UNIVERSITY OF CALIFORNIA

Los Angeles

Order-based Learning of Bayesian Networks: Regularized Cholesky Score and Distributed Data

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Statistics

by

Qiaoling Ye

2021

ABSTRACT OF THE DISSERTATION

Order-based Learning of Bayesian Networks:
Regularized Cholesky Score and Distributed Data

by

Qiaoling Ye
Doctor of Philosophy in Statistics
University of California, Los Angeles, 2021
Professor Arash Ali Amini, Co-chair
Professor Qing Zhou, Chair

Bayesian networks are a popular class of graphical models to encode conditional independence and causal relations among variables by *directed acyclic graphs* (DAGs). In this thesis, we focus on developing algorithms to estimate Bayesian network structures. We propose two structure learning methods, and both of them minimize regularized negative log-likelihood functions over the space of orderings.

First, we propose the *annealing on regularized Cholesky score* (ARCS) algorithm to learn Gaussian Bayesian networks. The scoring function of ARCS is derived from regularizing the Gaussian DAG likelihood, and its optimization is an alternative form of the sparse Cholesky decomposition, which depends on the choice of permutation (matrix) P. For this reason, we name our objective function the *regularized Cholesky* (RC) score of permutations. Essentially, minimizing the RC score is a joint optimization over a permutation P and a lower triangular matrix L, because the acyclic constraint of DAGs has been translated into a strictly lower triangular matrix given a permutation. ARCS uses simulated annealing to search over the permutation space and an effective first order method, called the proximal gradient algorithm, to compute the optimal DAG that is compatible with P. Combined, the two approaches allow us to quickly and effectively search over the space of DAGs without the need to verify the acyclicity constraint or to enumerate possible parent sets given a candidate topological sort. The annealing aspect of the optimization is able to consistently improve the accuracy of DAGs learned by greedy and deterministic search algorithms. Through extensive numerical tests, ARCS has demonstrated high structure learning accuracy and outperformed existing methods by a great margin when using observational and experimental data to learn Gaussian DAGs. As a byproduct, ARCS can accurately estimate Gaussian covariance matrix, and it has achieved higher test likelihood than other covariance estimation methods. In terms of theoretical results, we establish the consistency of our RC score in estimating topological sorts and DAG structures in the large-sample limit.

The second method we propose is the *distributed annealing on regularized likelihood score* (DARLS) algorithm, which generalizes the ARCS algorithm to learn a flexible family of DAGs from distributed data. To the best of our knowledge, it is the first method that uses distributed optimization to learn causal structures from data stored over different machines. DARLS searches over the space of topological sorts with simulated annealing strategy for a high-scoring causal graph, where the optimal graphical structure compatible with a sort is found by a distributed optimization method. We show that the estimate sequence generated by the distributed optimization method converges to a global optimizer of the overall score computed on all data across local machines. Additionally, we propose generalized linear DAG models where the conditional distributions of a Bayesian network is given by generalized linear models (GLMs) with canonical links. GLMs is a flexible family of distributions that take various types of data, and thus the use of it greatly increase the applicability of our DAG models. In our simulation studies, DARLS has demonstrated competing performance with distributed data against other existing methods using the overall data across local machines. It also exhibits higher predictive power than other methods in a real-world application for modeling protein-DNA binding networks using ChIP-Sequencing data.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

VITA

2012–2014	B.S., Department of Mathematics, Major in Applied Mathematics, University of California, Los Angeles.
2015–present	Expected Ph.D., Major in Statistics, University of California, Los Angeles.

PUBLICATIONS

Ye, Q., Amini, A., and Zhou, Q. (2020). Optimizing Regularized Cholesky Score for Order-Based Learning of Bayesian Networks. *IEEE Transactions on Pattern Analysis & Machine Intelligence, early access*, April 27 2020. DOI: 10.1109/TPAMI.2020.2990820.

Ye, Q., Amini, A., and Zhou, Q. (2021). Distributed Learning of Generalized Linear Causal Networks. *Manuscript under review.*

CHAPTER 1

Introduction

Bayesian networks are a class of graphical models, whose structure is represented by a directed acyclic graph. They are commonly used to model causal networks and conditional independence relations among random variables. The past decades have seen many successful applications of Bayesian networks in computational biology, social science, medical science, document classification, image processing, etc. We start this dissertation with some background of Bayesian networks and our contributions in learning Bayesian networks.

1.1 Background of Bayesian networks

A *Bayesian network* (BN) for a set of variables $\{X_1, \ldots, X_p\}$ consists of 1) a directed acyclic graph (DAG) $\mathcal{G}$ that encodes a set of conditional independence assertions among the variables, and 2) a set of local probability distributions associated with each variable. It can be considered as a recipe for factorizing a joint distribution of $\{X_1, \ldots, X_p\}$ with probability density

$$p(x_1, \ldots, x_p) = \prod_{j=1}^{p} p(x_j \mid \mathrm{PA}_j = pa_j), \tag{1.1}$$

where $\mathrm{PA}_j \subset \{X_1, \ldots, X_p\} \setminus \{X_j\}$ is the parent set of variable X_j and pa_j its value. The DAG $\mathcal{G}$ is denoted by $\mathcal{G} = (V, E)$, where $V = \{1, \ldots, p\}$ is the vertex set corresponding to the set of random variables and $E = \{(i, j) : i \in \mathrm{PA}_j\} \subset V \times V$ is the edge set. We use variable X_j and node j exchangeably throughout the thesis. DAGs contain no directed cycles, making the joint distribution in (1.1) well-defined.

Given a permutation π on $[p] := \{1, \ldots, p\}$, we permute a vector $v = (v_1, \ldots, v_p)$ according to π to obtain a relabeled vector $v_\pi = \big(v_{\pi(1)}, \ldots, v_{\pi(p)}\big)$. A *topological sort* of a DAG is a permutation of nodes such that if $a \in \mathrm{PA}_b$, then a precedes b in the order defined by π, denoted by $a \prec_\pi b$. By definition (1.1), every DAG has at least one topological sort.

A (directed) local Markov property (Lauritzen, 2004) of a BN (1.1) is commonly used to detect conditional independence among its variables. A joint probability distribution P with respect to a DAG obeys the local Markov property if any variable is conditionally independent of its non-descendants given its parents, that is,

$$X_i \perp\!\!\!\perp \mathrm{nd}(X_i) \mid \mathrm{PA}_i, \forall i \in [p], \tag{1.2}$$

where $\mathrm{nd}(X_i) \subset V$ denotes the set of non-descendants of X_i. A graphical test to detect conditional independence is through d-separation (Pearl, 1995).

Definition 1. *(d-separation). Let $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{Z}$ be disjoint subsets of nodes in a DAG $\mathcal{G}$. We say $\boldsymbol{A}$ and $\boldsymbol{B}$ are d-separated by $\boldsymbol{Z}$, if all paths between any node in $\boldsymbol{A}$ and $\boldsymbol{B}$ are blocked by $\boldsymbol{Z}$. A path is blocked by $\boldsymbol{Z}$ if there are nodes a, b, c on the path satisfying one of the following two conditions:*

1. *c is not a collider for $c \in \boldsymbol{Z}$, i.e., $a \to c \to b$ or $a \leftarrow c \leftarrow b$ or $a \leftarrow c \to b$.*
2. *c is a collider, i.e., $a \to c \leftarrow b$, and neither c nor its descendants are in $\boldsymbol{Z}$.*

Theorem 1. *(Verma and Pearl, 1988). Let $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{Z}$ be disjoint subsets of nodes in a DAG $\mathcal{G}$, and $\mathcal{G}$ generates a distribution P. If $\boldsymbol{A}$ and $\boldsymbol{B}$ are d-separated by $\boldsymbol{Z}$ in $\mathcal{G}$, then $\boldsymbol{A}$ and $\boldsymbol{B}$ are conditional independent given $\boldsymbol{Z}$ in distribution P.*

We write $\mathcal{I}(\mathcal{G})$ for the set of conditional independence (CI) statements implied by DAG $\mathcal{G}$, in the sense of d-separation. Two DAGs $\mathcal{G}_1$ and $\mathcal{G}_2$ are *Markov equivalent* if $\mathcal{I}(\mathcal{G}_1) = \mathcal{I}(\mathcal{G}_2)$. Similarly, we write $\mathcal{I}(P)$ for CIs that hold in distribution P. Theorem 1 implies that $\mathcal{I}(\mathcal{G}) \subset \mathcal{I}(P)$, i.e., $\mathcal{G}$ is an I-map for P. DAG $\mathcal{G}$ is *perfect* for (or *faithful* to) P if $\mathcal{I}(\mathcal{G}) = \mathcal{I}(P)$. i.e., the set of triples (X, Y, Z) that satisfy the d-separation criterion in $\mathcal{G}$ is one-to-one correspond to the set of conditional independencies $X \perp\!\!\!\perp Y \mid Z$ implied by the recursive decomposition of distribution (1.1).

1.2 Structure learning

As the relationships among variables in a BN are encoded in the underlying graph, it is an important task to estimate DAG structures from data. There are three main challenges in learning DAG structures from data. First, the number of DAGs grows super-exponentially in the number of nodes p. There is a total of $\binom{p}{2}$ pairs of nodes and each pair can be either connected or disconnected, so the number of DAGs with p variables, denoted by a_p, is lower bounded by $2^{\binom{p}{2}}$. The exact number is found by (Robinson, 1977) such that $a_p = \sum_{i=1}^{p}(-1)^{i+1}\binom{p}{i}2^{i(p-i)}a_{p-i}$ with an initial condition $a_0 = 1$. The second difficulty lies in the acyclicity constraint of DAGs, which requires careful design of the BN learning framework and imposes additional computational burdens in algorithms. Third, it is well-known that observational data can only recover DAGs up to an equivalence class, a set of BNs that have same CI statements, under common DAG models, including Gaussian and multinomial DAG models. The size of equivalence classes also grows super-exponentially with respect to p. Despite these difficulties, various methods have been put forward to estimate DAG structures from data, which can be categorized into three main algorithmic approaches.

Constraint-based methods. In constraint-based approaches, a set of CI tests is performed to detect the existence of edges. The CI tests usually rely on the following statement: if a DAG $\mathcal{G}$ is faithful to a distribution P, then there is no edge between X and Y if and only if there exists $Z \subset V \setminus \{X, Y\}$ such that X and Y are d-separated by Z. PC algorithm (Spirtes and Glymour, 1991) is a well-known example in this approach. It first estimates the skeleton of a DAG using a set of CI tests, and then identifies v-structures in the skeleton and finally orients the remaining edges such that no new CIs and no cycles are introduced. The constraint-based methods are often efficient and accurate in estimating graphs, especially when sample sizes are sufficiently big. However, they may require long computational time to learn large networks due to the size of CI tests to be performed.

Score-based methods. A network structure is identified by optimizing a score function through a certain searching strategy in this approach. The most common score function is based on the posteriori distribution of a network (Heckerman et al, 1995). Minimizing Bayesian information criterion has also been used as well, and it is equivalent to optimizing the minimum-description length score (Suzuki, 1993). The score-based search has been applied to three different search spaces: the DAG space (Heckerman et al, 1995; Gámez et al, 2011), the equivalence classes (Chickering, 2002; Heckerman et al, 1995) and the ordering space (or the space of topological sorts) (Larrañaga et al, 1996; Teyssier and Koller, 2005). Greedy search is a popular optimization technique, and it has been used by the greedy hill climbing (HC) algorithm (Gámez et al, 2011) over the DAG space and the greedy equivalence search (GES) algorithm (Chickering, 2002) over the equivalence class. Heuristic search strategies with Monte Carlo samples has been used to search over the topological sort space (Ellis and Wong, 2008; Zhou, 2011). Methods of the score-based approach usually reply on a pre-specified maximum incoming degree or additional tuning parameters to avoid fully-connected estimates. Similar to the previous approach, score-based methods become computationally impractical for large networks due to the size of the search space.

Hybrid methods. Hybrid approaches are proposed to integrate the merits of constraint-based and score-based methods, where a typical strategy is using a set CI tests to prune the search space, followed by a search for a high-scoring network structure. By removing as many edges as possible in the first step, the second step becomes faster than the unrestricted score-based methods. Max-min hill-climbing algorithm (Tsamardinos et al, 2006) is a well-known method of this kind, which uses the MMPC algorithm in the first step and the HC algorithm in the second step.

In this thesis, we focus on developing algorithms using the score-based approach. Our proposed algorithms learn DAG structures by searching over the topological sort space, and they are designed to handle the case where data is split over different machines, a common situation to store a large amount of data nowadays. In the remaining sections of this chapter, we discuss our order-based search and distribute learning of DAGs.

1.3 Order-based search

Searching over the topological sort to optimize scoring functions of DAGs has two major advantages. First, the existence of an ordering among nodes guarantees a graph structure that satisfies the acyclicity constraint. Second, the space of orderings is significantly smaller than the space of DAGs or of the equivalence classes. Consequently, several lines of research have developed efficient order-based methods for DAG learning. Some methods adopt a greedy search in conjunction with various operators that propose moves in the ordering space (Teyssier and Koller, 2005; Alonso-Barba et al, 2011; Scanagatta et al, 2015, 2017). A greedy search, however, may easily be trapped in a local minimum, and thus different techniques were proposed to perform a more global search (Silander and Myllymäki, 2006; Larrañaga et al, 1996; Bartlett and Cussens, 2013; Lee and van Beek, 2017). In particular, stochastic optimization, such as the genetic algorithm (Larrañaga et al, 1996; Champion et al, 2018; Scanagatta et al, 2017) and Markov chain Monte Carlo (Friedman and Koller, 2003; Ellis and Wong, 2008; Zhou, 2011), has been advocated as a promising way to perform global search over the ordering space. Under certain identifiability assumptions, sequential order search algorithms have been developed recently as well Ghoshal and Honorio (2018).

In spite of these methodological and algorithmic advances, there are a few difficulties in score-based learning of topological sorts for DAGs. First, the score of an ordering is usually defined by the score of the optimal DAG compatible with the ordering. The computational complexity of finding the optimal DAG given an ordering, typically by enumerating all possible parent sets for each node (Cooper and Herskovits, 1992), can be as high as $O(p^{k+1})$ for p nodes and a pre-specified maximum indegree of k. Such computation is needed for every ordering evaluated by a search algorithm, which becomes prohibitive when k is large. Second, although the ordering space is smaller than the graph space, optimization over orderings is still a hard combinatorial problem due to the NP-hard nature of structure learning of BNs (Chickering, 1996). It is not surprising that the performance of the above stochastic optimization algorithms degrades severely for large graphs.

Motivated by these challenges, we develop a new order-based method, annealing on regularized Cholesky score (ARCS), for learning Gaussian DAGs by optimizing a regularized likelihood score. Representing an ordering by the corresponding permutation matrix P, the weighted adjacency matrix of a Gaussian DAG can be coded into a lower triangular matrix L. We add a continuous and concave penalty function to the likelihood to encourage sparsity in L, and thus achieve the goal of structure learning. Instead of a prespecified maximum indegree, which is *ad hoc* in nature, we provide a principled data-driven way to determine the tuning parameters for the penalty function. Finding the optimal DAG given P is then reduced to p decoupled penalized regression problems, which are solved by proximal gradient, an efficient first-order method, without enumerating possible parent sets for any node. Searching over P is done by simulated annealing (SA). We may also incorporate informative initial orderings, learned by an existing method, by setting a low starting temperature. We also propose a constraint-based refinement step to removes false positive edges after annealing. Our numerical results demonstrate that this combined strategy substantially improves the accuracy of an estimated DAG. We note an interesting connection between our formulation and the sparse Cholesky factorization problem, and thus name our scoring function the *regularized Cholesky score* of orderings or permutations.

Regularizing likelihood with a continuous penalty function has been shown to be effective in learning Gaussian DAGs (Fu and Zhou, 2013; Aragam and Zhou, 2015; Zheng et al, 2018). These methods optimize a regularized likelihood score over the DAG space by continuous optimization. They are likely to be trapped in a suboptimal structure due to the nonconvexity of the DAG parameter space. Using DAGs learned by such methods to generate initial orderings, our method can significantly improve the accuracy in structure learning.

More recently, Champion et al. Champion et al (2018) developed a genetic algorithm optimizing over a triangular coefficient matrix and a permutation to learn Gaussian BNs. However, the authors did not provide a principled method to select the tuning parameter for the ℓ_1 penalty. Given a permutation, they optimize the network structure by an adaption of the least angle regression (Efron et al, 2004), which is closely related to the Lasso. In contrast, we use a more general and effective first-order method, the proximal gradient algorithm, which is applicable to many regularizers, including the ℓ_1 and concave penalties.

As shown by our numerical experiments, our method substantially outperforms their genetic algorithm.

1.4 Learning DAGs from distributed data

We also develope another order-based learning method, distributed annealing on regularized likelihood score (DARLS), to learn causal structures from data distributed over different machines. To the best of our knowledge, it is the first method using distributed optimization to learn causal structures.

With recent technology developments, distributed data storage has been used as a privacy protection mechanism for managing the large amount of data generated every day, raising a pressing need for distributed learning methods. There are 2.5×10^{18} bytes of data generated on the web every day (Mehmood et al, 2016). Storing such large data set on a single machine is impractical and has a high risk of privacy breach when data leakage occurs. Therefore, companies and researchers often separate sensitive data in practice, which advocates various distributed statistical and machine learning methods. A straightforward approach is to average local estimators for a global output, known as one-shot parameter averaging, but this method fails to obtain solutions with any desired suboptimality (Zinkevich et al, 2010; Shamir et al, 2014). To overcome drawbacks of one-shot averaging, communication-efficient algorithms that utilize multiple rounds of communication between local and central machines to generate a sequence of (global) estimates have been proposed (Zhang et al, 2013; Shamir et al, 2014; Jordan et al, 2018; Fan et al, 2019), and typically, parallel computation is used to reduce computational time. Communication-efficient algorithms are particularly useful in distributed optimization of multi-agent systems, such as electronic power systems, sensor networks and smart manufacturing (Molzahn et al, 2017; Yang et al, 2019).

In spite of these methodological advances, learning causal DAGs from data distributed across independent machines is still a challenging task. A main difficulty is integrating local information to form a global causal graph that satisfies the acyclicity constraint. Combining local DAGs by one-shot averaging is not feasible, because it cannot guarantee the combined estimate is a DAG. Hence, standard DAG learning algorithms cannot be easily adapted to the distributed data setting. To obtain an estimate of a network structure using distributed data, one may iterate over local data sets (once) and aggregate the local information by combining either local graphs or local p-values (Gou et al, 2007; Na and Yang, 2010; Tang et al, 2019). However, it is unclear if aggregating local estimates would be close to the global estimate on combined data. In this thesis, we propose a score-based learning which can effectively estimate DAGs from distributed data, where the objective is to maximize a regularized log-likelihood of the overall data. The central machine proposes a candidate topological sort π, and the score of π is evaluated via communications with local machines to optimize over DAGs compatible with π. The candidate sort π is then selected by simulated annealing. The convergence rate of our distributed optimization algorithm is $O(\log(n)/\sqrt{m})$ for a fixed true DAG, where n is the total sample size across all local machines and m is the smallest local sample size (Theorem 3, Section 3.3.1).

Another contribution of our work is the use of *generalized linear models* (GLMs) for local conditional distributions in BNs, which brings several advantages to causal structure learning. First, GLM is a flexible family of models for various data types beyond linear Gaussian models, greatly increasing the applicability of our proposed model. Second, most models in the GLM family lead to convex loss, which facilitates the optimization of objective functions in the structure learning problem. The objective function of our distributed learning is equivalent to a regularized likelihood of the overall data, which has been shown to be effective in learning both continuous and discrete DAGs (Fu and Zhou, 2013; Aragam and Zhou, 2015; Gu et al, 2019; Ye et al, 2020). Third, GLM DAG models lead to identifiability of underlying causal DAGs (Proposition 4, Section 3.1), while other common models, such as multinomial for discrete networks and Gaussian linear DAGs, are not identifiable in general. Under such identifiability, we establish the ℓ_2-consistency of a global maximizer DAG of our regularized likelihood score (Theorem 4, Section 3.3.2).

1.5 Outline

The remaining part of the dissertation is organized as follow:

- In Chapter 2, we propose annealing on regularized Choleksy score (ARCS) algorithm to learn Gaussian BNs, where a non-convex penalty is used to estimate sparse DAGs. The scoring function is derived from regularizing Gaussian DAG likelihood, and its optimization gives an alternative formulation of the sparse Cholesky factorization. We combine simulated annealing over permutation space with a fast proximal gradient algorithm, operating on triangular matrices of edge coefficients, to compute the score of any permutation. ARCS has demonstrated a remarkable performance in our exhaustive numerical tests, where we compare its structure accuracy, convariance matrix estimates and empirical loss to other methods. We also establish the consistency of our scoring function in estimating topological sorts and DAG structures in the large-sample limit.

- In Chapter 3, we propose distributed annealing on regularized likelihood score (DARLS) algorithm, which is the first method learning causal graphs from distributed data using iterative optimization that relies on multiple rounds of communication between local and central machines. The annealing strategy is used to search over the topological sort space, along with a distributed optimization method to compute the optimal graphical structure that is compatible with a sort. We establish the convergence of the distributed optimization method to a global optimizer of the overall score computed on all data across local machines. In our simulation studies, DARLS has demonstrated competing performance with distributed data against other existing methods using pooled data across local machines. DARLS also exhibits higher predictive power than other methods in a real-world application for modeling protein-DNA binding networks using ChIP-Sequencing data.

- Chapter 4 concludes the dissertation with a summary and future research directions.

CHAPTER 2

Minimizing Regularized Cholesky Score for Gaussian DAGs

In this chapter, we focus on learning Gaussian BNs from observational and experimental data using an order-based search. Gaussian BNs are equivalently represented by a set of linear structural equation models (SEMs) such that

$$X_j = \sum_{i \in \mathrm{PA}_j} \beta^0_{ij} X_i + \varepsilon_j, \quad j = 1, \ldots, p, \tag{2.1}$$

where $\varepsilon_j \sim \mathcal{N}(0, (\omega^0_j)^2)$ are mutually independent and independent of $\{X_i : i \in \mathrm{PA}_j\}$. Defining $B_0 := (\beta^0_{ij}) \in \mathbb{R}^{p \times p}$, where $\beta^0_{ij} = 0$ if $i \notin \mathrm{PA}_j$, $\varepsilon := (\varepsilon_1, \ldots, \varepsilon_p)^\top \in \mathbb{R}^p$, and $X := (X_1, \ldots, X_p)^\top \in \mathbb{R}^p$, we rewrite (2.1) as

$$X = B_0^\top X + \varepsilon. \tag{2.2}$$

The model has two parameters: 1) B_0 as a coefficient matrix, sometimes called the weighted adjacency matrix, where β^0_{ij} specifies a weight associated with the edge $i \to j$, and 2) $\Omega_0 := \mathrm{diag}((\omega^0_j)^2)$ as a noise variance matrix. The SEMs in (2.1) define a joint Gaussian distribution, $X \sim \mathcal{N}(0, \Sigma_0)$, where Σ_0 is positive definite and given by

$$\Sigma_0^{-1} = (I - B_0)\Omega_0^{-1}(I - B_0)^\top. \tag{2.3}$$

2.1 Acyclicity and permutations

The support of B_0 in (2.2) defines the structure of $\mathcal{G}$, and thus it must satisfy the acyclicity constraint so that $\mathcal{G}$ is indeed a DAG. To facilitate the development of our likelihood score for orderings, we express the acyclicity constraint on B_0 via permutation matrices. Let

$$\mathcal{G}\quad \begin{tikzcd} & X_2 \arrow[dl] \arrow[dr] & \\ X_3 \arrow[d] & & X_1 \\ X_4 & & \end{tikzcd}$$

$$B_0 = \begin{array}{c} \\ X_1 \\ X_2 \\ X_3 \\ X_4 \end{array} \begin{array}{c} \begin{array}{cccc} X_1 & X_2 & X_3 & X_4 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 \\ \beta^{\mathbf{0}}_{\mathbf{21}} & 0 & \beta^{\mathbf{0}}_{\mathbf{23}} & 0 \\ 0 & 0 & 0 & \beta^{\mathbf{0}}_{\mathbf{34}} \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{array} \qquad \pi = [\mathbf{4}, \mathbf{1}, \mathbf{3}, \mathbf{2}] \quad P_\pi = \begin{pmatrix} 0 & 0 & 0 & \mathbf{1} \\ \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 \\ 0 & \mathbf{1} & 0 & 0 \end{pmatrix}$$

$$B_\pi = P_\pi B_0 P_\pi^\top = \begin{array}{c} \\ X_4 \\ X_1 \\ X_3 \\ X_2 \end{array} \begin{array}{c} \begin{array}{cccc} X_4 & X_1 & X_3 & X_2 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \beta^{\mathbf{0}}_{\mathbf{34}} & 0 & 0 & 0 \\ 0 & \beta^{\mathbf{0}}_{\mathbf{21}} & \beta^{\mathbf{0}}_{\mathbf{23}} & 0 \end{pmatrix} \end{array}$$

Figure 2.1: An example DAG $\mathcal{G}$, its coefficient matrix B_0, and a permutation π. B_π permutes columns and rows of B_0 and is strictly lower triangular.

$\{e_1, \dots, e_p\}$ be the canonical basis of $\mathbb{R}^p$. To each permutation π on the set $[p]$, we associate a permutation matrix P_π whose i^{th} row is $e_{\pi(i)}^\top$. For a vector $v = (v_1, \dots, v_p)^\top$, we have

$$P_\pi v = v_\pi = (v_{\pi(1)}, \dots, v_{\pi(p)})^\top, \tag{2.4}$$

that is, P_π permutes the entries of v according to π. Since $P_\pi^\top P_\pi = I$, we can rewrite (2.2) as

$$P_\pi X = B_\pi^\top P_\pi X + P_\pi \varepsilon,$$

where $B_\pi := P_\pi B_0 P_\pi^\top$ is obtained by permuting the rows and columns of B_0 simultaneously according to π. Then, B_π will be a strictly lower triangular matrix if and only if π is the reversal of a topological sort of $\mathcal{G}$, i.e., $i \prec j$ in π for $j \in \mathrm{PA}_i$. See Figure 2.1 for an illustration. Under this reparametrization, the acyclicity constraint on B_0 translates to B_π being strictly lower triangular for some permutation π. Define $\Omega_\pi := P_\pi \Omega_0 P_\pi^\top$. Equivalently, one may think of node $\pi(i)$ as having been relabeled node i in B_π and Ω_π.

For simplicity, we drop the subscript π from P_π, B_π and Ω_π if no confusion arises. Therefore, throughout the chapter, P defines a permutation π, B and Ω label nodes according to π and we write the permuted SEM as

$$PX = B^\top PX + P\varepsilon. \tag{2.5}$$

Denote by $\mathrm{cov}(X)$ the covariance matrix of X. Then we have $\Sigma := \mathrm{cov}(PX) = P\Sigma_0 P^\top$, obtained by permuting the rows and columns of Σ_0 (2.3) according to P.

2.2 Regularized likelihood score

In this section, we construct the objective function to estimate BN structure given data from the Gaussian SEM (2.1). We first focus on observational data in Sections 2.2.1 and 2.2.2: We re-parametrize the negative log-likelihood as a Cholesky loss and then impose sparse regularization to define our scoring function over permutations and DAG structures. In Section 2.2.3, we discuss how to modify the likelihood function for experimental data.

2.2.1 Cholesky loss

Let $\mathbf{X} := [\mathbf{X}_1, \dots, \mathbf{X}_p] \in \mathbb{R}^{n \times p}$ be a data matrix where each row is an i.i.d. observation from (2.1). According to (2.5), we obtain a similar SEM on the data matrix:

$$\mathbf{X}P^\top = \mathbf{X}P^\top B + \mathbf{E}P^\top, \tag{2.6}$$

where each row of $\mathbf{E} \in \mathbb{R}^{n \times p}$ is an i.i.d. error vector from $\mathcal{N}(0, \Omega_0)$. In (2.6), $\mathbf{X}P^\top$ and $\mathbf{E}P^\top$ are $\mathbf{X}$ and $\mathbf{E}$ with columns permuted according to $P = P_\pi$. It then follows that each row of $\mathbf{X}P^\top$ is an i.i.d. observation from $\mathcal{N}(0, \Sigma)$ with $\Sigma^{-1} = (I - B)\Omega^{-1}(I - B)^\top$, and thus the negative log-likelihood of (2.6) is

$$\ell(B, \Omega, P \mid \mathbf{X}) = \frac{1}{2} \operatorname{tr}\left[P\mathbf{X}^\top \mathbf{X} P^\top (I - B)\Omega^{-1}(I - B)^\top\right] + \frac{n}{2}\log|\Omega|. \tag{2.7}$$

Recall that B and $\Omega = \operatorname{diag}((\omega_j)^2)$ are defined by permuting the rows and columns of B_0 and Ω_0 by the permutation matrix P. In particular, B is strictly lower triangular and we write its columns as $\beta_j \in \mathbb{R}^p$.

Denote by $L := (I - B)\Omega^{-\frac{1}{2}}$ a weighted coefficient matrix, where each column $L_j = (e_j - \beta_j)/\omega_j$ is a weighted coefficient vector for node $\pi(j)$. We define what we call the *Choleskly loss* function

$$\mathscr{L}_{\text{chol}}(L; A) := \frac{1}{2}\operatorname{tr}\left(ALL^\top\right) - \log|L|, \tag{2.8}$$

where $|L|$ denotes the determinant of L. Noting that $|L| = |(I - B)\Omega^{-\frac{1}{2}}| = |\Omega|^{-\frac{1}{2}}$ and denoting by $\widehat{\Sigma} := \frac{1}{n}\mathbf{X}^\top\mathbf{X}$ the sample covariance matrix, one can re-parametrize the negative log-likelihood (2.7) with L and P and connect it to the Cholesky loss:

Lemma 1. *The negative log-likelihood* (2.7) *for observational data can be re-parametrized as*

$$\ell(L, P) = n \cdot \mathscr{L}_{chol}(L; P\widehat{\Sigma}P^\top) = \frac{n}{2} \operatorname{tr}\left(P\widehat{\Sigma}P^\top LL^\top\right) - n \log |L|, \tag{2.9}$$

where $L = (I - B)\Omega^{-\frac{1}{2}}$ *is a lower triangular matrix and* P *is a permutation matrix.*

The reason for naming (2.8) the Cholesky loss is that it provides an interesting variational characterization of the Cholesky factor of the inverse of a matrix as the following proposition shows. Let $\mathcal{L}_p$ be the set of $p \times p$ lower triangular matrices with positive diagonal entries, and for any positive definite matrix M, let $\mathcal{C}(M)$ be its unique Cholesky factor, i.e., the unique lower triangular matrix L with positive diagonal entries such that $M = LL^\top$. Let $\lambda_{\min}(A)$ and $\|A\|_F$ denote, respectively, the minimum eigenvalue and the Frobenius norm of a matrix A.

Proposition 1 (Curvature)**.** *For any positive definite matrix* $A \in \mathbb{R}^{p\times p}$ *and lower triangular matrix* $L \in \mathcal{L}_p$,

$$\mathscr{L}_{chol}(L; A) - \mathscr{L}_{chol}(L^*; A) \geq \frac{1}{2}\lambda_{\min}(A)\|L - L^*\|_F^2,$$

where $L^* = \mathcal{C}(A^{-1})$. *Consequently,* L^* *is the unique minimizer of* $\mathscr{L}_{chol}(\,\cdot\,; A)$ *with optimal value*

$$\mathscr{L}^*_{chol}(A) := \mathscr{L}_{chol}(\mathcal{C}(A^{-1}); A) = \frac{1}{2}\left(p + \log |A|\right).$$

In particular, $\mathscr{L}^*_{chol}(A) = \mathscr{L}^*_{chol}(PAP^\top)$ *for any permutation matrix* P.

Proposition 1 states that $\mathcal{C}(A^{-1})$ is the unique minimizer of $\mathscr{L}_{\text{chol}}(\,\cdot\,; A)$ and bounds the curvature of the Cholesky loss near its minimum. The curvature bound will be used in the proof of consistency (cf. Theorem 2 in Section 2.3).

Now consider finding the maximum likelihood DAG for a fixed permutation P, which corresponds to minimizing $L \mapsto \ell(L, P)$ given by (2.9). Let $\ell^*(P)$ be the optimal value of this problem, i.e.,

$$\ell^*(P) := \min_{L \in \mathcal{L}_p} \ell(L, P).$$

Then, Proposition 1 implies

$$\ell^*(P) = n \cdot \mathscr{L}^*_{\text{chol}}(P\widehat{\Sigma}P^T) = n \cdot \mathscr{L}^*_{\text{chol}}(\widehat{\Sigma}), \tag{2.10}$$

showing that ℓ^* is invariant to permutations, hence maximum likelihood estimation does not favor any particular ordering. In other words, all the maximum likelihood DAGs corresponding to different permutations give the same Gaussian likelihood. Moreover, they will always be complete DAGs, which has negative implications for both computational and interpretability concerns. These motivate our development of sparse regularization for this problem.

2.2.2 Sparse regularization

To break the permutation equivalence of the maximum likelihood (2.10), we add a regularizer to the Cholesky loss to favor sparse DAGs. Under faithfulness stated in Definition 4 (Spirtes et al, 1993), the true DAG $\mathcal{G}$ in (2.1) and its equivalence class are the sparsest among all DAGs that can parameterize the joint distribution $\mathcal{N}(0, \Sigma_0)$. To start, let us point out some connections to the well-known "sparse Cholesky factorization" problem from linear algebra.

According to Proposition 1, the minimizer of $\ell(L, P)$ over L is the Cholesky factor of $(P\widehat{\Sigma}P^\top)^{-1} = P\widehat{\Sigma}^{-1}P^\top$. For a sparse $\widehat{\Sigma}^{-1}$, it is well-known that the choice of P greatly affects the sparsity of the resulting Cholesky factor. Heuristic approaches have been developed in numerical linear algebra to find a permutation that leads to a sparse factorization by trying to minimize the so-called "fill-in". An example is the maximum cardinality algorithm (Vandenberghe and Andersen, 2014).

From a statistical perspective, however, $\widehat{\Sigma}^{-1}$ is, in general, not sparse (due to noise) even if the inverse of population covariance matrix $\Sigma = \mathbb{E}[\widehat{\Sigma}]$ is so. In such cases, one can first estimate a sparse precision matrix and then use the sparse estimate as the input to the sparse Cholesky factorization problem. We take a more direct alternative approach by adding a sparsity-measuring penalty to the Cholesky loss.

Let $\rho_\theta : \mathbb{R} \mapsto [0, \infty)$ be a nonnegative and nondecreasing regularizer with some tuning parameter(s) θ. We consider the following penalized loss function:

$$f_\theta(L;P) := \; n \cdot \mathscr{L}_{\text{chol}}(L; P\widehat{\Sigma}P^\top) + \sum_{i>j} \rho_\theta(L_{ij}), \tag{2.11}$$

where the penalty is only applied to the off-diagonal entries of a lower triangular matrix L. The loss depends on the regularization parameter θ, and for simplicity we write $f_\theta(L;P)$ as $f(L;P)$. In this paper, we focus on the class of regularizers called the minimax concave penalty (MCP) (Zhang, 2010) which includes ℓ_1 and ℓ_0 as extreme cases; see (2.14) below. MCP is a sparsity-favoring penalty and adding it breaks the symmetry of the Cholesky loss w.r.t. permutations as in (2.10). As a result, the permutations leading to sparser lower-triangular factors L will have smaller loss values $f(L;P)$.

Let $\mathcal{P}_p$ be the set of $p \times p$ permutation matrices. Given $P \in \mathcal{P}_p$, the minimizer of $f(L;P)$ over L is a sparse DAG $\mathcal{G}(P)$ with a score $f(P)$ defined as

$$f(P) := \min_{L \in \mathcal{L}_p} f(L;P). \tag{2.12}$$

We minimize permutation score $f(P)$ over $\mathcal{P}_p$ to obtain an estimated ordering. The overall sparse BN learning problem is then

$$\min_{P\in\mathcal{P}_p} f(P) = \min_{P\in\mathcal{P}_p} \min_{L\in\mathcal{L}_p} \Big\{ \frac{n}{2} \operatorname{tr}\left(P\widehat{\Sigma}P^\top LL^\top\right) - n\log|L| + \sum_{i>j} \rho_\theta(L_{ij}) \Big\}. \tag{2.13}$$

In this formulation, the objective function only depends on the $p \times p$ sample covariance matrix $\widehat{\Sigma} = \frac{1}{n}\mathbf{X}^\top\mathbf{X}$. Thus, the computational complexity is determined by the dimension p once $\widehat{\Sigma}$ has been computed. In Section 2.4, we discuss our approach to solve this problem by optimizing over (L, P). It is worth noting that problem (2.13) can be considered both as 1) a penalized maximum likelihood BN estimator in the Gaussian case, and 2) a variational formulation of the sparse Cholesky factorization problem when the input matrix $\widehat{\Sigma}$ is noisy (hence its inverse usually not sparse). According to the second interpretation, we call $f(L;P)$ in (2.11) the *regularized Cholesky (RC) loss* function and $f(P)$ in (2.12) the *RC score* of a permutation P.

Throughout this chapter, let $\rho(\cdot) := \rho_\theta(\cdot)$ be the MCP with two parameters $\theta = (\gamma, \lambda)$ (Zhang,

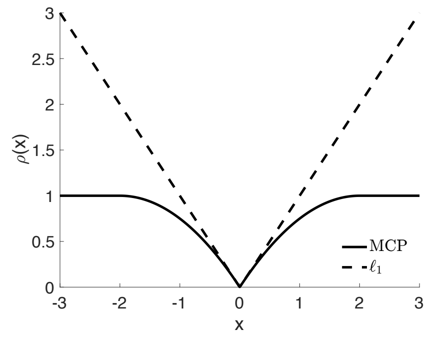


Figure 2.2: A comparison between the MCP (solid line) and the ℓ_1 penalty (dashed line).

2010):

$$\rho(x;\gamma,\lambda) = \begin{cases} \lambda|x| - \frac{x^2}{2\gamma}, & |x| < \gamma\lambda, \\ \frac{1}{2}\gamma\lambda^2, & |x| \geq \gamma\lambda, \end{cases} \tag{2.14}$$

where $\lambda \geq 0$ and $\gamma > 1$. Parameter λ measures the penalty level, while γ controls the concavity of the function. For a fixed value of λ, the MCP approaches the ℓ_1 penalty as $\gamma \to \infty$, and the ℓ_0 penalty as $\gamma \to 0^+$.

Figure 2.2 compares the MCP with $(\gamma, \lambda) = (2, 1)$ and the ℓ_1 penalty. The right derivative of MCP at zero is λ, which is the same as the derivative of the ℓ_1 penalty. The MCP function flats out when $|x| \geq \gamma\lambda$.

Remark 1. Aragam and Zhou Aragam and Zhou (2015) use an MCP regularized likelihood to estimate Gaussian DAGs as well. However, rather than searching over permutations, which automatically satisfies the acyclicity constraint, they perform a greedy coordinate descent to minimize the regularized loss over DAGs. Thus, at each update in their algorithm the acyclicity constraint must be carefully checked.

Remark 2. For a given permutation P, the minimizer of the RC loss $f(L; P)$ (2.12) estimates a sparse Cholesky factor of $P\Sigma_0^{-1}P^\top$, the precision matrix Σ_0^{-1} with rows and columns permuted. This is related to recent methods on covariance matrix estimation by Cholesky decomposition (Chen and Leng, 2015; Lee and Lee, 2018; Touchette et al, 2016; Verzelen, 2010; Li and Zhang, 2019), which make two main assumptions: (i) a fixed variable ordering is provided, i.e. P is given, and (ii) the precision matrix has certain sparse structures, say it is banded. See (Pourahmadi, 2011) for a recent review on covariance matrix estimation. The key differences between these methods and our BN learning approach are: (i) We impose sparsity on the Cholesky factor L, which encodes a DAG structure with P, while those precision matrix estimation methods either impose sparsity on Σ_0^{-1}, such as in (Chen and Leng, 2015; Lee and Lee, 2018), or assume L is banded (Verzelen, 2010). (ii) More importantly, instead of assuming a known ordering, we minimize the RC score (2.13) jointly over (L, P) to estimate an ordering and a sparse DAG structure. Once an estimated $\widehat{P}$ and the associated $\widehat{L}$ have been found, we can estimate the precision matrix Σ_0^{-1} by $\widehat{P}^\top \widehat{L}\widehat{L}^\top \widehat{P}$. Under the Gaussian SEM (2.1), our estimate of Σ_0^{-1} will be more accurate than the above precision matrix estimation methods when the true ordering is unknown. See Section 2.5.5 for relevant numerical comparisons.

2.2.3 Likelihood for experimental data

It is well-known that DAGs in the same Markov equivalence class are observationally equivalent, and thus we cannot distinguish such DAGs from observational data alone. However, experimental interventions can help distinguish equivalent DAGs and construct causal networks. Following Pearl (1995), intervention on a node X_j in a DAG is to impose a fixed external distribution on this node, denoted by $p(x_j \mid \bullet)$, independent of all X_{-j}, while keeping the structural equations (2.1) of the other nodes unchanged.

Suppose that our data $\mathbf{X} \in \mathbb{R}^{n \times p}$ consists of M blocks, where each block $\mathbf{X}^m \in \mathbb{R}^{n_m \times p}$ and $n = \sum_{m=1}^{M} n_m$. Denote by $X_{\mathcal{I}}^m \subset \{X_1, \ldots, X_p\}$ the set of variables under experimental interventions in block m. Then, the data for $X_j \in X_{\mathcal{I}}^m$ in this block are generated independently from the distribution $p(x_j \mid \bullet)$, while for $X_i \notin X_{\mathcal{I}}^m$ from the conditional distribution $[X_i \mid \mathrm{PA}_i]$. Note that multiple nodes could be intervened for a block of data, in which case $|X_{\mathcal{I}}^m| \geq 2$.

