UNIVERSITY OF CALIFORNIA

Los Angeles

Structural Learning of Gaussian DAGs from Network Data

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

by

Hangjian Li

2021

ABSTRACT OF THE DISSERTATION

Structural Learning of Gaussian DAGs from Network Data

by

Hangjian Li
Doctor of Philosophy in Statistics
University of California, Los Angeles, 2021
Professor Qing Zhou, Chair

Structural learning of Gaussian directed acyclic graphs (DAGs) or Bayesian networks has been studied extensively under the assumption that data are independent. But in real applications such as in biology and social studies observations generated from a Bayesian network model are often mutually dependent and their dependence can be model by a second network model.

In this dissertation, we generalize the existing Gaussian DAG framework by proposing a new Gaussian DAG model for dependent data which assumes the observations are correlated according to a given undirected network. Under this model, the dependent observations jointly follow a matrix normal distribution with variance represented by the Kronecker product of two positive definite matrices. The Cholesky factor of one of the matrices represent the DAG structure in the feature space while the other encodes the conditional independencies among the observations. We show that the proposed model also satisfies the desired score-equivalence property under common likelihood-based score functions.

Based on the proposed model, we develop a block coordinate descent algorithm to estimate the DAG structure given a topological ordering of the vertices. The proposed algorithm jointly estimates a sparse Bayesian network and the correlations among observations by optimizing a scoring function based on penalized likelihood. The algorithm is fast and can scale to networks with thousands of nodes. We also established finite-sample error bounds and large-sample consistency of the estimators. In particular, we show that under some mild conditions, the proposed method produces consistent estimators for the DAG structure and the sample covariances after one iteration. Extensive numerical experiments also demonstrate that by jointly estimating the DAG structure and the sample correlation, our method achieves much higher accuracy in structure learning than the competing algorithms. When the node ordering is unknown, through experiments on synthetic and real data, we show that our algorithm can be used to estimate the correlations between samples, with which we can de-correlate the dependent data to significantly improve the performance of classical DAG learning methods.

The dissertation of Hangjian Li is approved.

University of California, Los Angeles

2021

To my family and friends.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I'm really grateful to my advisor, Professor Qing Zhou, for his continuous support on my research projects. He has always been available and patient with my questions and ideas. I would like to thank Professor Oscar-Hernan Madrid-Padilla for many enlightening discussions and generous help on various projects. I would like to thank all my committee members for their valuable suggestions on my dissertation. I would also like to thank Dr. Zhanhao Peng, Dr. Qiaoling Ye, Dr. Carlos Cinelli, and many other friends from the Statistics Department for their help. Finally, I would like to thank my parents and Xiaoyi for their love and support.

VITA

2012–2016	B.S. (Applied Mathematics), UCLA, Los Angeles, California.
2017–2021	Assistant Editor, Journal of Statistics Software.
2016–present	Expected Ph.D. (Statistics), UCLA, Los Angeles, California.

CHAPTER 1

Introduction

Bayesian network (BN) has been a popular class of graphical models in statistical learning and causal inference, and it is widely used in various applications including genetics [Hus03, LKM96, ZBC13, VCC12], oncology [AFD18, SNB13], causal analysis [Pea95, YSW04], social network analysis [PC10, But08], etc. The increasing popularity of Bayesian network has spurred a wave of research on estimating its structure and parameters from data, especially under the high-dimensional setting when the number of variables is greater than the number of observations.

The classical problem of estimating the structure of BNs from observational data can be summarized as follows. We observe a data matrix $X \in \mathbb{R}^{n \times p}$ consisting of n i.i.d. samples from a BN of p variables. We want to estimate the structure of a BN, including the set of edges and the edge parameters, from X such that the underlying distribution of the data factorizes according to the BN. Most of the time, we hope to estimate a *sparse* BN so that more conditional independence information on the distribution is uncovered. The method should preferably be fast and scalable because the size of the space of BNs is combinatorial and scales super-exponentially. Moreover, the learning algorithm and the estimates should have good theoretical properties such as fast-convergence, finite-sample error bounds, large sample consistency, etc.

