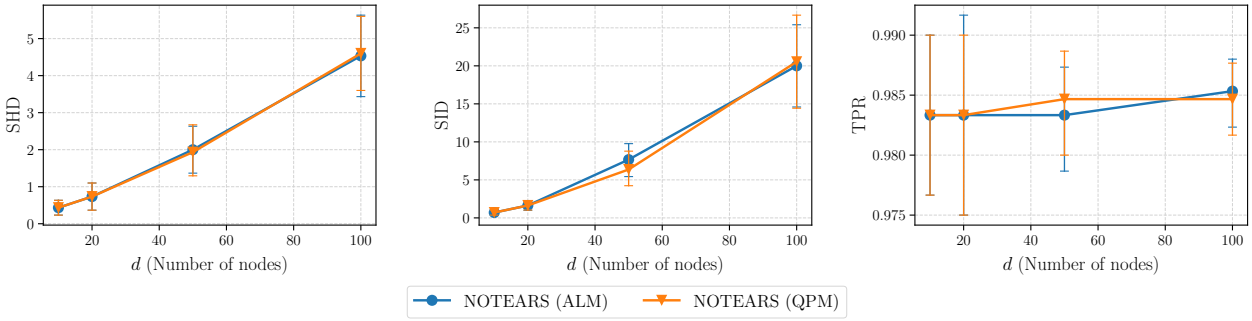


(b) NOTEARS-MLP.

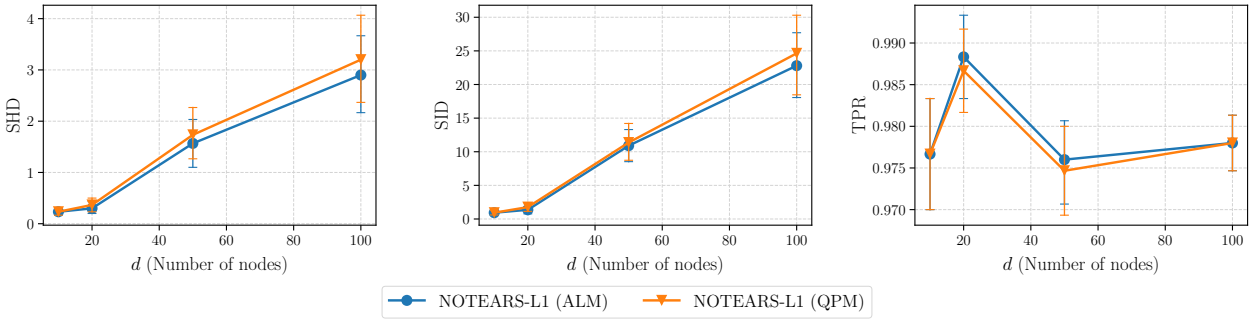


(c) NOTEARS-MLP-L1.

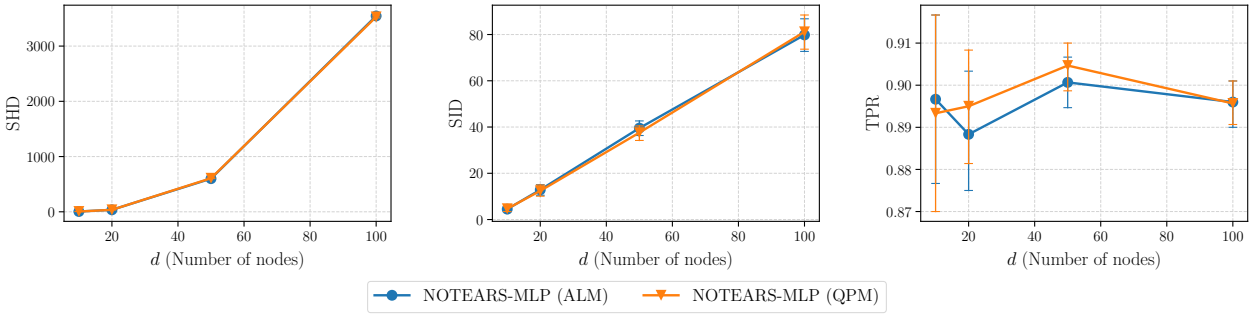
Figure 5: Optimization processes of the continuous constrained formulation using ALM and QPM on synthetic data. The ground truths are 10-node ER1 graphs and the sample size is $n = 1000$. Each data point corresponds to the k-th iteration. Shaded area denotes standard errors over 30 trials.