Let $\mathcal{I}_j \subset \{1, 2, \ldots, n\}$ be the set of observations for which X_j is under experimental intervention, and let $\mathcal{O}_j = \{1, 2, \ldots, n\} \backslash \mathcal{I}_j$ be its complement. By the truncated factorization formula (Robins, 1986; Pearl, 1995; Spirtes et al, 1993), the joint density of experimental data is

$$p(\mathbf{X}) = \prod_{j=1}^{p} \prod_{h \in \mathcal{O}_j} p(x_{hj} \mid pa_{hj}) \prod_{k \in \mathcal{I}_j} p(x_{kj} \mid \bullet), \tag{2.15}$$

where x_{hj} is the value of the j^{th} variable in the h^{th} observation and pa_{hj} is the value for its parents. Let $\mathbf{X}_{\mathcal{O}_{\pi(j)}}$ be the submatrix of $\mathbf{X}$ with rows in $\mathcal{O}_{\pi(j)}$ and t

$$\widehat{\Sigma}^j := \frac{1}{|\mathcal{O}_{\pi(j)}|} \mathbf{X}_{\mathcal{O}_{\pi(j)}}^{\top} \mathbf{X}_{\mathcal{O}_{\pi(j)}}$$

be the sample covariance matrix computed from data in these rows. Then the log-likelihood of experimental data can be re-parametrized into the Cholesky loss functions as well:

Lemma 2. *The negative log-likelihood for experimental data* (2.15) *can be written as*

$$\ell_{\mathcal{O}}(L, P) = \sum_{j=1}^{p} |\mathcal{O}_{\pi(j)}| \mathscr{L}_{chol}\left(L_j; P \widehat{\Sigma}^j P^{\top}\right), \tag{2.16}$$

where $L_j = (e_j - \beta_j)/\omega_j \in \mathbb{R}^p$, $L = (L_j) \in \mathcal{L}_p$, *and* $|L_j| := L_{jj}$ *in* $\mathscr{L}_{chol}(\cdot)$ (2.8).

Though experimental data likelihood $\ell_{\mathcal{O}}(L, P)$ in (2.16) is not identical to the observational $\ell(L, P)$ in (2.9), searching strategies described in Section 2.4 can be applied to both observational and experimental data.

2.3 Consistency

We now show that the estimator that minimizes the regularized Cholesky score (2.13) is consistent as the sample size $n \to \infty$, while the dimension of the problem p remains fixed (the classical asymptotic setting). We establish the result for a general class of regularizers that contains MCP as a special case. We show two levels of consistency: (1) permutation consistency, in the sense of recovering a topological sort that is consistent with a DAG in the true Markov equivalence class, and (2) structure consistency or support recovery, showing that the estimated DAG has the same support as a DAG in the true equivalence class. Theoretical results in this section are joint work in (Ye et al, 2020), and are included for completeness. Proof of Theorem 2 is provided in Sections 2.8.1 and 2.8.2

Let us give the high level ideas behind our proof of the consistency results. First, in the large-sample limit, the empirical loss $f(P_\pi)/n$ (2.12) converges to $F_{n,\pi}$ uniformly in π, where $F_{n,\pi}$ is its population version, obtained by replacing $\widehat{\Sigma}$ with Σ_0 in definition (2.11). Second, under certain identifiability assumptions, the population loss satisfies $F_{n,\pi^*} < F_{n,\pi}$, for any permutation π^* associated with the true Markov equivalence class and any π that is not. Together these two results allow us to guarantee both the permutation and structure consistency of the minimizers of the regularized Cholesky loss. We give a more detailed sketch in Section 2.3.3, after stating the main result.

To make the notion of permutation consistency more precise, we review some background on Markov equivalence classes and introduce the notion of *score-identifiability.*

2.3.1 Score-identifiability

Let P^* be the true data-generating distribution. Recall we write $\mathcal{I}(P^*)$ for the set of conditional independence (CI) statements that hold in P^* and similarly $\mathcal{I}(B)$ for the set of CIs, implied by DAG B, in the sense of d-separation. Then, B is an I-map for P^* if $\mathcal{I}(B) \subset \mathcal{I}(P^*)$ and is *perfect* for (or *faithful* to) P^* if $\mathcal{I}(B) = \mathcal{I}(P^*)$. Two DAGs B and B' are (Markov) equivalent if $\mathcal{I}(B) = \mathcal{I}(B')$. We write $\mathcal{E}(B)$ for the equivalence class of B.

By a well-known result, all the DAGs in a Markov equivalence class have the same skeleton (hence the same number of edges) and the same v-structures. We write $\mathcal{E}$ for a generic equivalence class and $\|\mathcal{E}\|_0$ for the number of edges of any DAG in the equivalence class. Consider the following definition:

Definition 2. *We say that a Markov equivalence class $\mathcal{E}$ is score-identifiable (for P^*) if for any DAG B which is an I-map for P^*, either $B \in \mathcal{E}$ or $\|B\|_0 > \|\mathcal{E}\|_0$.*

The case that this definition is ruling out is when there exists a DAG outside equivalence class $\mathcal{E}$ that has the same number of edges but a different set of v-structures. By definition, if a score-identifiable class exists, it is "the unique" sparsest equivalence class that is an I-map for P^*. We have the following which is essentially the same as (Chickering, 2002, Proposition 8):

Lemma 3. *If $\mathcal{E}$ is perfect for P^*, then $\mathcal{E}$ is score-identifiable.*

Therefore, score-identifiability is implied by, and thus no stronger than, faithfulness.

2.3.2 Consistent structure learning

With some abuse of notation, throughout this section, let $B = P_\pi^\top (I - A)\Omega^{-1/2} P_\pi$ denote a normalized DAG, where A is a strictly lower triangular matrix and Ω is a diagonal matrix with positive diagonal elements. Note that $L := (I - A)\Omega^{-1/2} \in \mathcal{L}_p$ is a lower triangular matrix with positive diagonals, and π is a reversed topological sort (RTS) of the DAG defined by the support of B (cf. Figure 2.1). Let $\mathfrak{D}_\pi$ be the set of DAGs whose RTS is consistent with permutation π, that is,

$$\mathfrak{D}_\pi = \{P_\pi^\top L P_\pi : L \in \mathcal{L}_p\}$$

and let K_π be the (unique) complete DAG reversely sorted by π. For every $B \in \mathfrak{D}_\pi$, we have $\text{supp}(B) \subset K_\pi$, where $\text{supp}(B)$ denotes the "off-diagonal" support of B, i.e., the set of indices of nonzero off-diagonal elements of B. Unless otherwise stated, the support of B refers to this off-diagonal support. We let B_π be the unique minimizer of the (unregularized) population Cholesky loss over $\mathfrak{D}_\pi$ in this section:

$$B_\pi = \underset{B \in \mathfrak{D}_\pi}{\arg\min}\, \mathscr{L}_{\text{chol}}(B; \Sigma_0) \tag{2.17}$$

and let $S_\pi = \text{supp}(B_\pi)$ be its (off-diagonal) support. It is not hard to see that the support of B_π is the (minimal) I-map corresponding to permutation π. We simply refer to B_π as the I-map associated with π.

Let $\widehat{\Sigma}_n$ be the sample covariance matrix. Consider the following permutation score

$$\widehat{F}_{n,\pi} := \min_{L \in \mathcal{L}_p} \Big[\mathscr{L}_{\text{chol}}(L; P_\pi \widehat{\Sigma}_n P_\pi^\top) + \frac{\rho_n(L)}{n}\Big], \tag{2.18}$$

for some permutation-invariant penalty $\rho_n = \rho_{\theta_n}$. Note that $\widehat{F}_{n,\pi} = f(P_\pi)/n$, where $f(\cdot)$ is the regularized Cholesky score in (2.12). We assume that ρ_n is defined over all $n \times n$ matrices (not just lower triangular ones). Then, we can alternatively write

$$\widehat{F}_{n,\pi} = \min_{B \in \mathfrak{D}_\pi} \Big[\mathscr{L}_{\text{chol}}(B; \widehat{\Sigma}_n) + \frac{\rho_n(B)}{n} := \mathcal{L}_n(B; \widehat{\Sigma}_n)\Big]. \tag{2.19}$$

Let $\widehat{B}_{n,\pi}$ be a (global) minimizer of $B \mapsto \mathcal{L}_n(B; \widehat{\Sigma}_n)$ over $\mathfrak{D}_\pi$ so that $\widehat{F}_{n,\pi} = \mathcal{L}_n(\widehat{B}_{n,\pi}; \widehat{\Sigma}_n)$. We estimate the permutation by minimizing $\pi \mapsto \widehat{F}_{n,\pi}$, with a minimizer denoted as $\widehat{\pi}_n$. Then, our estimated weighted adjacency matrix will be $\widehat{B}_{n,\widehat{\pi}_n}$, and its support defines the structure of an estimated DAG $\widehat{G}_n$.

We need some regularity conditions on the regularizer $\rho_n(\cdot)$ and the collection $\{B_\pi\}$ of I-maps of Σ_0. Let $\lambda_{\min} = \lambda_{\min}(\Sigma_0)$ be the minimum eigenvalue of Σ_0, and assume that $\rho_n(B) = \sum_{i \neq j} r_n(|B_{ij}|)$ for some $r_n : \mathbb{R}_+ \to \mathbb{R}_+$ that satisfies the following:

(R1) We say that r_n is (a_n, b_n)-flat if it is bounded by a_n and

$$r_n(t) = a_n \text{ for } t \geq b_n.$$

for some $b_n = O(1)$.

(R2) Assume that r_n is twice differentiable on $[0, b_n)$, with the derivatives at 0 interpreted as one-sided, and assume that r_n'' has a left limit at b_n. Moreover $r_n''(\cdot)/n$ is C_0-Lipschitz on $[0, b_n)$, for some constant $C_0 \geq 0$, and

$$|r_n''(0+)| \leq C_1 n, \quad |r_n''(b_n-)| = O(n),$$

with $C_1 \leq \lambda_{\min}/2$. Let $\lambda_n := r_n'(0+)$ be the right derivative of r_n at 0.

Consider the so-called $\beta_{\min}$ condition

$$2b_n \leq \min_\pi \tau(B_\pi), \tag{2.20}$$

where $\tau(B) = \min\{|B_{ij}| : (i,j) \in \operatorname{supp}(B)\}$. Assume further that

$$\liminf \frac{nb_n^2}{a_n} > \frac{2}{\lambda_{\min}} \max_\pi |S_\pi|. \tag{2.21}$$

Let us write $\Pi(\mathcal{E})$ for the collection of RTSs for some DAGs in equivalence class $\mathcal{E}$. If $\mathcal{E}^*$ is the true equivalence class, then $\Pi(\mathcal{E}^*)$ is the collection of true RTSs. We have the following consistency result:

Theorem 2. *Assume that P^*, or equivalently Σ_0 in* (2.3)*, has a (unique) score-identifiable Markov equivalence class $\mathcal{E}^*$, and let*

$$\widehat{\pi}_n \in \arg\min_{\pi} \widehat{F}_{n,\pi}.$$

(a) Assume that $r_n(\cdot)$ is (a_n, b_n)-flat for sequences that satisfy (2.20), (2.21), *$n^{-1/2}a_n \to \infty$ and $n^{1/4}b_n \to \infty$, and (R2) holds with $n^{-1}\lambda_n = O(1)$. Then, $\mathbb{P}\big(\widehat{\pi}_n \in \Pi(\mathcal{E}^*)\big) \to 1$ as $n \to \infty$. Moreover, $\widehat{B}_{n,\widehat{\pi}_n}$ is a $\sqrt{n}$-consistent estimate of $B_{\widehat{\pi}_n}$.*

(b) If in addition $n^{-1}\lambda_n \to 0$ and $n^{-1/2}\lambda_n \to \infty$, then we also have $\mathbb{P}\big(\widehat{G}_n \in \mathcal{E}^*\big) \to 1$ as $n \to \infty$, where $\widehat{G}_n$ is the DAG defined by $\mathrm{supp}(\widehat{B}_{n,\widehat{\pi}_n})$.

Part (a) of Theorem 2 establishes permutation consistency by guaranteeing that $\widehat{\pi}_n$ eventually does not leave $\Pi(\mathcal{E}^*)$, although it can move around in this set indefinitely. Part (b) establishes the structure consistency.

In the case of the MCP, we can take $r_n(t)/n = \rho(t; \gamma_n, \xi_n)$ giving $\lambda_n = r_n'(0+) = n\xi_n$. Then, (R1) holds with $a_n = \frac{1}{2}n\gamma_n\xi_n^2$ and $b_n = \gamma_n\xi_n$, and we have $r_n''(t)/n = 1/\gamma_n$ for $t \in [0, b_n)$. Assuming $1/\gamma_n \le \lambda_{\min}/2$ after proper re-scaling of the data or the regularizer, (R2) holds with $C_0 = 0$. Since the right-hand sides of (2.20) and (2.21) are constants, these conditions hold if γ_n is sufficiently large and $\xi_n \to 0$. Conditions for part (a) of the theorem hold if $\xi_n = O(1)$, $\sqrt{n}\gamma_n\xi_n^2 \to \infty$ and $n^{1/4}\gamma_n\xi_n \to \infty$, so it suffices to have $n^{-1/4} \ll \xi_n \lesssim 1$ and $\gamma_n \gtrsim 1$. For part (b), we need $\xi_n = o(1)$ and $\sqrt{n}\xi_n \to \infty$, that is, $n^{-1/2} \ll \xi_n \ll 1$. All the conditions are satisfied if $\gamma_n \gtrsim 1$ and $n^{-1/4} \ll \xi_n \ll 1$. In particular, if we let $\gamma_n \to \infty$ and $\gamma_n\xi_n \to 0$ at a rate slower than $n^{-1/4}$, then our consistency results apply to the capped ℓ_1 penalty, i.e. $r_n(t)/n = \xi_n t \wedge \gamma_n\xi_n^2$.

Remark 3. Aragam and Zhou Aragam and Zhou (2015) also provide asymptotic results for the estimator we consider here. Under appropriate conditions, they show that (i) in small neighborhoods (of radius $\sim n^{-1/2}$) of every B_π (2.17), there are "good" local minimizers of (2.19) (i.e., with correct support) and (ii) if one B_π has more edges than the other, the corresponding nearby local minimizer gives a higher value of the objective function. Their results, however, provide no guarantee for all local minimizers of the problem. In particular, their results are silent about the global minimizer(s) of (2.19). In contrast, we provide conditions, under which, any "global minimizer" of (2.19) is both permutation and structure consistent. Proving such global results requires significantly more technical effort. For example, even showing that a global minimizer is within a neighborhood of radius $\sim n^{-1/2}$ of some B_π is nontrivial, as the proof demonstrates.

2.3.3 Proof sketch

We give a brief sketch of the proof of Theorem 2 here. A detailed proof can be found in the supplement. For two symmetric matrices A and B of the same dimension, we write $A \succeq B$ if $A - B$ is positive semidefinite. First, we show that under (2.20), for any π and $B \in \mathfrak{D}_\pi$,

$$\mathcal{L}_n(B; \Sigma_0) - \mathcal{L}_n(B_\pi; \Sigma_0) \geq \frac{\lambda_{\min}}{2}\big(\|B - B_\pi\|_F^2 \wedge c_{n,\pi}\big), \tag{2.22}$$

where $c_{n,\pi} := b_n^2 - 2a_n|S_\pi|/(\lambda_{\min} n)$. Combined with (2.21), this implies that the population version of (2.19), namely,

$$F_{n,\pi} := \min_{B \in \mathfrak{D}_\pi} \mathcal{L}_n(B; \Sigma_0), \tag{2.23}$$

has B_π as its unique (isolated) minimizer. A so-called "basic inequality argument" further implies that $\mathcal{L}_n(\widehat{B}_{n,\pi}; \Sigma_0) - \mathcal{L}_n(B_\pi; \Sigma_0) = O_p(n^{-1/2})$. Together with (2.22), we get $\|\widehat{B}_{n,\pi} - B_\pi\|_F = O_p(n^{-1/4})$, that is, $\widehat{B}_{n,\pi}$ is $n^{1/4}$-consistent for B_π.

Using $n^{1/4} b_n \to \infty$, we conclude that $\|\widehat{B}_{n,\pi} - B_\pi\|_F < b_n$, eventually. Recalling assumption (2.20), this implies that for any DAG B which is between $\widehat{B}_{n,\pi}$ and B_π elementwise, the absolute coordinates $|B_{ij}|$ are either in $[0, b_n)$ or (b_n, ∞). Since $r_n(\cdot)$ is smooth over each of these intervals, we can apply a Taylor expansion of $\mathcal{L}_n(\,\cdot\,, \widehat{\Sigma}_n)$ around B_π. Taking into account the one-sided differentiability of the regularized loss at zero, we obtain a quadratic inequality for $\Delta_{n,\pi} = \widehat{B}_{n,\pi} - B_\pi$, where the quadratic term is controlled by the Hessian $\nabla^2 \mathcal{L}_n(\widetilde{B}_{n,\pi}; \widehat{\Sigma}_n)$ for some $\widetilde{B}_{n,\pi}$ that is between $\widehat{B}_{n,\pi}$ and B_π elementwise. A further argument shows that $\nabla^2 \mathcal{L}_n(\widetilde{B}_{n,\pi}; \widehat{\Sigma}_n) \succeq \frac{1}{2}\lambda_{\min} I_{p^2}$ when n is large, which together with the quadratic equality implies

$$\|\Delta_{n,\pi}\|_F \leq \frac{2}{\lambda_{\min}} \|\nabla \mathscr{L}_{\text{chol}}(B_\pi; \widehat{\Sigma}_n) - \nabla \mathscr{L}_{\text{chol}}(B_\pi; \Sigma_0)\|_F.$$

The right-hand side can be shown to be $O_p(n^{-1/2})$ from which we conclude the $\sqrt{n}$-consistency of $\widehat{B}_{n,\pi}$ for B_π. All the consistency arguments hold uniformly over π.

Equipped with $\sqrt{n}$-consistency, we then show that $\widehat{F}_{n,\pi} - F_{n,\pi} = O_p(n^{-1/2})$, uniformly in π. For any $\pi^* \in \Pi(\mathcal{E}^*)$ and any $\pi \notin \Pi(\mathcal{E}^*)$,

$$\begin{aligned}\widehat{F}_{n,\pi} - \widehat{F}_{n,\pi^*} &\geq F_{n,\pi} - F_{n,\pi^*} - O_p(n^{-1/2}) \\ &= \frac{a_n}{n}(|S_\pi| - |S^*_\pi|) - O_p(n^{-1/2}).\end{aligned}$$

See the supplement for the details of getting to the second line. Since by score-identifibility $|S_\pi| \geq |S_{\pi^*}| + 1$, and by assumption $n^{-1/2}a_n \to \infty$, with high probability (w.h.p.) $\widehat{F}_{n,\pi} - \widehat{F}_{n,\pi^*} > 0$ for all $\pi \notin \Pi(\mathcal{E}^*)$, when n is sufficiently large. This proves permutation consistency.

For the structure consistency, since $\widehat{B}_{n,\pi}$ is within a neighborhood of radius $O_p(n^{-1/2})$ around B_π, an argument similar to that of Lemma 1 in Fan and Li (2001) shows that $\text{supp}(\widehat{B}_{n,\pi}) = \text{supp}(B_\pi)$ for all π w.h.p. In particular, $\text{supp}(\widehat{B}_{n,\widehat{\pi}_n}) = \text{supp}(B_{\widehat{\pi}_n})$, w.h.p. and the proof is complete.

2.4 Optimization

We now describe how we solve the optimization problem (2.13). The main steps are outlined in Algorithm 1, where we use simulated annealing to search over the permutation space to minimize the RC score defined in (2.12). To obtain the RC score for a given permutation, we need to solve a continuous optimization problem (line 2 and 6) for which we propose a proximal gradient algorithm (Algorithm 2).

2.4.1 Searching over permutations

The ARCS algorithm is detailed in Algorithm 1. At each iteration, we propose a permutation P^* and decide whether to stay at the current permutation or move to the proposed one with probability α given in line 7. The probability is determined by the difference between the proposed and current scores $f(P^*) - f(\widehat{P})$ normalized by a temperature parameter T. For $T \to \infty$, the jumps are completely random and for $T \to 0^+$ completely determined by the

Algorithm 1 Annealing on regularized Cholesky score (ARCS).

```
Input: Dataset $\mathbf{X}$, initial permutation matrix $P_0$, constant $m$, a temperature schedule
       $\{T^{(i)}, i = 0, \dots, N\}$.
Output: Adjacency matrix $\widehat{B}$.
1: Select tuning parameters $(\gamma, \lambda)$ for $f(L; P)$ according to Algorithm 4 (Section 2.4.4).
2: $\widehat{P} \leftarrow P_0$, $\widehat{L} \leftarrow \arg\min_{L \in \mathcal{L}_p} f(L; \widehat{P})$ by Algorithm 2, $f(\widehat{P}) \leftarrow f(\widehat{L}; \widehat{P})$.
3: for $i = 0, \dots, N$ do
4:     $T \leftarrow T^{(i)}$.
5:     Propose $P^*$ by flipping a random length-$m$ interval in the permutation defined by $\widehat{P}$.
6:     $L^* \leftarrow \arg\min_{L \in \mathcal{L}_p} f(L; P^*)$ using Algorithm 2, $f(P^*) \leftarrow f(L^*; P^*)$.
7:     $\alpha \leftarrow \min\left\{1, \exp\left(-\frac{1}{T}[f(P^*) - f(\widehat{P})]\right)\right\}$.
8:     Set $(\widehat{P}, \widehat{L}, f(\widehat{P})) \leftarrow (P^*, L^*, f(P^*))$ with prob. $\alpha$.
9: end for
10: Refine adjacency matrix $\widehat{B}$ given $(\widehat{L}, \widehat{P})$ by Algorithm 3 (Section 2.4.3).
```

RC score $f(\cdot)$. The algorithm follows a temperature schedule which is often taken to be a decreasing sequence $T^{(0)} \geq T^{(1)} \geq \ldots \geq T^{(N)}$ allowing the algorithm to explore more early on and zoom in on a solution as time progresses.

The proposed permutation matrix P^* is constructed as follows. Let $\widehat{\pi}$ and π^* be the permutations associated with $\widehat{P}$ and P^* as in (2.4). We propose π^* by flipping (i.e., reversing the order of) a random interval of length m in the current permutation $\widehat{\pi}$. For example, with $m = 3$ we may flip $\widehat{\pi} = (1, 2, 3, 4, \dots, p)$ to $\pi^* = (1, 4, 3, 2, \dots, p)$ in the proposal. Equivalently, we flip a contiguous block of m rows of $\widehat{P}$ to generate P^*.

As a byproduct of evaluating the RC score for the proposed permutation P^*, we also obtain the corresponding lower triangular matrix L^*, representing the associated DAG. We keep track of these DAGs as well as the permutations throughout the algorithm (line 6).

Remark 4. There is no theoretical guarantee for ARCS to find a global minimizer using simulated annealing. In this sense, ARCS performs a heuristic search over the permutation space to learn a topological sort. For smaller DAGs, however, our numerical results in

Section 2.5.8 show that ARCS often finds solutions that are close to the global minimizers.

2.4.2 Computing RC score

We propose a proximal gradient algorithm to evaluate the RC score $f(P)$ at each permutation matrix P (line 2 and 6, Algorithm 1). This algorithm belongs to a class of first-order methods that are quite effective at optimizing functions composed of a smooth loss and a nonsmooth penalty (Parikh and Boyd, 2013a).

The RC score is obtained by minimizing the RC loss $f(L;P)$ over L as shown in (2.12). Recall that $\mathcal{L}_p$ is the set of $p \times p$ lower triangular matrices, and let

$$\rho(u) := \sum_{i>j} \rho(u_{ij}), \quad \text{for} \quad u = (u_{ij}) \in \mathcal{L}_p. \tag{2.24}$$

Note that we are leaving out the diagonal elements of u in defining $\rho(u)$. Then, the RC loss is $f(u;P) = \ell(u,P) + \rho(u)$, where $\ell(u,P)$ (2.9) is differentiable and $\rho(u)$ is nonsmooth. The idea of the proximal gradient algorithm is to replace $\ell(u,P)$ with a local quadratic function at the current estimate L and optimize the resulting approximation to $f(u;P)$ to get a new estimate L^+:

$$\begin{aligned} L^+ &= \operatorname*{arg\,min}_{u \in \mathcal{L}_p} \ell(L) + \nabla\ell(L)^\top (u-L) + \frac{1}{2t}\|u-L\|^2 + \rho(u) \\ &= \operatorname*{arg\,min}_{u \in \mathcal{L}_p} \frac{1}{2t}\|L - t\nabla\ell(L) - u\|^2 + \rho(u), \end{aligned} \tag{2.25}$$

where $\ell(L) = \ell(L,P), \nabla\ell(L) := \nabla_L \ell(L,P)$ is the gradient of $\ell(L,P)$ w.r.t. L, and $t > 0$ is a step size. Consider the proximal operator $\mathbf{prox}_\rho : \mathcal{L}_p \to \mathcal{L}_p$ associated with ρ defined by

$$\mathbf{prox}_\rho(x) := \operatorname*{arg\,min}_{u \in \mathcal{L}_p} \left(\rho(u) + \frac{1}{2}\|x-u\|^2 \right), \tag{2.26}$$

where $x \in \mathcal{L}_p$ and $\|\cdot\|$ is the usual Euclidean norm. Then, (2.25) is equivalent to

$$L^+ = \mathbf{prox}_{t\rho}\left(L - t\nabla\ell(L)\right), \tag{2.27}$$

where $\mathbf{prox}_{t\rho}(\cdot)$ is the proximal operator applied to the scaled function $t\rho(\cdot)$. Since $\rho(u)$ is

Algorithm 2 Compute the RC score by proximal gradient.

```
    Input: $P$, $L^{(0)} \in \mathcal{L}_p$, $t^{(0)} > 0$, $\kappa \in (0,1)$, max-iter, tol.
    Output: $L$.
1:  $k \leftarrow 0$, err $\leftarrow \infty$, $L \leftarrow L^{(0)}$.
2:  while $k <$ max-iter and err $>$ tol do
3:      Compute $\nabla \ell(L)$ using either Lemma 4 or 5.
4:      $t \leftarrow t^{(0)} / \|\nabla \ell(L)\|_F$.
5:      repeat
6:          $\widetilde{L} \leftarrow L - t\nabla \ell(L)$.
7:          $L^+_{ij} \leftarrow \mathbf{prox}_{t\rho}(\widetilde{L}_{ij})$ for $i > j$ (using Lemma 6).
8:          $L^+_{ii} \leftarrow \widetilde{L}_{ii}$.
9:          break if $\ell(L^+, P) \leq \ell(L, P) + \langle \nabla \ell(L), L^+ - L \rangle + \frac{1}{2t}\|L^+ - L\|_F^2$.
10:         $t \leftarrow \kappa t$.
11:     err $\leftarrow \max_j \delta(L^+_j, L_j)$ where $\delta(x,y) := \frac{\|x-y\|}{\max\{1, \|y\|\}}$.
12:     $L \leftarrow L^+$ and $k \leftarrow k+1$.
13: end while
```

separable across the coordinates $\{u_{ij}, i \geq j\}$, we have for $x \in \mathcal{L}_p$,

$$\left(\mathbf{prox}_\rho(x)\right)_{ij} = \begin{cases} \mathbf{prox}_\rho(x_{ij}), & i > j, \\ \mathbf{prox}_0(x_{ii}) = x_{ii}, & i = j. \end{cases}$$

The proximal operators on the RHS are univariate, and the distinction between the two cases is because we do not penalize the diagonal entries, i.e., $\rho(u_{ii}) = 0$.

The overall procedure is summarized in Algorithm 2. To choose the step size t normalized by $\|\nabla \ell(L)\|_F$ (line 4), we have used a line search strategy (Parikh and Boyd, 2013a), where we repeatedly reduce the step size by a factor $\kappa \in (0,1)$ until a quadratic upper bound is satisfied by the new update (line 9). To implement Algorithm 2, we need two more ingredients, $\nabla \ell(L)$ and the univariate $\mathbf{prox}_\rho(\cdot)$, both of which have nice closed-form expressions:

Lemma 4. *The gradient of $\ell(L,P)$ in (2.9) w.r.t. L is*

$$\nabla \ell(L) = n\left(\Pi_{\mathcal{L}}(P\widehat{\Sigma}P^\top L) - \operatorname{diag}\left(\{1/L_{ii}\}_{i=1}^p\right)\right),$$

where $\Pi_{\mathcal{L}} : A \mapsto (A_{ij}\mathbb{1}\{i \geq j\})_{p\times p}$ maps a matrix to its lower triangular projection.

Lemma 5. *The gradient of $\ell_{\mathcal{O}}(L,P)$ in (2.16) w.r.t. L_j is*

$$\nabla_{L_j}\ell_{\mathcal{O}}(L,P) = \left|\mathcal{O}_{\pi(j)}\right|\left(\Pi_j\left(P\widehat{\Sigma}^j P^\top L_j\right) - \frac{e_j}{L_{jj}}\right),$$

where $\Pi_j : v \mapsto (v_i\mathbb{1}\{i \geq j\})_{p\times 1}$ and $\{e_j\}$ is the canonical basis of $\mathbb{R}^p$.

Lemma 6. *Let ρ be the scalar MCP with parameter (γ,λ) defined in (2.14), and let ρ_1 be the same penalty for $\lambda = \gamma = 1$. Then, for any $t > 0$,*

$$\mathbf{prox}_{t\rho}(x) = \lambda\gamma\,\mathbf{prox}_{(t/\gamma)\rho_1}\left(\frac{x}{\lambda\gamma}\right), \tag{2.28}$$

and for any $\alpha > 0$,

$$\mathbf{prox}_{\alpha\rho_1}(x) = \begin{cases} 0, & \begin{array}{l} 0 \leq x < \min\{\alpha, 1\} \text{ or} \\ 1 < x < \sqrt{\alpha}; \end{array} \\ \dfrac{x-\alpha}{1-\alpha}, & \alpha < x \leq 1; \\ x, & \begin{array}{l} x > \max\{\alpha,1\} \text{ or} \\ 1 < \sqrt{\alpha} < x \leq \alpha. \end{array} \end{cases} \tag{2.29}$$

Moreover, $\mathbf{prox}_{\alpha\rho_1}(-x) = -\mathbf{prox}_{\alpha\rho_1}(x)$ for all $x \in \mathbb{R}$.

We have excluded two special cases in (2.29) in which the minimizer is not unique: 1) If $x = \alpha = 1$, $\mathbf{prox}_{\alpha\rho_1}(x) = [0,1]$; 2) If $x = \sqrt{\alpha} > 1$, $\mathbf{prox}_{\alpha\rho_1}(x) = \{0, \sqrt{\alpha}\}$. We set $\mathbf{prox}_{\alpha\rho_1}(x) = 0$ in our implementation if these special cases occur. The MCP has parameter $\gamma > 1$, and usually the step size $t < 1$. Thus, the cases with $\alpha < 1$ are the most common scenario in our numerical study.

2.4.3 Structure refinement after annealing

At the end of the annealing loop (line 9, Algorithm 1), a pair $(\widehat{L}, \widehat{P})$ is found. Accordingly, an estimated reversal of a topological sort is $\widehat{\pi} = \widehat{P}(1,\ldots,p)^\top$. Define $\widetilde{L} = \widehat{P}^\top\widehat{L}\widehat{P}$, and $\widehat{B}$

by $\widehat{B}_{ij} = -\widetilde{L}_{ij}/\widetilde{L}_{jj}$ for $i \neq j$ and $\widehat{B}_{ii} = 0$. Then, $\widehat{B}$ is the estimated weighted adjacency matrix for a DAG, i.e., an estimate for B_0. The support of $\widehat{B}$ gives the estimated parent sets $\widehat{pa}_j = \{i : \widehat{B}_{ij} \neq 0\}$ for $j = 1, \ldots, p$. The use of a continuous regularizer, i.e. MCP, eases our optimization problem; however, this may lead to more false positive edges compared to ℓ_0 regularization. To improve structure learning accuracy, we add a refinement step to remove some predicted edges by conditional independence tests, which borrows the strength from a constraint-based approach.

The refinement step outlined in Algorithm 3 is based on the following fact. If $k \prec j$ in a topological sort and there is no edge $k \to j$, then $X_k \perp X_j \mid \mathrm{PA}_j$, where PA_j is the parent set of X_j. For each $k \in \widehat{pa}_j$, we test the null hypothesis that X_k and X_j are conditionally independent given $\widehat{pa}_j \setminus \{k\}$ using the Fisher Z-score. We remove the edge $k \to j$ if the null hypothesis is not rejected at a given significance level. The conditional independence tests are performed in a sequential manner for the nodes in $\widehat{pa}_j$ according to the estimated topological sort: For $k_1, k_2 \in \widehat{pa}_j$, if $k_1 \prec k_2$ in the sort, we carry out the test for k_2 prior to that for k_1.

2.4.4 Selection of the tuning parameters

Before starting the iterations in Algorithm 1, we select and fix the tuning parameters $\theta = (\gamma, \lambda)$ of MCP (line 1), hence fixing a particular scoring function $f(L, P) = f_\theta(L, P)$ throughout the algorithm.

We use the Bayesian information criterion (BIC) (Schwarz, 1978) to select the tuning parameters, given an initial permutation P_0. The details are summarized in Algorithm 4. For every pair $(\gamma^{(i)}, \lambda^{(i)})$ over a grid of values, we evaluate the BIC score given in line 2, where $\|\widehat{L}(\cdot)\|_0$ is the number of nonzero entries in $\widehat{L}(\cdot)$, and then we output the one with the lowest BIC score. The regularization parameter in $\mathrm{BIC}(\theta)$ is adapted to $\log(\max\{n, p\})$, which works well for both low and high-dimensional data. To construct the grid, possible choices for the concavity parameter γ are $\{2, 10, 50, 100\}$ based on our tests. Note that $\gamma > 1$ is required in the definition of MCP (2.14), while the behavior of MCP for $\gamma \geq 100$

Algorithm 3 Constraint-based structure refinement.

```
    Input: $\mathbf{X}$, adjacency matrix $\widehat{B}$, significance level $\alpha$.
    Output: Adjacency matrix $\widehat{B}$.
 1: $Z_\alpha \leftarrow \Phi^{-1}(1-\frac{\alpha}{2})$, where $\Phi(x)$ is the CDF of $\mathcal{N}(0,1)$.
 2: for $j = 1, \ldots, p$ do
 3:     $\widehat{pa}_j \leftarrow \{i : \widehat{B}_{ij} \neq 0\}$.
 4:     for $k \in \widehat{pa}_j$ do
 5:         $\mathbf{s} \leftarrow \widehat{pa}_j \setminus \{k\}$.
 6:         $\mathbf{X}^j \leftarrow$ observations for which $j$ is not intervened
 7:         $n \leftarrow$ number of rows in $\mathbf{X}^j$.
 8:         $r_{j,k|\mathbf{s}} \leftarrow$ sample partial correlation between $X_j$ and $X_k$ given $X_\mathbf{s}$ based on $\mathbf{X}^j$.
 9:         $z \leftarrow \frac{1}{2}\sqrt{n-|\mathbf{s}|-3}\log\left(\frac{1+r_{j,k|\mathbf{s}}}{1-r_{j,k|\mathbf{s}}}\right)$.
10:         Remove $k$ from $\widehat{pa}_j$, if $|z| < Z_\alpha$.
11:     end for
12:     $\widehat{B}_{ij} \leftarrow 1$ if $i \in \widehat{pa}_j$ and $\widehat{B}_{ij} \leftarrow 0$ otherwise.
13: end for
```

is essentially the same as the ℓ_1 penalty. For the regularization parameter λ, we select 20 equi-spaced points from the interval $[0.1\sqrt{n}, \sqrt{n}]$. The choice of $\sqrt{n}$ often leads to an empty graph when the data are standardized, hence a natural end point.

2.5 Results on observational data

2.5.1 Methods and data

Recall that p is the number of variables and n is the number of observations. For a thorough evaluation of the algorithm, we simulated data for both $n > p$ and $n < p$ cases.

We used real and synthetic networks to simulate data. Real networks were downloaded from the Bayesian networks online repository (Scutari, Accessed: 2019). We duplicated some of them to further increase the network size. Synthetic DAG structures were constructed

Algorithm 4 Tuning parameter selection by BIC.

```
    **Input:** Initial permutation $P_0$ and a grid of values $\{\theta^{(i)}\} = \{(\gamma^{(i)}, \lambda^{(i)})\}$.
    **Output:** Optimal index $i^*$ in the grid.
1: Define $\widehat{L}(\theta) := \arg\min_{L \in \mathcal{L}_p} f_\theta(L; P_0)$ computed by Algorithm 2.
2: Let $\text{BIC}(\theta) := 2\ell\big(\widehat{L}(\theta), P_0\big) + \|\widehat{L}(\theta)\|_0 \log(\max\{n, p\})$.
3: Output $i^* = \arg\min_i \text{BIC}(\theta^{(i)})$.
```

using the `sparsebn` package (Aragam et al, 2019). Given a DAG structure, we sampled the edge coefficients β_{ij} uniformly from $[-0.8, -0.5] \cup [0.5, 0.8]$ and set the noise variance to one. We then calculated the covariance matrix according to (2.3) and normalized its diagonal elements to one. Consequently, the variances of $\{X_1, \ldots, X_p\}$ were identical. We used the following networks to generate observational data, denoted by the network name and (p, s_0), where s_0 is the number of edges after duplication: 4 copies of `Hailfinder` (224, 264), 1 copy of `Andes` (223, 338), 2 copies of `Hepar2` (280, 492), 4 copies of `Win95pts` (304, 448), 1 copy of `Pigs` (441, 592), and random DAGs, `rDAG1` (300, 300) and `rDAG2` (300, 600). The sample size $n = 200$ for real networks and $n = 240$ for synthetic graphs in the $n < p$ case. In the low-dimensional setting $n > p$, the sample size $n = 450, 500, 600, 600$ for `rDAG1`, `Win95pts`, `Pigs` and `rDAG2`, respectively, and $n = 400$ for the other networks.

In the observational data setting, we compared our algorithm with the following BN learning algorithms: the coordinate descent (CD) algorithm (Aragam and Zhou, 2015), the standard greedy hill climbing (HC) algorithm (Gámez et al, 2011), the greedy equivalence search (GES) (Chickering, 2002), the Peter-Clark (PC) algorithm (Spirtes and Glymour, 1991), the max-min hill-climbing (MMHC) algorithm (Tsamardinos et al, 2006), and the genetic algorithm (GA) (Champion et al, 2018).

The CD algorithm optimizes a regularized log-likelihood function by a blockwise update on (β_{ij}, β_{ji}) while checking the acyclicity constraint before each update. The HC algorithm performs a greedy search over the DAG space by starting from a certain initial state, performing a finite number of local changes and selecting the DAG with the best improvement in each local change. The GES algorithm searches over the equivalence classes and utilizes greedy search operators on the current state to find the next one, of which the output is an equivalence class of DAGs. The PC algorithm performs conditional independence tests to identify edges and orients edge directions afterwards. The MMHC algorithm constructs the skeleton of a Bayesian network via conditional independence tests and then performs a greedy hill climbing search to orient the edges via optimizing a Bayesian score. The GA decomposes graph estimation into two optimization sub-problems: node ordering search with mutation and crossover operators, and structure optimization by an adaption of the least angle regression (Efron et al, 2004).

Among these methods, PC is a constraint-based method, and MMHC is a hybrid method. Other methods, CD, HC, GES and GA, are all score-based, where CD and HC search over the DAG space, GES searches over the equivalence classes, and GA searches over the permutation space. Our method is a score-based search over the permutation space, similar to GA.

Our ARCS algorithm (Algorithm 1) may take an initial permutation P_0 provided by a local search method. In this study, we use the CD and GES algorithms to provide an initial permutation, and call the corresponding implementation ARCS(CD) and ARCS(GES). To partially preserve properties of the input initial permutation, we start with a low temperature $T^{(0)} = 1$. The output of the CD algorithm is a DAG for which we find a topological sort to define P_0. The GES algorithm outputs a completed partially directed acyclic graph (CPDAG). We then generate a DAG in the equivalence class of the estimated CPDAG, and initialize ARCS with a topological sort of this DAG.

We implemented the ARCS algorithm in Matlab, with source code available at `https://github.com/yeqiaoling/arcs_bn`. We used the following R packages for other methods: `sparsebn` (Aragam et al, 2019) for the CD algorithm, `rcausal` (Ramsey, 2015) for the GES, GIES (for experimental data) and PC algorithms, `bnlearn` (Scutari, 2010) for the MMHC and HC algorithms, and `GADAG` (Champion et al, 2018) for the GA. Among score-based methods, HC and GES used the BIC scoring function for Gaussian data; CD used an MCP-regularized likelihood scoring function, with an internal tuning parameter selection method; GA used an ℓ_1-regularized likelihood, for which we applied grid search for tuning parameter selection.

2.5.2 Accuracy metrics

Among all methods applied to observational data, ARCS, CD, HC and MMHC output DAGs, while the GES and PC algorithms output CPDAGs. Given these estimates, we need to evaluate the performance of each method. To standardize the performance metrics in observational data setting, we transfer an estimated DAG into its CPDAG before calculating the following metrics.

Define P, TP, FP, M, R as the numbers of estimated edges, true positive edges, false positive edges, missing edges and reversed edges, respectively, all with respect to CPDAGs. P is the number of edges in the estimated graph. FP is the number of edges in the estimated graph skeleton but not in the true skeleton. M counts the number of edges in the true skeleton but not in the skeleton of the estimated graph. TP reports the number of consistent edges, including edge orientation for directed edges, between the estimated CPDAG and the true CPDAG. Lastly, the number of reversed edges R = P − TP − FP and so R includes both incorrectly oriented edges and those edges that are oriented in one CPDAG but undirected in the other.

Denote by s_0 the number of edges in the true CPDAG. The overall accuracy of a method is measured by the structural Hamming distance (SHD) and Jaccard index (JI), where SHD = R + FP + M and JI = TP/(s_0 + P − TP). A method has better performance if it achieves a lower SHD and/or a higher JI.

2.5.3 Structure learning accuracy

We used large networks, where $p \in (200, 450)$ and $s_0 \in (250, 600]$, to simulate observational data with $n < p$ and $n > p$. For each setting (p, s_0, n), we generated 20 datasets, and ran CD, ARCS(CD), GES, ARCS(GES) and other methods (PC, HC, MMHC and GA) with a maximum time allowance of 10 minutes per dataset. The HC and MMHC algorithms had an upper-bound of the in-degree number as 2. We tried a higher maximum in-degree, but it resulted in a large FP. MMHC and PC were run with a significance level of 0.01 in conditional independence tests. We ran the CD algorithm with an MCP regularized

Table 2.1: Comparison between ARCS and initial estimates on observational data. ARCS improved GES and CD estimates, and ARCS(GES) achieved the best accuracy for every network.