Many DAG learning methods (we discuss them in Section 1.1.2) have been developed to solve the classical DAG learning problem and some do meet the requirements above. However, a key assumption of the existing BN models and the learning algorithms we know

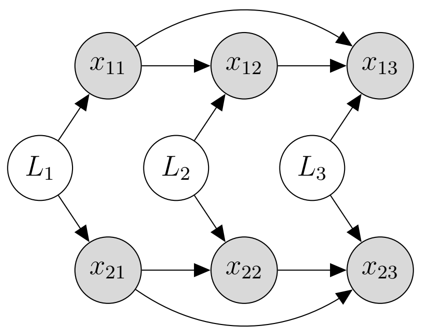


Figure 1.1: Causal Bayesian network with latent variables.

is *sample independence* — the n observations in X need to be i.i.d. In real applications, however, it is common for observations to be dependent as in network data. For example, when modeling the characteristics of an individual in a social network, the observed characteristics from different individuals can be dependent because they belong to the same social group such as friends, family, and colleagues, who often share similar characteristics. Another example appears when modeling a gene regulatory network from individuals that are potentially linked genetically. When estimating brain functional networks, we often have a matrix of fMRI measurements for each individual, $X \in \mathbb{R}^{T\times\nu}$, across T time points and ν brain regions of interests. The existence of correlations across both time points and brain regions often renders the estimates from standard graphical modeling methods inaccurate [KR20].

In causal Bayesian network (CBN) models, the dependence among samples is usually interpreted as a consequence of having unobserved confounders. For example, we can use the CBN shown in Figure 1.1 with observed variables $X = \{X_{11}, X_{12}, X_{13}, X_{21}, X_{22}, X_{23}\}$ and latent variables $\{L_1, L_2, L_3\}$ to model two dependent samples each with three features. The latent confounders do not have parents and their two children are both observed. Tian and Pearl [TP02] called this type of CBN the *semi-Markovian causal model.* In general it is difficult to learn the structure of the Bayesian network under the semi-Markovian framework with only observational data.

Motivated by these applications, we are interested in developing a BN model that can take into account the dependence between observations. Based on this model, we will develop a learning algorithm that can simultaneously infer the DAG structure and the sample associations. Moreover, since many real-world networks are sparse, we also want our method to be able to learn a sparse DAG and scale to large number of vertices. A sparsity constraint on the estimated DAG can also effectively prevent over-fitting and greatly improve the computational efficiency. Lastly, we would like to have theoretical guarantees on the consistency and finite-sample accuracy of the estimators. With these requirements in mind, we seek to

1. Develop a novel Bayesian network model for network data;
2. Develop a method that can jointly estimate a sparse DAG and the sample dependencies under the model;
3. Establish finite-sample error bound and consistency of our estimators;
4. Achieve good empirical performance on both synthetic and real data sets.

A key innovation in our proposed BN model for network data is the integration of both Bayesian network and undirected graphical model. The two types of graphical models have long been used to encode different types of conditional independencies and the undirected graphical models are well suited for the undirectional independence relations among observations. However, combined with the BN model the structural estimation problem of both networks became significantly more complicated due to the lack of independent data. Since the model we propose involves both a longitudinal (row-wise undirected graph) and a latitudinal (column-wise BN) networks, the estimation problem is always high-dimensional. In order to estimate sparse networks under the high-dimensional setting, we adopt the ℓ_1-regularization technique which has been applied separately to the structural learning of Gaussian graphical models [YL07, FHT08] and Bayesian networks [AZ15, FZ13] before. Our method combines the two by optimizing a score function with ℓ_1-regularization terms on both network structures. Intuitively, the estimator should exhibit similar properties of the existing estimators developed separately for learning Gaussian graphical model and classical BNs. In this dissertation, we will investigate this intuition and develop customized algorithm and estimators for this new model. For the rest of the introduction, we cover necessary background and introduce some previous results that our work build upon.