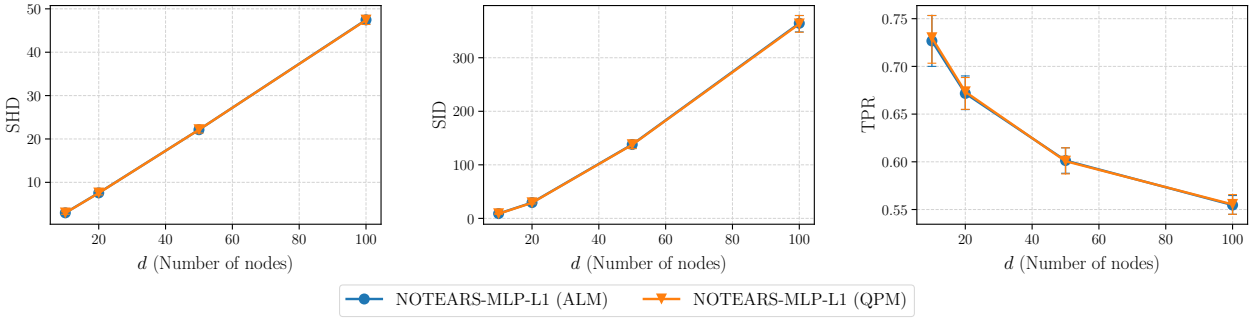
(a) NOTEARS.



(b) NOTEARS-L1.

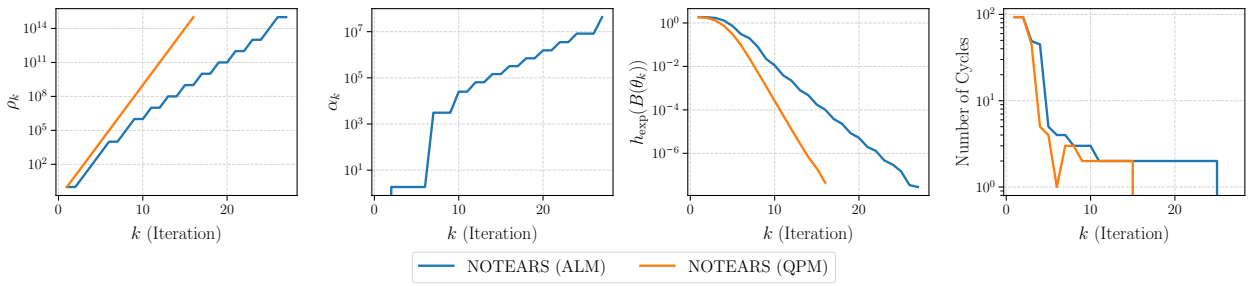


(c) NOTEARS-MLP.

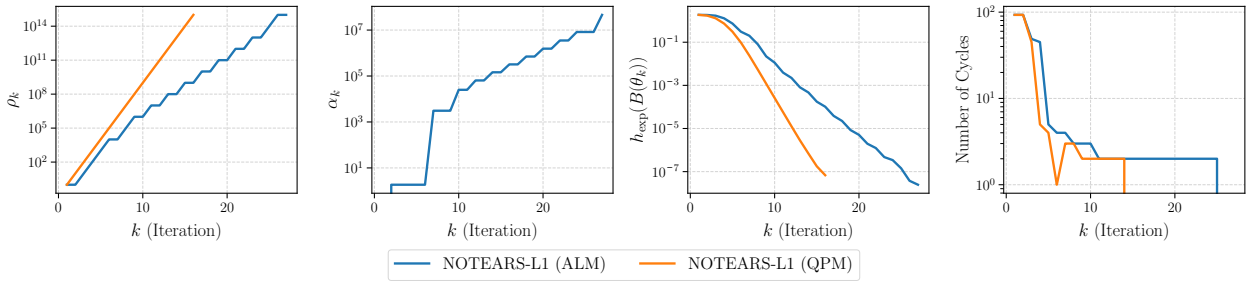


(d) NOTEARS-MLP-L1.