Network	Method	$n < p$					$n > p$				
(p, s_0)		TP	R	FP	SHD (sd)	JI (sd)	TP	R	FP	SHD (sd)	JI (sd)
`Hailfinder`	ARCS(GES)	203	42	16	76 (25)	0.64 (0.11)	237	21	15	42 (13)	0.79 (0.06)
(224, 264)	ARCS(CD)	183	58	33	114 (18)	0.51 (0.06)	195	60	41	110 (26)	0.54 (0.07)
	GES	180	53	10	94 (22)	0.56 (0.10)	228	26	18	54 (12)	0.74 (0.05)
	CD	145	93	31	150 (17)	0.38 (0.05)	150	94	34	148 (13)	0.38 (0.04)
`Andes`	ARCS(GES)	274	33	27	91 (33)	0.69 (0.10)	295	27	26	70 (29)	0.76 (0.08)
(223, 338)	ARCS(CD)	228	75	65	174 (37)	0.48 (0.07)	238	72	66	166 (63)	0.51 (0.10)
	GES	218	36	17	137 (18)	0.56 (0.06)	271	35	36	103 (32)	0.67 (0.09)
	CD	169	112	63	232 (15)	0.33 (0.03)	184	112	70	223 (35)	0.36 (0.05)
`Hepar2`	ARCS(GES)	300	119	64	255 (27)	0.45 (0.04)	309	121	74	257 (41)	0.45 (0.05)
(280,492)	ARCS(CD)	263	155	82	312 (37)	0.36 (0.04)	284	152	87	294 (55)	0.39 (0.06)
	GES	238	110	62	316 (24)	0.36 (0.04)	299	126	84	277 (28)	0.43 (0.04)
	CD	205	156	100	388 (20)	0.27 (0.03)	229	168	121	384 (31)	0.29 (0.04)
`Win95pts`	ARCS(GES)	348	69	34	134 (21)	0.63 (0.05)	379	58	38	107 (19)	0.70 (0.04)
(304, 448)	ARCS(CD)	301	104	59	206 (22)	0.49 (0.04)	318	114	111	241 (51)	0.48 (0.07)
	GES	236	78	21	232 (17)	0.43 (0.04)	320	88	64	192 (17)	0.53 (0.03)
	CD	177	169	47	317 (26)	0.27 (0.04)	187	172	39	300 (26)	0.28 (0.04)
`Pigs`	ARCS(GES)	446	102	27	172 (90)	0.62 (0.13)	466	107	46	172 (40)	0.63 (0.05)
(441, 592)	ARCS(CD)	401	145	47	239 (73)	0.51 (0.10)	437	136	57	212 (55)	0.56 (0.06)
	GES	432	112	38	198 (18)	0.58 (0.03)	467	122	57	182 (23)	0.61 (0.04)
	CD	324	224	106	374 (26)	0.35 (0.02)	334	243	175	433 (43)	0.33 (0.03)
`rDAG1`	ARCS(GES)	289	7	1	12 (7)	0.94 (0.03)	297	3	1	4 (6)	0.98 (0.03)
(300,300)	ARCS(CD)	253	43	9	56 (14)	0.72 (0.06)	264	35	20	56 (20)	0.75 (0.08)
	GES	274	8	1	27 (6)	0.89 (0.03)	293	4	2	9 (8)	0.96 (0.04)
	CD	189	104	21	132 (18)	0.45 (0.05)	196	98	24	127 (17)	0.47 (0.05)
`rDAG2`	ARCS(GES)	562	20	12	51 (12)	0.89 (0.03)	584	13	11	27 (14)	0.94 (0.03)
(300,600)	ARCS(CD)	481	90	52	171 (41)	0.65 (0.06)	491	99	106	215 (62)	0.61 (0.06)
	GES	470	29	6	136 (20)	0.74 (0.04)	558	21	17	58 (15)	0.88 (0.03)
	CD	382	159	59	276 (32)	0.47 (0.03)	379	165	61	283 (25)	0.46 (0.03)

In this and all subsequent tables in this chapter, reported results are rounded averages over 20 datasets.

likelihood, in which $\gamma = 2$ and λ was chosen by a default model selection mechanism. GA was run for a maximum of 10^4 iterations, using the default population size and the default rates of mutation and crossover. We tried a larger population size for GA, but it was too time-consuming.

Our methods, ARCS(CD) and ARCS(GES), initialized with permutations from CD and GES estimates, were run for a maximum of $N = 10^4$ iterations, with $T^{(0)} = 1$, $T^{(N)} = 0.1$, and reversal length $m = 4$ (Algorithm 1). The temperature decreased geometrically as $T^{(i)} = \alpha^i T^{(0)}$, where α is determined by $T^{(0)}, T^{(N)}$ and N. A p-value cutoff of 10^{-5} was used in the refinement step (Algorithm 3). For the networks we considered, on average, 500 tests were performed in the refinement step, and the cutoff was chosen by Bonferroni correction to control the familywise error rate at level 0.005. In fact, our results were almost identical for any p-value cutoff between 10^{-3} and 10^{-5}.

To perform a complete comparison, we applied the structure refinement step (Section 2.4.3) to other competing methods as well. It turned out that ARCS, CD, HC and MMHC benefitted from this additional step. Therefore, in this section, we report the results of these methods with the refinement step. Section 2.5.6 evaluates the effect of this structure refinement step, including results of these methods before refinement.

ARCS versus CD and GES. Table 2.1 reports the average performance metrics across 20 datasets for 7 networks (5 real and 2 random networks) using CD, ARCS(CD), GES and ARCS(GES). We were interested in the potential improvement of ARCS upon its initial permutations. It is indeed confirmed by the results in the table that ARCS(CD) and ARCS(GES) outperformed CD and GES, respectively, for every network, achieving lower SHDs and higher JIs. The reduction in SHD was close to or above 20% across networks. ARCS always increased TP, while maintaining or slightly reducing FP. The annealing process identified more TP edges, while the refinement step (Algorithm 3) cut down the FP edges given the ordering and parent sets learned through simulated annealing. In the case of $n < p$ for the `Pigs` network, the high standard deviation of the SHDs of ARCS was caused by a couple of outliers. If we excluded the corresponding data sets, the average SHD of ARCS(GES) and ARCS(CD) would decrease to 153 and 225 with standard deviations of 23 and 41, respectively.

ARCS(GES) versus other methods. We also compared ARCS(GES) with other existing methods, including HC, PC, MMHC and GA. Table 2.2 summarizes the average performance, where ARCS(GES) outperforms competing algorithms by a great margin. The HC algorithm tended to output a denser DAG than the truth, leading to a large FP. The PC algorithm had a relatively large number of reverse edges, causing a high SHD. The MMHC algorithm had a lower SHD than some other algorithms, but the SHD difference between ARCS(GES) and MMHC was still large. The PC and MMHC algorithms were slow for some networks, and thus are absent in the results for these networks. The GA was formulated in a similar way as ARCS(GES), but the TPs of GA estimates were much lower, resulting in large SHDs for the tested networks.

It is worth mentioning that ARCS(GES) outperformed other methods substantially for larger networks such as `Pigs` ($p = 441, s_0 = 592$). MMHC and PC failed to complete a single run on the `Pigs` network within 10 minutes, while HC and GA had very low accuracies. We suspect that the `Pigs` network has a certain structure that is particularly difficult to estimate, a hypothesis that merits more investigation.

Additional tests. We also compared ARCS with DAGs with NO TEARS (NOTEARS), a continuous optimization method for structure learning of BNs developed recently Zheng et al (2018). This method is not restricted to continuous data, and thus more general than ARCS. The current version of NOTEARS code online (Zheng, 2019) required a very large amount of memory for large DAGs such as those in Tables 2.1 and 2.2 (on 2016 MacBook Pro, 2.9 GHz Intel Core i5, 16 GB memory). Therefore, we restricted the comparison to relatively small DAGs. To this end, we generated observational data from 9 graphs (with at most 20 nodes), and then applied grid search to choose NOTEARS's tuning parameter since it required a pre-fixed tuning parameter. In particular, we applied a grid search on $\{5\times10^{-3}, 10^{-3}, 5\times10^{-4}, 10^{-4}, 10^{-5}\}$, where the highest accuracy, measured by SHD, occurred at tuning parameter $\lambda = 5 \times 10^{-4}$.

Figure 2.3 compares performances of NOTEARS and ACRS(GES) on observational data

Table 2.2: ARCS against other methods on observational data. ARCS(GES) achieved the best SHD and JI among all methods for every network.

Network	Method	$n < p$					$n > p$				
(p, s_0)		TP	R	FP	SHD (sd)	JI (sd)	TP	R	FP	SHD (sd)	JI (sd)
`Hailfinder`	ARCS(GES)	203	42	16	76 (25)	0.64 (0.11)	237	21	15	42 (13)	0.79 (0.06)
(224, 264)	HC	151	80	31	144 (13)	0.40 (0.04)	151	87	50	163 (19)	0.38 (0.06)
	PC	41	167	13	236 (7)	0.09 (0.02)	45	191	14	233 (14)	0.10 (0.03)
	GA	51	70	72	284 (14)	0.13 (0.02)	43	68	28	248 (13)	0.12 (0.03)
`Andes`	ARCS(GES)	274	33	27	91 (33)	0.69 (0.10)	295	27	26	70 (29)	0.76 (0.08)
(223, 338)	HC	142	112	62	258 (19)	0.28 (0.03)	148	121	85	275 (27)	0.27 (0.04)
	PC	84	169	13	267 (10)	0.16 (0.02)	85	202	16	269 (10)	0.15 (0.02)
	GA	57	87	72	352 (14)	0.12 (0.02)	36	84	33	335 (9)	0.08 (0.02)
	MMHC	147	88	3	194 (15)	0.34 (0.05)	171	98	4	171 (16)	0.39 (0.05)
`Hepar2`	ARCS(GES)	300	119	64	255 (27)	0.45 (0.04)	309	121	74	257 (41)	0.45 (0.05)
(280, 492)	HC	170	149	83	405 (20)	0.24 (0.03)	174	155	95	413 (19)	0.23 (0.02)
	PC	96	154	30	427 (13)	0.14 (0.02)	108	189	33	417 (15)	0.15 (0.02)
	GA	70	107	143	565 (14)	0.09 (0.01)	51	98	77	518 (14)	0.08 (0.02)
	MMHC	86	156	20	426 (15)	0.13 (0.02)	130	158	21	383 (15)	0.19 (0.02)
`Win95pts`	ARCS(GES)	348	69	34	134 (21)	0.63 (0.05)	379	58	38	107 (19)	0.70 (0.04)
(304, 448)	HC	147	177	54	355 (19)	0.22 (0.03)	148	212	111	410 (17)	0.19 (0.02)
	PC	124	207	28	352 (12)	0.18 (0.02)	134	264	27	341 (12)	0.18 (0.02)
	GA	83	114	57	422 (12)	0.13 (0.02)	41	86	12	419 (7)	0.07 (0.01)
	MMHC	157	147	4	295 (12)	0.26 (0.02)	177	191	3	274 (11)	0.28 (0.02)
`Pigs`	ARCS(GES)	446	102	27	172 (90)	0.62 (0.13)	466	107	46	172 (40)	0.63 (0.05)
(441, 592)	HC	353	174	102	341 (30)	0.41 (0.04)	367	173	132	357 (41)	0.41 (0.04)
	GA	79	187	91	605 (12)	0.09 (0.01)	46	150	22	569 (10)	0.06 (0.01)
`rDAG1`	ARCS(GES)	289	7	1	12 (7)	0.94 (0.03)	297	3	1	4 (6)	0.98 (0.03)
(300, 300)	HC	176	108	29	153 (13)	0.40 (0.03)	178	113	45	168 (18)	0.39 (0.04)
	PC	90	202	30	240 (11)	0.17 (0.02)	83	215	31	248 (11)	0.15 (0.02)
	GA	64	81	33	269 (9)	0.15 (0.02)	54	75	6	253 (10)	0.14 (0.03)
	MMHC	193	89	1	108 (8)	0.50 (0.03)	191	103	0	109 (8)	0.47 (0.03)
`rDAG2`	ARCS(GES)	562	20	12	51 (12)	0.89 (0.03)	584	13	11	27 (14)	0.94 (0.03)
(300, 600)	HC	253	180	51	398 (20)	0.31 (0.02)	248	201	76	428 (17)	0.28 (0.02)
	PC	194	307	7	413 (16)	0.21 (0.02)	175	389	6	431 (13)	0.18 (0.02)
	GA	108	140	63	555 (15)	0.14 (0.02)	42	96	5	563 (7)	0.06 (0.01)
	MMHC	277	156	0	323 (10)	0.37 (0.02)	284	191	0	316 (17)	0.36 (0.03)

If a method is absent for a network, that means, it took more than 10 minutes to run on a singe dataset, and thus is excluded from the comparison.

generated from 9 graphs. ARCS achieved higher accuracy than NOTEARS consistently for most graphs (8 out of 9), except `Sachs` ($p = 11, s_0 = 17$). The advantage of ARCS over NOTEARS grew with the size and complexity of the true DAG.

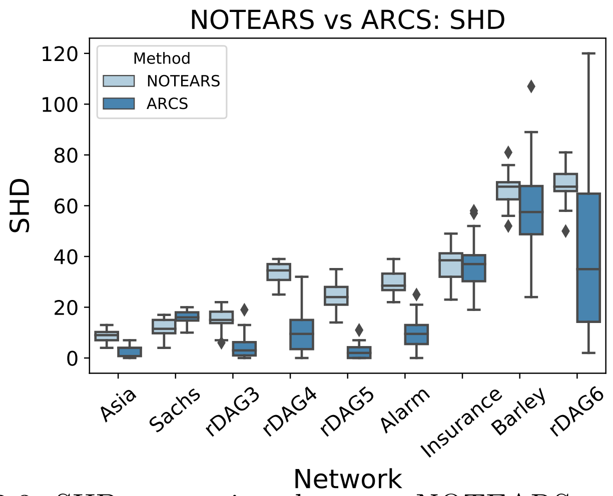


Figure 2.3: SHD comparison between NOTEARS and ARCS.

We also tested another order-based algorithm, linear structural equation model learning (LISTEN) (Ghoshal and Honorio, 2018), which estimates Gaussian DAG structure by a sequential detection of ordering. A key assumption of LISTEN is that the noise variables have equal variances. Moreover, the algorithm requires a prespecified regularization parameter for the score metric. To compare with this algorithm, we adapted our data generation process to satisfy the equal-variance assumption.

Code that implements LISTEN is available online (Ghoshal, 2019), which requires a pre-specified regularization parameter λ. We tried regularization values from $\{0.1, 0.01, 0.001\}$, where $\lambda = 0.001$ had the best performance. It is also the suggested value in the work of Ghoshal and Honorio (2018). The LISTEN code could not handle high-dimensional data, so we compared ARCS and LISTEN on the observational data with $n > p$ generated from `Hailfinder` and `Andes` networks as used in Table 2.1 in the main text. The performance of ARCS(GES) and LISTEN is shown in Figure 2.4. For `Hailfinder`, the average SHD of ARCS(GES) was 43, which was 12.42% of the SHD (346) achieved by LISTEN. For `Andes`, ARCS(GES) had an average SHD of 80, 18.18% of the average SHD (440) by LISTEN. This test demonstrates that ARCS(GES) outperformed LISTEN substantially on the tested networks.

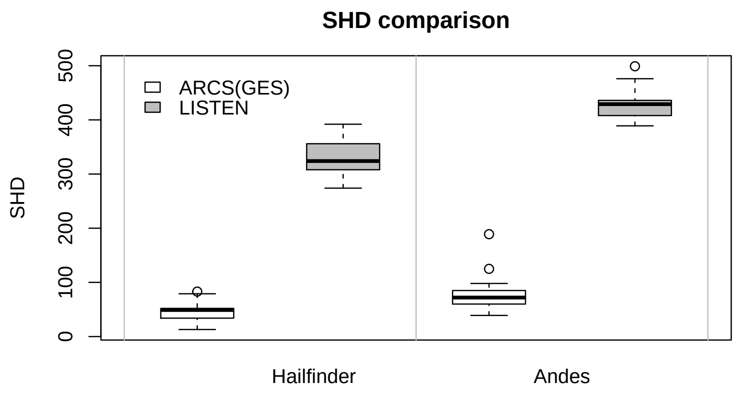


Figure 2.4: A comparison between ARCS(GES) and LISTEN.

2.5.4 Test data likelihood comparison

As discussed in Remark 2, an estimated DAG (or CPDAG) defines a multivariate Gaussian distribution for X via (2.3). To evaluate the accuracy in estimating the covariance matrix Σ_0, we compared test data log-likelihood as follows. Given an estimated DAG, we applied least-squares regression of each node on its parents to estimate B_0 and Ω_0. Then we simulated test data under the same Gaussian SEM with the true parameters (B_0, Ω_0) and calculated log-likelihood of the test data. A higher test data log-likelihood indicates a better estimate of Σ_0.

Figure 2.5 shows the test data log-likelihood of a few BN learning methods, including ARCS(GES), GES, CD, HC and GA, on four networks in the case of $n < p$. The case of $n > p$ shows a similar pattern and is thus omitted for brevity. We shifted test data log-likelihoods by the median of ARCS, so that the ARCS medians were always zero. The log-likelihood distributions for all other methods were below zero, indicating that ARCS achieved significantly higher accuracy in estimating Σ_0. In all these results, we normalized the log-likelihood by the sample size n and the dimension of the problem p.

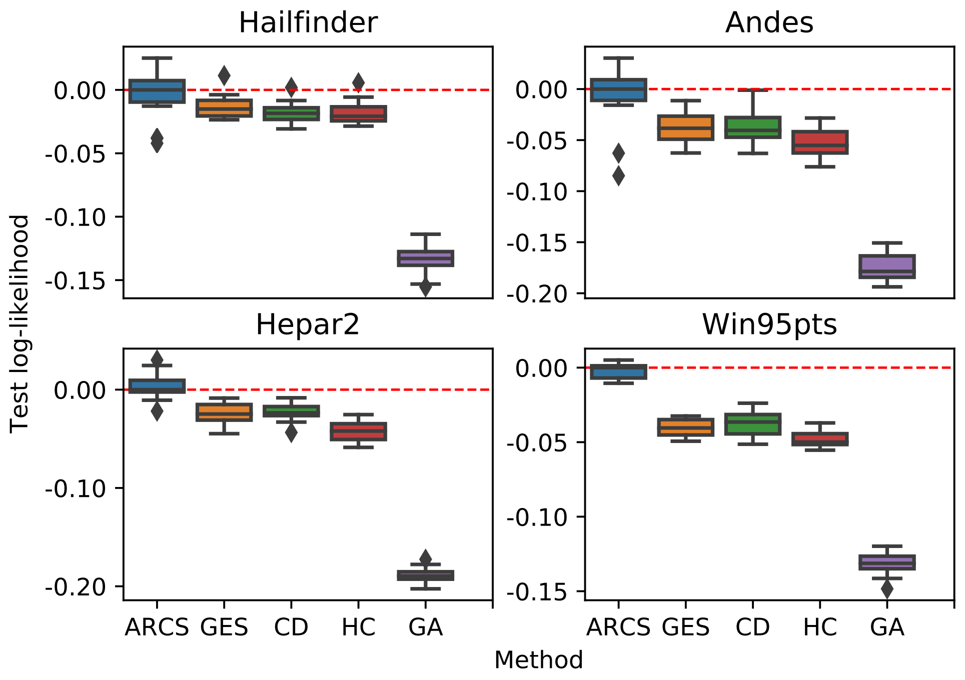


Figure 2.5: Test data log-likelihood comparison among BN learning methods. Log-likelihoods are shifted by the median of ARCS (the dashed line).

2.5.5 Precision matrix estimation

For a given P, the minimizer of the RC loss $f(L; P)$ (2.12) is a sparse Cholesky factor of $P\Sigma_0^{-1}P^\top$, the precision matrix Σ_0^{-1} after permuting rows and columns according to P. Instead of joint optimization over (L, P) in (2.13), one could first estimate a sparse precision matrix $\widehat{\Theta}$ and then learn a DAG structure though Cholesky decomposition of $\widehat{\Theta}$ as in (2.3). The Cholesky decomposition can be done with a given ordering or by searching for an order that leads to a sparse Cholesky factor. In this section, we compare ARCS(GES), or simply ARCS, against two precision matrix estimation methods, followed by Cholesky decomposition, to clarify their differences.

The first approach estimates a precision matrix via modified Cholesky decomposition (MCD) (Lee and Lee, 2018) which requires an input ordering of the variables. We provided MCD with ARCS initial and final orders, which we have called MCD and MCD*, respectively. The initial order of ARCS was found by GES. Given an estimated sparse precision matrix and the input order, we found the corresponding DAG structure encoded by the Cholesky factor, and then refined the DAG structure using Algorithm 3, the same refinement step used by ARCS. The comparison against MCD will show the advantage of joint minimization over (L, P), while the comparison against MCD* demonstrates the effectiveness of ARCS in terms of estimating precision matrices, since the two methods use exactly the same ordering in Cholesky decomposition.

The second approach is graphical Lasso (Friedman et al, 2008) followed by a Cholesky decomposition. Given a sparse precision matrix $\widehat{\Theta}$ estimated by graphical Lasso, we used the approximate minimum degree ordering algorithm (Amestoy et al, 1996), a heuristic common in numerical linear algebra, to find a permutation P such that the Cholesky factor of $P\widehat{\Theta}P^{\top}$ is sparse. We call this approach GC. We also applied the Cholesky decomposition to $\widehat{P}\widehat{\Theta}\widehat{P}^{\top}$, where $\widehat{P}$ is the permutation found by ARCS. We call this approach GC*. The comparison between ARCS and GC will highlight the importance of imposing sparsity on L in our formulation (2.13), rather than on Σ^{-1} as in GC. The difference between GC* and GC will show the effectiveness of the ordering obtained by ARCS, relative to the minimum degree ordering, in achieving a sparse Cholesky factorization.

We used R package `CovTools` to run MCD, the `glasso` package to run graphical Lasso, and MATLAB function `chol` for the approximate minimum degree algorithm. Based on a grid search on a sample dataset, we chose the tuning parameter of 0.001 for graphical Lasso. The upper bound on the bandwidth of the precision matrix, as required by `CovTools`, was set to 10.

Table 2.3 compares the structure recovery accuracy among ARCS, GC*, GC, MCD* and MCD. ARCS showed the best performance with a great margin, followed by GC* which used the permutation found by ARCS to complete the Cholesky decomposition. The observation that GC* consistently outperformed GC indicates that ARCS found a better topological sort than the minimum degree algorithm. Both MCD and MCD* suffered from high SHDs and low JIs, which shows that imposing sparsity on the precision matrix and decoupling its estimation from the order search would be suboptimal for DAG structure learning.

We also compared test data log-likelihood among these methods in Figure 2.6. Interest-

Table 2.3: ARCS against precision matrix estimation methods

Network	Method	$n < p$		$n > p$	
(p, s_0)		SHD(sd)	JI(sd)	SHD(sd)	JI(sd)
Hailfinder	ARCS	76 (25)	0.64 (0.11)	42 (13)	0.79 (0.06)
(224, 264)	GC*	128 (15)	0.53 (0.06)	82 (15)	0.64 (0.06)
	GC	297 (10)	0.09 (0.01)	236 (19)	0.16 (0.05)
	MCD*	290 (7)	0.03 (0.02)	304 (11)	0.06 (0.02)
	MCD	285 (10)	0.06 (0.02)	300 (12)	0.10 (0.03)
Andes	ARCS	91 (33)	0.69 (0.10)	70 (29)	0.76 (0.08)
(223, 338)	GC*	98 (24)	0.67 (0.08)	169 (19)	0.51 (0.05)
	GC	597 (114)	0.04 (0.05)	266 (75)	0.29 (0.17)
	MCD*	335 (16)	0.07 (0.05)	345 (25)	0.11 (0.04)
	MCD	322 (14)	0.09 (0.04)	321 (15)	0.14 (0.03)
Hepar2	ARCS	255 (27)	0.45 (0.04)	257 (41)	0.45 (0.05)
(280, 492)	GC*	392 (24)	0.32 (0.03)	328 (35)	0.38 (0.04)
	GC	546 (27)	0.16 (0.02)	497 (28)	0.19 (0.02)
	MCD*	594 (16)	0.02 (0.01)	659 (22)	0.02 (0.01)
	MCD	557 (13)	0.07 (0.02)	596 (15)	0.09 (0.02)
Win95pts	ARCS	134 (21)	0.63 (0.05)	107 (19)	0.70 (0.04)
(304, 448)	GC*	160 (17)	0.58 (0.04)	402 (11)	0.10 (0.02)
	GC	390 (14)	0.16 (0.02)	428 (7)	0.06 (0.01)
	MCD*	450 (4)	0.01 (0.01)	462 (7)	0.03 (0.01)
	MCD	446 (5)	0.03 (0.01)	453 (9)	0.06 (0.01)

ingly, although not designed for precision matrix estimation, ARCS achieved a much higher test data likelihood compared to all the precision matrix estimation methods. In particular, the fact that ARCS outperformed MCD* implies that our method did a better job at estimating precision matrices compared to MCD, even though the latter was provided with the same ordering finally used by ARCS.

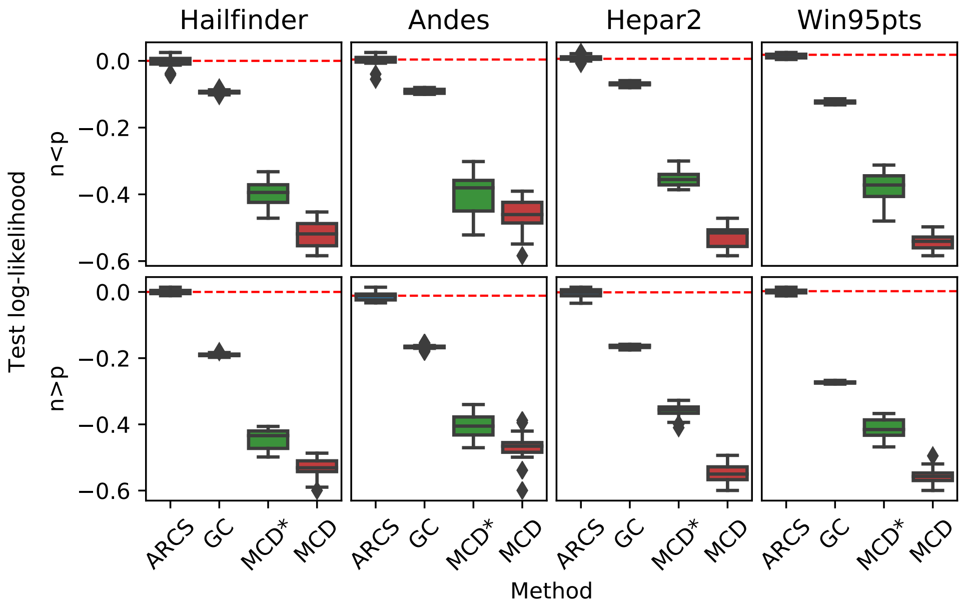


Figure 2.6: Test data log-likelihood comparison among ARCS and precision matrix estimation methods.

2.5.6 Effectiveness of refinement

Recall that we employed the constraint-based refinement step in ARCS after annealing (Section 2.4.3). To analyze the benefit of this post-processing step, we applied the same refinement step to other methods except PC and GA. We excluded PC because as a constraint-based method, it has already performed all conditional independence tests. The GA estimated graphs had the lowest TPs in Table 2.2, and thus further removing edges by the refinement step would not improve its overall accuracy. Note that GES outputs a CPDAG, so we applied the refinement step on a randomly chosen DAG in the equivalence class of the estimated CPDAG.

We summarize the mean SHDs before and after the refinement step for each method in Table 2.4. The refinement step reduced the SHDs for most of the methods included in this comparison. It worked particularly well for HC with a substantial decrease in SHD. The improvements for ARCS, CD and MMHC were quite substantial for some datasets while marginal for other datasets. There was no change in the GES estimates after the refinement

Table 2.4: Comparison on SHDs before (B) and after (A) the refinement step.

SHD	ARCS(GES)		ARCS(CD)		CD		HC		MMHC		GES
	B	A	B	A	B	A	B	A	B	A	
`Hailfinder`	129	76	165	114	151	150	344	144	–	–	94
`Andes`	125	91	256	174	237	232	356	258	203	194	137
`Hepar2`	290	255	425	312	411	388	513	405	430	426	316
`Win95pts`	205	134	293	206	319	317	544	355	316	295	232
`Pigs`	193	172	263	239	974	374	547	341	–	–	198
`rDAG1`	13	12	62	56	137	132	437	153	143	108	27
`rDAG2`	68	51	200	171	389	276	483	398	328	323	136
`Hailfinder`	43	42	112	110	148	148	335	163	–	–	54
`Andes`	71	70	174	166	224	223	341	275	181	171	103
`Hepar2`	273	257	304	294	385	384	505	413	389	383	277
`Win95pts`	110	107	244	241	300	300	519	410	295	274	192
`Pigs`	173	172	214	212	436	433	518	357	–	–	182
`rDAG1`	5	4	56	56	128	127	420	168	146	109	9
`rDAG2`	27	27	221	215	283	283	481	428	320	316	58

The top panel shows results for $n < p$ and the bottom for $n > p$.

step. It is observed that ARCS(GES) achieved the smallest SHD, after the refinement step was applied to all the methods. Moreover, ARCS(GES) without refinement had already outperformed the majority of the competing methods, except for the case `Hailfinder` ($n < p$) when compared against GES.

2.5.7 Effectiveness of BIC selection

Given an initial permutation, we used the BIC to choose tuning parameters (γ, λ) before applying the ARCS algorithm (Section 2.4.4). In Tables 2.2, the number of predicted edges (TP+R+FP) by ARCS(GES) is closer to s_0 than any other competing method in every

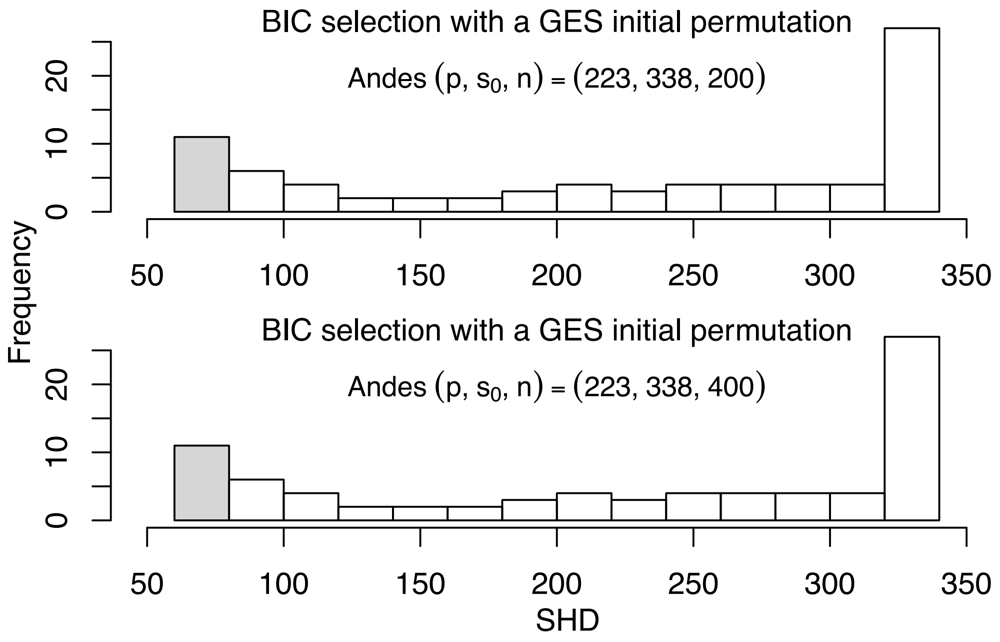


Figure 2.7: Performance of the BIC selected parameter among a grid of (γ, λ) given an initial permutation. Tuning parameters that lead to lower SHDs than the BIC selection are shown in gray.

network. This observation signifies the effectiveness of our parameter selection method by BIC. To further study its effect, we compared DAGs estimated with all values on a grid of (γ, λ) by ARCS(GES), or simply ARCS.

Figure 2.7 shows the histograms of the SHDs of ARCS estimates for a grid of tuning parameters. We used the `Andes` datasets with $(p, s_0) = (223, 338)$ and $n \in \{200, 400\}$ here. The shaded part of a histogram reports SHDs achieved with choices of tuning parameters on the grid that were smaller than the SHD corresponding to the BIC selected parameters. Each histogram has a high spike of large SHD, corresponding to large values of λ that generate almost empty graphs. The SHDs of ARCS with BIC selection corresponded to the 16th and 3rd percentiles in low and high sample sizes, respectively. These low percentiles confirm that the BIC selection works well for choosing the tuning parameters. Moreover, in our tests, the BIC usually selected $\gamma^* = 2$, the smallest provided value for γ. Since for small γ, the MCP is closer to the ℓ_0 penalty and far from the ℓ_1 norm, this choice of γ indicates the preference of concave penalties over ℓ_1 in estimating sparse DAGs. Some of CD's and GA's inferior performances, such as CD on the `Pigs` network (Table 2.1) and the overall performance of GA (Tables 2.2), were potentially caused by a bad choice of their tuning parameters. This demonstrates the importance of our data-driven selection scheme for a regularized likelihood method.

2.5.8 Empirical loss evaluation

To quantify the empirical loss and global search ability of our ARCS algorithm, we compared the BIC score of an ARCS estimate against the global minimum BIC score. A global minimum can be identified by integer linear programming (ILP) Cussens et al (2017). With a properly chosen upper bound on the parent size, and given sufficient computing budget, this method is guaranteed to find a global minimizer. We set the upper bound to the maximum parent size in the DAG estimated by ARCS(GES), and then calculated local BIC scores of all possible parent sets for each node, which were input to ILP implemented in the `GOBNILP` software package Cussens et al (2017).

We chose relatively small DAGs ($p \leq 20$) in this comparison and simulated datasets with sample size $n = 5p$. For each dataset, we computed the BIC score of the DAG estimated by ARCS(GES) and found the minimum BIC score by ILP. Define the relative BIC increase of ACRS compared to ILP as (BIC(ARCS) − BIC(ILP))/BIC(ILP). Figure 2.8 shows the relative increase distribution across 20 datasets generated from each of the selected DAGs. Since an upper bound on parent set size has been given to ILP, its minimum score may not be best possible BIC score. As expected, the BIC score of ARCS was slightly higher than the minimum score found by ILP. However, it is comforting to see the relative BIC increase of ARCS was less than 2% for almost all the datasets. Note that the loss function of ARCS may be quite different from the BIC when n is small. This is the reason why ARCS had the largest relative BIC increase for the smallest 8-node graph `Asia` for which the sample size $n = 40$.

Note that ILP is computationally intensive. In our tests, we allocated 20 hours to ILP for each single dataset generated from four graphs, the three reported in Figure 2.8 and `rDAG4`

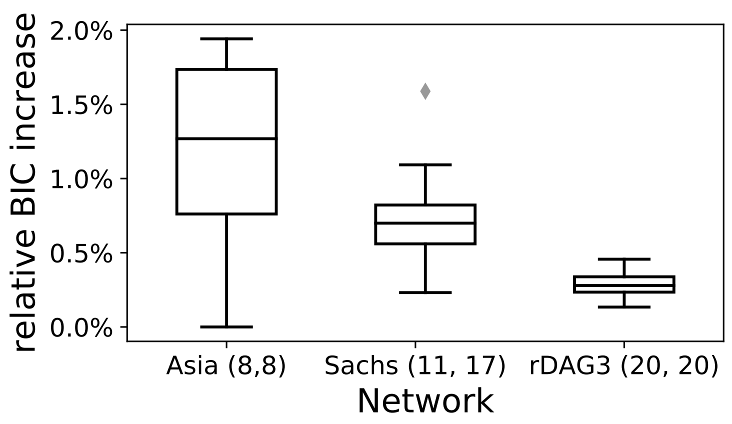


Figure 2.8: Distributions of the relative BIC increase. The numbers following each DAG report its (p, s_0).

$(p = 20, s_0 = 40)$. Yet it still failed to find an optimum within the time constraint for the largest graph `rDAG4`. Therefore, we excluded this graph from the figure. Meanwhile ARCS took less than one minute to complete every single run for these graphs.

2.6 Results on experimental data

To generate experimental datasets, we generated p blocks of observations, in each of which a single variable was under intervention. For each block, we generated 5 observations, and thus $n = 5p$. Networks in this experiment were smaller, with $p \leq 50$ and $s_0 \leq 100$ (see Table 2.5). Using these networks, we also simulated observational data of the same sample size, $n = 5p$, to study the effect of experimental interventions. Since the number of conditional independence tests were smaller in this setting, we used a p-value cutoff of 10^{-3} in each refinement test of the ARCS algorithm (Algorithm 3) to control the overall false discovery rate.

To assess the accuracy on experimental data, we compare an estimated DAG with the true one to calculate the numbers of false positives (FP), missing edges (M), reverse edges (R) and true positives (TP). FP and M follow the same calculations as in the observational settings. R counts the number of edges whose orientations are opposite between the two DAGs, and TP = P − R − FP. Note that the definitions of R and TP are different from those for observational data, because under the intervention setting used in this comparison, the true causal DAG is identifiable (Fu and Zhou, 2013). The structural Hamming distance (SHD) and the Jaccard index (JI) are then calculated as in the observational case.

2.6.1 Comparison on experimental data

In this setting, we compared ARCS with the CD algorithm (Fu and Zhou, 2013) and the greedy interventional equivalence search (GIES) algorithm (Hauser and Bühlmann, 2012), both of which can handle experimental interventions. We initialize ARCS with CD and GIES estimates and call them ARCS(CD) and ARCS(GIES), respectively.

Table 2.5 compares the CD, ARCS(CD), GIES and ARCS(GIES) algorithms, averaging over 20 datasets for each type of networks. Both ARCS(GIES) and ARCS(CD) achieved dramatic improvements upon GIES and CD algorithms for every single network. This observation is consistent with the findings from the observational data and further confirms that the ARCS algorithm is a powerful tool for improving local estimates. Different from the observational data results (Table 2.1), ARCS(CD) usually had better performance than ARCS(GIES) on experimental data. The standard deviation, relative to the mean, of the results here was comparable to that of the observational data results, and thus is not reported in Table 2.5 for brevity.

We also compared the performance of our method ARCS(CD) on experimental and observational data with the same sample size $n = 5p$. Figure 2.9 plots the SHDs of ARCS(CD) on 20 datasets, with a side-by-side comparison between observational and experimental data. For some networks, such as `rDAG6` and `Ins.`, the estimated DAGs using experimental data had much lower SHDs than using the observational data. For some small networks, such as `Asia` ($p = 8, s_0 = 8$), ARCS(CD) achieved a low SHD on observational data and the improvement when using experimental data was not substantial.

Because estimated DAGs did not have the same number of predicted edges, we further compared the reversed edge proportion (R/P). The DAGs estimated by ARCS(CD) had a lower R/P on the experimental than the observational data for all networks. The decrease in R/P with experimental interventions was remarkable, as Figure 2.9 shows. This finding

Table 2.5: Performance comparison on experimental data.

Network (p, s_0, n)	Method	P	TP	R	FP	SHD	JI
`Asia`	CD	14	6	1	7	8	0.47
$(8, 8, 40)$	ARCS(CD)	5	5	1	0	**4**	0.53
	GIES	9	1	6	2	9	0.07
	ARCS(GIES)	5	5	1	0	**4**	**0.53**
`Sachs`	CD	23	9	5	10	17	0.34
$(11, 17, 55)$	ARCS(CD)	12	10	2	1	8	0.50
	GIES	17	4	10	4	17	0.13
	ARCS(GIES)	12	10	2	1	**8**	**0.51**
`Ins.`	CD	56	30	12	14	36	0.39
$(27, 52, 135)$	ARCS(CD)	54	40	7	7	**18**	**0.63**
	GIES	72	14	34	24	62	0.13
	ARCS(GIES)	57	38	10	10	24	0.56
`Alarm`	CD	51	35	9	7	16	0.57
$(37, 46, 185)$	ARCS(CD)	48	43	3	2	**5**	**0.86**
	GIES	70	6	40	24	65	0.05
	ARCS(GIES)	51	40	6	6	12	0.70
`Barley`	CD	85	52	17	16	48	0.45
$(48, 84, 240)$	ARCS(CD)	103	67	13	23	**41**	**0.58**
	GIES	146	21	58	67	129	0.10
	ARCS(GIES)	122	44	35	44	84	0.28
`rDAG5`	CD	55	39	10	6	17	0.60
$(50, 50, 250)$	ARCS(CD)	51	48	2	1	**3**	**0.90**
	GIES	87	7	43	37	80	0.05
	ARCS(GIES)	52	47	3	2	5	0.86
`rDAG6`	CD	101	70	17	13	43	0.54
$(50, 100, 250)$	ARCS(CD)	106	94	5	7	**12**	**0.86**
	GIES	155	15	81	58	143	0.06
	ARCS(GIES)	159	59	36	64	105	0.34

The best SHD and JI for each network are highlighted in boldface.