1.1 Bayesian networks

Graphical models are widely used to represent conditional independence among random variables. The two common branches of graphical models are directed and undirected graphical models. An undirected graphical model (UGM) is a graph $G = (V, E)$, where the vertices V represent random variables and the undirected edges $E \subseteq V \times V$ encode the conditional independence among the variables. A directed graphical model $\mathcal{G} = (V, E)$ consists of nodes V and directed edges $E \subseteq V \times V$. Bayesian network is a particular type of directed graphical model whose structure is represented by a directed acyclic graph (DAG). We first review the concept of Bayesian networks.

Suppose $X = (X_1, \dots, X_p)$ is a p dimensional random vector follows distribution $\mathbb{P}$. Let $\mathcal{G} = (V, E)$ be a DAG whose p vertices are identified with the components in X. Define the following terminology associated with $\mathcal{G}$.

- There is a *directed path* from X_i to X_j, denoted by $X_i \to X_j$, if we can go from vertex X_i to vertex X_j following the directed edges.
- There is a *path* between X_i and X_j if either $X_i \to X_j$ or $X_j \to X_i$.
- If $(X_i, X_j) \in E$, the X_i is a *parent* of X_j.
- The *ancestors* of X_j is denoted as $\text{an}(X_j) = \{X_i : X_i \to X_j\}$.
- The *descendants* of X_i is denoted as $\text{de}(X_i) = \{X_j : X_i \to X_j\}$.
- The *non-descendants* of X_j is denoted as $\text{nd}(X_i) = V \setminus (\text{de}(X_i) \cup \{X_i\})$.

Definition 1 (Bayesian network)**.** *Given a distribution $\mathbb{P}$ over variables $\{X_1, \ldots, X_p\}$ whose density function is f, we define a Bayesian network of $\mathbb{P}$ as a DAG $\mathcal{G} = (V, E)$ such that f factorizes according to $\mathcal{G}$:*

$$f(X_1, \ldots, X_p) = \prod_{X_j \in V} f_j(X_j \mid \Pi_j^{\mathcal{G}}), \tag{1.1}$$

where $\Pi_j^{\mathcal{G}} = \{X_i \in V : (X_i, X_j) \in E\}$ is the set of parents of X_j and $f_j(X_j \mid \Pi_j^{\mathcal{G}})$ is the conditional probability density for $\left[j \mid \Pi_j^{\mathcal{G}}\right]$.

We say a DAG $\mathcal{G}$ and a distribution $\mathbb{P}$ are *compatible* if $\mathbb{P}$ admits such a factorization in (1.1) according to $\mathcal{G}$. What really makes a BN model useful in practice is its ability to encode conditional independence relations in a compatible distribution with the associated DAG structure. And how a DAG encodes the conditional independence is made clear by the *local Markov property* (LMP).

Definition 2 (local Markov property)**.** *We say a joint distribution $\mathbb{P}$ of $X = (X_1, \ldots, X_p)$ satisfies the local Markov property with respect to a DAG $\mathcal{G}$ if*

$$X_i \perp nd(X_i) \mid \Pi_i^{\mathcal{G}}, \quad \forall i = 1, \ldots, p. \tag{1.2}$$

LMP is a direct consequence of the factorization property in (1.1). Therefore, a BN defined on the tuple $(\mathcal{G}, \mathbb{P})$ always satisfies LMP. It turns out that we can use LMP property to discover more conditional independence in the underlying distribution through the structure of a compatible DAG. This involves another concept of graph separation called *d-separation*.

Definition 3 (d-separation)**.** *A path γ from X_i to X_j is said to be blocked by $S \subset V$, if the path contains a vertex X_k such that either (1) or (2) holds:*

1. *$X_k \in S$ and the arrows in γ do not meet at X_k.*

2. *$X_k \cup de(X_k) \notin S$ and the arrows in γ meet at X_k.*

Two subsets of vertices A and B are d-separated by S if all paths from A to B are blocked by S. We denote it as $A \perp\!\!\!\perp_d B \mid S$.

We can directly identify conditional independence relations in $\mathbb{P}$ by looking for d-separated vertices in a compatible DAG $\mathcal{G}$ thanks to the following theorem.