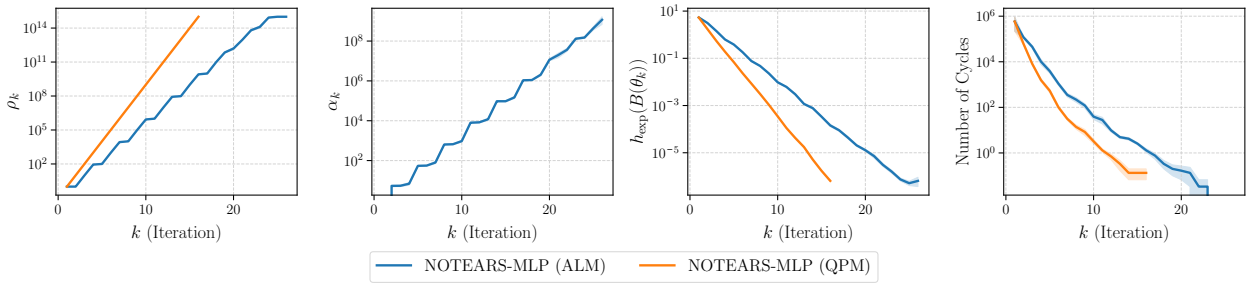
Figure 6: Empirical results of the continuous constrained formulation using ALM and QPM on synthetic data. The ground truths are ER1 graphs and the sample size is $n = 1000$. Lower is better, except for TPR. Error bars denote standard errors over 30 trials.



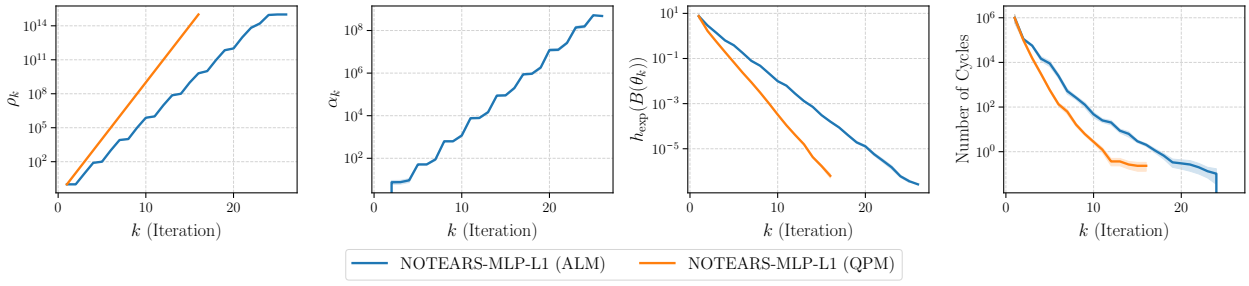
(a) NOTEARS.



(b) NOTEARS-L1.

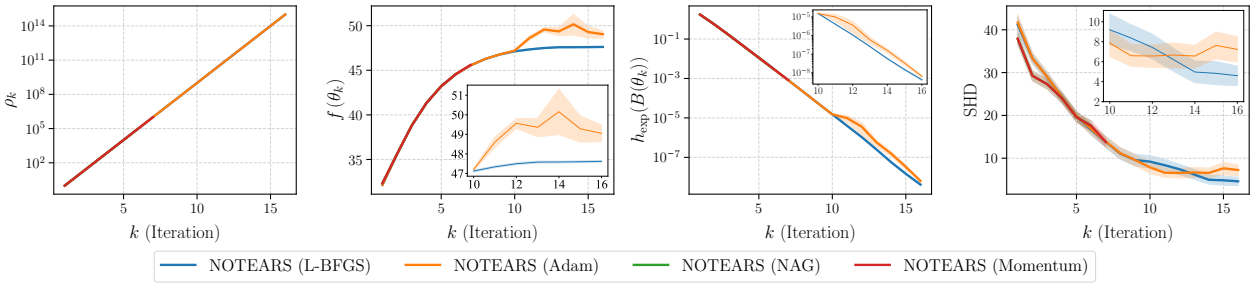


(c) NOTEARS-MLP.