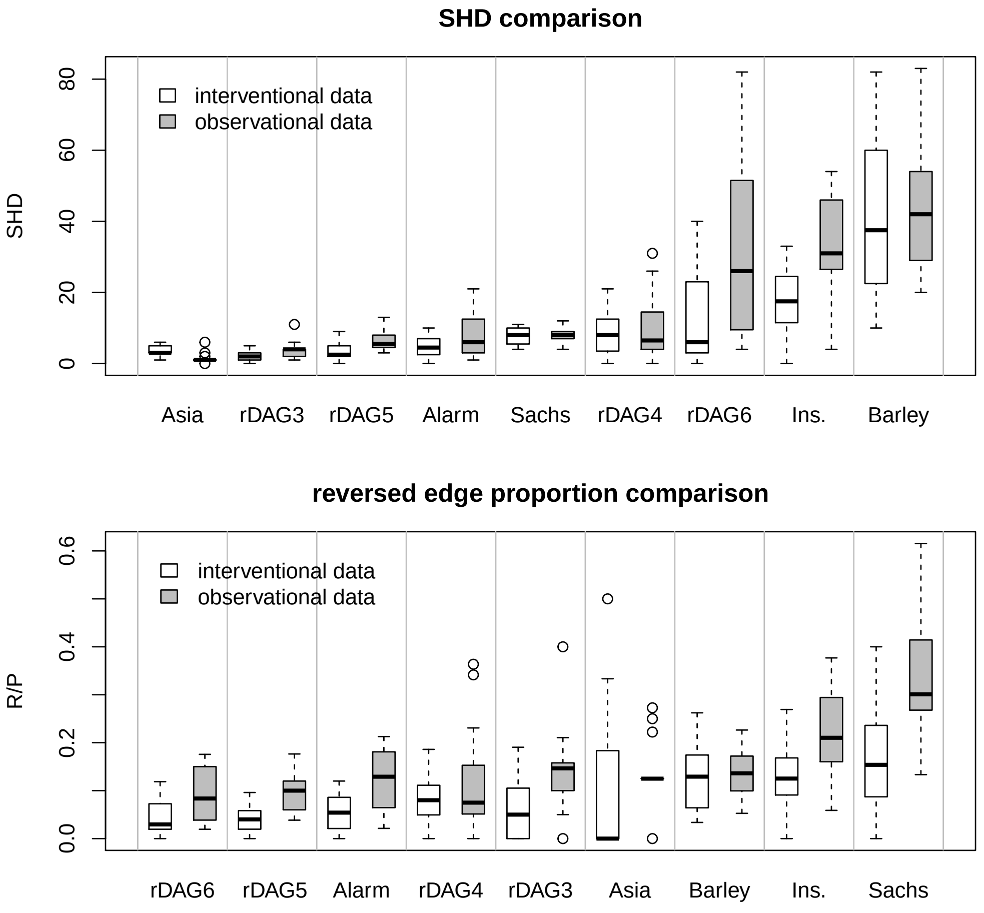


Figure 2.9: Comparison of SHD and reversed edge proportion between experimental and observational data with ARCS(CD).

supports the idea that experimental interventions help correct the reversed edges and distinguish equivalent DAGs. Note that for the 20 observational datasets generated from `Asia` ($p = 8$, $s_0 = 8$), ARCS(CD) output 16 estimated DAGs with P = 8 and R = 1, resulting in a very thin interquantile range in the boxplots of `Asia` in Figure 2.9.

2.6.2 Random initialization with a high temperature

Recall that we started ARCS(CD) and ARCS(GIES) with $T^{(0)} = 1$. To test its global search ability over the permutation space, we may initialize the annealing process with a random permutation and a high temperature, which we denote by ARCS(RND). For a random initial

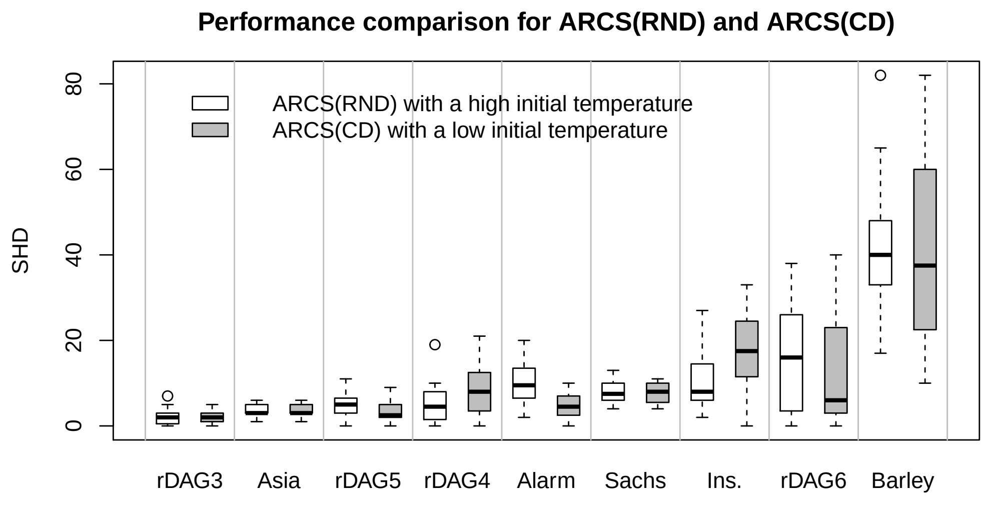


Figure 2.10: A comparison between ARCS(RND) with a high initial temperature and ARCS(CD) on experimental data.

permutation, we do not need to preserve its properties, so we use a high initial temperature $T^{(0)} = 100$ to help the algorithm traverse the search space.

We compared ARCS(RND) and ARCS(CD) on experimental data for 9 networks. We chose ARCS(CD) due to its superior performance on these networks in the experimental setting (Table 2.5). As shown in Figure 2.10, ARCS(RND) and ARCS(CD) had comparable performances on all networks. Both of them learned DAGs with small SHDs. ARCS(RND) was slightly better on `rDAG4` and `Ins.`, and slightly worse on `Alarm` and `rDAG6`. For the other networks, the two methods were quite comparable, demonstrating the effectiveness of ARCS(RND).

The networks in this experiment had $p \leq 50$. For $p = 50$, there are $50! \approx 3 \times 10^{64}$ possible permutations. ARCS(RND) managed to learn a network structure within 10^4 iterations, which is much smaller than $p!$. However, the performance of ARCS(RND) was not competitive on large networks. The reason is that for large p, it takes much more time for the annealing to thoroughly search the huge permutation space. Therefore, for large networks, it is better to initialize the ARCS algorithm with estimates from other local methods and choose a low temperature. Given a good initial estimate, ARCS searches over the permutation space and improves the accuracy of the initial estimate as demonstrated in Tables 2.1 and 2.5. This study suggests that, by searching over the permutation space under a regularized likelihood framework, our ARCS algorithm is a promising approach to the challenging problem of DAG structure learning.

2.7 Discussion

In this chapter, we have developed a method to learn Gaussian BN structures by minimizing the MCP regularized Cholesky score over topological sorts, through a joint iterative update on a permutation matrix and a lower triangular matrix. We search over the permutation space and optimize the network structure encoded by a lower triangular matrix given a topological sort. This approach relates BN learning problem to sparse Cholesky factorization, and provides an alternative formulation for the order-based search. The proposed regularized Cholsky score is shown to be consistent for estimating topological sorts and DAG structures. Our method can serve as an improvement of a local search or a stand-alone method with a best-guess initial permutation. Although we formulated this order-based search for Gaussian BNs, it can potentially be extended to discrete BNs and other scoring functions. A main difference in the extension to discrete data is the proximal gradient step, where we can borrow the regularized multi-logit model in Gu et al (2019) or develop a continuous regularizer for multinomial likelihood.

With a proper temperature schedule, simulated annealing may search over the permutation space effectively, and there are several interesting aspects to investigate in the annealing process. For instance, various operators of moving from one permutation to another have been proposed for greedy order-based search, which could better guide the annealing process in traversing the search space. The numerical results in this paper demonstrate the advantage and potential application of local search and global annealing in learning BNs. Left as future work are theoretical properties of our method in high-dimensional settings, such as the consistency of the regularized Cholesky score when $p \gg n$.

2.8 Proofs

The proof of the consistency result (Section 2.3) is provided in Sections 2.8.1, which uses some auxiliary results (Section 2.8.2). The consistency proofs are joint work in (Ye et al, 2020). Section 2.8.3 consists of proofs of other results in this chapter.

2.8.1 Consistency proof

For a matrix B, $\|B\|_F$ denotes its Frobenius norm and $\|B\|$ its ℓ_2 operator norm. We also use $\|B\|_{\max} = \max_{i,j} |B_{ij}|$. We start with some intermediate results. The following shows that for suitable regularizers, the I-map B_π is also the minimizer of the population version of the regularized Cholesky loss:

Proposition 2. *(Population consistency) Under* (2.20)*, for all* π *and* $B \in \mathfrak{D}_\pi$*,*

$$\mathcal{L}_n(B;\Sigma_0) - \mathcal{L}_n(B_\pi;\Sigma_0) \geq \frac{\lambda_{\min}}{2} \min\Big\{\|B - B_\pi\|_F^2, b_n^2 - \frac{2}{\lambda_{\min}}\frac{a_n}{n}|S_\pi|\Big\}. \tag{2.30}$$

If, in addition, (2.21) *holds, then for any* $\varepsilon > 0$*, when* n *is sufficiently large, we have*

$$\inf_{\pi} \inf_{B\in\mathfrak{D}_\pi:\, \|B-B_\pi\|_F \geq \varepsilon} \Big[\mathcal{L}_n(B;\Sigma_0) - \mathcal{L}_n(B_\pi;\Sigma_0)\Big] > 0. \tag{2.31}$$

In particular, B_π *is the unique solution of* (2.23) *for all* π*, and*

$$F_{n,\pi} = \big[\min_{B\in\mathfrak{D}_\pi} \mathscr{L}_{chol}(B;\Sigma_0)\big] + \frac{a_n}{n}|S_\pi| = \mathscr{L}^*_{chol}(\Sigma_0) + \frac{a_n}{n}|S_\pi|.$$

Using Proposition 2, we can show that the estimated DAG $\widehat{B}_{n,\pi}$ is consistent for B_π:

Proposition 3. *(Consistency) Under assumptions* (2.20) *and* (2.21)*,*

(a) $\widehat{B}_{n,\pi}$ *is* $n^{1/4}$*-consistent, uniformly in* π*.*

(b) If in addition (R2) holds, then $\widehat{B}_{n,\pi}$ *is a* $\sqrt{n}$*-consistent estimator of* B_π*, that is,* $\widehat{B}_{n,\pi} = B_\pi + O_p(n^{-1/2})$*, uniformly in* π*. Moreover,*

$$\rho_n(\widehat{B}_{n,\pi}) - \rho_n(B_\pi) = O_p\Big(\frac{\lambda_n}{\sqrt{n}}\Big) + O_p(1). \tag{2.32}$$

Here, and elsewhere, statement such as "$\widehat{B}_{n,\pi} = B_\pi + O_p(n^{-1/2})$, uniformly in π" should be interpreted as $\sup_\pi |\widehat{B}_{n,\pi} - B_\pi| = O_p(n^{-1/2})$.

We also need the following formula for the (unrestricted) Hessian of the Cholesky loss, a $p^2 \times p^2$ matrix,

$$\nabla^2 \mathscr{L}_{\text{chol}}(L; A) = I_p \otimes A + \operatorname{diag}\Big(\frac{1}{L_{ii}^2} e_i e_i^T, i = 1, \dots, p\Big) \tag{2.33}$$

where $\otimes$ is the Kronecker product and $e_i \in \mathbb{R}^p$ is ith basis vector of $\mathbb{R}^p$. The second term is a block diagonal matrix whose ith diagonal block is $e_i e_i^T / L_{ii}^2$, that is, a diagonal matrix with only a single nonzero entry, $1/L_{ii}^2$, on its diagonal at the ith position. See Section 2.8.2.2 for details.

Proof of Theorem 2. By Proposition 2, B_π is the solution of (2.23), hence

$$\begin{aligned} |\widehat{F}_{n,\pi} - F_{n,\pi}| &= |\mathcal{L}_n(\widehat{B}_{n,\pi}; \widehat{\Sigma}_n) - \mathcal{L}_n(B_\pi; \Sigma_0)| \\ &\le |\mathcal{L}_n(\widehat{B}_{n,\pi}; \widehat{\Sigma}_n) - \mathcal{L}_n(\widehat{B}_{n,\pi}; \Sigma_0)| + |\mathcal{L}_n(\widehat{B}_{n,\pi}; \Sigma_0) - \mathcal{L}_n(B_\pi; \Sigma_0)|. \end{aligned}$$

The first term is bounded above by $\frac{1}{2}|\operatorname{tr}(\widehat{B}_{n,\pi}\widehat{B}_{n,\pi}^T(\widehat{\Sigma}_n - \Sigma_0))| = O_p(n^{-1/2})$ by CLT and since $\|\widehat{B}_{n,\pi}\|_F = O_p(1)$. The second term above, call it T_2, is bounded above by

$$\begin{aligned} T_2 &\le |\mathscr{L}_{\text{chol}}(\widehat{B}_{n,\pi}; \Sigma_0) - \mathscr{L}_{\text{chol}}(B_\pi; \Sigma_0)| + O_p\Big(\frac{\lambda_n}{n}\frac{1}{\sqrt{n}}\Big) + O_p\Big(\frac{1}{n}\Big) \\ &\le \Big(\|\Sigma_0\| + \frac{4}{\min_i (B_\pi)_{ii}^2}\Big)\|\widehat{B}_{n,\pi} - B_\pi\|_F^2 + O_p\Big(\frac{1}{\sqrt{n}}\Big) \end{aligned}$$

where the first equality uses (2.32) in Proposition 3 and the second inequality is a straightforward consequence of $\nabla \mathscr{L}_{\text{chol}}(B_\pi; \Sigma_0) = 0$ and the Hessian formula (2.33), assuming that n is sufficiently large so that $(\widehat{B}_{n,\pi})_{ii} \ge \frac{1}{2}(B_\pi)_{ii}$. Since $\|\widehat{B}_{n,\pi} - B_\pi\|_F^2 = O_p(n^{-1})$ by Proposition 3, we have $T_2 = O_p(n^{-1/2})$. It follows that $|\widehat{F}_{n,\pi} - F_{n,\pi}| = O_p(n^{-1/2})$ uniformly in π.

Let π^* be such that $B_{\pi^*} \in \mathcal{E}^*$ and consider π such that $B_\pi \notin \mathcal{E}^*$. Then,

$$\begin{aligned} \widehat{F}_{n,\pi} - \widehat{F}_{n,\pi^*} &\ge F_{n,\pi} - F_{n,\pi^*} - 2 \max_{\pi' \in \{\pi, \pi^*\}} |\widehat{F}_{n,\pi'} - F_{n,\pi'}| \\ &= \big[\mathscr{L}^*_{\text{chol}}(\Sigma_0) + \frac{a_n}{n}|S_\pi|\big] - \big[\mathscr{L}^*_{\text{chol}}(\Sigma_0) + \frac{a_n}{n}|S^*_\pi|\big] - O_p(n^{-1/2}) \\ &= \frac{a_n}{n}(|S_\pi| - |S^*_\pi|) - O_p(n^{-1/2}) \\ &\ge \frac{a_n}{n} - O_p(n^{-1/2}) \end{aligned}$$

since by score-identifibility $|S_\pi| \geq |S_{\pi^*}| + 1$. Since $n^{-1/2}a_n \to \infty$ by assumption, for sufficiently large n, we have $\widehat{F}_{n,\pi} - \widehat{F}_{n,\pi^*} > 0$ for all π such that $B_\pi \notin \mathcal{E}^*$. This proves permutation consistency. For the second assertion, since $\widehat{B}_{n,\pi}$ is within a neighborhood of radius $O_p(n^{-1/2})$ around B_π, an argument similar to that of Lemma 1 in Fan and Li (2001) shows that $\mathrm{supp}(\widehat{B}_{n,\pi}) = \mathrm{supp}(B_\pi)$ for all π w.h.p. In particular, $\mathrm{supp}(\widehat{B}_{n,\widehat{\pi}_n}) = \mathrm{supp}(B_{\widehat{\pi}_n})$, w.h.p. and the proof is complete. $\square$

2.8.2 Proofs of auxiliary results

2.8.2.1 *Proof of Proposition 2*

Noting that $\lambda_{\min}(P_\pi \Sigma_0 P_\pi^T) = \lambda_{\min}(\Sigma_0) =: \lambda_{\min}$, we have by Proposition 1,

$$\mathscr{L}_{\mathrm{chol}}(B;\Sigma_0) - \mathscr{L}_{\mathrm{chol}}(B_\pi;\Sigma_0) \geq \frac{\lambda_{\min}}{2}\|B - B_\pi\|_F^2. \tag{2.34}$$

Consider the set

$$\mathcal{B} := \{B \in \mathfrak{D}_\pi : \ |B_{ij}| < b_n \text{ for some } (i,j) \in S_\pi\}.$$

For any $B \in \mathcal{B}^c$, at least $|S_\pi|$ of its elements are in the flat part of the penalty, whence $\rho_n(B) \geq a_n|S_\pi|$. Since by assumption (2.20), all the nonzero elements of B_π are in the flat part of the penalty, $\rho_n(B_\pi) = a_n|S_\pi|$. Thus, for any $B \in \mathcal{B}^c$,

$$\mathcal{L}_n(B) - \mathcal{L}_n(B_\pi) \geq \mathscr{L}_{\mathrm{chol}}(B;\Sigma_0) - \mathscr{L}_{\mathrm{chol}}(B_\pi;\Sigma_0)$$

which combined with (2.34), gives one of the inequalities in (2.30).

Now consider $B \in \mathcal{B}$. Then, $|B_{ij}| < b_n$ for some $(i,j) \in S_\pi$. Since $|(B_\pi)_{ij}| \geq 2b_n$ by assumption (2.20), it follows from (2.34) that

$$\mathscr{L}_{\mathrm{chol}}(B;\Sigma_0) - \mathscr{L}_{\mathrm{chol}}(B_\pi;\Sigma_0) \ \geq \ \frac{\lambda_{\min}}{2}(B_{ij} - (B_\pi)_{ij})^2 \ \geq \ \frac{\lambda_{\min}}{2}b_n^2.$$

We then have

$$\mathcal{L}_n(B) - \mathcal{L}_n(B_\pi) \geq \frac{\lambda_{\min}}{2}b_n^2 - \frac{\rho_n(B_\pi)}{n} = \frac{\lambda_{\min}}{2}b_n^2 - \frac{a_n}{n}|S_\pi|$$

which gives the second inequality in (2.30). The proof is complete.

2.8.2.2 *The Hessian*

The Hessian formula (2.33) should be interpreted as giving the correct quadratic term in the Taylor expansion when applied to $\mathrm{vec}(\Delta)$ for any lower triangular matrix Δ:

$$\langle \mathrm{vec}(\Delta), \nabla^2 \mathscr{L}_{\mathrm{chol}}(L; A)\, \mathrm{vec}(\Delta)\rangle = \mathrm{tr}(\Delta^T A \Delta) + \sum_i \frac{\Delta_{ii}^2}{L_{ii}^2} \tag{2.35}$$

using $\mathrm{vec}(\Delta)^T (I_p \otimes A)\, \mathrm{vec}(\Delta) = \mathrm{vec}(\Delta)^T \mathrm{vec}(A\Delta I_p) = \mathrm{tr}(\Delta^T A \Delta)$. In general $\mathrm{vec}(AXB) = (B^T \otimes A)\, \mathrm{vec}(X)$. That (2.35) is the correct quadratic term can be verified by Taylor expanding $t \mapsto \mathscr{L}_{\mathrm{chol}}(L + t\Delta; A)$. The Hessian is locally Lipschitz:

Lemma 7. *Let $L \in \mathcal{L}_p$ be such that $\min_i L_{ii} = m > 0$. For any L', such that $L'_{ii} \geq L_{ii}/2$ for all i, we have*

$$\|\nabla^2 \mathscr{L}_{chol}(L'; A) - \nabla^2 \mathscr{L}_{chol}(L; A)\| \leq \frac{6}{m^3} \max_i |(L' - L)_{ii}|.$$

Proof. The result follows from the elementary inequality

$$\Big|\frac{1}{x^2} - \frac{1}{y^2}\Big| \leq \Big|\frac{1}{x} + \frac{1}{y}\Big|\Big|\frac{y - x}{xy}\Big| \leq \frac{3}{x}\frac{2}{x^2}|y - x|$$

which holds for $y \geq x/2 > 0$. ☐

2.8.2.3 *Proof of Proposition 3*

Since $\mathscr{L}_{\mathrm{chol}}(B; \widehat{\Sigma}_n) \to \infty$ as $B \to \infty$, and the penalty $\rho_n(B)/n$ is bounded, it not hard to see that $\widehat{B}_{n,\pi}$ is eventually in a compact set (uniformly in π), with high probability. That is, we can choose R, independent of π and n, such that for sufficiently large n, $\|\widehat{B}_{n,\pi}\|_F \leq R$ for all π. The empirical loss uniformly converges to the population loss over this compact set:

$$\begin{aligned}\sup_{\|B\|_F \leq R} |\mathcal{L}_n(B; \widehat{\Sigma}_n) - \mathcal{L}_n(B; \Sigma_0)| &= \frac{1}{2} \sup_{\|B\|_F \leq R} \left|\mathrm{tr}\left(BB^T(\widehat{\Sigma}_n - \Sigma_0)\right)\right| \\ &\leq \frac{R^2}{2} \|\widehat{\Sigma}_n - \Sigma_0\| = O_p(n^{-1/2})\end{aligned} \tag{2.36}$$

by the CLT. Since $\widehat{B}_{n,\pi}$ minimizes $\mathcal{L}_n(\,\cdot\,; \widehat{\Sigma}_n)$, we have $0 \leq \mathcal{L}_n(B_\pi, \widehat{\Sigma}_n) - \mathcal{L}_n(\widehat{B}_{n,\pi}; \widehat{\Sigma}_n)$. Adding $\mathcal{L}_n(\widehat{B}_{n,\pi}; \Sigma_0) - \mathcal{L}_n(B_\pi; \Sigma_0)$ to both sides, we obtain

$$\mathcal{L}_n(\widehat{B}_{n,\pi}; \Sigma_0) - \mathcal{L}_n(B_\pi; \Sigma_0) \leq \max_{B \in \{B_\pi, \widehat{B}_{n,\pi}\}} |\mathcal{L}_n(B, \widehat{\Sigma}_n) - \mathcal{L}_n(B, \Sigma_0)| = O_p(n^{-1/2})$$

where the last equality is by (2.36). From (2.30), it follows that

$$\frac{\lambda_{\min}}{2} \min\Big\{\|\widehat{B}_{n,\pi} - B_\pi\|_F^2,\, b_n^2\Big(1 - \frac{2}{\lambda_{\min}} \frac{a_n}{n b_n^2} |S_\pi|\Big)\Big\} = O_p(n^{-1/2})$$

Since by assumption, $1 - 2a_n|S_\pi|/(\lambda_{\min} n b_n^2) \geq c > 0$ eventually and $b_n^2 \gg n^{-1/2}$, it follows that $\|\widehat{B}_{n,\pi} - B_\pi\|_F^2 = O_p(n^{-1/2})$, establishing $n^{1/4}$-consistency. It is not hard to argue, by inspecting the argument there, that the convergence is uniform in π, that is, $\sup_\pi |\widehat{B}_{n,\pi} - B_\pi| = O_p(n^{-1/4})$.

Next, since $n^{1/4} b_n \to \infty$ by assumption, eventually $\|\widehat{B}_{n,\pi} - B_\pi\|_F < b_n$. Since we either have $(B_\pi)_{ij} = 0$ or $|(B_\pi)_{ij}| \geq 2b_n$, the deviations $|[\widehat{B}_{n,\pi} - B_\pi]_{ij}|$ are either in the interval $[0, b_n)$ or (b_n, ∞), on both of which the loss is twice continuously differentiable. Then, the $\sqrt{n}$-consistency follows by a standard Taylor expansion argument. A technical issue is dealing with the one-sided differentiability of the loss function at zero. The following lemma, whose proof is deferred to Section 2.8.2.4, provides the quadratic expansion needed for the analysis:

For a function $f : \mathbb{R}^d \to \mathbb{R}$, and $x, u \in \mathbb{R}^d$, let $f'(x; u)$ denote its directional derivative at x along the vector u. Recall that a necessary condition for x to be a (local) minimum of f is that $f'(x; u) \geq 0$ for all u for which the directional derivative exists.

Lemma 8. *Consider the functions $\ell, \bar{\ell}, h : \mathbb{R}^d \to \mathbb{R}$ where $h(x) = \frac{1}{n}\sum_i r_n(|x_i|)$ with r_n twice differentiable on $[0, \infty)$ and both ℓ and $\bar{\ell}$ twice differentiable on $\mathbb{R}^d$. Let*

$$f(x) = \ell(x) + h(x), \quad \bar{f}(x) = \bar{\ell}(x) + h(x).$$

Assume that $x, \bar{x} \in \mathbb{R}^d$ are such that $f'(x; u) \geq 0$ and $\bar{f}'(\bar{x}; u) \geq 0$ for all $u \in \mathbb{R}^d$. Moreover, assume that x_i and $\bar{x}_i$ are on the same side of the origin, i.e., $x_i \bar{x}_i \geq 0$, for all $i = 1, \dots, d$. Let $w = x - \bar{x}$. Then,

$$\langle \nabla \ell(\bar{x}) - \nabla \bar{\ell}(\bar{x}), w\rangle + \langle w, \nabla^2 f(\tilde{x}) w\rangle \leq 0 \tag{2.37}$$

for some $\tilde{x}$ on the line segment between x and $\bar{x}$. In particular, if $\nabla^2 f(\tilde{x}) \succeq \alpha^2 I_d$, then

$$\|w\| \leq \alpha^{-2} \|\nabla \ell(\bar{x}) - \nabla \bar{\ell}(\bar{x})\|. \tag{2.38}$$

Now, let $S = \{i : \bar{x}_i \neq 0\}$. If $\|\tilde{x}\|_{\max} < b_n$ and $r_n'(\bar{x}_i) = 0$ for all $i \in S$, we also have

$$\big|n[h(x) - h(\bar{x})]\big| \leq \lambda_n \sum_{i \in S^c} |w_i| + \frac{\|w\|^2}{2} \sup_{t \in [0,\, b_n)} |r_n''(t)|. \tag{2.39}$$

where $\lambda_n = r_n'(0+)$.

The Hessian in Lemma 8 is given by $\nabla^2 f(\tilde{x}) = \nabla^2 \ell(\tilde{x}) + \nabla^2 h(\tilde{x})$ where

$$\nabla^2 h(\tilde{x}) = n^{-1} \operatorname{diag}\big(r_n''(|\tilde{x}_i|), i = 1, \dots, d\big).$$

When either x_i or $\bar{x}_i$ is nonzero, $|\tilde{x}_i| > 0$ hence, $r_n''(|\tilde{x}_i|)$ is well-defined. When $x_i = \bar{x}_i = 0$, we can define $r_n''(|\tilde{x}_i|)$ arbitrarily as far as the validity of (2.37) is concerned. To be specfic, let $r_n''(|\tilde{x}_i|) = r_n''(0) := r_n''(0+)$ in that case.

Now assume that $\bar{x}$ is such that either $\bar{x}_i = 0$ or $|\bar{x}_i| \geq 2b_n$, and $\|x - \bar{x}\|_{\max} < b_n$. Under assumption (R2), r_n''/n is C_0-Lipschitz on $[0, b_n)$, hence

$$n^{-1}\big|r_n''(|\tilde{x}_i|) - r_n''(|\bar{x}_i|)\big| \leq C_0\big||\tilde{x}_i| - |\bar{x}_i|\big| \leq C_0|\tilde{x}_i - \bar{x}_i|$$

where $r_n''(|\bar{x}_i|)$ is interpreted as $r_n''(0+)$ if $\bar{x}_i = 0$. It follows that $\|\nabla^2 h(\tilde{x}) - \nabla^2 h(\bar{x})\| \leq C_0\|\tilde{x} - \bar{x}\|_{\max}$. Assume that $|r_n''(b_n-)| \leq |r_n''(0+)|$. Under assumptions (R1) and (R2), we have $b_n \leq K$ and $|r_n''(b_n-)|/n \leq K$ for some $K > 0$, and

$$\sup_{t \in [0, b_n)} |r_n''(t)| \leq |r_n''(b_n-)| + \sup_{t \in [0, b_n)} |r_n''(t) - r_n''(b_n-)| \leq nK + C_0 nK. \tag{2.40}$$

A similar bound holds when $|r_n''(0+)| \leq |r_n''(b_n-)|$ with nK replaced with $C_1 n$ in (2.40).

Fix π and let $\omega^2 \leq \min_{\pi,i}(B_\pi)_{ii}$. With high probability, eventually $\widehat{B}_{n,\pi}$ will be in the neighborhood

$$\mathcal{N} := \big\{B : \; B_{ii} \geq (B_\pi)_{ii}/2, \; \|B - B_\pi\|_{\max} < \min\{b_n, R_1\}\big\},$$

for any fixed constant $R_1 > 0$. We apply Lemma 8 with $f \equiv \mathcal{L}_n(\,\cdot\,; \widehat{\Sigma}_n)$, $\ell \equiv \mathscr{L}_{\text{chol}}(\,\cdot\,; \widehat{\Sigma}_n)$, $\bar{\ell} \equiv \mathscr{L}_{\text{chol}}(\,\cdot\,; \Sigma_0)$, $\bar{f} \equiv \mathcal{L}_n(\,\cdot\,; \Sigma_0)$ and $h \equiv \rho_n$, all with domains restricted to $\mathbb{R}^{K_\pi}$, and we take $x \equiv \widehat{B}_{n,\pi}$ and $\bar{x} \equiv B_\pi$ We conclude that

$$\langle \nabla \mathcal{L}_n(B_\pi; \widehat{\Sigma}_n) - \nabla \mathcal{L}_n(B_\pi; \Sigma_0), \Delta_{n,\pi} \rangle \; + \; \langle \Delta_{n,\pi}, \nabla^2 \mathcal{L}_n(\widetilde{B}_{n,\pi}; \widehat{\Sigma}_n) \Delta_{n,\pi} \rangle \leq 0$$

where $\Delta_{n,\pi} = \widehat{B}_{n,\pi} - B_\pi$ and $\widetilde{B}_{n,\pi}$ is between $\widehat{B}_{n,\pi}$ and B_π, elementwise. See the discussion after Lemma 8 for the existence of $\nabla^2 \mathcal{L}_n(\widetilde{B}_{n,\pi}; \widehat{\Sigma}_n)$, whose smallest eigenvalue we now control. Since $\Delta_{n,\pi} = o_p(1)$, it follows that $\widetilde{B}_{n,\pi}$ belongs to $\mathcal{N}$ for large n and

$$\begin{aligned}\|\nabla^2 \mathcal{L}_n(\widetilde{B}_{n,\pi}; \widehat{\Sigma}_n) - \nabla^2 \mathcal{L}_n(B_\pi; \widehat{\Sigma}_n)\| &\le \|\nabla^2 \mathscr{L}_{\text{chol}}(\widetilde{B}_{n,\pi}; \widehat{\Sigma}_n) - \nabla^2 \mathscr{L}_{\text{chol}}(B_\pi; \widehat{\Sigma}_n)\| \\ &\quad + \|\nabla^2 \rho_n(\widetilde{B}_{n,\pi}) - \nabla^2 \rho_n(B_\pi)\| \\ &\le \frac{6}{\omega^6} \max_i |(\widetilde{B}_{n,\pi} - B_\pi)_{ii}| + C_0 \|\widetilde{B}_{n,\pi} - B_\pi\|_{\max},\end{aligned}$$

using Lemma 7. Both terms are $o_p(1)$ by the consistency established earlier. It follows that

$$\nabla^2 \mathcal{L}_n(\widetilde{B}_{n,\pi}; \widehat{\Sigma}_n) = \nabla^2 \mathcal{L}_n(B_\pi; \widehat{\Sigma}_n) + o_p(1) = \nabla^2 \mathcal{L}_n(B_\pi; \Sigma_0) + o_p(1)$$

where the second equality is since $\nabla^2 \mathcal{L}_n(B_\pi; \widehat{\Sigma}_n) - \nabla^2 \mathcal{L}_n(B_\pi; \Sigma_0) = \frac{1}{2} \operatorname{tr}(B_\pi B_\pi^T (\widehat{\Sigma}_n - \Sigma_0)) = o_p(1)$ by the LLN.

Since $r_n''(|(B_\pi)_{ij}|)$ is either 0 or $r_n''(0+)$, (R2) implies $\nabla^2 \rho_n(B_\pi) \succeq -C_1 I_{p^2}$. Using (2.33),

$$\nabla^2 \mathcal{L}_n(B_\pi; \Sigma_0) = \nabla^2 \mathscr{L}_{\text{chol}}(B_\pi; \Sigma_0) + \nabla^2 \rho_n(B_\pi) \;\succeq\; I_p \otimes \Sigma_0 - C_1 I_{p^2} \;\succeq\; \frac{1}{2}\lambda_{\min} I_{p^2}$$

where we have also used assumption $C_1 \le \lambda_{\min}/2$. It follows that $\nabla^2 \mathcal{L}_n(\widetilde{B}_{n,\pi}; \widehat{\Sigma}_n) \succeq \frac{1}{2}\lambda_{\min} I_{p^2}$, eventually. Then, inequality (2.38) of Lemma 8 gives

$$\begin{aligned}\|\Delta_{n,\pi}\|_F &\le \frac{2}{\lambda_{\min}} \|\nabla \mathscr{L}_{\text{chol}}(B_\pi; \widehat{\Sigma}_n) - \nabla \mathscr{L}_{\text{chol}}(B_\pi; \Sigma_0)\|_F \\ &\le \frac{2}{\lambda_{\min}} \|(\widehat{\Sigma}_n - \Sigma_0) B_\pi\|_F = O_p(n^{-1/2})\end{aligned}$$

by the CLT. This $O_p(n^{-1/2})$ term can be taken to be independent of π, showing that the convergence is uniform. This completes the proof of $\sqrt{n}$-consistency.

For (2.32), we apply inequality (2.39) of Lemma 8 and use (2.40) to obtain

$$|\rho_n(\widehat{B}_{n,\pi}) - \rho_n(B_\pi)| \le \lambda_n \sum_{(i,j) \in K_\pi \setminus S_\pi} |[\Delta_{n,\pi}]_{ij}| + \frac{\|\Delta_{n,\pi}\|_F^2}{2} O_p(n).$$

The first term above is bounded by $\lambda_n \sqrt{|K_\pi \setminus S_\pi|} \|\Delta_{n,\pi}\|_F$, hence

$$\begin{aligned}|\rho_n(\widehat{B}_{n,\pi}) - \rho_n(B_\pi)| &\le p\lambda_n \|\Delta_{n,\pi}\|_F + \frac{\|\Delta_{n,\pi}\|_F^2}{2} O_p(n) \\ &= p\lambda_n O_p(n^{-1/2}) + O_p(n^{-1}) O_p(n),\end{aligned}$$

since $\|\Delta_{n,\pi}\|_F = O_p(n^{-1/2})$ by $\sqrt{n}$-consistency. This completes the proof of (2.32).

2.8.2.4 *Proof of Lemma 8*

Let us extend r_n to entire $\mathbb{R}$ by setting $r_n(t) := r_n(|t|)$. Consider $t, \bar{t} \in \mathbb{R}$ that are on the same side of the origin, i.e., $t\bar{t} \geq 0$. Then, we have

$$r_n(t) = r_n(\bar{t}) + r_n'(\bar{t}; \delta) + \frac{1}{2}\delta^2 r_n''(\widetilde{t}) \tag{2.41}$$

where $\delta = t - \bar{t}$ and $\widetilde{t}$ is between t and $\bar{t}$. Here, r_n' is the directional derivative of r_n at $\bar{t}$ along δ. If $\bar{t} \neq 0$, then r_n is differentiable at $\bar{t}$ and $r_n'(\bar{t}; \delta) = r_n'(\bar{t})\delta = r_n'(|\bar{t}|)\text{sign}(\bar{t})\delta$. If $\bar{t} = 0$, we have $r_n'(\bar{t}; \delta) = r_n'(0+)|\delta|$. Note that when $t \neq \bar{t}$, then $\widetilde{t}$ is strictly between t and $\bar{t}$ and hence $r_n''(\widetilde{t}) = r_n''(|\widetilde{t}|)$ is well-defined, even when $\bar{t} = 0$. In the case $t = \bar{t} = 0$, we can define $r_n''(\widetilde{t}) = r_n''(0+)$, and (2.41) still holds.

We can now write

$$f(x) = f(\bar{x}) + f'(\bar{x}; w) + \frac{1}{2}\langle w, \nabla^2 f(\tilde{x}^{(1)})w\rangle,$$
$$f(\bar{x}) = f(x) + f'(x; -w) + \frac{1}{2}\langle w, \nabla^2 f(\tilde{x}^{(2)})w\rangle$$

where $\tilde{x}^{(i)}, i = 1, 2$ lies on the line segment between x and $\bar{x}$. By Darboux's theorem, there is $\tilde{x}$ on the same line segment such that $\langle w, \frac{1}{2}[\nabla^2 f(\tilde{x}^{(1)}) + \nabla^2 f(\tilde{x}^{(2)})]w\rangle = \langle w, \nabla^2 f(\tilde{x})w\rangle$. Summing the two equalities, we get

$$0 = f'(\bar{x}; w) + f'(x; -w) + \langle w, \nabla^2 f(\tilde{x})w\rangle.$$

Using the assumption $f'(x; -w) \geq 0$,

$$f'(\bar{x}; w) + \langle w, \nabla^2 f(\tilde{x})w\rangle \;\leq\; 0 \;\leq\; \bar{f}'(\bar{x}; w)$$

where the second inequality is again by assumption. Note that $f'(z; u) = \langle \nabla \ell(z), u\rangle + h'(z; u)$, for any $z, u \in \mathbb{R}^d$, and similarly $\bar{f}'(z; u) = \langle \nabla \bar{\ell}(z), u\rangle + h'(z; u)$. Hence $f'(z; u) - \bar{f}'(z; u) = \langle \nabla \ell(z) - \nabla \bar{\ell}(z), u\rangle$ for any $z, u \in \mathbb{R}^d$ and we obtain

$$\langle \nabla \ell(\bar{x}) - \nabla \bar{\ell}(\bar{x}), w\rangle + \langle w, \nabla^2 f(\tilde{x})w\rangle \leq 0$$

proving (2.37). For (2.38), we note that

$$\alpha^2\|w\|^2 \;\leq\; \langle w, \nabla^2 f(\tilde{x})w\rangle \;\leq\; \langle \nabla \bar{\ell}(\bar{x}) - \nabla \ell(\bar{x}), w\rangle \;\leq\; \|\nabla \bar{\ell}(\bar{x}) - \nabla \ell(\bar{x})\|\|w\|$$

giving the desired inequality.

For (2.39), using $n[h(x) - h(\bar{x})] = nh'(\bar{x}; w) + \frac{1}{2}\langle w, \nabla^2 h(\tilde{x}) w\rangle$ for some $\tilde{x}$ in the line segment between x and $\bar{x}$, we have

$$n[h(x) - h(\bar{x})] = \sum_{i \in S} r_n'(\bar{x}_i) w_i + \lambda_n \sum_{i \in S^c} |w_i| + \frac{1}{2} \sum_i r_n''(\tilde{x}_i) w_i^2.$$

The first term is zero by assumption. The last term is bounded as in the statement of the lemma, since $|\tilde{x}_i| \in [0, b_n)$ for all i, by assumption, and $r_n''(\tilde{x}_i) = r_n''(|\tilde{x}_i|)$. The proof is complete.

2.8.3 Proofs of other results

2.8.3.1 *Proof of Lemma 1*

Recall $B = (\beta_j) \in \mathbb{R}^{p \times p}$ and $\Omega = \operatorname{diag}(\omega_j^2) \in \mathbb{R}^{p \times p}$. The negative log-likelihood is

$$\begin{aligned} \ell(B, \Omega, P_\pi \mid X) &= \frac{1}{2} \sum_{j=1}^{p} \Big\{ n \log \omega_j^2 + \frac{1}{\omega_j^2} \|\mathbf{X}_{\pi(j)} - \mathbf{X} P_\pi^\top \beta_j\|^2 \Big\}, \\ &= \frac{n}{2} \log |\Omega| + \frac{1}{2} \|(\mathbf{X} P_\pi^\top - \mathbf{X} P_\pi^\top B) \Omega^{-\frac{1}{2}}\|^2. \end{aligned}$$

The second term is equal to

$$\frac{1}{2} \operatorname{tr}(P_\pi \mathbf{X}^\top \mathbf{X} P_\pi^\top (I - B) \Omega^{-1} (I - B)^\top) = \frac{n}{2} \operatorname{tr}(P_\pi \widehat{\Sigma} P_\pi^\top L L^\top)$$

recalling that $L = (I - B)\Omega^{-\frac{1}{2}}$ and $\widehat{\Sigma} = \frac{1}{n} \mathbf{X}^\top \mathbf{X}$. Because B is strictly lower triangular, the determinant of $(I - B)$ is $|I - B| = 1$ and $|L| = |(I - B)\Omega^{-\frac{1}{2}}| = -\frac{1}{2}|\Omega|$. Putting the pieces together and writing P for P_π finishes the proof.

2.8.3.2 *Proof of Proposition 1*

Since A is positive definite, A^{-1} is positive definite as well. By Cholesky decomposition, $A^{-1} = CC^\top$ where $C = \mathcal{C}(A^{-1})$. Therefore, $A = C^{-\top} C^{-1}$. Letting $R = C^{-1} L$, we have $\log |R| = -\log |C| + \log |L|$, where $\log |C|$ is a constant. Recall that $\mathcal{L}_p$ is the set of lower triangular matrices with positive diagonals. Since $C \in \mathcal{L}_p$ and $L \in \mathcal{L}_p$, we have $R \in \mathcal{L}_p$.

Also, $\operatorname{tr}(ALL^\top) = \operatorname{tr}(RR^T) = \|R\|_F^2$. Then,

$$\mathscr{L}_{\text{chol}}(L; A) = \frac{1}{2}\big[\|R\|_F^2 - 2\log|R|\big] - \log|C|.$$

Let $L^* := C$ and $R^* = C^{-1}L^* = I_p$. Set $\Delta = L - L^*$ and $U = C^{-1}\Delta$. Note that $R = U + I_p$, hence

$$a := 2\big[\mathscr{L}_{\text{chol}}(L; A) - \mathscr{L}_{\text{chol}}(L^*; A)\big] = \|R\|_F^2 - 2\log|R| - \|R^*\|_F^2 + 2\log|R^*|.$$

We have

$$\|R\|_F^2 - \|R^*\|_F^2 = \|U + I_p\|_F^2 - p = \|U\|_F^2 + 2\langle U\rangle I_p + \|I_p\|_F^2 - p = \|U\|_F^2 + 2\operatorname{tr}(U)$$

and $\log|R| = \log|U + I_p| = \sum_i \log(U_{ii} + 1)$. Hence,

$$a = \|U\|_F^2 + 2\sum_i [U_{ii} - \log(1 + U_{ii})] \geq \|U\|_F^2$$

using the inequality $\log(1 + x) \leq x$. We have

$$\|\Delta\|_F = \|CU\|_F \leq \|C\|\|U\|_F$$

which gives $\|U\|_F \geq \|\Delta\|_F / \|C\|$. We also note that $A^{-1} = CC^T$, hence $[\lambda_{\min}(A)]^{-1} = \|A^{-1}\| = \|C\|^2$. Putting the pieces together proves the inequality.