Theorem 4. *[VP90] For any three disjoint subsets of nodes (X, Y, Z) in a Bayesian network of distribution $\mathbb{P}$ whose structure is a DAG $\mathcal{G}$, we have*

$$X \perp\!\!\!\perp_p Y \mid Z \implies X \perp Y \mid Z.$$

Therefore, the structure of a Bayesian network defines a set of conditional independence constraints that the underlying distribution must satisfy. This is called the *soundness of d-separation*: any distribution $\mathbb{P}$ that factorizes according to $\mathcal{G}$ must satisfy all the conditional independence implied by d-separation in $\mathcal{G}$.

1.1.1 Structural Equivalence of DAGs

Two DAGs are considered *Markov equivalent* if they encode the same set of conditional independence constraints implied by the Markov property. Let the *skeleton* of a DAG $\mathcal{G}$ be the undirected graph obtained after removing all directions of the edges in $\mathcal{G}$. A v-structure in a DAG $\mathcal{G}$ is a node triplet (i, j, k) such that $i \to j \leftarrow k$ and i, k are not adjacent in $\mathcal{G}$. Verma and Pearl [VP90] proved the following theorem for identifying equivalent DAGs.

Theorem 5 (Markov equivalence)**.** *Two DAGs $\mathcal{G}$ and $\mathcal{G}'$ are Markov equivalent if they have the same skeleton and v-structures.*

Figure 1.2 is an example of three Markov equivalent DAGs with four vertices and four edges. In this dissertation we denote two Markov equivalent DAGs $\mathcal{G}$ and $\mathcal{G}'$ as $\mathcal{G} \backsimeq \mathcal{G}'$. This equivalence relation defines a set of *equivalence class* $\mathcal{E}$ over DAGs [Chi03] and one

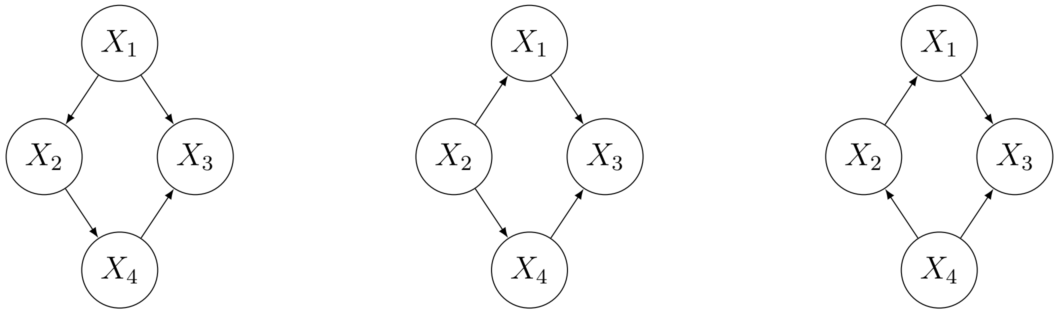


Figure 1.2: Markov equivalent DAGs.

can not distinguish the DAGs within the same equivalence class purely from observational data [VP90]. The equivalence class can be summarized by a *partially directed acyclic graph* (PDAG) which contains both directed and undirected edges but no directed cycles. In an equivalence class $\mathcal{E}$, all edges in a DAG $\mathcal{G} \in \mathcal{E}$ can be categorized as either *compelled* or *reversible*. A directed edge $X \rightarrow Y \in E(\mathcal{G})$ is compelled if $X \rightarrow Y \in \mathcal{G}', \forall\ \mathcal{G}' \in \mathcal{E}$. The edge $X \rightarrow Y$ is reversible if it is not compelled, meaning there exists another DAG $\mathcal{G}' \in \mathcal{E}$ with an edge $Y \rightarrow X$. If we orient all compelled edges in a PDAG and leave the reversible edges undirected, we will obtain a complete PDAG (CPDAG). A CPDAG is a unique representation of an equivalence class of DAGs [Chi03].

The existence of Markov equivalent DAGs plays an important role in the problem of structural learning of BNs because it implies that two DAGs can possess exactly the same statistical independence information even if some of their directed edges differ. As a result, we are not able to always uniquely identify a DAG from observational data.

1.1.2 Structural Learning Algorithms

Many structure learning algorithms for BNs have been developed, which can be largely categorized into three groups: score-based, constraint-based, and hybrid of the two.