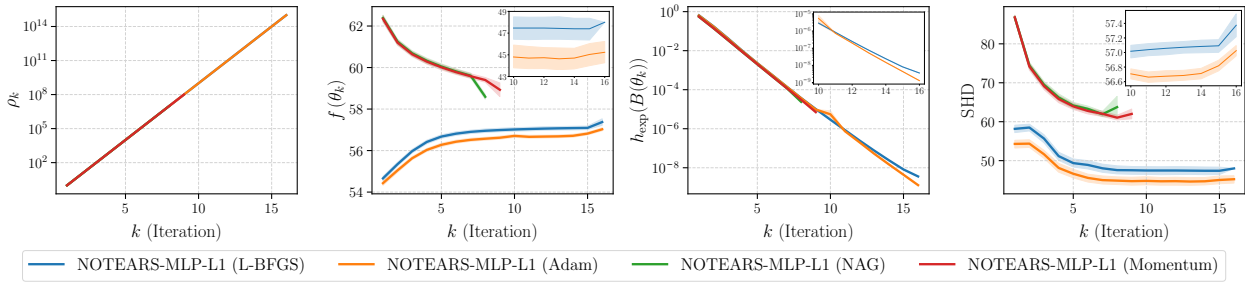


(d) NOTEARS-MLP-L1.

Figure 7: Optimization processes of the continuous constrained formulation using ALM and QPM on a real dataset by Sachs et al. (2005). Each data point corresponds to the k-th iteration. Shaded area denotes standard errors over 30 trials.

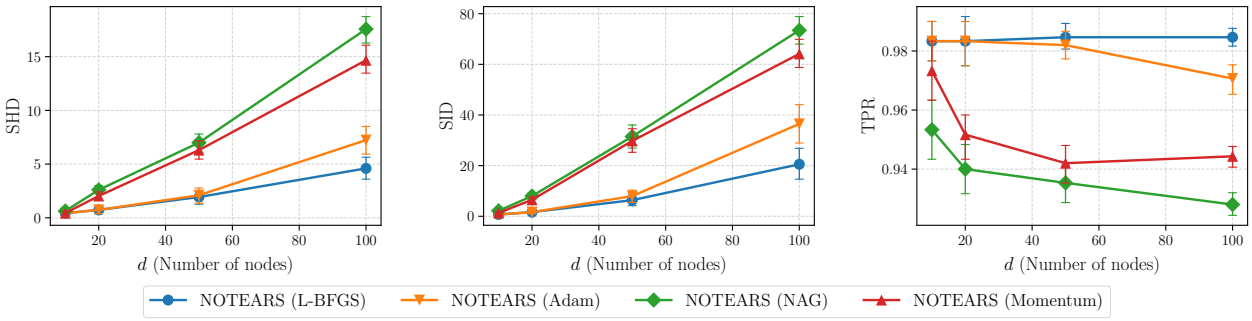


(a) NOTEARS.

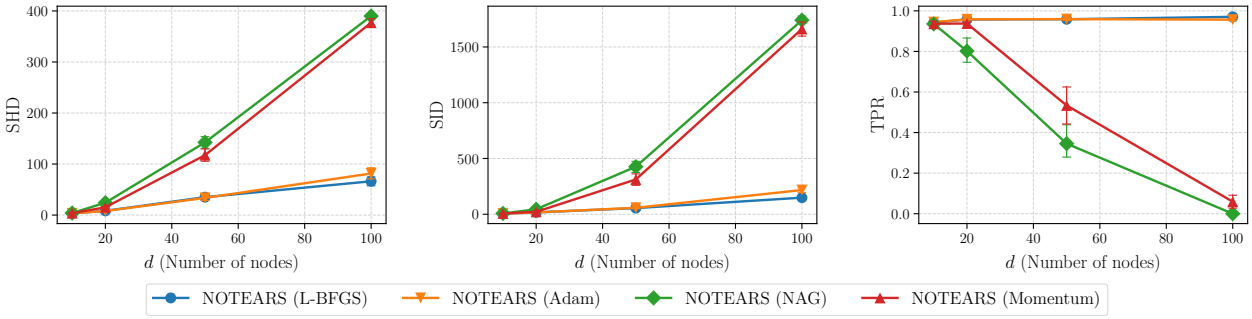


(b) NOTEARS-MLP-L1.