Substituting $L^* = \mathcal{C}(A^{-1})$ into $\mathscr{L}_{\text{chol}}(L; A)$, we obtain the minimum value:

$$\mathscr{L}^*_{\text{chol}}(A) := \mathscr{L}_{\text{chol}}(L^*; A) = \frac{1}{2}\operatorname{tr}(AA^{-1}) - \log|A^{-\frac{1}{2}}| = \frac{1}{2}\left(p + \log|A|\right).$$

The assertion $\mathscr{L}^*_{\text{chol}}(A) = \mathscr{L}^*_{\text{chol}}(PAP^\top)$ follows by noting that $|P\widehat{\Sigma}P^T| = |P||\widehat{\Sigma}||P^\top| = |\widehat{\Sigma}|$.

2.8.3.3 *Proof of Lemma 2*

Recall $B = (\beta_j) \in \mathbb{R}^{p\times p}$, $\Omega = \operatorname{diag}(\omega_j^2) \in \mathbb{R}^{p\times p}$, we have the following experimental data log-likelihood:

$$\ell_{\mathcal{O}}(B, \Omega, P_\pi \mid \mathbf{X}) = \sum_{j=1}^{p}\Big[\frac{1}{2\omega_j^2}\|\mathbf{X}_{\mathcal{O}_{\pi(j)},\pi(j)} - \mathbf{X}_{\mathcal{O}_{\pi(j)}}P_\pi^\top\beta_j\|^2 + \frac{|\mathcal{O}_{\pi(j)}|}{2}\log\omega_j^2\Big].$$

We have

$$\|\mathbf{X}_{\mathcal{O}_{\pi(j)},\pi(j)} - \mathbf{X}_{\mathcal{O}_{\pi(j)}} P_\pi^\top \beta_j\|^2 = \|\mathbf{X}_{\mathcal{O}_{\pi(j)}} P_\pi^\top (e_j - \beta_j)\|^2 = \omega_j^2 |\mathcal{O}_{\pi(j)}| \operatorname{tr}\left(P_\pi \widehat{\Sigma}^j P_\pi^\top L_j L_j^\top\right)$$

recalling that $|\mathcal{O}_{\pi(j)}|\widehat{\Sigma}^j = \mathbf{X}_{\mathcal{O}_{\pi(j)}}^\top \mathbf{X}_{\mathcal{O}_{\pi(j)}}$ and $L = (L_j) \in \mathbb{R}^{p\times p}$ with columns $L_j = (e_j - \beta_j)/\omega_j \in \mathbb{R}^p$. Simplifying P_π to P, and noting that $\log \omega_j^2 = -2\log L_{jj} = -2\log|L_j|$, we have

$$\ell_{\mathcal{O}}(L, P) = \sum_{j=1}^p |\mathcal{O}_{\pi(j)}| \left[\frac{1}{2}\operatorname{tr}\left(P\widehat{\Sigma}^j P^\top L_j L_j^\top\right) - \log|L_j|\right]$$

which is the desired result.

2.8.3.4 *Proof of Lemma 3*

Pick $B^* \in \mathcal{E}$. Assume that $B \notin \mathcal{E}$, then $\mathcal{I}(B) \subset \mathcal{I}(P^*) = \mathcal{I}(B^*)$ where the equality is by perfectness. Thus, B is an I-map for B^* and according to Meek-Chickering theorem (Chickering, 2002), DAG B^* can be converted to DAG B by a sequence of edge additions and covered edge reversals. Assume in addition that $\|B\|_0 \le \|B^*\|_0$. Then, the aforementioned sequence consists only of covered edge reversals. However, this implies that B and B^* belong to the same equivalence class, a contradiction. Hence, $\|B\|_0 > \|B^*\|_0 = \|\mathcal{E}\|_0$.

2.8.3.5 *Proof of Lemma 4*

Fix P and let $Z = \frac{1}{\sqrt{n}}\mathbf{X}P^\top$ so that $Z^\top Z = \frac{1}{n}P\mathbf{X}^\top\mathbf{X}P^\top = P\widehat{\Sigma}P^\top$. We can rewrite (2.9) as $\ell(L) = \ell^1(L) + \ell^2(L)$ where

$$\ell^1(L) := \frac{n}{2}\|ZL\|_F^2, \quad \text{and} \quad \ell^2(L) := -n\log|L|.$$

To compute the gradient of ℓ^1 w.r.t. L, we perturb L to $L + t\delta$ where $\delta \in \mathcal{L}_p$ and $t \in \mathbb{R}$. We have

$$\ell^1(L + t\delta) - \ell^1(L) = nt\langle ZL, Z\delta\rangle + \frac{nt^2}{2}\|Z\delta\|_F^2 = n\langle Z^\top ZL, \delta\rangle\, t + o(t)$$

where $\langle A, B\rangle = \operatorname{tr}(A^\top B)$ for two matrices A and B. Be definition, $\Pi_{\mathcal{L}}$ maps a $p\times p$ matrix to its lower triangular projection. Since $\delta \in \mathcal{L}_p$, $\langle Z^\top ZL, \delta\rangle = \langle \Pi_{\mathcal{L}}(Z^\top ZL), \delta\rangle$, and thus $\nabla\ell^1(L) = n\Pi_{\mathcal{L}}(Z^\top ZL)$.

For the second term, $\ell^2 = -n\sum_{i=1}^{p} \log L_{ii}$, and thus $\nabla \ell^2(L) = -n \operatorname{diag}\left(\{1/L_{ii}\}_{i=1}^{p}\right)$ where $L \in \mathcal{L}_p$. Putting the pieces together gives the desired result.

2.8.3.6 *Proof of Lemma 5*

From (2.16), we have $\ell_{\mathcal{O}} = \ell_{\mathcal{O}}^1 + \ell_{\mathcal{O}}^2$, where

$$\ell_{\mathcal{O}}^1 := \frac{1}{2}\sum_{j=1}^{p} |\mathcal{O}_{\pi(j)}| L_j^\top P \widehat{\Sigma}^j P^\top L_j, \quad \text{and} \quad \ell_{\mathcal{O}}^2 := -\sum_{j=1}^{p} |\mathcal{O}_{\pi(j)}| \log L_{jj}.$$

Note that each term in $\ell_{\mathcal{O}}^1$ is a quadratic form in L_j. The gradient of $\ell_{\mathcal{O}}^1$ w.r.t. to L_j is $|\mathcal{O}_{\pi(j)}| \Pi_j(P\widehat{\Sigma}^j P^\top L_j)$, since $L_{ij} = 0$ for $i < j$. It is also easy to see that $\nabla_{L_j} \ell_{\mathcal{O}}^2 = -|\mathcal{O}_{\pi(j)}| \frac{e_j}{L_{jj}}$. Putting the pieces together, the gradient of $\ell_{\mathcal{O}}$ w.r.t. L_j is $|\mathcal{O}_{\pi(j)}| \left(\Pi_j(P\widehat{\Sigma}^j P^\top L_j) - \frac{e_j}{L_{jj}}\right)$.

2.8.3.7 *Proof of Lemma 6*

Recall the MCP function is defined as

$$\rho(x) = \begin{cases} \lambda|x| - \frac{x^2}{2\gamma}, & |x| < \gamma\lambda, \\ \frac{1}{2}\gamma\lambda^2, & |x| \geq \gamma\lambda. \end{cases}$$

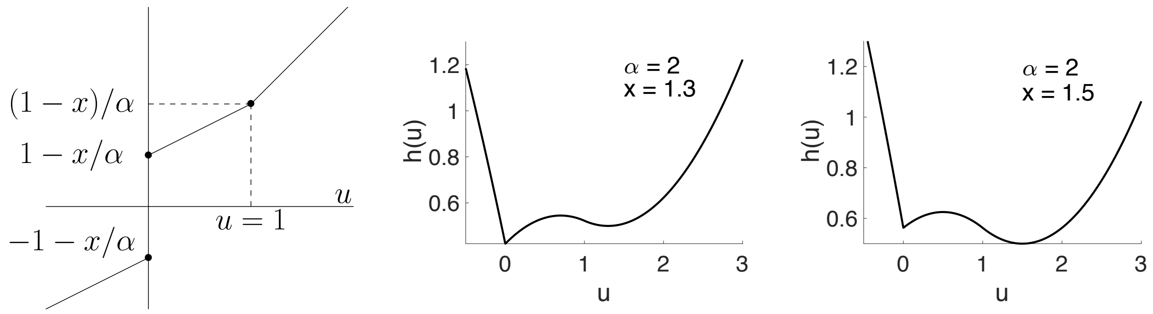
It is not hard to see that $\rho(\lambda\gamma x) = \gamma\lambda^2 \rho_1(x)$. It follows that

$$\begin{aligned} &\mathbf{prox}_{t\rho}(\lambda\gamma x) = \arg\min_u \left[\rho(u) + \frac{1}{2t}(u - \lambda\gamma x)^2\right] \\ =&\lambda\gamma \cdot \arg\min_v \left[\rho(\lambda\gamma v) + \frac{1}{2t}(\lambda\gamma v - \lambda\gamma x)^2\right] = \lambda\gamma \cdot \arg\min_v \left[\gamma\lambda^2\rho_1(v) + \frac{\gamma^2\lambda^2}{2t}(v - x)^2\right] \\ =&\lambda\gamma \cdot \arg\min_v \left[\rho_1(v) + \frac{\gamma}{2t}(v - x)^2\right] = \lambda\gamma\, \mathbf{prox}_{(t/\gamma)\rho_1}(x), \end{aligned}$$

where the second equality uses the change of variable $u = \lambda\gamma v$ and the fourth uses the fact that $\arg\min$ is invariant to rescaling the objective. This establishes (2.28).

Due to the symmetry of ρ_1, we have $\mathbf{prox}_{\alpha\rho_1}(-x) = -\mathbf{prox}_{\alpha\rho_1}(x)$, which is easy to see by a change of variable $u = -v$ in the defining optimization. Thus, it is enough to consider $x \geq 0$ which we assume in the following.

The function $h(u) = \rho_1(u) + \frac{1}{2\alpha}(u - x)^2$ is continuous and decreasing on $(-\infty, 0]$, hence it achieves its minimum of $\frac{x^2}{2\alpha}$ over this interval at $u = 0$. Over $(0, \infty)$, the function is



(a) (b) (c)

Figure 2.11: (a) The derivative $h'(u)$ for $u \in (0, \infty)$ in a typical case. Note that h' is discontinuous at 0. (b) and (c) Plots of $h(u)$ for two values of (α, x) that lead to case 4.

differentiable with derivative $h'(u) = (1-u)_+ + \frac{1}{\alpha}(u-x)$ which is piecewise linear (or affine) with a break at $u = 1$. The first segment of h' is a line connecting $(0, 1 - x/\alpha)$ to $(1, (1-x)/\alpha)$. The next segment is an always-increasing section starting at $(1, (1-x)/\alpha)$ and increasing with slope $1/\alpha$. See Figure 2.11(a). The behavior of h' for $u \in (0, 1)$ determines the minimizer. We have four cases:

1. When both $1 - x/\alpha > 0$ and $(1-x)/\alpha > 0$, that is $x < \min\{\alpha, 1\}$, h' is increasing in $[0, 1]$. Hence, $h' > 0$ over $[0, \infty)$ and the overall minimum of h occurs at $u_1 = 0$.

2. When $1 - x/\alpha \le 0 < (1-x)/\alpha$, that is, $\alpha \le x < 1$, then h has a single critical point at $u_2 = (x-\alpha)/(1-\alpha)$ before which it decreases and after which it increases. Hence, this is its unique minimizer.

3. When both $1 - x/\alpha < 0$ and $(1-x)/\alpha < 0$, that is, $x > \max\{\alpha, 1\}$, then, the only critical point occurs in the second linear segment and is $u_3 = x$ which is the unique minimizer.

4. When $(1-x)/\alpha < 0 \le 1 - x/\alpha$, that is, $1 < x \le \alpha$, then both of the points u_2 and u_3 in cases 2 and 3 are critical points. The function drops in $(-\infty, u_1)$ where $u_1 = 0$, increases in (u_1, u_2), drops in (u_2, u_3) and increases in (u_3, ∞). Thus, both u_1 and u_3 are local minima (while u_2 is a local maximum). The global minimum is determined by comparing $h(u_1) = x^2/(2\alpha)$ and $h(u_3) = 1/2$. That is, if $x < \sqrt{\alpha}$, the global minimum

is $u_1 = 0$ (Figure 2.11b); if $x > \sqrt{\alpha}$, the global minimum is $u_3 = x$ (Figure 2.11c); if $x = \sqrt{\alpha}$, minimum is $\{u_1, u_3\} = \{0, x\}$, which is not unique.

What left is the special case when both $1 - x/\alpha = 0$ and $(1 - x)/\alpha = 0$, i.e. $\alpha = x = 1$, indicating the first segment of $h'(u)$, $u \geq 0$, is flat as zero. Hence, minimizer of h is any value in the interval $[0, 1]$. We merge case 4 into cases 1 and 3 by revising the domains of x, and thus complete the derivation of (2.29).

CHAPTER 3

Leaning Generalized Linear Causal Graphs from Distribtuted Data

In this chapter, we consider the task of learning causal structures from data stored on multiple machines, and propose a novel structure learning method called distributed annealing on regularized likelihood score (DARLS) to solve this problem. DARLS searches over the space of topological sorts for a high-scoring causal graph, and it is a generalization of the ARCS algorithm that we discussed in Chapter 2 in two aspects. First, it recovers a flexible family of DAGs, known as generalized linear DAGs, which allow various distributions for variables in a BN. Second, the DARLS algorithm considers the distributed data storage. To the best of our knowledge, DALRS is the first method to learn causal structures from distributed data using iterative optimization that relies on multiple rounds of communication between local and central machines. In Section 3.1., we first introduce generalized linear DAG models, which uses generalized linear models for the conditional distributions of BNs. Then we propose the DARLS algorithm in Section 3.2 and study some theoretical results in Section 3.3. We show simulation tests in Section 3.4 and apply DARLS on ChIP-Sequencing data to model protein-DNA binding networks in Section 3.5. Lastly, we summarize our proposed DARLS algorithm in Section 3.6.

3.1 Generalized linear DAG models

Let us denote by $x^j \in \mathbb{R}^{d_j}$ a realization of variable X_j, where $d_j = 1$ for a numerical X_j and $d_j = r_j - 1$ for a categorical variable X_j with r_j classes, using the one-hot encoding. Under the BN model (1.1), let $\beta_{ij} \in \mathbb{R}^{d_i \times d_j}$ encode the influence of X_i on X_j and $\beta_{ij} = 0$ if

$X_i \notin \mathrm{PA}_j$. Put

$$\beta_j := [\beta_{0j}, \beta_{1j}, \dots, \beta_{pj}] \in \mathbb{R}^{(d+1)\times d_j}, \quad x := [1, x^1, \dots, x^p] \in \mathbb{R}^{d+1}, \tag{3.1}$$

where $\beta_{0j} \in \mathbb{R}^{1\times d_j}$ and $d = \sum_{i=1}^p d_i$. Here and elsewhere, $[x, y]$ denotes the vertical concatenation of two vectors or matrices x and y. We define a *generalized linear DAG* (GLDAG) as the Bayesian network (1.1) with conditional densities given by GLMs with canonical links, that is,

$$p(x^j \mid pa_j, \beta_j) = \exp\big(\langle \beta_j^\top x, x^j\rangle - b_j(\beta_j^\top x)\big) + c_j(x^j), \quad j \in [p] \tag{3.2}$$

where b_j and c_j are both functions from $\mathbb{R}^{d_j}$ to $\mathbb{R}$. Note that $\beta_j^\top x = \sum_{i\in \mathrm{PA}_j} \beta_{ji}^\top x^i$ only depends on pa_j. GLDAG models allow for many common distributions via the choice of the log partition-function $b_j(\cdot)$. Examples include the Bernoulli distribution for $b_j(\theta) = \log(1+e^\theta)$, constant-variance Gaussian for $b_j(\theta) = \theta^2/2$, Poisson for $b_j(\theta) = \exp(\theta)$, Gamma for $b_j(\theta) = -\log(-\theta)$ and the multinomial for $b_j(\theta) = \log\big(1 + \sum_{l=1}^{d_j} e^{\theta_l}\big)$. Note that in the multinomial case $b_j(\cdot)$ is a multivariate function, operating on a vector $\theta = (\theta_l)$, in contrast to the other example for which $b_j(\cdot)$ is a scalar function. The Bernoulli and multinomial choices above give rise to logistic and multi-logit regression models for each node.

We collect all the parameters of model (3.2) in a matrix $\beta \in \mathbb{R}^{(d+1)\times d}$ which is obtained by horizontal concatenation of β_j, $j = 1, \dots, p$, each of which is as defined via (3.1). We say a GLDAG (3.2) is continuous if all the variables are continuous. Recall in this case, $d_j = 1$ for all $j \in [p]$ and thus β is a $(p+1)\times p$ matrix. We re-write the log pdf of (3.2) as

$$L(x;\beta) = \sum_{j=1}^p \big[\log c_j(x^j) + x^j(\beta^\top x)_j - b_j\big((\beta^\top x)_j\big)\big], \tag{3.3}$$

where $\beta^\top x \in \mathbb{R}^p$ $\beta_{ij} \neq 0$ if and only if $X_i \to X_j$. Next, we show continuous GLDAGs are identifiable.

Definition 3. *(Identifiability). Suppose we are given a joint distribution $L(X) = L(X_1, \dots, X_p)$ that has been generated from an unknown GLDAG model* (3.2) *with a graph $\mathcal{G}_0$. If the distribution $L(X)$ cannot be generated by any GLDAG model with a different graph $\mathcal{G} \neq \mathcal{G}_0$, then we say $\mathcal{G}_0$ is identifiable from $L(X)$.*

It is well-known that linear Gaussian DAGs and multinomial DAGs in general are not identifiable. In contrast, continuous GLDAG models (3.3) are identifiable under mild assumptions:

Proposition 4. *Suppose the joint distribution $L(X)$ is defined by the log-pdf $L(x;\beta)$ with a DAG $\mathcal{G}_0$ according to* (3.3) *such that $\beta_{ij} \neq 0$ if and only if $i \in PA_j$ in $\mathcal{G}_0$. If $L(x;\beta)$ is second-order differentiable with respect to x and the first-order derivative of $b_j(\cdot)$ exists and is not constant for all j, then $\mathcal{G}_0$ is identifiable from $L(X)$.*

Proposition 4 establishes the identifiability of continuous GLDAG models (3.3), expanding the class of identifiable DAG models in the literature. DAGs generated from linear Gaussian structural equations models with equal variance can be fully identified (Peters and Bühlmann, 2014), which is a special case of the GLDAG models with $b_j(\theta) = \theta^2/2$, $\forall j$. Other distribution constraints are imposed on DAG models to identify causal graphs, such as the additive noise model, $X_j = f_j(\mathrm{PA}_j) + \varepsilon_j$, assuming nonlinear f_j and/or non-Gaussian ε_j (Shimizu et al, 2006; Hoyer et al, 2008; Peters et al, 2014).

3.2 Distributed DAG learning

In this section we construct the objective function using distributed data and propose a simulated annealing search combined with an iterative optimization method to learn causal DAG structures. We start with the definition of topological sorts for DAGs. Given a permutation π on $[p]$, we permute a vector $v = (v_1, \ldots, v_p)$ according to π to obtain a relabeled vector $v_\pi = \big(v_{\pi(1)}, \ldots, v_{\pi(p)}\big)$. Recall a *topological sort* of a DAG is a permutation of nodes such that if $a \in \mathrm{PA}_b$, then a precedes b in the order defined by π, denoted by $a \prec_\pi b$. By definition (1.1), every DAG has at least one topological sort.

Let $\{x_h\}_{h=1}^n$ be an i.i.d. sample of size n from model (3.2). We also let x_h^j represent the observed value of the j-th variable (X_j) in the h-th data point. Consider a subset $\mathcal{I} \subset [n]$. The the normalized negative log-likelihood based on subsample $\{x_h\}_{h\in\mathcal{I}}$ is given, up to an additive constant, by

$$\ell_{\mathcal{I}}(\beta) := \frac{1}{|\mathcal{I}|}\sum_{h\in\mathcal{I}}\sum_{j=1}^{p}\big[b_j(\beta_j^\top x_h) - \langle \beta_j^\top x_h, x_h^j\rangle\big]. \tag{3.4}$$

Note that in this notation, $\ell_{[n]}$ denotes the normalized negative log-likelihood based on the entire sample of size n.

3.2.1 Local and global objective functions

We consider the case that the overall data is stored in K different servers, where each local machine $\mathcal{M}_k$ holds its private data $\{x_h\}_{h\in\mathcal{I}_k}$ and communicates with a central machine $\mathcal{C}$. Let $n_k = |\mathcal{I}_k|$ be the sample size in $\mathcal{M}_k$ so that $\sum_{k=1}^K n_k = n$. The normalized negative log-likelihood based on the entire data can be decomposed as $\ell_{[n]}(\beta) = \sum_{k=1}^K \frac{n_k}{n}\ell_{\mathcal{I}_k}(\beta)$. We ideally would like to estimate β by minimizing a regularized loss function of the form

$$\min_{\pi\in\mathcal{P}} f(\pi), \quad \text{where} \quad f(\pi) := \min_{\beta\in\mathcal{D}(\pi)}\sum_{k=1}^{K}\frac{n_k}{n}\ell_{\mathcal{I}_k}(\beta) + \rho(\beta), \tag{3.5}$$

where $\mathcal{D}(\pi) \subset \mathbb{R}^{(d+1)\times d}$ is the set of DAGs whose topological sort is compatible with permutation π, $\mathcal{P}$ is the set of all permutations on $[p]$ and $\rho(\cdot)$ is an appropriate regularizer. Note that $\mathcal{D}(\pi)$ is a linear subspace of $\mathbb{R}^{(d+1)\times d}$.

For any fixed π, we use distributed computing to evaluate $f(\pi)$. That is, instead of directly working with the objective function in (3.5), we rely on local versions of it to guide a distributed algorithm that divides the task of computing $f(\pi)$ among the K local machines. In particular, we consider the local objective functions

$$f_k(\pi) := \min_{\beta\in\mathcal{D}(\pi)} F_k(\beta), \quad \text{where} \quad F_k(\beta) := \ell_{\mathcal{I}_k}(\beta) + \rho(\beta).$$

The global version can be rewritten as $f(\pi) = \min_{\beta\in\mathcal{D}(\pi)} F(\beta)$ where $F(\beta) := \ell_{[n]}(\beta) + \rho(\beta)$. Typically, each of F and F_k is nonsmooth due to the presence of the regularizer ρ, but the difference $h_k := F_k - F = \ell_{\mathcal{I}_k} - \ell_{[n]}$ is often smooth. The gradient of h_k is used to guide iterations in each local machine. More precisely, given the current (global) estimate β, local machine $\mathcal{M}_k$ performs the update

$$\varphi_{k,\pi}(\beta) := \underset{\xi \in \mathcal{D}(\pi)}{\arg\min} \left[F_k(\xi) - \langle \nabla h_k(\beta), \xi \rangle\right] = \underset{\xi \in \mathcal{D}(\pi)}{\arg\min} \left[F_k(\xi) - \langle \nabla \ell_{\mathcal{I}_k}(\beta) - \nabla \ell_{[n]}(\beta), \xi \rangle\right]. \tag{3.6}$$

Let $\beta_\pi^{(t)}$ be the global estimate of the algorithm at iteration t. At the next iteration, $t+1$, we obtain local estimates $\beta_{k,\pi}^{(t+1)} = \varphi_{k,\pi}(\beta_\pi^{(t)})$ for $k = 1, \ldots, K$. These are then passed to the central machine $\mathcal{C}$ to compute the next global estimate by averaging, i.e., $\beta_\pi^{(t+1)} = \sum_{k=1}^K \frac{n_k}{n} \beta_{k,\pi}^{(t+1)}$.

The above approach is essentially a version of the DANE algorithm (Zhang et al, 2013; Shamir et al, 2014; Jordan et al, 2018; Fan et al, 2019). Note that to calculate local updates $\beta_{k,\pi}^{(t+1)}$ only the global gradient $\nabla \ell_{[n]}(\beta_\pi^{(t)})$ needs to be communicated to each local machine. In Section 3.3.1, we show that for a sufficiently large minimum sample size per machine, i.e. $\min_k n_k$, the sequence $\{\beta_\pi^{(t)}\}_{t \geq 0}$ thus produced will converge to a global minimizer $\widehat{\beta}_\pi$ of $F(\cdot)$ over $\mathcal{D}(\pi)$.

Choosing the regularizer. Recall $\beta_{ij} \neq 0$ if and only if $i \in \mathrm{PA}_j$. Given a topological sort π, we apply group regularization on $\beta_{ij} \in \mathbb{R}^{d_i \times d_j}$ for all pairs (i, j) such that $i \prec_\pi j$ and set β_{ij} to zero otherwise to learn sparse DAGs. We choose $\rho(\beta) = \lambda \sum_j \sum_{i \prec_\pi j} \rho_g(\beta_{ij})$ to be the overall regularizer, in which $\rho_g(\cdot)$ is a nonnegative and nondecreasing group regularizer and $\lambda > 0$ is a tuning parameter. Group Lasso (i.e., group ℓ_2) is a natural extension of Lasso regularization and it demonstrates remarkable performance in grouped variable selection (Yuan and Lin, 2007). Hence, we consider the convex penalty $\rho_g(\beta_{ij}) = \|\beta_{ij}\|_F$ in this paper, where $\|\beta_{ij}\|_F$ is the Frobenius norm of matrix β_{ij}.

3.2.2 Optimization

The DAG learning problem (3.5) is to search for $(\pi \in \mathcal{P}, \beta \in \mathcal{D}(\pi))$ using distributed data. We propose the *distributed annealing on regularized likelihood score* (DARLS) algorithm to learn causal graphs from distributed data, which applies annealing strategies to search over the permutation space, coupled with a distributed optimization method. Joint optimization

Algorithm 5 Distributed annealing on regularized likelihood score (DARLS).

```
    Input: $\{x_h\}_{h=1}^n$ distributed over $K$ machines, $\pi_0$, a temperature schedule $\{T^{(i)}\}_{i=0}^N$, $\tau$.
    Output: $\widehat{\pi}, \widehat{\beta}$.
1: Select tuning parameter $\lambda$ by BIC selection.
2: $\widehat{\pi} \leftarrow \pi_0$, compute $(\widehat{\beta}, f(\widehat{\pi}))$ by Algorithm 6.
3: for $i = 0, \dots, N$ do
4:     $T \leftarrow T^{(i)}$.
5:     Central machine $\mathcal{C}$ proposes $\pi^*$ by flipping a random interval (length up to $\tau$) in $\widehat{\pi}$.
6:     Compute $(\beta^*, f(\pi^*))$ using Algorithm 6.
7:     $\mathcal{C}$ sets $(\widehat{\pi}, \widehat{\beta}, f(\widehat{\pi})) \leftarrow (\pi^*, \beta^*, f(\pi^*))$ with prob. $\min\left\{1, \exp\left(-\frac{1}{T}[f(\pi^*) - f(\widehat{\pi})]\right)\right\}$.
8: end for
9: Refine the causal structure implied by $\widehat{\beta}$.
```

over the topological sort space and the DAG space has demonstrated great effectiveness in learning BNs, such as (Ye et al, 2020; Larrañaga et al, 1996; Friedman and Koller, 2003; Teyssier and Koller, 2005; Ellis and Wong, 2008; Zhou, 2011; Alonso-Barba et al, 2011; Scanagatta et al, 2015, 2017; Champion et al, 2018).

Main steps of DARLS algorithm are outlined in Algorithm 5. At each annealing iteration, a permutation π^* is proposed based on current $\widehat{\pi}$ (line 5) and is accepted with probability according to simulated annealing given a decreasing temperature schedule. We use the Bayesian information criterion (BIC) to select tuning parameter λ for the group Lasso penalty before the annealing process (line 1), and refine the DAG structure implied by annealing estimator $\widehat{\beta}$ after the search (line 9). Note that DARLS algorithm can be applied to any objective function as long as the gradient w.r.t. β has a closed-form expression.

We use a *distributed optimization* method (Algorithm 6) to learn the optimal DAG structure from distributed data given a permutation, which allows multiple rounds of communications between local and the central machines to update and synthesize information. In each round of communication, a global estimate (line 6) is formed by averaging local values (3.6) weighted by local sample sizes (line 5).

Algorithm 6 Using distributed optimization to compute the global permutation score.

```
    Input: $\pi$, $\beta^{(0)} \in \mathcal{D}(\pi)$, number of iteration $T$.
    Output: $\widehat{\beta} \in \mathcal{D}(\pi), f(\pi)$.
1: Central processor $\mathcal{C}$ broadcasts $\pi$ to local machines $\{\mathcal{M}_k\}_{k=1}^K$.
2: for $t = 0, 1, \dots, T-1$ do
3:     Each machine $\mathcal{M}_k$ computes $\nabla \ell_{\mathcal{I}_k}(\beta^{(t)})$ and sends to $\mathcal{C}$.
4:     $\mathcal{C}$ computes $\nabla \ell_{[n]}(\beta^{(t)}) = \sum_{k=1}^K \frac{n_k}{n} \nabla \ell_{\mathcal{I}_k}(\beta^{(t)})$ and broadcasts it to local machines.
5:     Each $\mathcal{M}_k$ calculates and sends the minimizer $\beta_k^{(t+1)} = \varphi_{k,\pi}(\beta^{(t)})$ (3.6) to $\mathcal{C}$.
6:     $\mathcal{C}$ computes $\beta^{(t+1)} = \sum_{k=1}^K \frac{n_k}{n} \beta_k^{(t+1)}$ and broadcasts it to local machines.
7: end for
8: Each $\mathcal{M}_k$ reports $n_k F_k\left(\beta^{(T)}\right)$ to $\mathcal{C}$, and $\mathcal{C}$ sets $\widehat{\beta} \leftarrow \beta^{(T)}$ and $f(\pi) \leftarrow \sum_{k=1}^K \frac{n_k}{n} F_k\left(\beta^{(T)}\right)$.
```

The proximal gradient algorithm, with steps outlined in Algorithm 7, is used to perform a local update (3.6) in the distributed optimization (line 5, Algorithm 6). The objective (3.6) is equivalent to minimize $\ell_{\mathcal{I}_k}(\xi) - \langle \nabla h_k(\beta^{(t)}), \xi \rangle + \rho(\xi)$ with respect to ξ over $\mathcal{D}(\pi)$ given a current global estimate $\beta^{(t)}$. Define $\widetilde{\ell}_{\mathcal{I}_k}(\xi) := \ell_{\mathcal{I}_k}(\xi) - \langle \nabla h_k(\beta^{(t)}), \xi \rangle$ a surrogate to the global likelihood $\ell_{[n]}(\xi)$ (Jordan et al, 2018). A proxy to (3.6) is found by minimizing a quadratic approximation of $\widetilde{\ell}_{\mathcal{I}_k}(\xi)$ at $\beta^{(t)}$ with a regularization:

$$\beta_{k,\pi}^{(t+1)} := \underset{\xi \in \mathcal{D}(\pi)}{\arg\min} \left[\widetilde{\ell}_{\mathcal{I}_k}(\xi) + \left\langle \nabla \widetilde{\ell}_{\mathcal{I}_k}(\beta^{(t)}), \xi - \beta^{(t)} \right\rangle + \frac{1}{2s} \|\xi - \beta^{(t)}\|_F^2 + \rho(\xi) \right], \tag{3.7}$$

where $s > 0$ is a step size. Recall the proximal operator $\mathbf{prox}_\rho(x) = \arg\min_u \left(\rho(u) + \frac{1}{2}\|x - u\|_F\right)$ define in (2.26). Equation (3.7) can be re-written as

$$\beta_{k,\pi}^{(t+1)} = \mathbf{prox}_{s\rho}\left(\beta^{(t)} - s\nabla \widetilde{\ell}_{[n]}(\beta^{(t)})\right), \tag{3.8}$$

where $\mathbf{prox}_{s\rho}(\cdot)$ is the proximal operator applied on a scaled function $s\rho(\cdot)$. The update (3.8) is known as a *proximal gradient update*. When $\rho(\cdot)$ is the group Lasso penalty, the update (3.8) has a closed-form expression, known as a *block soft thresholding*:

$$\left(\mathbf{prox}_{s\rho}(u)\right)_{ji} = \left(1 - \frac{s}{\|u_{ji}\|_F}\right)_+ u_{ji}.$$

We use a backward line search to compute the step size of s, which shrinks initial value s_0 until a proper step size is found (Parikh and Boyd, 2013b).

Algorithm 7 Use the proximal gradient algorithm to compute local permutation scores.

```
Input: $\{x_h\}_{h\in\mathcal{I}_k}$, $\pi$, $\beta^{(t-1)} \in \mathcal{D}(\pi)$, $\nabla h_k(\beta^{(t-1)})$, $s_0 > 0$, $\kappa \in (0,1)$, max-iter, tol.
Output: $\beta^{(t)}_{k,\pi}$.
1: iter $\leftarrow 0$, err $\leftarrow \infty$, $u \leftarrow \beta^{(t-1)}$.
2: while iter $<$ max-iter and err $>$ tol do
3:     $\nabla\widetilde{\ell}_{\mathcal{I}_k}(u) \leftarrow \nabla\ell_{\mathcal{I}_k}(u) - \nabla h_k(u)$
4:     $s \leftarrow s_0/\|\nabla\widetilde{\ell}_{\mathcal{I}_k}(u)\|_F$.
5:     repeat
6:         $\widetilde{u} \leftarrow u - s\nabla\widetilde{\ell}_{\mathcal{I}_k}(u)$.
7:         $u^+ \leftarrow \mathbf{prox}_{s\rho}(\widetilde{u})$.
8:         break if $\widetilde{\ell}_{\mathcal{I}_k}(u^+) \le \widetilde{\ell}_{\mathcal{I}_k}(u) + \left\langle \nabla\widetilde{\ell}_{\mathcal{I}_k}(u), u^+ - u \right\rangle + \frac{1}{2s}\|u^+ - u\|_F^2$.
9:         $s \leftarrow \kappa s$.
10:    err $\leftarrow \max_{ji} d\left(u^+_{ji}, u_{ji}\right)$ where $d(x,y) := \frac{\|x-y\|_F}{\max\{1,\|y\|_F\}}$.
11:    $u \leftarrow u^+$ and iter $\leftarrow$ iter $+1$.
12: end while
13: $\beta^{(t)}_{k,\pi} \leftarrow u$.
```

3.2.3 Selection of the tuning parameter

We use BIC grid search to select tuning parameter λ used in the group Lasso penalty, given an initial permutation π_0 (line 1, Algorithm 5). To construct the grid, we select 20 equal-space points of $\lambda^{(i)}$ in the log scale from the interval $[0.01, 0.1]$, where $\lambda = 0.1$ is sufficiently large for an empty graph in our test. We select the tuning parameter $\lambda^{(i)}$ that minimizes the BIC score, BIC $= 2\ell_{[n]}(\widehat{\beta}^{(i)}) + (\log n)\mathcal{N}(\widehat{\beta}^{(i)})$, where $\widehat{\beta}^{(i)} \in \mathcal{D}(\pi_0)$ is the minimizer of $F(\beta)$ with penalty parameter $\lambda^{(i)}$ and it is computed by Algorithm 6, and $\mathcal{N}(\widehat{\beta}^{(i)})$ is the number of free parameters in $\widehat{\beta}^{(i)}$.

3.2.4 Structure refinement after annealing

A GLDAG representation $\widehat{\beta}$ is provided at the end of the DARLS annealing search, from which we can recover a causal structure (line 9, Algorithm 5). Let W be a $p \times p$ weighted adjacency matrix of a DAG such that $W_{ij} := \|\widehat{\beta}_{ij}\|_F$. The use of a group Lasso regularizer helps to eliminate some edges in learning a DAG, but it may still result in false positive edges. Hence, we further refine estimated structures by setting W_{ij} to zero if $|W_{ij}| < \alpha \max_{ij} |W_{ij}|$. One can adjust the value of α to achieve a desired sparsity level, especially when having prior knowledge. In our simulation tests, we fix $\alpha = 0.1$ to remove edges whose weights are relatively small compared to others.

3.3 Theoretical guarantees

In this section, we study the convergence of the iterative distributed optimization algorithm (Algorithm 6) and establish the consistency of the global minimizer of (3.5). The convergence results (Theorem 3) is a joint work in (Ye et al, 2021), and it is included for completeness.

3.3.1 Distributed estimate convergence

Recall the local iteration functions $\varphi_{k,\pi}$ defined in (3.6). The overall iteration function for the distributed algorithm can be written as $\Phi_\pi(\cdot) := \sum_k \frac{n_k}{n} \varphi_{k,\pi}(\cdot)$ (line 6, Algorithm 6). Let $\Sigma := \mathbb{E}[x_h x_h^\top]$ be the population second-moment matrix of the model. For a matrix β, let $\mathbb{B}_F(\beta; r)$ denote the Frobenius ball of radius r centered at β. We consider the case of numerical variables, i.e. $d_j = 1$ for all j. The following theorem provides convergence guarantees on the distributed optimization algorithm represented by Φ_π for any fixed π. Let $\widehat{\beta}_\pi$ be any global minimizer of $\ell_{[n]}(\cdot) + \rho(\cdot)$ over $\mathcal{D}(\pi)$, where $\rho(\beta)$ is a convex regularizer. Let $\Omega := \cup_\pi \mathcal{D}(\pi)$ be the parameter space of GLDAGs. Recall that $\{x_h\}$ is an i.i.d. sample from a GLDAG model (3.2).

Theorem 3. *Assume that the coordinates of x_h are T-bounded, that is, $|x_h^j| \le T$ for all $h \in [n]$ and $j \in [p]$. Let $\theta \in \Omega$ be any GLDAG parameter and $r > 0$, and set $R_1^* = \max_j \|\theta_j\|_1$*

and $r_p := 2r\sqrt{p}$. *Let algo* : $pg_p = \inf_{|t|\le T(r_p+R_1^*)} b''(t)$, *and assume that* $b''(\cdot)$ *is* $\bar{b}_p$*-Lipschitz on* $[-Tr_p, Tr_p]$. *Define*

$$\zeta_n := \Big(T^3 \frac{\psi\big(\bar{b}_p(r + \frac{R_1^*}{\sqrt{p}})\big) + b''(0)}{\underline{b}_p \lambda_{\min}(\Sigma)}\Big) \frac{p^{3/2}\log(np)}{\sqrt{m}},$$

where $\psi(x) := \max\{x, \sqrt{x}\}$ *and* $m := \min_k |\mathcal{I}_k|$. *Assume further that* $np \ge \max\{K+1, 3\}$. *There exist constants* $c_1, C_1, C > 0$ *such that if* $C_1 T^2 \sqrt{p^2 \log(np)/m} \le \lambda_{\min}(\Sigma)$, *then with probability at least* $1 - 3(np)^{-c_1} - \mathbb{P}(\|\widehat{\beta}_\pi - \theta\|_F > r)$,

$$\|\Phi_\pi(\beta) - \widehat{\beta}_\pi\|_F \le C\zeta_n \, \|\beta - \widehat{\beta}_\pi\|_F, \quad \textit{for all}\, \beta \in \mathbb{B}_F(\widehat{\beta}_\pi, r).$$

Theorem 3 applies to any $\theta \in \Omega$. It is natural to take θ to be β^*_π, the minimizer of the population loss $\mathbb{E}[\ell_{[n]}(\cdot)]$ over $\mathcal{D}(\pi)$. Since $\widehat{\beta}_\pi$ is a consistent estimate of β^*_π for any π (Theorem 4, Section 3.3.2), we have that $\mathbb{P}(\|\widehat{\beta}_\pi - \beta^*_\pi\|_F > r)$ goes to zero as n grows. Thus, with high probability, the iteration operator $\Phi_\pi(\cdot)$ will be a contraction: the sequence $\{\beta_\pi^{(t)}\}_{t\ge 0}$ produced by the distributed algorithm converges geometrically to $\widehat{\beta}_\pi$ if $C\zeta_n < 1$. For fixed p, and for sufficiently large r such that $\theta \in \mathbb{B}_F(\widehat{\beta}_\pi, r)$, one can always satisfy the condition of $C\zeta_n < 1$ by taking m (the minimum sample size per machine) large enough. Hence, Theorem 3 provides a quantitative lower bound on m for the geometric convergence to kick in.

Theorem 3 is proved by establishing the uniform concentration of the Hessian of the GLDAG model (3.2) around its expectation over certain balls in the parameter space, and then invoking a general convergence result for the DANE algorithm which we derive in the Supplementary material (cf. Theorem 6). Note that establishing such uniform concentration in the GLDAG model is challenging due to the highly dependent and nonlinear relation among the coordinates $\{x_h^j\}_{j=1}^p$. A technical tool in establishing the concentration of the Hessian is the Ledoux–Talagrand contraction theorem. In order to extend the argument to the multi-logit and generally vector-valued DAG models with $d_j > 1$, one needs a multivariate extension of the contraction theorem which is not available in literature at the moment. This extensions is, in principle, possible and we leave it for the future work.

3.3.2 Consistency

Recall that $F(\beta) = \ell_{[n]}(\beta) + \lambda_n \sum_{i,j} \|\beta_{ij}\|_F$ is the global regularized negative log-likelihood in this paper, and Ω is the GLDAG parameter space. The optimization problem (3.5) is equivalent to $\min_{\beta \in \Omega} F(\beta)$. Denote by $\widehat{\beta} \in \Omega$ a global minimizer of $F(\beta)$ and $\beta^* \in \Omega$ the true parameter with the true DAG $\mathcal{G}^*$. We impose assumptions (A1) and (A2) to establish consistency results of $\widehat{\beta}$, where $p(x \mid \beta)$ is the joint density of a GLDAG parameterized by β. We also establish the consistency of $\widehat{\beta}_\pi$, used in Theorem 3, for any fixed π.

(A1) The graph $\mathcal{G}^*$ is identifiable. The true parameter β^* is faithful to $\mathcal{G}^*$ and is an interior point of Ω. The log-likelihood $p(x^j \mid pa_j, \beta_j)$ is strictly concave in β_j and continuously three times differentiable with respect to β_j for all $j \in [p]$.

(A2) There exist a neighborhood of the true parameter β^*, denoted by $\mathrm{nb}(\beta^*)$, and functions M_{jkl} such that $\left| \frac{\partial^3}{\partial \beta_j \partial \beta_k \partial \beta_l} \log p(x \mid \beta) \right| \leq M_{jkl}(x)$ for all $\beta \in \mathrm{nb}(\beta^*)$ and $\mathbb{E}_{\beta^*}\left[M_{jkl}(x)\right] < \infty$ for all j, k, l, where β^* and β are regarded as vectors.