Score-based

Score-based methods search for the optimal DAG by maximizing a scoring function such as minimum description length [Roo17], BIC [E 78], and Bayesian scores [HGC95, CH92a] with various search strategies, such as order-based search [SCC16, SNM07a, YAZ19], greedy search [RGS17, Chi03], and coordinate descent [FZ13, AZ15, GFZ19]. The greedy equivalence search (GES) algorithm developed by Meek [MM97] and further explored by Chickering [Chi03] is one of the most successful score-based algorithms. It is a two-phase algorithm that searches through the space of equivalence classes of DAGs by maximizing a Bayesian scoring function and returns partially directed DAG (PDAG) that represents an equivalence class $\mathscr{E}$. An equivalence class $\mathscr{E}$ is a *perfect map* of distribution $\mathbb{P}$ if (1) every conditional independence constraint in $\mathbb{P}$ is implied by the d-separations in DAGs $\mathcal{G} \in \mathscr{E}$ and (2) every conditional independence implied by the structure of $\mathcal{G} \in \mathscr{E}$ holds in $\mathbb{P}$. Meek in 1997 [MM97] proved that GES is optimal in the limit of large data sets as it returns the perfect map of the underlying distribution $\mathbb{P}$ of the data.

Constraint-based

Constraint-based methods, such as the PC algorithm and the Fast Causal Inference algorithm in [SGS00], perform conditional independence tests among variables to construct a skeleton of a Bayesian network before orienting the edges following certain rules. Due to the existence of Markov equivalence classes, some of the edges can not be oriented and these algorithms will return a PDAG. One issue with constraint-based algorithms is that they tend to work well only when $n \gg p$ and under-perform compared to their score-based competitors when $p > n$.

Hybrid methods

There are also hybrid methods such as the max-min hill-climbing in [TBA06] and H2PC in [GAE14] that combine the above two approaches. They first prune the search space using a constraint-based method and then search for a DAG structure by optimizing a score function.

1.1.3 Gaussian Bayesian Networks

In order for Bayesian network to be a useful modeling tool, we need to parameterize the conditional densities in (1.1). In this dissertation, we focus on the Gaussian Bayesian network where we assume the variables $\{X_1, \dots, X_p\}$ follow a multivariate Gaussian distribution. In this case, given a DAG $\mathcal{G} = (V, E)$ we can parameterize the corresponding Bayesian network as a system of *linear structural equations* with Gaussian noises:

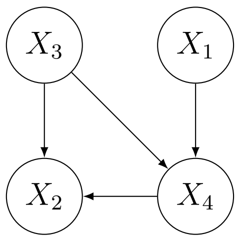
$$X_j = \sum_{i \in \Pi_j} \beta_{ij} X_i + \varepsilon_j, \quad j = 1, \dots, p, \tag{1.3}$$

where $\varepsilon_j \sim \mathcal{N}(0, \omega_j^2)$ and $\varepsilon_j \perp \Pi_j$. The $p \times p$ matrix $B = (\beta_{ij})$ is the weighted adjacency matrix (WAM) of $\mathcal{G}$ such that $\beta_{ij} \neq 0$ iff $(i, j) \in E$. $\beta_{jj} = 0$. Let $\Omega = \operatorname{diag}(\omega_1^2, \dots, \omega_p^2)$. Equation (1.3) implies that the random vector $X = (X_1, \dots, X_p) \sim \mathcal{N}(0, \Psi)$, where $\Psi = (I - B)^{-\top} \Omega (I - B)^{-1}$. The representation of the distribution of X in (1.3) is known as the linear structural equation model (SEM). And the Gaussian Bayesian network on X can be completely characterized by an SEM with the corresponding DAG structure determined by the WAM B. Therefore, the problem of structural learning of Gaussian BNs is the same as estimating (B, Ω) in (1.3). Figure 1.3 gives an example of a Gaussian Bayesian network model represented by four linear structural equations. $X = (X_1, X_2, X_3, X_3) \sim \mathcal{N}(0, \Sigma)$ and

$$X = B^\top X + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \Omega),$$

where $\Omega = \operatorname{diag}(\omega_1^2, \omega_2^2, \omega_3^2, \omega_4^2)$ and $\Sigma = (I - B^*)^{-\top} \Omega (I - B^*)^\top$.