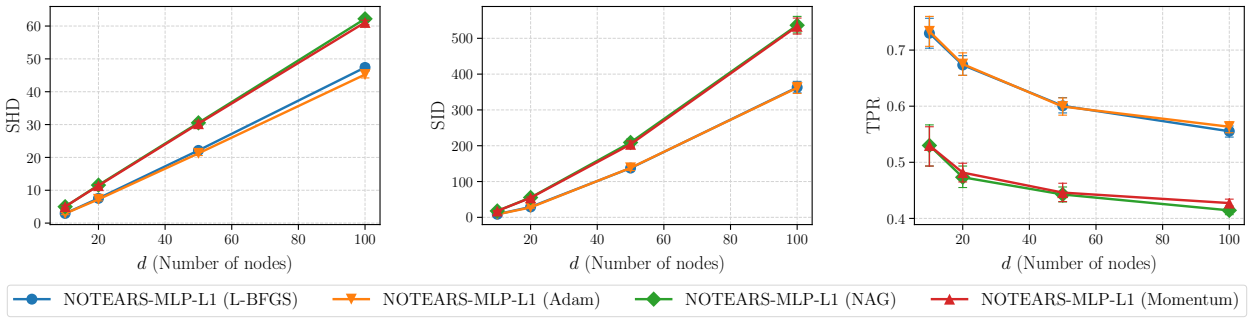
Figure 8: Optimization processes of different optimization algorithms for solving the QPM subproblems of NOTEARS and NOTEARS-MLP-L1 on synthetic data. The ground truths are 100-node ER1 graphs and the sample size is $n = 1000$. Each data point corresponds to the k-th iteration. Shaded area denotes standard errors over 30 trials. The blue line overlaps with the orange line in the first panel.



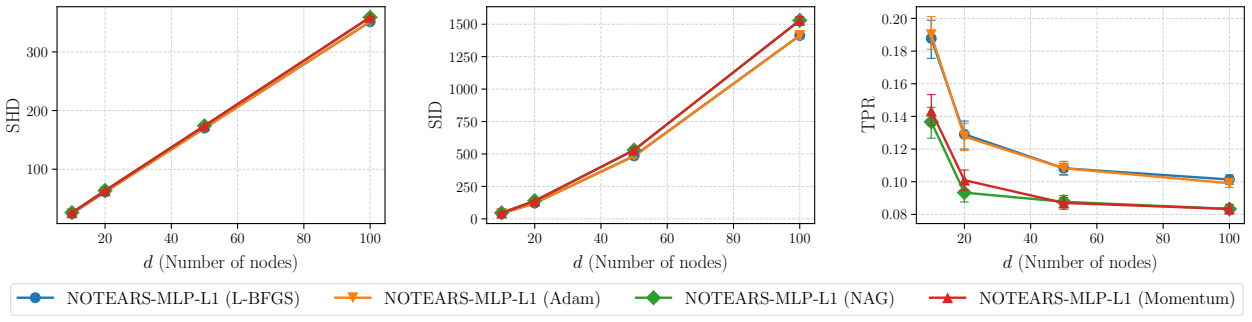
(a) NOTEARS with ER1 graphs.



(b) NOTEARS with SF4 graphs.

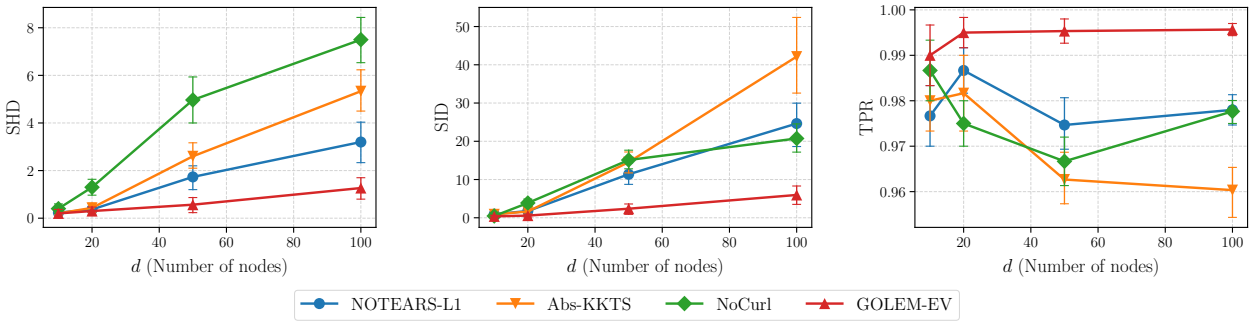


(c) NOTEARS-MLP-L1 with ER1 graphs.