Theorem 4. *Assume (A1) and (A2) hold, and $\lambda_n \sqrt{n} \to 0$. Then for any π, we have $\|\widehat{\beta}_\pi - \beta^*_\pi\|_F = \mathcal{O}_p(n^{-1/2})$, where $\widehat{\beta}_\pi$ is any global minimizer of $F(\beta)$ over $\mathcal{D}(\pi)$ and β^*_π is the minimizer of the population loss $\mathbb{E}[\ell_{[n]}(\beta)]$ over $\mathcal{D}(\pi)$. Moreover, there is a global minimizer $\widehat{\beta}$ of $F(\beta)$ over Ω, such that $\|\widehat{\beta} - \beta^*\|_F = \mathcal{O}_p(n^{-1/2})$.*

Theorem 4 confirms that the group Lasso regularized estimator is $\sqrt{n}$-consistent, where (A1) assumes identifiability (Definition 3) and faithfulness (Definition 4) of the model and (A2) is a standard regularity condition. The strict concavity assumption in (A1) is satisfied under the GLDAG model (3.2) if the Hessian of $b_j(\beta_j^\top x)$ is positive definite.

3.4 Numerical experiments

We first use the multinomial logistic (multi-logit) model as an example to illustrate GLDAG models. Then we assess the structure accuracy of DARLS (Algorithm 5) by comparing it to other existing methods using distributed simulation data. We also examine the performance of distributed optimization (Algorithm 6) in terms of accuracy and computational efficiency.

3.4.1 Multi-logit GLDAG models

Suppose categorical random variable X_j has r_j possible states $\{1,\ldots,r_j\}$, and the level of X_j is encoded by $d_j = r_j - 1$ dummy variables, i.e., $x_j = \{0,1\}^{d_j}$, for $j \in [p]$. Let us consider the multinomial logistic (multi-logit) model for $\{X_1,\ldots,X_p\}$:

$$\mathbb{P}\left(X_j = k \mid \mathrm{PA}_{X_j} = pa_{x_j}\right) = \frac{\exp\left(\langle \beta_j^k, x\rangle\right)}{1+\sum_{l=1}^{d_j}\exp\left(\langle \beta_j^l, x\rangle\right)}, \quad k = 1,\ldots,d_j, \tag{3.9}$$

where $\beta_j^l \in \mathbb{R}^{d+1}$ is the coefficient vector for X_j being level l. Model (3.9) is specified by $d_j = r_j - 1$ equations, reflecting the constraint that conditional probabilities of all states of X_j sum up to one. One can verify that multi-logit (3.9) belongs to the GLDAG family (3.2), by taking

$$b(\theta_j) = \log\Big(1+\sum_{l=1}^{d_j}\exp\theta_{jl}\Big), \quad \text{and} \quad c(x_j) = 0,$$

where $\theta_{jk} := \langle \beta_j^k, x\rangle$ for $k = 1,\ldots,d_j$ and $\theta_j = [\theta_{j1},\ldots,\theta_{jd_j}]$. Let $y_{hjl} := \mathcal{I}(x_h^j = l)$ be an indicator variable. Following (3.4), the normalized likelihood under the multi-logit DAG model (3.9) is

$$\ell_{[n]}(\beta) = \frac{1}{n}\sum_{j=1}^{p}\sum_{h=1}^{n}\left[\sum_{k=1}^{d_j} y_{hjk}\langle \beta_j^k, x_h\rangle - \log\left(1+\sum_{l=1}^{d_j}\exp\left(\langle \beta_j^l, x_h\rangle\right)\right)\right], \tag{3.10}$$

where $\beta = \{\beta_j^k,\ j = 1,\ldots,p,\ k = 1,\ldots,d_j\}$ collects all parameters. Let $(\beta_j^l)_0 \in \mathbb{R}$ be the first entry (i.e., the intercept coefficient) in β_j^l, and $(\beta_j^l)_i \in \mathbb{R}^{d_i}$ be the i-th block related to X_i in β_j^l, for $i = 1,\ldots,p$. Defining intercept parameter $\beta_{j0} := [(\beta_j^1)_0,\ldots,(\beta_j^{d_j})_0] \in \mathbb{R}^{d_j}$ and coefficient matrix $\beta_{ji} := [(\beta_j^1)_i,\ldots,(\beta_j^{d_j})_i] \in \mathbb{R}^{d_j\times d_i}$, the gradients of (3.10) w.r.t β_{j0} and β_{ji}

for $i, j \in [p]$ are

$$\frac{\partial \ell(\beta)}{\partial \beta_{j0}} = \frac{1}{n}\sum_{h=1}^{n} \begin{pmatrix} y_{hj1} - P(X_j = 1 \mid \beta, pa_{x_j}) \\ \cdot \\ \cdot \\ y_{hjd_j} - P(X_j = d_j \mid \beta, pa_{x_j}) \end{pmatrix} \in \mathbb{R}^{d_j}, \quad \text{and}$$

$$\frac{\partial \ell(\beta)}{\partial \beta_{ji}} = \frac{1}{n}\sum_{h=1}^{n} \begin{pmatrix} \left(y_{hj1} - P(X_j = 1 \mid \beta, pa_{x_j})\right) x_h^i \\ \cdot \\ \cdot \\ \left(y_{hjd_j} - P(X_j = d_j \mid \beta, pa_{x_j})\right) x_h^i \end{pmatrix} \in \mathbb{R}^{d_j \times d_i},$$

which are used in the distributed optimization algorithm (line 6, Algorithm 5). We use the multi-logit DAG model to simulate categorical data in our numerical tests, which will be discussed in the next subsection.

3.4.2 Methods and data

Denote by s_0 the number of edges in a graph, the following `networks` (p, s_0) were downloaded from the Bayesian networks repository (Scutari, Accessed: 2019), licensed under the Creative Commons Attribution-Share Alike License, to simulate our large data sets: `Asia` (8, 8), `Sachs` (11, 17), `Child` (20, 25), `Insurance` (27, 52), `Alarm` (37, 46), `Hailfinder` (56, 66) and `Hepar2` (70, 123). Logistic GLDAG and multinomial models were used to generate 20 data sets for each network under two settings: $n = 100p, K = 10$ and $n = 10,000, K = 20$, where a total of n observations were randomly assigned to K local machines. In particular, binary data were generated under logistic GLDAGs with parameters $\{\beta_{ji}\}$ uniformly sampled from $[-1.5, -0.8] \cup [0.8, 1.5]$. To examine the robustness of our method against violations of GLM assumptions, multinomial data were also simulated from contingency tables provided in the BN repository with modifications made to ensure (1) the number of states per variable was at most three, and (2) marginal probability of any state was at least 10% for every variable by merging states. However, due to the high structure complexity of `Hailfinder` and `Hepar2`, we kept the original distribution of a few nodes in these two networks, resulting in marginal probability less than 10% for some states.

We compared DARLS algorithm to the following five causal structure learning methods: the standard greedy hill climbing (HC) algorithm (Gámez et al, 2011), the Peter-Clark (PC) algorithm (Spirtes and Glymour, 1991), the max-min hill-climbing (MMHC) algorithm (Tsamardinos et al, 2006), the fast greedy equivalence search (FGES) (Chickering, 2002; Ramanan and Natarajan, 2020), and the NOTEARS algorithm (Zheng et al, 2018). Since none of these competing algorithms can handle distributed data, we applied standard methods on each independent data set $\{x_h\}_{h\in\mathcal{I}_k}$ to obtain local estimate A_k, and then constructed a global graph using $\{A_k\}_{k=1}^K$ by counting the occurrence of each edge direction and sequentially adding the edges to an empty graph subject to the acyclicity constraint. In particular, given $\{A_k\}_1^K$, we counted occurrences of each edge direction, and there are three possible orientations between i and j: $i \to j$, $i \leftarrow j$, or $i - j$ (undirected). We ranked all directions in a descending order of their counts, and then sequentially added these edges to an empty graph, as long as it would not introduce a directed cycle (a cycle consisted with all directed edges). At last, we applied Meek's rule to maximally orient directed edges, and hence constructed a global structure that satisfied the acyclicity constraint. HC global estimates had too many edges using this approach, because its local machines lacked consensus on structure estimation, resulting in a large number of candidate edges. To solve this problem, we counted the number of "no edges" between i and j when using the HC algorithm, and we would not add connection between i and j if majority of local graphs had no edges between them. In this way, the sparsity of graphs estimated by HC was reduced and became more reasonable.

We implemented the DARLS algorithm in Matlab and used the following packages to run competing methods: `bnlearn` (Scutari, 2010) for the MMHC and HC algorithms, `pcalg` (Kalisch et al, 2012) for the PC algorithms and `rcausal` (Ramsey, 2015) for the FGES. The NOTEARS method was run with online Python code (Zheng, 2019). These competing methods were applied on each local data set using a 2016 MacBook Pro (2.9 GHz Intel Core i5, 16 GB memory), since they could not handle distributed data. DARLS learned causal structures through parallel computation, so it was run on UCLA's Hoffman2 Cluster.

In the DARLS algorithm (Algorithm 5), we started with temperature $5 \cdot 10^{-2}$ and gradually decreased it to $5 \cdot 10^{-5}$ with a total iteration of 10^3. A significance level of 0.01 was used for the PC algorithm to generate graphs with desires sparsities. FGES was applied with a significance level of 0.1, which was the default value. For MMHC and HC methods, the maximum number of parents for a node was set to be three.

3.4.3 Structure learning accuracy

We computed the Structural Hamming Distance (SHD) between an estimated CPDAG and the true CPDAGs (if a DAG is estimated, we converted it to CPDAG), and used it to assess each method's performance. SHD counts the number of edge insertions, deletions or flips in order to transform the estimated graph to the true one, and thus lower SHD corresponds to higher structure learning accuracy. Since GLDAG is identifiable, we compared DAGs for DARLS's SHD on binary data generated by GLDAG models. We explain the calculation of SHD in the next paragraph.

Let P, TP, FP, M, R be the numbers of estimated edges, true positive edges, false positive edges, missing edges and reverse edges, respectively. P is the number of edges in the estimated graph. FP is the number of edges in the estimated graph skeleton but not in the te skeleton, and M counts the number of edges in the true skeleton but not in the skeleton of the estimated graph. TP reports the number of consistent edges between the estimated DAG/CPDAG and the true DAG/CPDAG, where consistent edges have the same direction between two nodes: there are two edge orientations between two nodes in DAGs and three directions in CPDAGs. Lastly, the number of reversed edges $\mathrm{R} = \mathrm{P} - \mathrm{TP} - \mathrm{FP}$. We then define structural Hamming distance (SHD) and Jaccard index (JI): $\mathrm{SHD} = \mathrm{R} + \mathrm{FP} + \mathrm{M}$ and $\mathrm{JI} = \mathrm{TP}/(s_0 + \mathrm{P} - \mathrm{TP})$. A method has higher structure learning accuracy if it achieves a lower SHD and/or a higher JI.

Figure 3.1 compares DARLS using distributed data over K machines to (1) the best performance among the above five competing methods using the same distributed data (best-

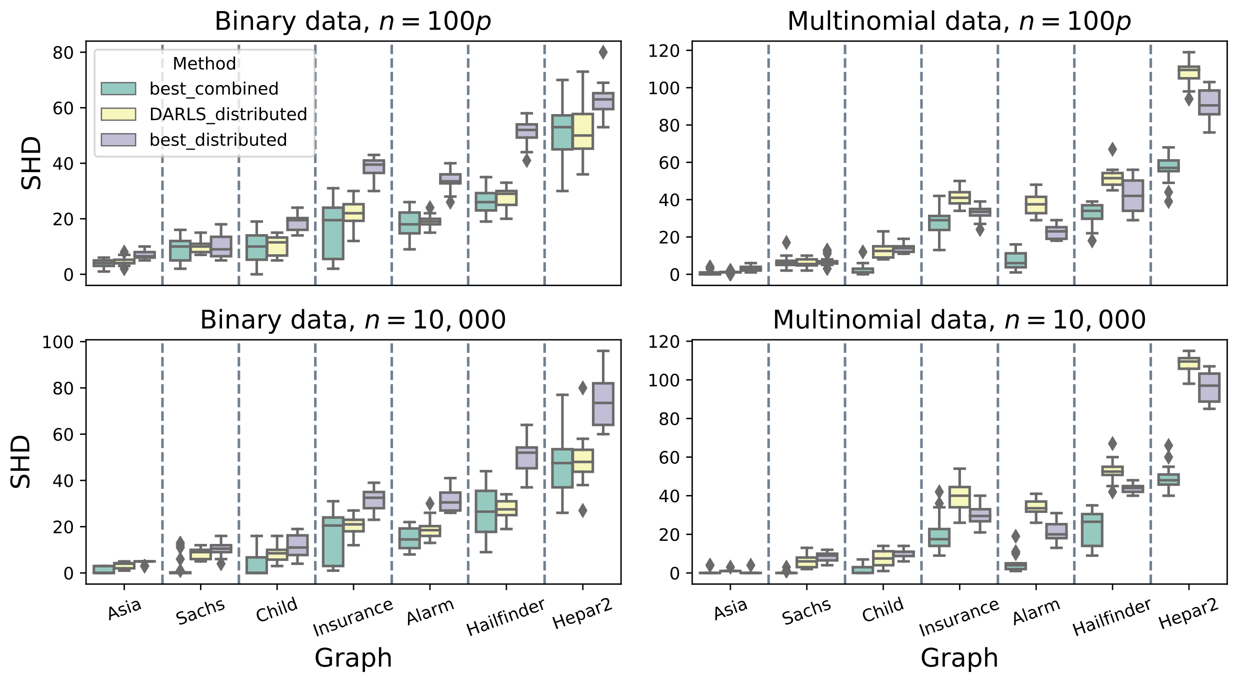


Figure 3.1: SHD comparison between DARLS on distributed data and the best method using combined or distributed data.

distributed method) and (2) the best among all six methods, including DARLS, using the pooled data across all local machines (best-combined method). The best-combined method is expected to achieve the lowest SHD. However, DARLS and the best-combined method had substantial overlaps in the distribution of SHD, especially when using binary data to learn DAG structures, indicating highly competitive performance of our algorithm using distributed data compared to the best method on combined data. DARLS consistently outperformed the best-distributed method when using binary data to learn DAGs across all networks.

In the multinomial data setting where GLM assumption was violated, DARLS exhibited higher structure accuracy than the best-distributed method for smaller graphs, but its improvement diminished for larger graphs (`Insurance`, `Alarm`, `Hailfinder` and `Hepar2`). However, it is comforting to see DARLS achieved higher accuracy than most of the competing methods using distributed data (Tables 3.1 and 3.2). Moreover, variability in SHD of DARLS was smaller than that of the best competing methods, showing its higher consistency across data sets.

We also summarize average values of TP, R, FP, M, SHD and JI over 20 data sets for each graph and each method. Table 3.1 shows results for $n = 100p, K = 10$, allowing n to grow with p, which includes binary and multinomial data experiments. In the binary data setting, DARLS had the lowest SHDs for every graphs, and its JI was the highest for most networks (except `Sachs`). Using multinomial data, its SHDs and JIs were the best for three graphs, `Asia`, `Sachs` and `Child`. For larger networks, though DARLS was not the best method, its performance was still better than the most of the competing methods. NOTEAR estimates were too sparse to be competitive, even we decreased the penalty tuning parameter to 10^{-4} while the suggested value was 10^{-1}, and its ratio of TP versus (R + FP) was also much lower than that of other methods. Hence, we only showed its results for the two small graphs, `Asia` and `Sachs`. PC had difficulty generating estimates within a reasonable time limit for the last two networks, `Hailfinder` and `Hepar2`, so we removed its results from comparisons using these two networks.

Table 3.2 shows big data set results where $n = 10,000$ and $K = 20$. The results were similar to those in Table 3.1, where DARLS consistently achieved the highest accuracy when using binary data. When the underlying data generation model is multinomial, its improvements diminished slightly, especially in large graphs.

3.4.4 Distributed optimization accuracy and computational time

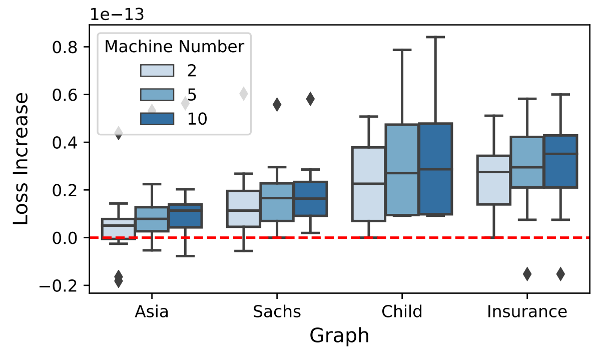
To quantify the accuracy of distributed optimization using Algorithm 6, we compared permutation scores $f(\pi)$ (3.5) under various values of $K \in \{1, 2, 5, 10\}$ for a fixed π. For each value of K, we fixed the tuning parameter λ, permutation π and all internal computation parameters to ensure only K varied in the calculation of $f(\pi)$. Let $f^{(K)}$ be the value of $f(\pi)$ computed by K local machines, and $\Delta f^{(K)} := f^{(K)} - f^{(1)}$ be the relative increase in the loss $f^{(K)}$. We compared $\{\Delta f^{(K)}, K = 2, 5, 10\}$ across selected networks in Figure 3.2a, where values of $\Delta f^{(K)}$ were in the order of 10^{-13}, verifying $f(\pi)$ was essentially identical using either the overall ($K = 1$) or distributed data ($K \geq 2$). Since the number of iterations were fixed, $\Delta f^{(K)}$ increased with the greater network size.

Table 3.1: DARLS against other methods on simulation data with $n = 100p$ and $K = 10$.

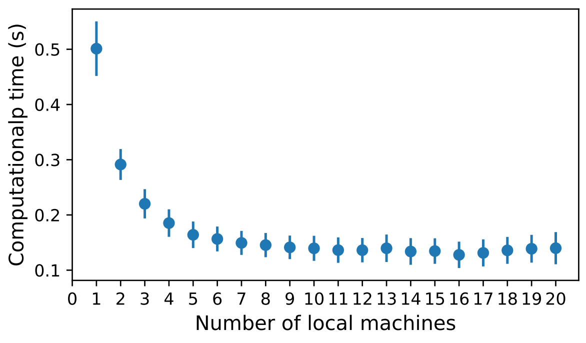
Network	Method	Binary Data						Multinomial Data					
(n, p, s_0)		TP	R	FP	M	SHD (sd)	JI (sd)	TP	R	FP	M	SHD (sd)	JI (sd)
Asia	DARLS	3.7	3.1	0.3	1.1	4.6 (1.6)	0.33 (0.15)	6.9	0.1	0.1	1.0	1.1 (0.6)	0.85 (0.09)
(800, 8, 8)	HC	0.8	0.8	0.0	6.3	7.2 (0.8)	0.10 (0.10)	4.9	0.5	0.1	2.6	3.2 (1.3)	0.59 (0.18)
	MMHC	2.6	3.0	1.4	2.4	6.7 (1.3)	0.22 (0.06)	1.9	3.6	0.1	2.5	6.2 (0.9)	0.17 (0.11)
	PC	2.1	3.1	0.8	2.7	6.7 (1.4)	0.19 (0.10)	1.4	3.8	0.1	2.9	6.7 (0.7)	0.12 (0.05)
	FGES	2.5	3.8	3.5	1.7	9.0 (2.4)	0.17 (0.09)	5.8	1.6	6.0	0.7	8.2 (1.7)	0.38 (0.10)
	NOTEARS	0.5	0.9	4.2	6.6	11.8 (1.1)	0.04 (0.05)	0.5	1.0	4.4	6.5	11.9 (1.7)	0.04 (0.07)
Sachs	DARLS	7.5	7.2	0.5	2.4	10.0 (2.1)	0.31 (0.10)	11.1	1.9	0.0	4.0	6.0 (2.4)	0.60 (0.18)
(1100, 11, 17)	HC	3.6	0.5	0.0	12.9	13.3 (1.4)	0.21 (0.09)	7.8	1.1	0.0	8.1	9.2 (1.7)	0.44 (0.12)
	MMHC	8.8	3.5	1.8	4.7	10.0 (4.7)	0.44 (0.26)	10.2	1.2	0.1	5.5	6.8 (2.5)	0.58 (0.17)
	PC	4.7	6.5	1.4	5.8	13.7 (2.4)	0.20 (0.12)	7.0	1.1	0.1	8.9	10.1 (2.3)	0.40 (0.15)
	FGES	3.2	9.3	5.2	4.5	19.1 (3.1)	0.11 (0.07)	2.0	12.1	2.9	2.9	17.9 (2.9)	0.07 (0.07)
	NOTEARS	0.0	2.6	4.8	14.3	21.8 (1.4)	0.00 (0.00)	0.0	4.5	7.1	12.5	24.1 (3.8)	0.00 (0.00)
Child	DARLS	15.1	7.4	0.5	2.5	10.4 (3.5)	0.47 (0.14)	12.9	7.2	0.8	4.8	12.9 (4.2)	0.41 (0.16)
(2000, 20, 25)	HC	6.2	5.3	0.0	13.4	18.8 (2.8)	0.21 (0.10)	11.1	6.2	0.1	7.7	14.0 (2.1)	0.36 (0.07)
	MMHC	9.2	12.7	6.2	3.1	22.1 (3.7)	0.21 (0.07)	10.2	8.3	0.5	6.5	15.3 (1.8)	0.30 (0.06)
	PC	6.5	14.4	4.7	4.0	23.1 (3.3)	0.15 (0.05)	4.5	7.5	0.3	13.1	20.9 (1.0)	0.14 (0.03)
	FGES	6.5	14.0	10.2	4.5	28.8 (4.7)	0.13 (0.06)	11.4	11.2	4.7	2.5	18.3 (2.8)	0.29 (0.10)
Insurance	DARLS	30.9	16.8	0.6	4.2	21.6 (5.0)	0.45 (0.10)	14.6	17.8	4.0	19.6	41.5 (4.6)	0.20 (0.07)
(2700, 27, 52)	HC	11.0	11.3	0.0	29.6	41.0 (4.1)	0.18 (0.07)	19.8	9.3	0.9	22.9	33.1 (3.8)	0.32 (0.07)
	MMHC	18.3	26.7	4.5	7.0	38.2 (3.7)	0.22 (0.05)	12.8	14.4	0.5	24.7	39.6 (2.3)	0.19 (0.04)
	PC	15.8	26.9	3.5	9.2	39.6 (2.9)	0.19 (0.03)	8.2	15.2	0.9	28.6	44.8 (2.2)	0.12 (0.04)
	FGES	18.4	24.9	9.3	8.7	42.9 (5.1)	0.22 (0.05)	20.7	16.9	14.7	14.4	46.0 (5.1)	0.25 (0.04)
Alarm	DARLS	27.4	15.3	0.2	3.3	18.8 (2.3)	0.45 (0.05)	17.1	17.9	8.4	10.9	37.3 (5.9)	0.24 (0.07)
(3700, 37, 37)	HC	10.9	18.8	0.0	16.3	35.0 (4.7)	0.17 (0.08)	25.4	13.3	3.0	7.3	23.6 (5.9)	0.42 (0.11)
	MMHC	17.9	25.8	10.3	2.4	38.5 (5.7)	0.22 (0.05)	24.1	12.4	0.9	9.5	22.9 (3.5)	0.41 (0.08)
	PC	20.1	22.8	7.9	3.1	33.8 (3.2)	0.26 (0.04)	9.3	25.9	1.0	10.8	37.7 (3.3)	0.13 (0.05)
	FGES	19.2	22.1	10.3	4.7	37.1 (4.4)	0.25 (0.05)	38.3	3.6	19.1	4.0	26.8 (4.5)	0.56 (0.06)
Hailfinder	DARLS	39.2	23.0	0.8	3.8	27.5 (3.5)	0.44 (0.05)	16.9	14.3	2.6	34.8	51.7 (4.9)	0.21 (0.05)
(5600, 56, 66)	HC	14.8	38.2	0.0	13.0	51.2 (4.4)	0.14 (0.05)	24.6	18.4	2.0	23.1	43.5 (3.1)	0.29 (0.04)
	MMHC	16.4	44.6	19.6	5.0	69.2 (6.4)	0.13 (0.02)	15.1	19.8	1.9	31.1	52.8 (2.7)	0.17 (0.03)
	FGES	23.0	39.0	13.0	4.0	56.0 (6.5)	0.20 (0.04)	33.9	17.8	9.8	14.3	42.0 (8.8)	0.37 (0.12)
Hepar2	DARLS	72.9	37.9	1.6	12.2	51.7 (10.0)	0.45 (0.07)	17.4	29.0	2.5	76.5	108.1 (6.6)	0.11 (0.05)
(7000, 70, 123)	HC	60.2	29.9	0.0	32.9	62.8 (5.9)	0.39 (0.04)	33.4	20.4	1.6	69.2	91.3 (7.8)	0.23 (0.06)
	MMHC	53.0	45.2	19.1	24.7	89.0 (7.6)	0.28 (0.04)	34.9	20.1	21.9	68.0	110.1 (5.7)	0.21 (0.03)
	FGES	75.8	36.8	18.9	10.4	66.2 (9.9)	0.43 (0.05)	58.5	17.9	37.6	46.6	102.1 (7.0)	0.33 (0.03)

Table 3.2: DARLS against other methods on simulation data with $n = 10,000$ and $K = 20$.

Network	Method	Binary Data						Multinomial Data					
(p, s_0)		TP	R	FP	M	SHD (sd)	JI (sd)	TP	R	FP	M	SHD (sd)	JI (sd)
Asia	DARLS	4.5	3.0	0.0	0.5	3.5 (1.3)	0.43 (0.17)	7.0	0.0	0.1	1.0	1.1 (0.4)	0.87 (0.04)
(8, 8)	HC	3.3	3.5	0.0	1.2	4.7 (0.7)	0.29 (0.09)	7.8	0.1	0.1	0.0	0.2 (0.9)	0.97 (0.13)
	MMHC	3.1	4.8	0.3	0.1	5.2 (1.0)	0.25 (0.08)	4.8	1.2	0.5	2.0	3.8 (1.3)	0.50 (0.10)
	PC	3.0	4.8	0.6	0.1	5.5 (1.5)	0.24 (0.13)	2.5	3.5	0.3	2.0	5.8 (0.7)	0.21 (0.05)
	FGES	2.8	5.0	3.0	0.1	8.2 (2.2)	0.19 (0.13)	5.8	2.2	5.0	0.0	7.3 (2.0)	0.39 (0.12)
	NOTEARS	0.2	0.6	2.4	7.2	10.2 (1.1)	0.02 (0.04)	0.6	0.9	3.8	6.5	11.2 (1.7)	0.05 (0.07)
Sachs	DARLS	8.7	7.0	0.0	1.2	8.3 (2.3)	0.37 (0.12)	10.9	3.1	0.1	3.0	6.2 (3.3)	0.57 (0.24)
(11, 17)	HC	6.8	4.0	0.0	6.2	10.2 (2.8)	0.34 (0.17)	6.7	5.9	0.0	4.5	10.3 (4.3)	0.33 (0.25)
	MMHC	6.8	9.2	0.8	1.1	11.1 (3.5)	0.27 (0.18)	3.1	11.2	0.1	2.8	14.0 (1.1)	0.11 (0.04)
	PC	5.9	10.2	0.7	0.9	11.8 (4.4)	0.25 (0.26)	8.8	1.1	0.1	7.0	8.3 (2.5)	0.49 (0.17)
	FGES	2.4	13.8	4.8	0.8	19.4 (3.5)	0.07 (0.05)	4.4	11.8	2.9	0.8	15.5 (4.5)	0.16 (0.19)
	NOTEARS	0.0	2.3	3.5	14.7	20.6 (1.7)	0.00 (0.00)	0.1	2.8	5.7	14.2	22.6 (2.4)	0.00 (0.01)
Child	DARLS	16.9	7.2	0.1	0.9	8.2 (3.7)	0.54 (0.16)	18.1	4.0	0.6	2.9	7.5 (4.3)	0.63 (0.20)
(20, 25)	HC	13.3	6.8	0.0	4.9	11.7 (4.9)	0.44 (0.20)	14.9	5.0	0.1	5.0	10.2 (2.3)	0.50 (0.10)
	MMHC	10.2	14.6	4.3	0.3	19.2 (3.9)	0.24 (0.09)	12.2	8.2	0.5	4.7	13.3 (3.9)	0.38 (0.14)
	PC	8.1	16.2	3.9	0.7	20.8 (4.5)	0.19 (0.11)	6.0	10.8	0.6	8.2	19.6 (2.5)	0.17 (0.07)
	FGES	11.7	12.8	9.7	0.6	23.0 (5.0)	0.25 (0.08)	14.4	9.9	3.9	0.6	14.4 (2.9)	0.38 (0.08)
Insurance	DARLS	31.9	15.4	0.3	4.7	20.4 (4.3)	0.47 (0.08)	15.7	16.1	3.5	20.2	39.8 (7.6)	0.23 (0.09)
(27, 52)	HC	20.4	11.9	0.0	19.7	31.6 (4.9)	0.32 (0.09)	22.5	10.7	0.8	18.9	30.2 (4.9)	0.36 (0.09)
	MMHC	22.1	27.1	4.1	2.9	34.0 (5.3)	0.27 (0.07)	16.6	15.2	0.9	20.2	36.3 (2.9)	0.25 (0.05)
	PC	17.9	30.3	3.4	3.8	37.5 (4.4)	0.21 (0.05)	7.3	20.7	0.9	24.0	45.6 (2.8)	0.10 (0.04)
	FGES	23.9	23.6	11.8	4.5	39.9 (7.0)	0.28 (0.05)	23.1	16.6	20.4	12.3	49.2 (6.0)	0.26 (0.04)
Alarm	DARLS	27.1	16.2	0.1	2.8	19.1 (4.1)	0.44 (0.09)	18.1	17.6	6.2	10.2	34.0 (4.0)	0.26 (0.06)
(37, 46)	HC	14.5	20.6	0.0	10.9	31.5 (4.9)	0.22 (0.08)	27.1	12.3	2.4	6.5	21.2 (4.5)	0.45 (0.10)
	MMHC	17.6	27.8	14.5	0.7	43.0 (5.7)	0.20 (0.05)	24.6	13.9	2.3	7.4	23.6 (2.6)	0.40 (0.05)
	PC	18.9	26.4	13.3	0.8	40.5 (5.3)	0.22 (0.06)	11.6	25.2	1.8	9.2	36.2 (2.4)	0.16 (0.03)
	FGES	23.5	21.2	18.9	1.3	41.4 (7.7)	0.28 (0.06)	36.6	4.4	25.6	5.0	35.0 (6.0)	0.49 (0.06)
Hailfinder	DARLS	39.0	22.6	0.8	4.3	27.7 (4.6)	0.44 (0.07)	16.6	15.2	3.5	34.2	52.9 (5.5)	0.20 (0.06)
(56, 66)	HC	15.7	34.4	0.0	15.9	50.4 (6.6)	0.16 (0.08)	23.8	18.0	1.7	24.2	44.0 (2.3)	0.28 (0.03)
	MMHC	17.1	45.6	42.8	3.3	91.7 (8.7)	0.11 (0.02)	14.3	22.4	4.8	29.3	56.4 (4.1)	0.16 (0.03)
	FGES	21.7	41.0	28.1	3.3	72.5 (8.5)	0.16 (0.03)	28.1	23.9	17.6	14.0	55.6 (6.3)	0.26 (0.07)
Hepar2	DARLS	75.5	35.8	1.3	11.7	48.8 (10.4)	0.47 (0.08)	18.1	28.6	3.2	76.3	108.1 (4.8)	0.12 (0.04)
(70, 123)	HC	49.5	30.1	0.0	43.5	73.5 (11.0)	0.33 (0.09)	28.1	19.8	1.9	75.1	96.8 (7.8)	0.20 (0.06)
	MMHC	51.0	55.9	46.5	16.1	118.5 (10.1)	0.23 (0.03)	35.0	25.1	51.5	62.9	139.4 (8.7)	0.18 (0.03)
	FGES	76.4	34.3	39.9	12.3	86.5 (11.9)	0.39 (0.04)	56.8	20.0	74.5	46.2	140.7 (11.0)	0.26 (0.02)



(a) Relative loss distribution.



(b) Computational time distribution.

Figure 3.2: Accuracy and computational time comparison for $f(\pi)$. Computational time comparison, with mean and standard deviations plotted, is performed over 20 data sets generated from `Insurance`.

We also compared computational time of finding $f^{(K)}$ for $K \in [20]$ using 20 `Insurance` data sets. In each test, the same data were split and distributed to different number K of local machines, with all other parameters fixed to solve the optimization problem (3.5). We show how computational time varied with respect to K in Figure 3.2b. As a merit of distributing a complicated task, computing $f(\pi)$ required less time when using more machines. However, the reduction in computational time reached approximately a stable level after $K = 10$, which indicates a trade-off between parallel computation and communication overhead.

3.5 Real data application

In this section, we apply our methods to the ChIP-Seq data generated by Chen et al (2008). The data set contains the DNA binding sites of 12 transcription factors (TFs) in mouse embryonic stem cells: *Smad1, Stat3, Sox2, Pou5f1, Nanog, Esrrb, Tcfcp2l1, Klf4, Zfx, E2f1, Myc,* and *Mycn.* For each TF, an association strength score, which is the weighted sum of the corresponding ChIP-Seq signal strength, was calculated for each of the 18,936 genes Ouyang et al (2009). Roughly speaking, this score can be understood as a measure of the binding strength of a TF to a gene. Following the same preprocessing in Wang and Zhou (2021), the genes with zero association scores were removed from our analysis. Accordingly,

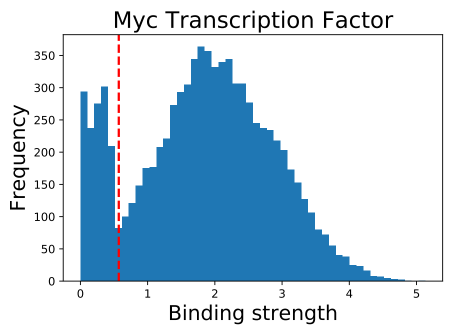


Figure 3.3: Histogram of TF Mycn. Red dotted line separates two clusters.

our observed data matrix, of size $n \times p = 8462 \times 12$, contains the association scores of 12 TFs over 8,462 genes. We aim to build a causal network that reveals how these 12 TFs affect each other's binding to genes.

The associate scores of a TF were discretized before network estimation. Distributions of the binding strength of TF in Chip-Seq were bimodal with two different-size clusters: a smaller one with samples mostly gathered around zero and a larger one with a center greater than zero. Thus, we applied the K-means clustering (with $K = 2$) on its log-transformed data to determine cut-off values to discretize the data into binary values. Figure 3.3 illustrates a TF discretization pattern.

3.5.1 Test data likelihood comparison

We distributed this data set across $K = 20$ local machines and applied DARLS, HC, MMHC, PC and FGES on distributed data to learn the protein-DNA binding network; we excluded the NOTEARS algorithm because its performance was not competitive. Local estimates of a competing method were combined to construct a global graph as we did in the simulation study.

Test data likelihood under multinomial DAG models in ten-fold cross validation was used to assess the accuracy of estimated networks, since the true network structure is unknown. Denote by g the likelihood values using training under multinomial DAG models:

$$g_{j|pa_j} = \prod_{k=1}^{r_j} (p^k_{j|pa_j})^{n^k_{j|pa_j}},$$

where $n^k_{j|pa_j}$ is the number of $X_j = k$ given $\mathrm{PA}_j = pa_j$, and $p^k_{j|pa_j}$ is the corresponding conditional probability. Hence, the overall test data log-likelihood is

$$\log g = \sum_j \sum_{pa_j} \log(g_{j|pa_j}) = \sum_j \sum_{pa_j} \sum_{k=1}^{r_j} n^k_{j|pa_j} \log(\hat{p}^k_{j|pa_j}),$$

where $\hat{p}^k_{j|pa_j}$ is an estimated conditional probability using training samples. Denoting by $\tilde{n}_{pa_j}$ the number of training samples such that $\mathrm{PA}_j = pa_j$, we let $\hat{p}^k_{j|pa_j} := (\tilde{n}^k_{j|pa_j} + \alpha_{jkpa_j})/(\tilde{n}_{pa_j} + \alpha_{j\cdot pa_j})$, where pseudo count α_{klpa_j} is used to ensure numerical validity and $\alpha_{j\cdot pa_j} = \sum_k \alpha_{jkpa_j}$. We define $\alpha_{jkpa_j} := \alpha/(r_j q_j)$, where α is a fixed small number (taking value of 1 in our test) and q_j is the number of states for PA_j. Denote by $\widetilde{g}$ the training data likelihood under the multinomial DAG model, and we can find its value as how we compute g. We further computed $\mathrm{BIC} = -2\log\widetilde{g} + \log(\widetilde{n})\mathcal{N}(\widehat{\mathcal{G}})$, where $\widetilde{n}$ is the training sample size and $\mathcal{N}(\widehat{\mathcal{G}})$ is the number of multinomial parameters for estimated graph $\widehat{\mathcal{G}}$. Test likelihood g and BIC score are used to measure accuracy of each method.

Denoting by g_{D} and $\mathrm{BIC}_{\mathrm{D}}$ the DARC's test data likelihood and BIC, respectively, and using them as benchmarks, we summarize the log-likelihood difference $\Delta(\log g) = \log g - \log g_{\mathrm{D}}$ and the BIC difference $\Delta\mathrm{BIC} = \mathrm{BIC} - \mathrm{BIC}_{\mathrm{D}}$ for each method in Table 3.3. Structure learning methods achieve different sparsity of their estimators by varying internal tuning parameters, such as penalty parameter in DARLS and significance levels in MMHC and PC algorithms. To ease the comparison, we controlled the sparsity of estimated networks such that every method produced two graphs, with roughly 17 and 28 edges, respectively. The number of edge is represented be $|E|$. FGES was reported with around 12 edges because it had difficulty generating output close to 17 edges. In both cases, DARLS achieved the highest test data likelihood and the smallest BIC, outperforming other methods by a substantial margin. Note that $\log g$ is the test log-likelihood while BIC is using the training data. Thus the magnitude of Δ BIC is much larger than $\Delta(\log g)$. The value of $\exp\{-\Delta\mathrm{BIC}/(2\widetilde{n})\}$ is also reported in the table as an approximation to the normalized marginal likelihood ratio (NLR) $(P(\widetilde{X} \mid \widehat{\mathcal{G}})/P(\widetilde{X} \mid \widehat{\mathcal{G}}_D))^{1/\widetilde{n}}$, where $\widetilde{X}$ denotes training data, between estimated DAGs $\widehat{\mathcal{G}}$ by a competing method and $\widehat{\mathcal{G}}_D$ by DARLS. All the NLRs are substantially < 1, showing that $\widehat{\mathcal{G}}_D$ fit the data much better. Note that the likelihood and BIC here are calculated under

Table 3.3: Test data log-likelihood and BIC comparison on the ChIP-Seq dataset.

Method	Sparse				Moderate			
	$\|E\|$	$\Delta(\log g)$	ΔBIC	NLR	$\|E\|$	$\Delta(\log g)$	ΔBIC	NLR
DARLS	16.5	0.0	0.0	1.000	27.5	0.0	0.0	1.000
HC	17.0	−15.0	1458.4	0.384	29.0	−14.3	877.1	0.562
MMHC	17.0	−12.7	1418.4	0.394	29.5	−21.9	764.7	0.605
PC	17.0	−19.4	1458.4	0.384	29.0	−14.4	644.9	0.655
FGES	11.5	−27.4	1426.9	0.392	28.0	−17.1	728.9	0.620

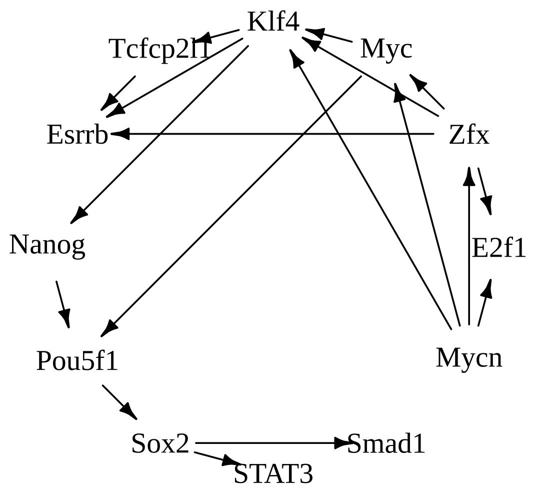
the multinomial model, instead of the GLDAG model. Thus, the superior performance of $\widehat{\mathcal{G}}_D$ learned by DARLS on this real-world data suggests that our proposed GLDAG model is a good approximation to the underlying data generation mechanism.

3.5.2 Protein-DNA binding networks estimated by DARLS

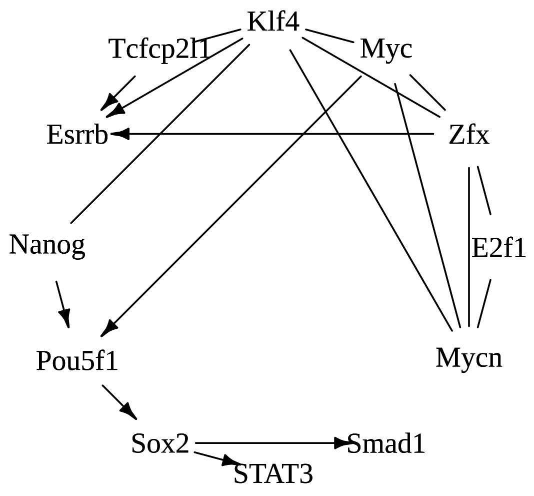
To gain more scientific insights, we show in Figure 3.4 the sparser DAG ($P = 17$) and its converted CPDAG, learned by DARLS from the complete data set ($n = 8,462$) distributed over $K = 20$ local machines. One interesting observation is the directed path Nanog→Pou5f1→Sox2 in the estimated CPDAG, among the three core regulators in the gene regulatory network in mouse embryonic stem cells (Chen et al, 2008; Zhou et al, 2007). It is well-known that many genes are co-regulated by Pou5f1, Sox2 and Nanog. The estimated path suggests that Nanog binding would cause the binding of Pou5f1, which then may cause Sox2 binding. This provides new clue for how the three TFs work together to co-regulate downstream genes.

3.6 Discussion

We have developed the DARLS algorithm to learn causal networks from distributed data, which incorporates a distributed optimization method in simulated annealing. To the best of knowledge, it is the first method learning causal graphs from data distributed over multiple



(a) DAG.



(b) CPDAG.

Figure 3.4: DAG and the converted CPDAG learned by DARLS with $\lambda = 0.06$ and refinement parameter $\alpha = 0.3$.

machines, using distributed optimization that relies on multiple rounds of communication between local and central machines. The GLDAG model we proposed includes a family of flexible distributions besides linear Gaussian models, and thus can be applied to different types of data. Our primary focus is on the big and distributed data case, with large n but moderate p. However, generalizing the convergence and consistency results to allow diverging p is interesting theoretically and left as future work.

3.7 Proofs

We first present proofs of main results in Section 3.7.1, where Section 3.7.1.2 is a joint work in (Ye et al, 2021), and then show proofs of lemmas in Section 3.7.2. Proofs of other results are moved to the end of this chapter, in Section 3.7.3.

3.7.1 Proofs of main results

3.7.1.1 *Proof of Proposition 4*

Proof. We first show the skeleton of a continuous GLDAG (3.3) defined by β is unique in the following lemma, and then prove the identifiability stated in Proposition 4. Proofs of this and following lemmas are provided in Section 3.7.2. Since x^j is reduced to a scalar under continuous GLDAG models, we use x_j rather than x^j for a realization of X_j in this proof.

Lemma 9. *Suppose the joint distribution $L(X)$ is defined by the log-pdf $L(x;\beta)$ with a DAG $\mathcal{G}_0$ according to* (3.3) *such that $\beta_{ij} \neq 0$ if and only if $i \in PA_j$ in $\mathcal{G}_0$. If $L(x;\beta)$ is second-order differentiable with respect to x, then any GLDAG that defines $L(X)$ has the same skeleton.*

Let $\mathcal{G}$ be the DAG defined by the support of β. Suppose there is another GLDAG $\widetilde{\mathcal{G}}$ parameterized by $\widetilde{\beta} \neq \beta$, which generates the same distribution. Since DAGs contains no cycles, we can find leave nodes in DAGs. If a same leave node X_i exists in both graphs and $\mathrm{PA}_i^{\mathcal{G}} = \mathrm{PA}_i^{\widetilde{\mathcal{G}}} =: S$, then $\beta_{ki} = \widetilde{\beta}_{ki}$ for all $k \in S$ because $\beta_{ki} = \frac{\partial^2 L(x;\beta)}{\partial x_k \partial x_i} = \frac{\partial^2 L(x;\widetilde{\beta})}{\partial x_k \partial x_i} = \widetilde{\beta}_{ki}, \forall k \in S$ (see proof of Lemma 9 for derivations). So we can safely remove the corresponding $(i+1)$-th rows and i-th columns from both coefficient matrices, and call the remaining coefficient matrices β and $\widetilde{\beta}$, respectively defining DAGs $\mathcal{G}$ and $\widetilde{\mathcal{G}}$, again. If we repeat this process, we will end up with a leave node X_z in $\mathcal{G}$ such that (1) X_z has child in $\widetilde{\mathcal{G}}$ or (2) $\mathrm{PA}_z^{\mathcal{G}} \neq \mathrm{PA}_z^{\widetilde{\mathcal{G}}}$, or otherwise this leave node in $\mathcal{G}$ will be removed.

Case 1. Let us consider the first case that X_z has child node(s) in $\widetilde{\mathcal{G}}$; that is, consider $\mathrm{ch}(z) = \emptyset$ in $\mathcal{G}$, but $\mathrm{ch}(z) \neq \emptyset$ in $\widetilde{\mathcal{G}}$, where $\mathrm{ch}(z)$ is the children set of X_z. Taking the second derivative w.r.t. x_z, we have

$$\frac{\partial^2 L(x;\beta)}{\partial x_z^2} = \frac{c_z''(x_z)}{c(x_z)} + \beta_{zz} - \sum_{k \in \mathrm{ch}(z)} \beta_{zk}^2 b_k''\big((\beta^\top x)_k\big) = \frac{c_z''(x_z)}{c_z(x_z)} - \sum_{k \in \mathrm{ch}(z)} \beta_{zk}^2 b_k''\big((\beta^\top x)_k\big),$$

where the second equation is because $\beta_{zz} = 0$. Since $\text{ch}(z) = \emptyset$ in $\mathcal{G}$, but $\text{ch}(z) \neq \emptyset$ in $\widetilde{\mathcal{G}}$, we have

$$\frac{\partial^2 L(x;\beta)}{\partial x_z^2} - \frac{\partial^2 L(x;\widetilde{\beta})}{\partial x_z^2} = \sum_{k \in \text{ch}(z)} \widetilde{\beta}_{zk}^2 b_k''\big((\widetilde{\beta}^\top x)_k\big) \neq 0,$$

where the inequality is because $b'(\cdot)$ is not constant. However, it contradicts to $L(x;\beta) = L(x;\widetilde{\beta})$. Therefore, X_z has no child node(s) in $\widetilde{\mathcal{G}}$ either.

Case 2. The second case is that $\text{PA}_z^{\mathcal{G}} \neq \text{PA}_z^{\widetilde{\mathcal{G}}}$. Since X_z must be a leave node in both $\mathcal{G}$ and $\widetilde{\mathcal{G}}$ (as case 1 cannot be true), different parent sets of X_z in $\mathcal{G}$ and $\widetilde{\mathcal{G}}$ means that X_z has different skeletons in $\mathcal{G}$ and $\widetilde{\mathcal{G}}$, which contradicts to Lemma 9.

Combining these two cases, we can remove X_z from both DAGs. We keep removing such leave nodes, until β and $\widetilde{\beta}$ are empty, which means that β and $\widetilde{\beta}$ are identical.

□

3.7.1.2 *Proof of Theorem 3*

Proof. To simplify, we write $x_h^\top = x_{h*}^\top$ for the i-th row of the data and $x_{h\ell}$ is equivalent to x_h^ℓ. We also fix π and drop π from $\widehat{\beta}_\pi$, Φ_π, and simplify $b_j(\cdot)$ to $b(\cdot)$ for $j \in [p]$ in the proof. Define

$$\mathcal{B} := \{\beta \in \mathbb{R}^{(p+1)\times p} : \max_{1 \le \ell \le p} \|\beta_\ell - \theta_\ell\|_1 \le r_p\}.$$

Consider the negative log-likelihood based on rows of the data matrix indexed by $\mathcal{I} \subset [n]$,

$$\ell_{\mathcal{I}}(\beta) = \frac{1}{|\mathcal{I}|} \sum_{h \in \mathcal{I}} \sum_{\ell=1}^{p} b(x_h^\top \beta_\ell) - x_{h\ell}\, x_h^\top \beta_\ell.$$

We have, for $\ell = 1, \ldots, p$,

$$[\nabla^2 \ell_{\mathcal{I}}(\beta)]_{(\ell\ell)} = \frac{1}{|\mathcal{I}|} \sum_{h \in \mathcal{I}} b''(x_h^\top \beta_\ell)\, x_h x_h^\top, \tag{3.11}$$

where $[\cdot]_{(\ell\ell)}$ denotes the ℓth $(p+1) \times (p+1)$ block of a matrix. Note that the Hessian of $\ell_{\mathcal{I}}$ is a block-diagonal matrix in $\mathbb{R}^{p(p+1)\times p(p+1)}$ with diagonal blocks given above. Let $\mathcal{I}_0 = [n] = \{1, 2, \ldots, n\}$ and note that $F_k = \ell_{\mathcal{I}_k} + \rho$, $F = \ell_{\mathcal{I}_0} + \rho$ and $h_k = \ell_{\mathcal{I}_k} - \ell_{\mathcal{I}_0}$.

Let $\varphi_h^\ell(t) := b''(t + x_h^\top \theta_\ell) - b''(x_h^\top \theta_\ell)$. For any $v \in \mathbb{R}^{p+1}$, consider the $(p+1) \times (p+1)$ matrix $A_\mathcal{I}^\ell(v)$ with entries:

$$[A_\mathcal{I}^\ell(v)]_{ij} := \frac{1}{|\mathcal{I}|} \sum_{h \in \mathcal{I}} \varphi_h^\ell(x_h^\top v) x_{hi} x_{hj} - \mathbb{E}[\varphi_h^\ell(x_h^\top v) x_{hi} x_{hj}].$$

Let $\widehat{\Sigma}_\mathcal{I} := \frac{1}{|\mathcal{I}|} \sum_{h \in \mathcal{I}} x_h x_h^\top$ and note that

$$\nabla^2 \ell_\mathcal{I}(\beta) - \mathbb{E}\, \nabla^2 \ell_\mathcal{I}(\beta) = \operatorname{diag}\big(A_\mathcal{I}^\ell(\beta_\ell - \theta_\ell),\ \ell = 1, \ldots, p\big) + \nabla^2 \ell_\mathcal{I}(\theta) - \mathbb{E}\, \nabla^2 \ell_\mathcal{I}(\theta). \tag{3.12}$$

The following lemma provides uniform control on the Frobenius norm of $A_\mathcal{I}^\ell(v)$ as v varies in the ℓ_1 ball of radius r_p.

Lemma 10. *For $\mathcal{I}$ with $|\mathcal{I}| \geq m$, we have with probability at least $1 - e^{-t^2}$,*

$$\sup_{\|v\|_1 \leq r_p} \Big(\sum_{1 \leq i,j \leq p} [A_\mathcal{I}^\ell(v)]_{ij}^2 \Big)^{1/2} \leq CT \sqrt{\frac{p^2 \log p}{m}} \big(\kappa + \sqrt{\kappa}(t + \sqrt{\log p})\big)$$

where $\kappa = \overline{b}_p T^2 r_p$.

Let $g(t) = b''(t) - b''(0)$ and consider the $(p+1) \times (p+1)$ matrix $B_\mathcal{I}^\ell$ with entries:

$$[B_\mathcal{I}^\ell]_{ij} := \frac{1}{|\mathcal{I}|} \sum_{h \in \mathcal{I}} g(x_h^\top \theta_\ell) x_{hi} x_{hj} - \mathbb{E}[g(x_h^\top \theta_\ell) x_{hi} x_{hj}].$$

We note that

$$\nabla^2 \ell_\mathcal{I}(\theta) - \mathbb{E}\, \nabla^2 \ell_\mathcal{I}(\theta) = \operatorname{diag}\big(B_\mathcal{I}^\ell, \ell = 1, \ldots, p\big) + b''(0)\big(\widehat{\Sigma}_\mathcal{I} - \mathbb{E}[\widehat{\Sigma}_\mathcal{I}]\big). \tag{3.13}$$

Let us write $R_1^* = \max_\ell \|\theta_\ell\|_1$. The next lemma provides control on the Frobenius norm of $B_\mathcal{I}^\ell$.

Lemma 11. *For $\mathcal{I}$ with $|\mathcal{I}| \geq m$, we have with probability at least $1 - e^{-t^2}$,*

$$\|B_\mathcal{I}^\ell\|_F \leq CT \sqrt{\frac{p^2 \log p}{m}} \big(\kappa_* + \sqrt{\kappa_*}(t + \sqrt{\log p})\big)$$

where $\kappa_* = \overline{b}_p T^2 R_1^*$.

The proof of Lemma 11 is identical to that of Lemma 10 and is omitted. Next, we control the deviation of $\widehat{\Sigma}_\mathcal{I}$ from its mean.

Lemma 12. *Assume that $p \geq 2$. For $\mathcal{I}$ with $|\mathcal{I}| \geq m$, we have with probability at least $1 - e^{-t^2}$,*

$$\|\widehat{\Sigma}_{\mathcal{I}} - \mathbb{E}[\widehat{\Sigma}_{\mathcal{I}}]\|_F \leq CT^2 \frac{p(t + \sqrt{\log p})}{\sqrt{m}}.$$

Recall that, by assumption, $|\mathcal{I}_k| \geq m$ for all $k = 1, \dots, K$. Then, $K \leq n/m$. We also have $|\mathcal{I}_0| = n \geq m$. Recall that we assume $K + 1 \leq np$. Let us take $t = \sqrt{(2 + c_1)\log(np)}$ in Lemma 10. Then, on an event $\mathcal{G}_1$ with $\mathbb{P}(\mathcal{G}_1) \geq 1 - p(K+1)(np)^{-(2+c_1)} \geq 1 - (np)^{-c_1}$, we have

$$\max_{0 \leq k \leq K} \max_{1 \leq \ell \leq p} \sup_{\|v\|_1 \leq r_p} \|A^{\ell}_{\mathcal{I}_k}(v)\|_F \leq CT\kappa \sqrt{\frac{p^2 \log p}{m}} \Big(1 + \sqrt{\frac{\log(np)}{\kappa}}\Big) =: \alpha_n$$

where $\kappa = \bar{b}_p T^2 r_p$ and C is a different constant from the one in Lemma 12. Letting $\psi(x) = x \vee \sqrt{x}$, we can further bound α_n as

$$\alpha_n \leq CT \sqrt{\frac{p^2 \log p}{m}} \psi(\kappa) 2\sqrt{\log(np)} = 2CT^3 \sqrt{p}\, \psi(\bar{b}_p r) \frac{p \log(np)}{\sqrt{m}}$$

assuming $np \geq 3$. Here, we are using $\psi(\kappa) \leq T^2 \sqrt{p}\, \psi(\bar{b}_p r)$ since $r_p = r\sqrt{p}$ and $T, p \geq 1$. By an almost identical argument, using Lemma 11, on an event $\mathcal{G}_2$ with $\mathbb{P}(\mathcal{G}_2) \geq 1 - (np)^{-c_1}$,

$$\max_{0 \leq k \leq K} \max_{1 \leq \ell \leq p} \|B^{\ell}_{\mathcal{I}_k}\|_F \leq \alpha^*_n$$

where $\alpha^*_n \leq 2CT^3 \psi(\bar{b}_p R^*_1) \frac{p \log(np)}{\sqrt{m}}$.

Similarly, taking $t = \sqrt{(1 + c_1)\log(np)}$ in Lemma 12, on an event $\mathcal{G}_3$, with $\mathbb{P}(\mathcal{G}_3) \geq 1 - (np)^{-c_1}$, we have

$$\max_{0 \leq k \leq K} \|\widehat{\Sigma}_{\mathcal{I}_k} - \mathbb{E}[\widehat{\Sigma}_{\mathcal{I}_k}]\|_F \leq CT^2 \sqrt{\frac{p^2 \log(np)}{m}} := \gamma_n,$$

for some other constant C. Since $\mathbb{E}[\widehat{\Sigma}_{\mathcal{I}_k}] = \Sigma := \mathbb{E}[xx^\top]$ for all k, on event $\mathcal{G}_3$ we have $\|\widehat{\Sigma}_{\mathcal{I}_k} - \Sigma\|_o p \leq \gamma_n$ for all $k = 0, 1, \dots, K$. For the rest of the proof, we work on the good event $\mathcal{G} = \mathcal{G}_1 \cap \mathcal{G}_2 \cap \mathcal{G}_3$.

Recall that for any $\beta \in \mathcal{B}$, we have $\|\beta_\ell - \theta_\ell\|_1 \le r_p$. Using (3.12), we obtain

$$\begin{aligned}
&\sup_{\beta \in \mathcal{B}} \|\nabla^2 \ell_{\mathcal{I}_k}(\beta) - \mathbb{E}[\nabla^2 \ell_{\mathcal{I}_k}(\beta)]\|_op \\
&\quad \le \sup_{\beta\in\mathcal{B}} \max_\ell \|A^\ell_{\mathcal{I}_k}(\beta_\ell - \theta_\ell)\|_op + \max_\ell \|B^\ell_{\mathcal{I}_k}\|_op + |b''(0)| \|\widehat{\Sigma}_{\mathcal{I}_k} - \mathbb{E}[\widehat{\Sigma}_{\mathcal{I}_k}]\|_op \\
&\quad \le \max_{\|v\|_1 \le r_p} \max_\ell \|A^\ell_{\mathcal{I}_k}(v)\|_op + \alpha_n^* + |b''(0)|\gamma_n \\
&\quad \le \alpha_n + \alpha_n^* + b''(0)\gamma_n
\end{aligned}$$

for all $k = 0, \dots, K$, on $\mathcal{G}$. The last line above follows from $\|\cdot\|_op \le \|\!|\cdot|\!\|_F$ and the fact that $b''(0) \ge 0$ by convexity. Next, we note that $\mathbb{E}[\nabla^2 \ell_{\mathcal{I}_k}(\beta)] =: M_\beta$ is the same for all $k = 0, \dots, K$. Then, on $\mathcal{G}$, we have

$$\begin{aligned}
\|\nabla^2 h_k(\beta)\|_op &= \|\nabla^2 \ell_{\mathcal{I}_k}(\beta) - \nabla^2 \ell_{\mathcal{I}_0}(\beta)\|_op \\
&\le \|\nabla^2 \ell_{\mathcal{I}_k}(\beta) - M_\beta\|_op + \|\nabla^2 \ell_{\mathcal{I}_0}(\beta) - M_\beta\|_op \\
&= \|\nabla^2 \ell_{\mathcal{I}_k}(\beta) - \mathbb{E}[\nabla^2 \ell_{\mathcal{I}_k}(\beta)]\|_op + \|\nabla^2 \ell_{\mathcal{I}_0}(\beta) - \mathbb{E}[\nabla^2 \ell_{\mathcal{I}_0}(\beta)]\|_op \\
&\le 2\alpha_n + 2\alpha_n^* + 2b''(0)\gamma_n
\end{aligned}$$

for all $\beta \in \mathcal{B}$ and $k = 1, \dots, K$.

Recalling that $\underline{b}_p = \inf_{|t| \le T(r_p + R_1^*)} b''(t) > 0$, by the convexity of $b(\cdot)$, $\underline{b}_p \ge 0$. We further assume that $\underline{b}_p > 0$. For any $\beta \in \mathcal{B}$, we have $|x_h^\top \beta_\ell| \le T\|\beta_\ell\|_1 \le T(r_p + R_1^*)$ by Hölder inequality, followed by the triangle inequality. It follows from (3.11) that, for all $\beta \in \mathcal{B}$ and k,

$$[\nabla^2 \ell_{\mathcal{I}_k}(\beta)]_{(jj)} \succeq \underline{b}_p \widehat{\Sigma}_{\mathcal{I}_k} \succeq \underline{b}_p(\lambda_{\min}(\Sigma) - \gamma_n) I_{p+1} \succeq \frac{1}{2} \underline{b}_p \lambda_{\min}(\Sigma) I_{p+1}$$

where the second inequality is by Weyl's theorem and the last by assumption $\gamma_n \le \lambda_{\min}(\Sigma)/2$. It follows that for any $\beta \in \mathcal{B}$, we have $\nabla^2 \ell_{\mathcal{I}_k}(\beta) \succeq \frac{1}{2}\underline{b}_p \lambda_{\min}(\Sigma) I_{p(p+1)}$.

We are now ready to apply convergence Theorem 6 (see Section 3.7.3). we need to establish properties of the loss functions over a (vector) ℓ_2 balls, or equivalently matrix Frobenius-norm balls. Consider the event $\mathcal{G}_0 = \{\|\!|\widehat{\beta} - \theta|\!\|_F \le r\}$ and let us work on $\mathcal{G} \cap \mathcal{G}_0$ for the rest of the proof. For any $\beta = (\beta_\ell)$ for which $\|\!|\beta - \widehat{\beta}|\!\|_F \le r$, we have $\|\!|\beta - \theta|\!\|_F \le 2r$, by the triangle inequality. Then, $\max_\ell \|\beta_\ell - \widehat{\beta}_\ell\|_2 \le 2r$ hence, $\max_\ell \|\beta_\ell - \theta_\ell\|_1 \le r_p = 2r\sqrt{p}$.

We conclude that $\mathbb{B}_F(\widehat{\beta}; r) \subset \mathcal{B}$, where $\mathbb{B}_F(\widehat{\beta}; r)$ denotes the Frobenius-norm ball of radius r centered at $\widehat{\beta}$.

It follows that on the good event $\mathcal{G} \cap \mathcal{G}_0$ and over $\mathbb{B}_F(\widehat{\beta}; r)$, h_k is δ-smooth with $\delta = 2\alpha_n + 2\alpha_n^* + 2b''(0)\gamma_n$ and $\ell_{\mathcal{I}_k}$ is ρ-strongly convex with $\rho = \frac{1}{2}\underline{b}_p \lambda_{\min}(\Sigma)$. We have

$$\frac{\delta}{\rho} \leq \frac{4}{\underline{b}_p \lambda_{\min}(\Sigma)}(\alpha_n + \alpha_n^* + b''(0)\gamma_n)$$

where $\alpha_n + \alpha_n^* + b''(0)\gamma_n \lesssim T^3\big[\psi(\bar{b}_p r) + \frac{\psi(\bar{b}_p R_1^*)}{\sqrt{p}} + b''(0)\big]p^{3/2}\log(np)/\sqrt{m}$. The bound can be simplified using $\psi(x) + \psi(y) \lesssim \psi(x+y)$ for $x, y \geq 0$. The proof is complete. $\square$

3.7.1.3 *Proof of Theorem 4*

Definition 4. *(Faithfulness.) A joint distribution $L(X)$ is faithful to a DAG $\mathcal{G}$ if the conditional independence statements in L have a one-to-one correspondence to the d-separations in $\mathcal{G}$.*

Proof. Let $\mathcal{S}(\beta)$ be the set of all topological sorts implied by β. We observe that (Gu et al, 2019):

Lemma 13. *For any $\theta \in \Omega$, there exists a $\delta(\theta)$ such that if $\beta \in \Omega$ and $\|\beta - \theta\|_F < \delta(\theta)$, then $\mathcal{S}(\beta) \bigcap \mathcal{S}(\theta) \neq \emptyset$.*

We say that β is in the neighborhood of θ, denoted by $\mathrm{nb}(\theta)$, if $\|\beta - \theta\|_F < \delta(\theta)$. Hence θ and parameters in $\{\beta : \beta \in \mathrm{nb}(\theta)\}$ share at least one topological sort by Lemma 13.

Case 1. Let us consider $\beta \in \mathrm{nb}(\beta_\pi^*)$ and $\beta \neq \beta_\pi^*$. Consider a permutation $\pi \in \mathcal{S}(\beta) \bigcap \mathcal{S}(\beta_\pi^*)$, which is compatible with β and β_π^*. If $p(x \mid \beta) = p(x \mid \beta_\pi^*)$ for DAG $\mathcal{G}$, then β and β_π^* decompose the joint distribution of $\mathcal{G}$ according to a common permutation π and have the same joint distribution. In this case, β and β_π^* must be identical since $\mathcal{G}$ is identifiable, which contradicts $\beta \neq \beta_\pi^*$. Thus we have $p(x|\beta) \neq p(x|\beta_\pi^*)$ for some x and $\beta \in \mathrm{nb}(\beta_\pi^*)$. Therefore, we have $\mathbb{E}_{\beta_\pi^*}[\log(p(x|\beta_\pi^*)] > \mathbb{E}_{\beta_\pi^*}[\log(p(x|\beta)]$, which indicates that the Fisher information matrix is positive definite, i.e.,

$$I(\beta_\pi^*) = \mathbb{E}\left[\frac{\partial}{\partial \beta_\pi^*}\log p(x \mid \beta_\pi^*)\frac{\partial}{\partial \beta_\pi^*}\log p(x \mid \beta_\pi^*)^\top\right] \succ 0,$$

where $I_{ik}(\beta) = \mathbb{E}_\beta \left[-\frac{\partial^2}{\partial \beta_i \partial \beta_k} \log(p(x \mid \beta) \right]$ with θ reshaped as a vector.

Letting $L(\beta) := -n\ell(\beta)$ be an unnormalized log-likelihood and $\rho_{\lambda_n}(\beta) = \lambda_n \sum_{ij} \|\beta_{ij}\|_F$ be a group Lasso regularizer, the problem (3.5) is equivalent to

$$\max_{\beta \in \Omega} -nF(\beta) = \max_{\beta \in \Omega} L(\beta) - n\rho_{\lambda_n}(\beta).$$

Let $a_n = n^{-1/2}$ and we consider $u \in \{u : \beta^*_\pi + a_n u \in \mathcal{D}(\pi)\}$. Without loss of generality, we assume $\|u\|_F \leq C$ for a fixed positive constant C. Define $\mathcal{A} = \{(j,i) : (\beta^*_\pi)_{ji} \neq 0\}$, we have

$$\begin{aligned} D(u) :=& L(\beta^*_\pi + a_n u) - n\rho_{\lambda_n}(\beta^*_\pi + a_n u) - L(\beta^*_\pi) + n\rho_{\lambda_n}(\beta^*_\pi) \\ \leq& L(\beta^*_\pi + a_n u) - L(\beta^*_\pi) - n\lambda_n \sum_{(j,i)\in\mathcal{A}} \left(\|(\beta^*_\pi)_{ji} + a_n u_{ji}\|_F - \|(\beta^*_\pi)_{ji}\|_F \right), \\ \leq& L(\beta^*_\pi + a_n u) - L(\beta^*_\pi) + n\lambda_n a_n \sum_{(j,i)\in\mathcal{A}} \|u_{ji}\|_F, \\ =& a_n \nabla L^\top(\beta^*_\pi) u - \frac{n}{2} a_n^2 u^\top I(\beta^*_\pi) u (1 + o_p(1)) + n\lambda_n a_n \sum_{(j,i)\in\mathcal{A}} \|u_j\|_F, \end{aligned} \tag{3.14}$$

where the second inequality is by the triangle inequality and the last equation is by the Taylor expansion and assumption (A2). We multiply $(\sqrt{n}a_n)^{-1}$ to both sides of (3.14), then we have

$$\frac{D(u)}{\sqrt{n}a_n} \leq \frac{1}{\sqrt{n}} \nabla L^\top(\beta^*_\pi) u - \frac{1}{2}\sqrt{n} a_n u^\top I(\beta^*_\pi) u (1 + o_p(1)) + \lambda_n \sqrt{n} \sum_{(j,i)\in\mathcal{A}} \|u_{ji}\|_F,$$

where β^*_π and u are regarded as vectors. Since $\frac{\|\nabla L(\beta^*_\pi)\|_F}{\sqrt{n}} = \mathcal{O}_p(1)$ by the central limit theorem, the second term dominates the first term uniformly in $\|u\|_F = C$ by choosing a significantly large C. The third term is also dominated by the first term uniformly because $\lambda_n\sqrt{n} = o(1)$ by assumption. Hence, for any $\varepsilon > 0$, there exist some constant C such that

$$\mathbb{P}\left(\inf_{\|u\|_F = C} F(\beta^*_\pi + a_n u) > F(\beta^*_\pi) \right) \geq 1 - \varepsilon, \tag{3.15}$$

where $a_n = n^{-1/2}$. Note that the arguments in this case apply to subspace $\mathcal{D}(\pi) \subset \Omega$. Hence, denoting by $\widehat{\beta}_\pi$ a global minimizer of $F(\beta)$ over $\mathcal{D}(\pi)$ and β^*_π the minimizer of the population loss $\mathbb{E}(\ell_{[n]}(\beta))$ over $\mathcal{D}(\pi)$, we have $\|\widehat{\beta}_\pi - \beta^*_\pi\|_F = \mathcal{O}(n^{-1/2})$ for all π.

Case 2. Over the DAG space Ω, consider $\beta \notin \text{nb}(\beta^*)$. By pointwise convergence in probability of strictly convex function $F(\beta)$ over subspace $\mathcal{D}(\pi) \subset \Omega$, we have $\widehat{\beta}_\pi \to_p \beta^\pi$ where β^π

is a unique minimizer of the limiting function of $F(\beta)$ over $\mathcal{D}(\pi)$ (Andersen and Gill, 1982, Appendix Corollary II.2.). Denote by $\mathcal{M} = \{\beta^\pi : \beta^\pi \neq \beta^*, \beta^\pi = \arg\min_{\beta \in \mathcal{D}(\pi)} \mathbb{E}[F(\beta)]\}$ which is a finite set. Consider $\beta^i \in \mathcal{M}$, and we have

$$\begin{aligned} F(\beta^*) - F(\beta^i) &\leq \frac{1}{n}\left(L(\beta^i) - L(\beta^*) - n\lambda_n \sum_{(j,i)\in\mathcal{A}} \left(\|\beta^i_{ji}\|_F - \|\beta^*_{ji}\|_F \right) \right), \\ &\to_p \mathbb{E}_{\beta^*}\left[\log \frac{p(x \mid \beta^i)}{p(x \mid \beta^*)} \right] - \lambda_n \mathcal{O}(1), \end{aligned}$$

where the first term is negative and the second term is $o(n^{-1/2})$ since $\lambda_n\sqrt{n} = o(1)$. Therefore, for any $\varepsilon > 0$, we have

$$\mathbb{P}\left(\inf_{\mathcal{M}} F(\beta^i) > F(\beta^*) \right) \geq 1 - \varepsilon, \tag{3.16}$$

for a sufficiently large n.

Combining (3.15) and (3.16), we have shown that there is a unique global maximizer $\widehat{\beta} = \arg\min_{\beta\in\Omega} F(\beta)$, such that $\|\widehat{\beta} - \beta^*\|_F = \mathcal{O}_p(n^{-1/2})$. □

3.7.2 Proofs of technical lemmas

3.7.2.1 *Proof of Lemma 9*

Proof. Taking the derivative of (3.3) with respect to x_i, the only terms that contribute are $k = i$ and $k \in \mathrm{ch}(i)$, where $\mathrm{ch}(i)$ is the set of indices of the children of X_i. That is,

$$\frac{\partial L(x;\beta)}{\partial x_i} = \frac{c_i'(x_i)}{c_i(x_i)} + (\beta^\top x)_i + \sum_{k\in\mathrm{ch}(i)} \left[x_k - b_k'\big((\beta^\top x)_k\big) \right] \beta_{ik},$$

where $b_k'(\cdot)$ is the first derivative of $b_k(\cdot)$. Taking the derivative w.r.t. x_j for $j \neq i$. If $j \in \mathrm{ch}(i)$, one of the terms in the sum contributes, otherwise, none will contribute. Therefore, we get

$$\frac{\partial^2 L(x;\beta)}{\partial x_j \partial x_i} = \beta_{ji} + 1\{j \in \mathrm{ch}(i)\} \left[\beta_{ij} - b_j''\big((\beta^\top x)_j\big)\beta_{jj} \right] = \beta_{ji} + \beta_{ij},$$

where $b_j''(\cdot)$ is the second derivative of $b_j(\cdot)$ and the second equation is because $B_{ij} = 0$ if $j \notin \mathrm{ch}(i)$ and $B_{jj} = 0$. Thus, this mixed derivative is equal to

$$\frac{\partial^2 L(x;\beta)}{\partial x_j \partial x_i} = \begin{cases} \beta_{ij} \neq 0, & \text{if } j \in \mathrm{ch}(i); \\ \beta_{ji} \neq 0, & \text{if } i \in \mathrm{ch}(j); \\ 0, & \text{otherwise,} \end{cases}$$

Suppose there is another GLDAG structural equation models $\widetilde{\beta}$ generating the same distribution as β, and let $\mathcal{G}$ and $\widetilde{\mathcal{G}}$ be the respective DAGs implied by β and $\widetilde{\beta}$. Since $\frac{\partial^2 L(x;\beta)}{\partial x_j \partial x_i} = \frac{\partial^2 L(x;\widetilde{\beta})}{\partial x_j \partial x_i}$, if $\beta_{ij} \neq 0$, then we have have $\widetilde{\beta}_{ij} = \beta_{ij}$ or $\widetilde{\beta}_{ij} = \beta_{ji}$. It means that all non-adjacent nodes in $\widetilde{\mathcal{G}}$ remains non-adjacent in $\mathcal{G}$, i.e., $\widetilde{\mathcal{G}}$ has the same skeleton as $\mathcal{G}$. □

3.7.2.2 *Proof of Lemma 10*

Proof. We will need the following contraction principle:

Theorem 5 (Ledoux–Talagrand; cf. Theorem 3.2.1 of Giné and Nickl (2021))**.** *Let $\{\varepsilon_h\}_{h=1}^n$ be a sequence of independent Radecmacher (i.e., symmetric Bernoulli) variables, and let $\phi_h : \mathbb{R} \to \mathbb{R}$ be contraction (i.e., 1-Lipschitz) mappings that vanish at zero ($\phi_h(0) = 0$). Let $F : [0, \infty) \to \mathbb{R}$ be a nonnegative, nondecreasing convex function. Then,*

$$\mathbb{E}\, F\Big(\frac{1}{2} \sup_{t \in T} \Big| \sum_{h=1}^n \varepsilon_h \phi_h(t_h) \Big|\Big) \leq \mathbb{E}\, F\Big(\Big| \sup_{t \in T} \sum_{h=1}^n \varepsilon_h t_h \Big|\Big)$$

where $t = (t_1, \ldots, t_n)$ and T is any bounded subset of $\mathbb{R}^n$.

Recall that $b''(\cdot)$ is assumed to be $\bar{b}_p$-Lipschitz on $[-Tr_p, Tr_p]$. Then, the same holds for φ. It follows that, $\frac{1}{\bar{b}_p T^2}\varphi(\cdot) x_{hi} x_{hj}$ is 1-Lipschitiz on the same interval, conditioned on $\mathcal{T}$. To simplify the notation, let us fix $\mathcal{I}$ and ℓ and drop the dependence on $\mathcal{I}$ and ℓ and write $A_{ij}(v) = [A^{\ell}_{\mathcal{I}}(v)]_{ij}$.

Let $\{\varepsilon_h\}_{h \in \mathcal{I}}$ be an i.i.d. sequence of Rademacher variables, independent of $\{x_h\}_{h \in \mathcal{I}}$.

Then,

$$\begin{aligned}
\mathbb{E}\exp\Big(\lambda \sup_{\|v\|_1\le r_p}|A_{ij}(v)|\Big) &\le \mathbb{E}\exp\Big(2\lambda \sup_{\|v\|_1\le r_p}\Big|\frac{1}{|\mathcal{I}|}\sum_{h\in\mathcal{I}}\varepsilon_h\varphi_h^\ell(x_h^\top v)x_{hi}x_{hj}\Big|\Big)\\
&\le \mathbb{E}\exp\Big(4\bar{b}_pT^2\lambda \sup_{\|v\|_1\le r_p}\Big|\frac{1}{|\mathcal{I}|}\sum_{h\in\mathcal{I}}\varepsilon_h x_h^\top v\Big|\Big)\\
&\le \mathbb{E}\exp\Big(4\bar{b}_pT^2 r_p\lambda\Big\|\frac{1}{|\mathcal{I}|}\sum_{h\in\mathcal{I}}\varepsilon_h x_h\Big\|_\infty\Big)\\
&= \mathbb{E}\, e^{\kappa\lambda\|\xi\|_\infty}
\end{aligned}$$

where $\xi := \frac{1}{|\mathcal{I}|}\sum_{h\in\mathcal{I}}\varepsilon_h x_h$ and $\kappa := 4\bar{b}_pT^2r_p$. The first inequality above is a symmetrization inequality and the second inequality is by Ledoux–Talagrand contraction (Theorem 5), applied with $F(x) = e^{4\bar{b}_pT^2\lambda x/|\mathcal{I}|}$ and $\phi_h(\cdot) = \frac{1}{\bar{b}_pT^2}\varphi_h^\ell(\cdot)x_{hi}x_{hj}$ which are 1-Lipschitz and vanish at zero.

We recall Lemma 2.3.3 of Giné and Nickl (2021): Let $\xi = (\xi_1, \dots, \xi_p)$ be vector with each coordinate a sub-Gaussian variable (not necessarily independent) and $2 \le p < \infty$. Then,

$$\|\|\xi\|_\infty\|_{\psi_2} \le 4\sqrt{\log p}\cdot \max_{i\le p}\|\xi_i\|_{\psi_2}.$$

where $\|\cdot\|_{\psi_2}$ denotes the sub-Gaussian norm.

Let ξ be as defined earlier, with coordinates $\xi_i = \frac{1}{|\mathcal{I}|}\sum_{h\in\mathcal{I}}\varepsilon_h x_{hi} \in \mathbb{R}$. Each ξ_i is zero-mean and sub-Gaussian, and

$$\|\xi_i\|_{\psi_2}^2 \lesssim \frac{1}{|\mathcal{I}|^2}\sum_{h\in\mathcal{I}}\|\varepsilon_h x_{hi}\|_{\psi_2}^2 \lesssim \frac{T^2}{|\mathcal{I}|}$$

using that for a random variable Y in $[a, b]$, we have $\|Y\|_{\psi_2}^2 \lesssim (b-a)^2$. Recalling the assumption $|\mathcal{I}| \ge m$, we obtain $\|\xi_i\|_{\psi_2} \lesssim T/\sqrt{m}$, hence

$$\|\|\xi\|_\infty\|_{\psi_2} \le a := c_0T\sqrt{\log p/m},$$

for a numerical constant $c_0 > 0$. This gives $\mathbb{E}\left[e^{\lambda(\|\xi\|_\infty - \mathbb{E}\|\xi\|_\infty)}\right] \le e^{c_1a^2\lambda^2}$ for some numerical constant $c_1 > 0$.

Recall that $\mathbb{E}\|\xi\|_\infty \lesssim \|\|\xi\|_\infty\|_{\psi_2}$. That is, letting $M := \mathbb{E}\|\xi\|_\infty$, we have $M \le c_2a$ for some numerical constant $c_2 > 0$ and

$$\begin{aligned}\mathbb{P}\Big(\sup_{\|v\|_1 \le r_p} |A_{ij}(v)| \ge (\kappa + t)c_2 a\Big) &\le \mathbb{P}\Big(\sup_{\|v\|_1 \le r_p} |A_{ij}(v)| \ge \kappa M + t c_2 a\Big) \\ &\le \mathbb{E}\exp\Big(\lambda\big[\sup_{\|v\|_1 \le r_p} |A_{ij}(v)| - (\kappa M + c_2 a t)\big]\Big) \\ &\le e^{-\lambda c_2 a t}\, \mathbb{E}\, e^{\kappa\lambda(\|\xi\|_\infty - \mathbb{E}\,\|\xi\|_\infty)} \le e^{-\lambda c_2 a t + c_1 \kappa a^2 \lambda^2}\end{aligned}$$

where the second line is by Markov inequality (i.e., Chernoff bounding). Minimizing the RHS over $\lambda > 0$, we get

$$\mathbb{P}\Big(\sup_{\|v\|_1 \le r_p} |A_{ij}(v)| \ge c_2(\kappa + t)a\Big) \le \exp\Big(-\frac{c_2^2\, t^2}{2c_1\kappa}\Big).$$

Changing t to $\sqrt{2c_1\kappa}t/c_2$ and using the union bound

$$\mathbb{P}\Big(\max_{1 \le i,j \le p}\sup_{\|v\|_1 \le r_p} |A_{ij}(v)| \ge c_3(\kappa + \sqrt{\kappa}t)a\Big) \le p^2 \exp(-t^2).$$

Next, change t to $t + \sqrt{2\log p}$ to obtain

$$\mathbb{P}\Big(\max_{1 \le i,j \le p}\sup_{\|v\|_1 \le r_p} |A_{ij}(v)| \ge c_4(\kappa + \sqrt{\kappa}t + \sqrt{\kappa \log p})a\Big) \le \exp(-t^2).$$

The result follows by noting that

$$\sup_{\|v\|_1 \le r_p}\Big(\sum_{1 \le i,j \le p} A_{ij}^2(v)\Big)^{1/2} \le p \max_{1 \le i,j \le p}\sup_{\|v\|_1 \le r_p} |A_{ij}(v)|.$$

□

3.7.2.3 *Proof of Lemma 12*

Proof. Let $Z = (Z_{ij}) = \widehat{\Sigma}_{\mathcal{I}} - \mathbb{E}[\widehat{\Sigma}_{\mathcal{I}}]$ and note that $Z_{ij} = \frac{1}{|\mathcal{I}|}\sum_{h\in\mathcal{I}}(x_{hi}x_{hj} - \mathbb{E}[x_{hi}x_{hj}])$. Let $\{\varepsilon_h\}_{h\in\mathcal{I}}$ be an i.i.d. sequence of Rademacher variables, independent of $\{x_h\}_{h\in\mathcal{I}}$. We have

$$\mathbb{E}\exp\big(\lambda|Z_{ij}|\big) \le \mathbb{E}\exp\big(2\lambda|\xi|\big), \quad \xi := \frac{1}{|\mathcal{I}|}\sum_{h\in\mathcal{I}} \varepsilon_h x_{hi} x_{hj},$$

by a symmetrization argument. We also have $a := \|\xi\|_{\psi_2} \lesssim \frac{T^2}{\sqrt{m}}$. As in the proof of Lemma 10, let $M := \mathbb{E}\,|\xi|$, and note that $M \le c_2 a$ for some numerical constant $c_2 > 0$. Then,

$$\begin{aligned}\mathbb{P}\Big(|Z_{ij}| \ge (2+t)c_2 a\Big) &\le \mathbb{P}\Big(|Z_{ij}| \ge 2M + tc_2 a\Big) \\ &\le \mathbb{E}\exp\Big(\lambda\big[|Z_{ij}| - (2M + c_2 at)\big]\Big) \\ &\le e^{-\lambda c_2 at}\,\mathbb{E}\,e^{2\lambda(|\xi| - \mathbb{E}\,|\xi|)} \le e^{-\lambda c_2 at + c_1 2a^2\lambda^2}.\end{aligned}$$

Minimizing the RHS over $\lambda > 0$ and proceeding with the same argument as in the proof of Lemma 10 (with κ set to 2), we obtain

$$\mathbb{P}\Big(\max_{1\le i,j\le p} |Z_{ij}| \ge c_4(2 + \sqrt{2}t + \sqrt{2\log p})a\Big) \le \exp(-t^2).$$

Combined with $\|Z\|_F \le p \cdot \max_{1\le i,j\le p}|Z_{ij}|$, and using $2 \le 2\sqrt{2\log p}$ for $p \ge 2$, the result follows. □

3.7.2.4 *Proof of Lemma 13*

Proof. If DAG $\mathcal{G}$ is an empty graph, then the statement hold trivially. Otherwise let $\delta(\theta) = 1/2\min_{\theta_{ji}\neq 0}\|\theta_{ji}\|_F$. For $\theta_{ji} \neq 0$, if $\beta \in \Omega$ and $\|\beta - \theta\|_F < \delta(\theta)$, then we have $\beta_{ji} \neq 0$ as well, since otherwise $\|\beta - \theta\|_F \ge \|\theta_{ji}\|_F \ge \delta(\theta)$. It indicates that for any existing edge in θ, there must be an edge in β. Thus the graphs represented by β and θ share at least one topological sort. □

3.7.3 Convergence of the DANE algorithm

The convergence of the DANE algorithm for general distributed supervised learning has been studied in (Zhang et al, 2013; Shamir et al, 2014; Jordan et al, 2018; Fan et al, 2019). To make the thesis self-contained, we state and prove a concrete convergence result in Theorem 6 below, based on the ideas in (Fan et al, 2019, Theorem 2.1), simplifying the statement and including a missing assumption needed in their argument (Assumption (C1)). The proof of Theorem 3 (Section 3.7.1.2) verifies that the assumptions in Theorem 6 are satisfied by our GLDAG models in a distributed setting, and thus, our algorithm (Algorithm 6) achieves the desired geometric rate of convergence.