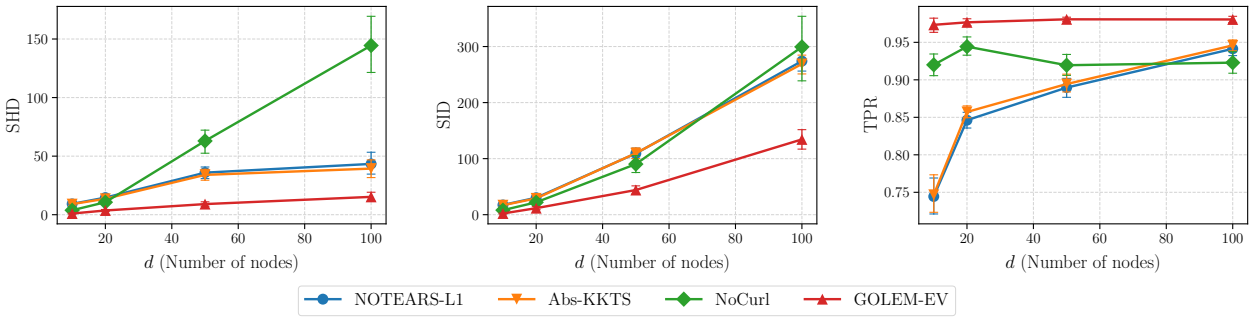


(d) NOTEARS-MLP-L1 with SF4 graphs.

Figure 9: Empirical results of different optimization algorithms for solving the QPM subproblems of NOTEARS and NOTEARS-MLP-L1 on synthetic data. The sample size is $n = 1000$. Lower is better, except for TPR. Error bars denote standard errors over 30 trials.

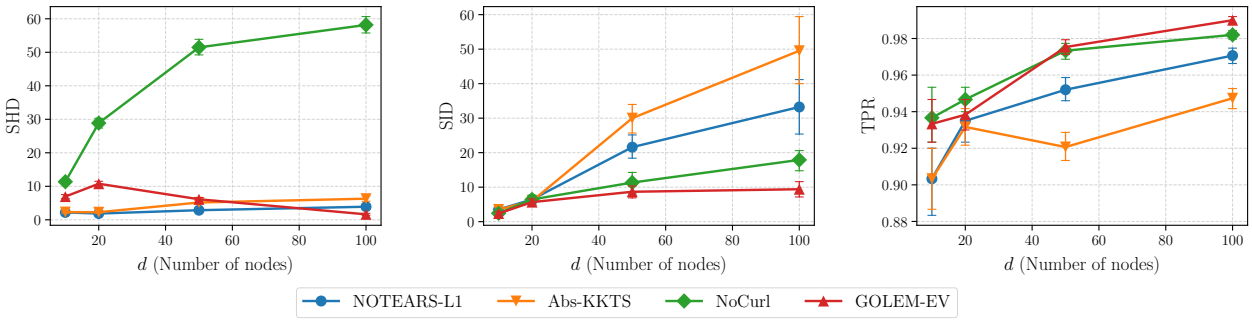


(a) ER1 graphs.

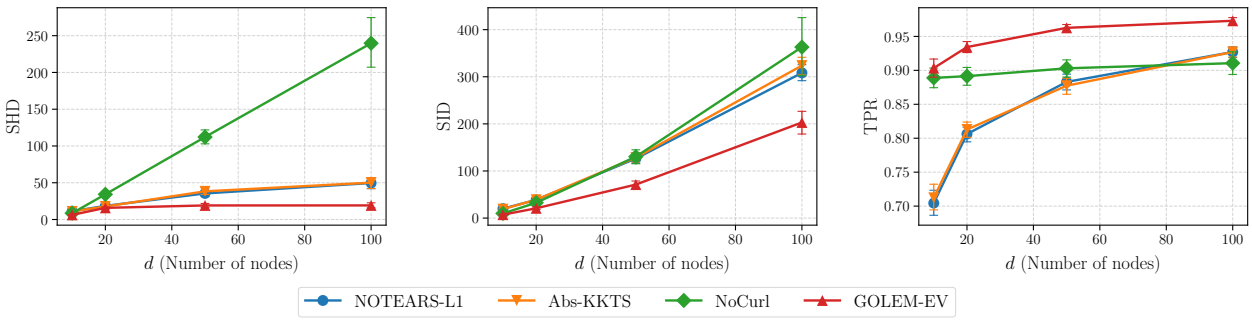


(b) SF4 graphs.

Figure 10: Empirical results of different structure learning methods on synthetic data with sample size $n = 1000$. Lower is better, except for TPR. Error bars denote standard errors over 30 trials.



(a) ER1 graphs.



(b) SF4 graphs.

Figure 11: Empirical results of different structure learning methods on synthetic data with sample size $n = 3d$. Lower is better, except for TPR. Error bars denote standard errors over 30 trials.