Theorem 6. *Assume that F and F_k are convex, with F_k ρ-strongly convex on $B(\widehat{\beta}_\pi, r)$ where $\widehat{\beta}_\pi$ is a global minimizer of F over $\mathcal{D}(\pi)$ and $r > 0$. Recall $h_k = F_k - F$. Assume further that*

(C1) Local update (3.6) *has a unique solution for $\beta \in B(\widehat{\beta}_\pi, r)$,*

(C2) h_k is δ-smooth on $B(\widehat{\beta}_\pi, r)$.

and $\delta/\rho < 1$. Then, $\widehat{\beta}_\pi$ is a fixed point of $\varphi_{k,\pi}$ and

$$\|\varphi_{k,\pi}(\beta) - \widehat{\beta}_\pi\| \leq \frac{\delta}{\rho}\|\beta - \widehat{\beta}_\pi\|, \quad \forall \beta \in B(\widehat{\beta}_\pi, r).$$

Condition (C2) by definition means that ∇h_k is δ-Lipschitz on $B(\widehat{\beta}_\pi, r)$. It is guaranteed if

$(C2')$ h_k is almost everywhere second-order differentiable with $\|\nabla^2 h_k(\beta)\|_o p \leq \delta$ for $\beta \in B(\widehat{\beta}_\pi, r)$,

where $\|\cdot\|_o p$ is an operator norm.

Proof. **$\widehat{\beta}_\pi$ is a fixed point of $\varphi_{k,\pi}$.** For any β, by first-order optimality criterion (Fermat's rule), (3.6) is equivalent to the sub-differential of the objective function at $\varphi_{k,\pi}(\beta)$ containing zero, that is,

$$0 \in \partial F_k(\varphi_{k,\pi}(\beta)) - \nabla h_k(\beta) \iff \nabla h_k(\beta) \in \partial F_k(\varphi_{k,\pi}(\beta)). \tag{3.17}$$

By uniqueness of the solution, if $\nabla h_k(\beta) \in \partial F_k(u)$, then we should have $\varphi_{k,\pi}(\beta) = u$. Therefore, it is enough to verify that $\nabla h_k(\widehat{\beta}_\pi) \in \partial F_k(\widehat{\beta}_\pi)$. From the fact that $\widehat{\beta}_\pi$ minimizes F over $\mathcal{D}(\pi)$, we have $0 \in \partial F(\widehat{\beta}_\pi)$. Since $F = F_k - h_k$ and h_k is differentiable, $\partial F = \partial F_k - \nabla h_k$ and the result follows.

$\varphi_{k,\pi}$ is locally a contraction. Let $C = B(\widehat{\beta}_\pi, r)$ and pick any $\beta \in C$. From the optimality condition (3.17), we have $\nabla h_k(\beta) \in \partial F_k(\varphi_{k,\pi}(\beta))$ and $\nabla h_k(\widehat{\beta}_\pi) \in \partial F_k(\varphi_{k,\pi}(\widehat{\beta}_\pi)) = \partial F_k(\widehat{\beta}_\pi)$. It follows from Lemma 14 below that

$$\|P_C(\varphi_{k,\pi}(\beta)) - \widehat{\beta}_\pi\| \leq \rho^{-1}\|\nabla h_k(\beta) - \nabla h_k(\widehat{\beta}_\pi)\| \leq \frac{\delta}{\rho}\|\beta - \widehat{\beta}_\pi\|.$$

Since $\|\beta - \widehat{\beta}_\pi\| \leq r$ and $\delta/\rho < 1$, we have $\|P_C(\varphi_{k,\pi}(\beta)) - \widehat{\beta}_\pi\| < r$ which implies $\varphi_{k,\pi}(\beta) \in C$, hence $P_C(\varphi_{k,\pi}(\beta)) = \varphi_{k,\pi}(\beta)$ and the proof is complete. □

3.7.3.1 *Local strong monotonicity*

We recall that for a proper closed convex function f, ρ-strong convexity of f is equivalent to ∂f being strongly monotone, that is,

$$\langle g_x - g_y, x - y\rangle \geq \rho\|x - y\|^2$$

for all $x, y \in \operatorname{dom}\partial f$ and $g_x \in \partial f(x)$ and $g_y \in \partial f(y)$. Cauchy-Schwarz inequality then implies that $\|x - y\| \leq \rho^{-1}\|g_x - g_y\|$. Lemma B1 in [Fan, Guo and Wang, 2019] suggests that this inequality holds if f is convex everywhere, ρ-strictly convex on a ball $B(x, r)$ and in addition $\rho^{-1}\|g_x - g_y\| \leq r$. We state this result a bit more generally:

Lemma 14. *Assume that $f : \mathbb{R}^d \to \mathbb{R}$ is convex on $\mathbb{R}^d$ and ρ-strongly convex on $C := B(x, r)$ with $r > 0$. Then, for any $y \in \mathbb{R}^d$, $g_y \in \partial f(y)$ and $g_x \in \partial f(x)$, we have*

$$\langle g_y - g_x, P_C(y) - x\rangle \geq \rho\|P_C(y) - x\|^2.$$

Proof. Let $z = P_C(y)$ where $C = B(x, r)$. The case $y \in C$ follows from strong monotonicity of ∂f on C. Assume then that $y \notin C$. It is not hard to see that $z - x = \alpha(y - z)$ for some $\alpha > 0$. By convexity of f on $\mathbb{R}^d$:

$$\langle g_y - g_z, z - x\rangle = \alpha\langle g_y - g_z, y - z\rangle \geq 0.$$

By strong convexity of f on C, $\langle g_z - g_x, z - x\rangle \geq \rho\|z - x\|^2$. Adding the two inequalities give the desired result. □

CHAPTER 4

Summary and Discussion

We have developed two order-based methods to learn Bayesian networks in this dissertation. In this chapter, we summarize these proposed algorithms in Section 4.1 and discuss future works in Section 4.2.

4.1 Overviews

We first propose annealing on regularized Cholesky score (ARCS) to estimate Gaussian DAGs, whose weighted adjacency matrix is encoded into a lower triangular matrix in conjunction with a permutation. The scoring function of ARCS is derived from regularizing Gaussian DAG likelihood, and its optimization gives an alternative formulation of the sparse Cholesky factorization problem from a statistical viewpoint. We establish the consistency of the scoring function in estimating topological sorts and DAG structures in the large-sample limit. In terms of optimization, we combine simulated annealing over permutation space with a fast proximal gradient algorithm to compute the score of any permutation. Combined, the two approaches allow us to quickly and effectively search over the space of DAGs without the need to verify the acyclicity constraint or to enumerate possible parent sets given a candidate topological sort. The annealing aspect of the optimization is able to consistently improve the accuracy of DAGs learned by greedy and deterministic search algorithms. Through extensive numerical comparisons, we show that ARCS outperformed existing methods by a substantial margin, demonstrating its great advantage in structure learning of Bayesian networks from both observational and experimental data. Though ARCS is not designed for estimating covariance matrix, it achieved higher test data likelihood than other covariance matrix estimation methods in our tests.

The second structure learning method we developed is distributed annealing on regularized likelihood score (DARLS), which considers the task of learning causal structures from data stored on multiple machines. To the best of our knowledge, it is the first method that uses distributed optimization to learn causal structures from multiple data sets stored across different machines. The objective function of DARLS is a summation of local data likelihood applied with a group ℓ_2 penalty, which is equivalent to a regularized likelihood based on the pooled data across local machines. DARLS applies annealing strategy to searches over the topological sort space for a high-scoring causal graph, where the optimal graphical structure that is compatible with a sort is found by a distributed optimization method. We establish the convergence of the distributed optimization method to a global optimizer of the overall score computed on all data across local machines. We also propose GLDAG models, where GLMs include a family of flexible distributions and can be applied to different types of data. We show that continuous GLDAGs are identifiable, while other common DAG models, such as Gaussian DAGs and multinomial DAGs, are typically not identifiable. In our simulation studies, DARLS has demonstrated competing performance with distributed data against other existing methods using pooled data across local machines. DARLS also exhibits higher predictive power than other methods in a real-world application for modeling protein-DNA binding networks using ChIP-Sequencing data.

Several computational techniques are used in our methods to facilitate network structures learning. First, we provide a principled data-driven way to determine the tuning parameters of the penalty function. BIC model selection is used to achieved this goal. Second, we use the proximal gradient algorithm, an efficient first-order method to find the optimal DAG given a topological sort. The application of the proximal gradient algorithm on our regularized likelihood score avoids the need of enumerating possible parent sets for any node when computing the optimal DAG given a topological sort. Third, refinement steps are applied to remove false positive edges after annealing. We perform constraint-based conditional independence tests to refine DAG structures in the ARCS algorithm and use some threshold values to remove small edge weights in the DARLS algorithm. These techniques help our order-based learning methods achieve high accuracy in numerical tests.

4.2 Future directions

In this section, we discuss a few directions of future advances in structure learning of Bayesian networks, with a more thorough discussion on the last two directions.

First, we are interested in extending our order-based methods to learn DAG structures from data generated from continuous and discrete variables. Given the development of our algorithms, coping with mixed data types will be straightforward. The main challenge would be modeling the interactions between continuous and discrete variables.

Second, our current theoretical results are based on the classical setting, with p fixed and n approaching infinity. Generalizing convergence and consistency results in the BN learning problem to the case that allows diverging p (i.e., the high-dimensional setting) is theoretically interesting and left as future work.

It is worthwhile investigating tensor representations that can encode (sparse) multinomial DAGs. The main difficulty is encoding the conditional independence among variables and reducing the number of parameters at the same time. Another research direction is to relax the permutation matrix space to its convex hull. Hence, a combinatorial problem searching over permutation matrices is reduced to a continuous optimization problem searching over doubly stochastic matrices. In the remainder of this dissertation, we discuss how to use tensor representation and permutation relaxation in BN learning problems.

4.2.1 Tensor representations of multinomial DAG models

Let us consider categorical variables $\{X_1, \dots, X_p\}$, where X_i has r_i levels denote by $[r_i] := \{1, \dots, r_i\}$, and further assume variables are indexed by a topological sort π, that is, $(1, 2, \dots, p)$ is a topological sort of DAG $\mathcal{G}$.

Definition 5. *(Tensor). Let $\mathcal{I} \subset N$, where N is the set of natural numbers. A tensor is a mapping $\Theta : r_{\mathcal{I}} \to \mathbb{R}$, where $r_{\mathcal{I}} = \times_{i \in \mathcal{I}} [r_i]$ is a Cartesion product of $[r_i]$. The cardinality $|\mathcal{I}|$*

is called tensor dimension.

Denote by $\Pi_i := \{X_1, X_2, \ldots, X_{i-1}\}$ a set of nodes that precedes X_i. The conditional distribution of $[X_i \mid \Pi_i]$ can be encoded by a tensor $\Theta^i \in \mathbb{R}^{r_i \times r_1 \times r_2 \times \ldots \times r_{i-1}}$ where each entry can be written as

$$\Theta^i_{x_i, x_1, \ldots, x_{i-1}} = \mathbb{P}(X_i = x_i \mid X_1 = x_1, \ldots, X_{i-1} = x_{i-1}), \tag{4.1}$$

and $\sum_{x_i=1}^{r_i} \Theta^i_{x_i, x_1, \ldots, x_{i-1}} = 1$ for any $(x_1, \ldots, x_{i-1})$. To simplify the notation, we let $\xi_{a:b} := (x_a, x_{a+1}, \ldots, x_{b-1}, x_b)$ denote some fixed values of $(X_a, X_{a+1}, \ldots, X_{b-1}, X_b)$ for $a \leq b$, and rewrite $\Theta^i_{x_i|\xi_{1:i-1}} := \Theta^i_{x_i, x_1, \ldots, x_{i-1}}$ (4.1). If X_i and X_k are conditional independent given $\Pi_i \setminus \{X_k\}$ for $X_k \in \Pi_i$, then $\mathbb{P}(X_i = x_i \mid X_1 = x_1, \ldots, X_{i-1} = x_{i-1})$ is irrelevant to the value of X_k for any x_i. That is,

$$X_i \perp\!\!\!\perp X_k \mid \Pi_i \setminus \{X_k\} \quad \Rightarrow \quad \Theta^i_{x_i|\xi_{1:k-1}, x_k = l, \xi_{k+1:i-1}} = c_i, \ \forall\, l \in [r_k], x_i \in [r_i], \tag{4.2}$$

for some constant $c_i \in [0,1]$ such that $\sum_{i=1}^{r_i} c_i = 1$. Note that (4.2) indicates that the conditional probability of $\mathbb{P}(X_i = x_i \mid \Pi_i)$ is reduced to $\mathbb{P}(X_i = x_i \mid \Pi_i \setminus X_k)$ for all x_i, reflecting the fact that $[X_i \perp\!\!\!\perp X_k \mid \Pi_i \setminus \{X_k\}]$. We illustrate such conditional independence by using tensor representations in the following example.

Example 1. Suppose we are given a DAG $\mathcal{G}$: $U \leftarrow E \rightarrow A$ (Lauritzen, 2004, Chapter 4), where

1. U represents the use of physical punishment, having states Yes (y) or No (n),
2. E indicates the childhood experience of physical punishment, having states Yes (y) or No (n).
3. A is the political affiliation with three possible states: Danish Society Democratic Party (s), the left (l) or right (r) of this party.

The DAG $\mathcal{G}$ has following marginal and conditional distributions:

$$P(E=y)=0.68,\quad \mathbb{P}(E=n)=0.32,$$
$$P(U=y\mid E=y)=0.6,\quad P(U=n\mid E=y)=0.4,$$
$$P(U=y\mid E=n)=0.3,\quad P(U=n\mid E=n)=0.7,$$
$$P(A=l\mid E=y)=0.12,\quad P(A=s\mid E=y)=0.34,\quad P(A=r\mid E=y)=0.54,$$
$$P(A=l\mid E=n)=0.37,\quad P(A=s\mid E=n)=0.27,\quad P(A=r\mid E=y)=0.36.$$

A set of tensors $\{\Theta^E\in\mathbb{R}^2,\Theta^U\in\mathbb{R}^{2\times 2},\Theta^A\in\mathbb{R}^{3\times 2\times 2}\}$ is used to encode the structure of this discrete DAG $\mathcal{G}$, such that,

$$\Theta^E=(0.68,0.32)^\top,\quad \Theta^U=\begin{pmatrix}0.6 & 0.3\\ 0.4 & 0.7\end{pmatrix},\quad \Theta^A=\begin{pmatrix}\Theta^A_{E=y,U=y} & \Theta^A_{E=y,U=n}\\ \Theta^A_{E=n,U=y} & \Theta^A_{E=n,U=n}\end{pmatrix},\tag{4.3}$$

where $\Theta^A_{E=y,U=y}=\Theta^A_{E=y,U=n}=(0.12,0.34,0.54)^\top$, as well as $\Theta^A_{E=n,U=y}=\Theta^A_{E=n,U=n}=(0.37,0.27,0.36)^\top$. In (4.3), Θ^E specifies the marginal distribution of E and each columns of Θ^U reports the conditional distribution of $[U\mid E]$. The conditional independence of $[U\perp\!\!\!\perp A\mid E]$ can be observed from Θ^A, where the conditional probabilities of A given $\{E,U\}$ only depend on the value of E.

4.2.2 Discussion on structure learning using tensors

Suppose we have a data set $\{x_h\}_{h\in[n]}$ where each row x_h are i.i.d. observations of X. Let $\Xi_{1:i-1}\in\mathbb{R}^{r_1\times r_2\times r_{i-1}}$ be a tensor where the k-th dimension encodes possible outcomes of X_k for $k\in[i-1]$. Suppose $\xi_{1:i-1}$ is a realization of $\Xi_{1:i-1}$, then we let $N_{x_i|\xi_{1:i-1}}$ be the size of subsamples where we observe $X_i=x_i$ and $\Xi_{1:i-1}=\xi_{1:i-1}$. The log-likelihood of $\{x_h\}_{h\in[n]}$ under the multinomial distribution is

$$\sum_{i=1}^{p}\sum_{\xi_{1:i-1}=1}^{z_i}\sum_{x_i=1}^{r_i}N_{x_i|\xi_{1:i-1}}\log(\Theta^i_{x_i|\xi_{1:i-1}}),\tag{4.4}$$

and we have a constraint that $\sum_{x_i=1}^{r_i}\Theta^i_{x_i|\xi_{1:i-1}}=1$ for any $\xi_{1:i-1}$ that is an observation of $\Xi_{1:i-1}$ and any $i\in[p]$. Let $z_i=\prod_{k=1}^{i-1}r_k$, and recall that $\Theta^i\in\mathbb{R}^{r_i\times r_1\times\ldots\times r_{i-1}}$. Then the number of free parameters of Θ^i is $(z_i - 1)r_i$, leading to a total of $\sum_{i=1}^{p}(z_i - 1)r_i$ free parameters of $\{\Theta^i\}_{i=1}^{p}$ (4.4). The number of parameters grows in the order of $\mathcal{O}(r^p)$ if each variable has r states.

The score-based algorithm aims to optimize an objective, such as the multinomial data likelihood (4.4). However, the tensor expression becomes prohibitively complicated when learning structures of large DAGs. One may use tensor decomposition to reduce the number of parameters of a tensor. Tensor rank-one, or the CP (CANDECOMP/PARAFAC) decomposition, has been used to decompose a probability table into a series of tables, so that the original table can be expressed as the sum of the series of tables. Such decomposition is equivalent to adding one artificial parent to all random variables and deleting all edges between the variables (Tichavský and Vomel, 2007, 2018). However, tensor decomposition cannot guarantee to reduce the edge numbers in a DAG. Investigating a tensor representation with fewer parameters while maintaining interpretability of DAGs is the main difficulty using tensors to learn network structures and is left as future work.

4.2.3 Relaxation of permutation matrices

The other research direction is to relax the permutation space to its convex hull when learning Gaussian DAGs. Recall that, by Lemma 1, the negative log-likelihood of Gaussian Bayesian networks (2.7) can be reparameterized as $\ell(L, P) := \frac{n}{2}\operatorname{tr}\left(P\widehat{\Sigma}P^\top LL^\top\right) - n\log|L|$, where L is a lower triangular matrix and P is a permutation matrix. Further, to break the permutation equivalence of the maximum likelihood (2.10), we add a regularizer ρ_θ on the lower triangular matrix to favor sparse DAGs, and in Chapter 2, we solve the following optimization problem

$$\min_{P\in\mathcal{P}_p}\min_{L\in\mathcal{L}_p} \ell(L, P) + \sum_{i>j}\rho_\theta(L_{ij}), \tag{4.5}$$

where $\mathcal{P}_p$ is the set of $p \times p$ permutation matrices, $\mathcal{L}_p$ is the set of $p \times p$ lower triangular matrices and the penalty ρ_θ is only applied to the off-diagonal entries of L. Instead of searching over permutation space, we can relax $\mathcal{P}_p$ to the convex hull of permutation matrices, and hence the combinatorial problem searching over $\mathcal{P}_p$ is converted to a continuous optimization problem search its convex hull. In this section, we discuss how to recover a topological sort with relaxation of permutation matrices.

Definition 6. *A doubly stochastic matrix is a square matrix $A = (a_{ij})$ such that $a_{ij} \geq 0$ for all i, j, and $\sum_i a_{ij} = \sum_j a_{ij} = 1$.*

It is well-known that the convex hull of the set of $p \times p$ permutation matrices is the set of doubly stochastic matrices, $\mathcal{D}_p := \{S : S \geq 0, S1 = 1, S^\top 1 = 1\}$. By replacing $\mathcal{P}_p$ with $\mathcal{D}_p$ in (4.5), we obtain a relaxation of the problem which is separately convex in $P \in \mathcal{D}_p$ and $L \in \mathcal{L}_p$. A convex function usually achieves its minimum over a convex domain at an interior point; however, the permutation matrices are extreme points of $\mathcal{D}_p$. To force the solution to be closer to the extreme points, we add a penalty applied on the Frobenius norm of the doubly stochastic matrix. Since the extreme points achieves the maximum Frobenious norm over $\mathcal{D}_p$, the relaxed optimization problem is

$$\min_{S \in \mathcal{D}_p} \min_{L \in \mathcal{L}_p} \ell(L, S) + \sum_{i>j} \rho_\theta(L_{ij}) - \gamma \|S\|_F^2, \tag{4.6}$$

where $\| \cdot \|_F^2$ is the matrix Frobenius norm and $\gamma > 0$ is a tuning parameter. Note that, if one wants to keep the convexity of the objective, the value of γ can not be arbitrarily large. More precisely, because $\ell(L, S) - \gamma \|S\|_F^2 = \left\langle \text{vec}(S), \left(\frac{n}{2}\widehat{\Sigma} \otimes LL^\top - \gamma I\right) \text{vec}(S) \right\rangle$, the problem (4.6) for a fixed L remains convex if

$$\gamma \leq \lambda_{\min}\left(\frac{n}{2}\widehat{\Sigma} \otimes LL^\top\right) = \frac{n}{2}\lambda_{\min}(\widehat{\Sigma})\lambda_{\min}(LL^\top). \tag{4.7}$$

To solve the optimization problem (4.6), we propose an alternating minimization approach, that is, given $\left(S^{(t)}, L^{(t)}\right)$, we perform the following updates until convergence:

$$L^{(t+1)} = \arg\min_{L \in \mathcal{L}_p} \ell(L, S^{(t)}) + \sum_{i>j} \rho_\theta(L_{ij}), \tag{4.8}$$

$$S^{(t+1)} = \arg\min_{S \in \mathcal{D}_p} \ell(L^{(t+1)}, S) - \gamma \|S\|_F^2. \tag{4.9}$$

Assuming a sufficiently small γ that satisfied (4.7) and the usual ℓ_1 or ℓ_2 regularizer, these problems, (4.8) and (4.9), are second order cone programs and they can be solved efficiently by convex optimization methods.

Permutation retrieval. Assuming we have a total of T iterations, the final estimate of (4.9), $S^{(T)}$ is a doubly stochastic matrix. There are two main approaches to convert a doubly stochastic matrix S to a permutation matrix P_π. The first strategy is to match ranking of coordinates of Sv and $P_\pi v$ where $v \in \mathbb{R}^p$ is a monotonically increasing vector (Fogel et al, 2015). Suppose we generate a monotonic random vector v and compute Sv. Then for each v, we can find a permutation matrix P_{π^+} such that $P_{\pi^+}Sv$ is monotonically increasing. Under this operation, the ordering defined by P_{π^+} is equivalent to the ranking of coordinates of Sv. Lastly, we can project S to the permutation space by finding π which is the inverse of permutation π^+.

The second approach is to minimize the distance between S and the permutation space, that is, $\min_{P\in\mathcal{P}_p} \|S-P\|_F^2$ where $\mathcal{P}_p$ is the set of $p\times p$ permutation matrices. This optimization problem can be simplified as

$$\max_{P\in\mathcal{P}_p} \text{vec}(S)^\top \text{vec}(P). \tag{4.10}$$

which is straightforward to solve.

4.2.4 Numerical studies

We evaluate the accuracy of recovered permutation and discuss numerical results in this section. Once a topological sort is found, the network estimation problem is simplified to a set of regression problems, and thus we leave the task of learning network structures (via permutation relaxation) as future work.

Data. We used real and synthetic networks to simulate data, where real networks were downloaded from the Bayesian networks online repository (Scutari, Accessed: 2019). The data generation process is the same as the one in Section 2.5.1. That is, given a DAG structure, we sampled the edge coefficients β_{ij} uniformly from $[-0.8, -0.5] \cup [0.5, 0.8]$ and set the noise variance to one. We then calculated the covariance matrix according to (2.3) and normalized its diagonal elements to one. Following networks were used to generate 20 data sets, denoted by the network name and (p, s_0), where s_0 is the number of edges: `Asia`

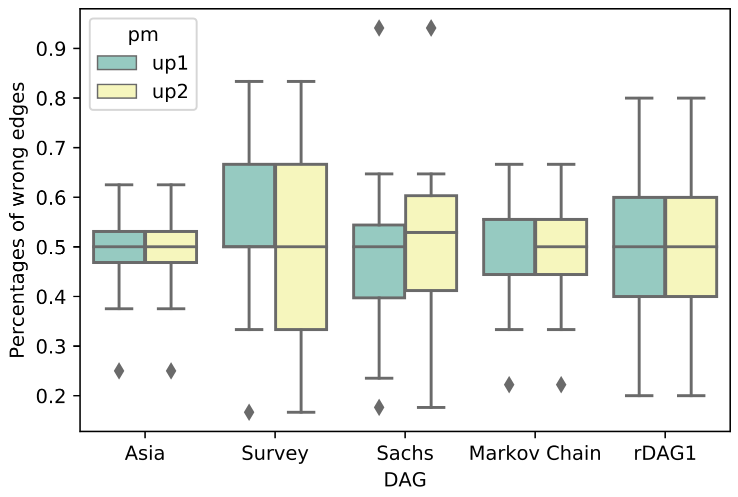


Figure 4.1: Wrong percentage of edges induced by permutation matrix (pm). up1 is $u(\widehat{P})$ with $\widehat{P}$ is found by matching orders of $\widehat{P}v$ and $S^{(T)}v$ for a monotonic random vector v and up2 is computed with $\widehat{P}$ found by minimizing the distance of S to the permutation space.

(8, 8), `Sachs` (11, 17), `Survey` (6, 6), and synthetic networks, `Markov Chain` (10, 9), `rDAG1` (10, 10).

Tuning parameters. In our numerical tests, we consider $\rho_\theta(L_{ij}) = \theta\|L_{ij}\|_2^2$, the ℓ_2 penalty with tuning parameter $\theta = 0.1$. We also set $\gamma = 2$, which is sufficient for (4.9) to be a convex optimization problem.

Evaluation metric. To evaluate the correctness of the estimated ordering $\widehat{P}$, we permute rows and columns of the true adjacency matrix B^* by the order defined by $\widehat{P}$ to obtain $B^\pi := PB^*P^\top$. We then compute the percentage of upper entries of B_π, $u_{B^*}(\widehat{P}) := \left(\sum_{i<j} \|B^\pi_{ij}\|_0\right) / \|B^*\|_0$, where $\|\cdot\|_0$ is the l_0 norm. This metric $u_{B^*}(\widehat{P})$, or simply $u(\widehat{P})$, measure the false edge percentage of B^* induced by $\widehat{P}$. Since the number of wrong edges under a random permutation follows a binary distribution, with number of trials s_0 and probability of success 0.5, we then have $\mathbb{E}_P(u) = 0.5$ and $\mathrm{SD}_P(u) = 0.5/\sqrt{s_0}$.

Numerical results. Figure 4.1 shows the distributions of $u(\widehat{P})$, where $\widehat{P}$s are estimated by the approach discussed in Section 4.2.3. Unfortunately, the median of $u(\widehat{P})$ is around

0.5, which means that $\widehat{P}$s are not better than random guesses. There are several directions to investigate, and they are left as future work:

1. We observe that the value of each entries of $S^{(T)}$ is around $1/p$, and thus $S^{(T)}$ is far away from extreme points of $\mathcal{D}_p$. Therefore, we need to increase the value of γ, indicating a need for solving a non-convex optimization problem (4.9).

2. We used iterative optimization over $L \in \mathcal{L}_p$ (4.8) and $S \in \mathcal{D}_p$ (4.9), and then retrieved a permutation matrix after convergence of these estimators. In contrast, a recently proposed method iteratively optimizes over $L \in \mathcal{L}_p$ and $P \in \mathcal{P}_p$, while carrying out permutation relaxation and retrieval at each iteration (Dallakyan and Pourahmadi, 2021). It is worthwhile to check if such iterative updates would improve accuracy of our recovered topological sorts or not.

Bibliography

Alonso-Barba JI, delaOssa L, Puerta JM (2011) Structural Learning of Bayesian Networks Using Local Algorithms Based on the Space of Orderings. Soft Computing 15(10):1881–1895

Amestoy PR, Davis TA, Duff IS (1996) An Approximate Minimum Degree Ordering Algorithm. SIAM Journal on Matrix Analysis and Application 17(4):886–905

Andersen PK, Gill RD (1982) Cox's Regression Model for Counting Processes: A Large Sample Study. The Annals of Statistics 10(4):1100–1120

Aragam B, Zhou Q (2015) Concave Penalized Estimation of Sparse Gaussian Bayesian Networks. Journal of Machine Learning Research 16:2273–2328

Aragam B, Gu J, Zhou Q (2019) Learning Large-Scaled Bayesian Networks with the `sparsebn` Package. Journal of Statistical Software 91(11):1–38

Bartlett M, Cussens J (2013) Advances in Bayesian Network Learning using Integer Programming. in Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence pp 182–191

Champion M, Picheny V, Vignes M (2018) Inferring Large Graphs Using ℓ_1-Penalized Likelihood. Statistics and Computing 28:905–921

Chen X, Xu H, Yuan P, Fang F, Huss M, Vega VB, Wong E, Orlov YL, Zhang W, Jiang J, Loh YH, Yeo HC, Yeo ZX, Narang V, Govindarajan KR, Leong B, Shahab A, Ruan Y, Bourque G, Sung WK, Clarke ND, Wei CL, Ng HH (2008) Integration of External Signaling Pathways with the Core Transcriptional Network in Embryonic Stem Cells. Cell 133:1106–17

Chen Z, Leng C (2015) Local Linear Estimation of Covariance Matrices via Cholesky Decomposition. Statistica Sinica 15(1249-1263)

Chickering DM (1996) Learning Bayesian Networks is NP-Complete. In: Learning from Data, Lecture Notes in Statistics, Springer

Chickering DM (2002) Optimal Structure Identification with Greedy Search. Journal of Machine Learning Research 3:507–554

Cooper GF, Herskovits E (1992) A Bayesian Method for the Induction of Probabilistic Networks from Data. Machine Learning 9:309–347

Cussens J, J́'arvisalo M, Korhonen JH, Bartlett M (2017) Bayesian Network Structure Learning with Integer Programming: Polytopes, Facets and Complexity. Journal of Artificial Intelligence Research 58:185–229

Dallakyan A, Pourahmadi M (2021) Learning Bayesian Networks through Birkhoff Polytope: A Relaxation Method. arXiv:2107.01658

Efron B, Hastie T, Johnstone I, Tibshirani R (2004) Least Angle Regression. The Annals of Statistics 32(2):407–499

Ellis B, Wong WH (2008) Learning Causal Bayesian Network Structures from Experimental Data. Journal of the American Statistical Association 103(482):778–789

Fan J, Li R (2001) Variable selection via nonconcave penalized likelihood and its oracle properties. Journal of the American statistical Association 96(456):1348–1360

Fan J, Guo Y, Wang K (2019) Communication-efficient accurate statistical estimation. arXiv:190604870

Fogel F, Jenatton R, Bbach F, d'Asparemont A (2015) Convext Relaxations for Permutation Problems. SIAM Journal on Matrix Analysis and Application 36(4):1465–1488

Friedman J, Hastie T, Tibshirani R (2008) Sparse Inverse Covariance Estimation with the Graphical Lasso. Biostatistics 9(3):432–441

Friedman N, Koller D (2003) Being Bayesian about Network Structure. A Bayesian Approach to Structure Discovery in Bayesian Networks. Machine Learning 50:95–125

Fu F, Zhou Q (2013) Learning Sparse Causal Gaussian Networks with Experimental Intervention: Regularization and Coordinate Descent. Journal of the American Statistical Association 108:288–300

Gámez JA, Mateo JL, Puerta JM (2011) Learning Bayesian Networks by Hill Climbing: Efficient Methods Based on Progressive Restriction of the Neighborhood. Data Mining and Knowledge Discovery 22:106–148

Ghoshal A (2019) Listen: Linear structural equation model learning. URL `https://bitbucket.org/asish_geek/listen/src/master/`, accessed: 2019-03-13

Ghoshal A, Honorio J (2018) Learning Linear Structural Equation Models in Polynomial Time and Sample Complexity. in Proceedings of Machine Learning Research 84:1466–1475

Giné E, Nickl R (2021) Mathematical foundations of infinite-dimensional statistical models. Cambridge university press

Gou K, Gong XJ, Zhao Z (2007) Learning Bayesian Network Structure from Distributed Homogeneous Data. In: ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, pp 250–254

Gu J, Fu F, Zhou Q (2019) Penalized Estimation of Directed Acyclic Graphs from Discrete Data. Statistics and Computing 29:161–176

Hauser A, Bühlmann P (2012) Characterization and Greedy Learning of Interventional Markov Equivalence Classes of Directed Acyclic Graphs. The Journal of Machine Learning Research 13:2409–2464

Heckerman D, Geiger D, Chickering DM (1995) Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. Machine Learning 20:197–243

Hoyer PO, Janzing D, Mooij J, Peters J, Schölkopf B (2008) Nonlinear Causal Discovery with Additive Noise Models. Preceedings of 21st International Conference on Neural Information Processing Systems pp 689–696

Jordan MI, Lee JD, Yang Y (2018) Communication-Efficient Distributed Statistical Inference. Journal of the American Statistical Association 114(526):668–681

Kalisch M, M´achler M, Colombo D, Maathuis MH, B´uhlmann P (2012) Causal Inference Using Graphical Models with the R Package pcalg. Journal of Statistical Software 47(11):1–26

Larrañaga P, Poza M, Yurramendi Y, Murga RH, Kuijpers CMH (1996) Structure Learning of Bayesian Networks by Genetic Algorithms: A Performance Analysis of Control Parameters. IEEE Transactions on Pattern Analysis and Machine Intelligence 18(9):912–926

Lauritzen SL (2004) Graphical Models. Oxford University Press

Lee C, van Beek P (2017) Metaheuristics for Score-and-Search Bayesian Network Structure Learning. in Proceedings of the 30th Canadian Conference on Artificial Intelligence pp 129–141

Lee K, Lee J (2018) Estimating Large Precision Matrices via Modified Cholesky Decomposition. Statistica Sinica Preprint No: SS-2018-0476

Li Q, Zhang XS (2019) Bayesian Estimation of Large Precision Matrix Based on Cholesky Decomposition. Acta Mathematica Sinica, English Series 35:619–631

Mehmood A, Natgunanathan I, Xiong Y, Hua G, Guo S (2016) Protection of Big Data Privacy. IEEE Access 4:1821–1834

Molzahn DK, Dörfler F, Sandberg H, Low SH, Chakrabarti S, Baldick R, Lavaei J (2017) A Survey of Distributed Optimization and Control Algorithms for Electric Power Systems. IEEE Transactions on Smart Grid 8(6):2941–2962

Na Y, Yang J (2010) Distributed Bayesian Network Structure Learning. In: IEEE International Symposium on Industrial Electronics, pp 1607–1611

Ouyang Z, Zhou Q, Wong WH (2009) Chip-Seq of Transcription Factors Predicts Absolute and Differential Gene Expression in Embryonic Stem Cells. Preceedings of the National Academic of Science of the United State of America 106(51):21,521–6

Parikh N, Boyd S (2013a) Proximal Algorithms. Foundations and Trends in Optimization 1(3):123–231

Parikh N, Boyd S (2013b) Proximal algorithms. Foundations and Trends in Optimization 1(3):123–231

Pearl J (1995) Causal Diagrams for Empirical Research. Biometrika 82:669–710

Peters J, Bühlmann P (2014) Identifiability of Gaussian Structural Models with Equal Error Variances. Biometrika 101(1):219–228

Peters J, Mooij JM, Janzing D, Schölkopf B (2014) Causal Discovery with Continuous Additive Noise Models. Journal of Machine Learning Research 15:2009–2053

Pourahmadi M (2011) Covariance Estimation: The GLM and Regularization Perspectives. Statistical Science 26:369–387

Ramanan N, Natarajan S (2020) Causal Learning from Predictive Modeling for Observational Data. Frontiers in Big Data 3:34

Ramsey JD (2015) Scaling up Greedy Causal Search for Continuous Variables. arXiv:150707749

Robins J (1986) A New Approach to Causal Inference in Mortality Studies with a Sustained Exposure Period - Application to Control of the Healthy Worker Survivor Effect. Mathematical Modelling 7:1393–1512

Robinson RW (1977) Counting Unlabeled Acyclic Digraphs. Lectures Notes in Mathematics 622: Combinatorial Mathematics V, CHC pp 28–43

Scanagatta M, de Campos CP, Corani G, Zaffalon M (2015) Learning Bayesian Networks with Thousands of Variables. in Advances in Neural Information Processing Systems pp 1864–1872

Scanagatta M, Corani G, Zaffalon M (2017) Improved Local Search in Bayesian Networks Structure Learning. in Proceedings of Machine Learning Research 73:45–56

Schwarz G (1978) Estimating the Dimension of a Model. The Annals of Statistics 6(2):461–464

Scutari M (2010) Learning Bayesian Networks with the bnlearn R Package. Journal of Statistical Software 35(3):1–22

Scutari M (Accessed: 2019) Bayesian network repository. URL `http://www.bnlearn.com/bnrepository/`, accessed: 2019-01-21

Shamir O, Srebro N, Zhang T (2014) Communication-Efficient Distributed Optimization using an Approximate Newton-type Method 32(2):1000–1008

Shimizu S, Hoyer PO, Hyvärinen A, Kerminen A (2006) A Linear Non-Gaussian Acyclic Model for Causal Discovery. Journal of Machine Learning Research 7(72):2003–2030

Silander T, Myllymäki P (2006) A Simple Approach for Finding the Globally Optimal Bayesian Network Structure. in Proceedings of the 22nd Conference Annual Conference on Uncertainty in Artificial Intelligence pp 445–452

Spirtes P, Glymour C (1991) An Algorithm for Fast Recovery of Sparse Causal Graphs. Social Science Computer Review 9(1):62–72

Spirtes P, Glymour C, Scheines R (1993) Causation, Prediction, and Search. Springer-Verlag, New York

Suzuki J (1993) A Construction of Bayesian Networks from Databases Based on an MDL Scheme. in Proceedings of the Ninth International Conference on Uncertainty in Artificial Intelligence pp 266–273

Tang Y, Wang J, Nguyen M, Altintas I (2019) PEnBayes: A Multi-Layered Ensemble Approach for Learning Bayesian Network Structure from Big Data. Sensors 19(4400)

Teyssier M, Koller D (2005) Ordering-Based Search: A Simple and Effective Algorithm for Learning Bayesian Networks. in Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence pp 584–590

Tichavský P, Vomel J (2007) Exploiting Tensor Rank-One Decomposition in Probabilistic Inference. Kybernetika 43:747–764

Tichavský P, Vomel J (2018) Representations of Bayesian Networks by Low-Rank Models. In: Machine Learning Research, vol 72, pp 463–474

Touchette S, Gueaieb W, Lanteigne E (2016) Efficient Cholesky Factor Recovery for Column Reordering in Simultaneous Localisation and Mapping. Journal of Intelligent & Robotic Systems 84:859–875

Tsamardinos I, Brown LE, Aliferis CF (2006) The Max-Min Hill-Climbing Bayesian Network Structure Learning Algorithm. Machine Learning 65(1):31–78

Vandenberghe L, Andersen MS (2014) Chordal Graphs and Semidefinite Optimization. Foundations and Trends in Optimization 1(4):241–433

Verma T, Pearl J (1988) Causal Network: Semantics and Expressiveness. Proceedings of the Fourth Annual Conference on Uncertainty in Artificial Intelligence pp 352–359

Verzelen N (2010) Adaptive Estimation of Covariance Matrices via Cholesky Decomposition. Electronic Journal of Statistics 4:1113–1150

Wang B, Zhou Q (2021) Causal Network Learning with N0n-intertible Functional Relationships. Computational Statistics and Data Analysis 156:107,141

Yang T, Yi X, Wu J, Yuan Y, Wu D, Meng Z, Hong Y, Wang H, Lin Z, Johansson KH (2019) A Survey of Distributed Optimization. Annual Reviews in Control 47:278–305

Ye Q, Amini AA, Zhou Q (2020) Optimizing Regularized Cholesky Score for Order-Based Learning of Bayesian Networks. IEEE Transactions on Pattern Analysis & Machine Intelligence, early access DOI: 10.1109/TPAMI.2020.2990820

Ye Q, Amini AA, Zhou Q (2021) Distributed Learning of Generalized Linear Causal Networks

Yuan M, Lin Y (2007) Model Selection and Estimation in Regression with Grouped Variables. Journal of Royal Statistical Society, Series B 68(1):49–67

Zhang CH (2010) Nearly Unbiased Variable Selection under Minimax Concave Penalty. The Annals of Statistics 38(2):894–942

Zhang Y, Duchi JC, Wainwright MJ (2013) Communication-Efficient Algorithms for Statistical Optimization. Journal of Machine Learning Research 14:3321–3363

Zheng X (2019) Dags with no tears. URL `https://github.com/xunzheng/notears`, accessed: 2019-09-17

Zheng X, Aragam B, Pavikumar P, Xing EP (2018) Dags with NO TEARS: Continuous Optimization for Structure Learning. in Advances in Neural Information Processing Systems

Zhou Q (2011) Multi-Domain Sampling with Applications to Structural Inference of Bayesian Networks. Journal of the American Statistical Association 106:1317–1330

Zhou Q, Chipperfield H, Melton DA, Wong WH (2007) A Gene Regulatory Network in Mouse Embryonic Stem Cells. Preceedings of the National Academy of Sciences of the United States of America 104(42):16,438–16,443

Zinkevich MA, Weimer M, Smola A, Li L (2010) Parallelized Stochastic Gradient Descent. In: Advances in Neural Information Processing Systems, vol 23, pp 2595–2603