Any random variable following multivariate Gaussian distribution can be represented by an SEM, and thus, can be modeled by a Bayesian network. Let $X = (X_1, \dots, X_p) \sim \mathcal{N}(0, \Psi)$. We pick a permutation of the node labels π and for each $j \in [p]$, project $X_{\pi^{-1}(j)}$ onto $\{X_{\pi^{-1}(1)}, X_{\pi^{-1}(2)}, \dots, X_{\pi^{-1}(j-1)}\}$. Collect all the regression coefficients and form matrix $B = \{\beta_{ij}\}$ where β_{ij} is the coefficient of X_i after regressing X_j on all the proceeding variables.



$$B^* = \begin{pmatrix} 0 & 0 & 0 & -4 \\ 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 3 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

$$\begin{aligned} X_1 &= \varepsilon_1 \\ X_3 &= \varepsilon_3 \\ X_2 &= 2X_3 + X_4 + \varepsilon_2 \\ X_4 &= -4X_1 + 3X_3 + \varepsilon_4 \end{aligned}$$

Figure 1.3: Linear structural equations.

In this case, B will be a *strictly upper-triangular* matrix. Consider the Gaussian Bayesian network over p variables in (1.3). Let P_π be a permutation matrix defined by the permutation function $\pi \in \mathcal{P}$, where $\mathcal{P}$ is the set of all permutation functions of indices $\{1, 2, \ldots, p\}$. If B is a weighted adjacency matrix, there exists a permutation π such that $P_\pi B$ is a strictly upper-triangular matrix. $P_\pi B$ is the matrix obtained after permuting the rows and columns of B according to π. And the permutation π is called a *topological sort* of the DAG defined by B.

Definition 6 (Topological sort)**.** *A topological sort of a DAG $\mathcal{G} = (V, E)$ is a permutation π of $\{1, \ldots, p\}$ such that if there is a path from i to j in $\mathcal{G}$, then $\pi^{-1}(i) < \pi^{-1}(j)$.*

Example 1. *The topological sort of B^* from Figure 1.3 is $\pi = [1, 4, 2, 3]$.*

$$B^* = \begin{pmatrix} 0 & 0 & 0 & -4 \\ 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 3 \\ 0 & 1 & 0 & 0 \end{pmatrix} \quad \xrightarrow{\pi=[1,4,2,3]} \quad P_\pi B^* = \begin{pmatrix} 0 & 0 & -4 & 0 \\ 0 & 0 & 3 & 2 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

With different permutations of the labels, the sequential projection method mentioned in the previous paragraph will yield different SEMs, and thus, different DAGs, for the same Gaussian random variable X. For a p dimensional Gaussian variable X, without extra assumption we have at least $p!$ different DAGs defined by the set of all permutations $\mathcal{P}$. In practice, we usually look for the *sparsest* DAG which contains the most conditional independence information about the underlying distribution with the fewest parameters.

1.2 An overview of score-based methods

Score-based methods for learning Bayesian networks usually consist of two components: a *score function* over the space of DAGs and a *search algorithm* that searches over the space for a DAG achieving the maximum score. The problem can be formalized as follows. Given a data matrix $X \in \mathbb{R}^{n \times p}$ containing n i.i.d. observations, find a DAG $\widehat{\mathcal{G}}$ such that

$$\widehat{\mathcal{G}} = \underset{\mathcal{G} \in \mathbb{D}_p}{\arg\max} \; g(\mathcal{G}; X),$$

where $\mathbb{D}_p = \{\mathcal{G}(V, E) \mid |V| = p, E \text{ is acyclic}\}$ is the space of DAGs with p nodes and g is a score function. With observational data, a score function evaluates the fitness of conditional independence constraints encoded by a candidate DAG on the observed data set. There are also score functions designed for causal DAGs that takes the causal semantics of the edges into evaluation [SGS00, VP90, DS93, HS95]. But in this dissertation, we will not consider causal BNs.

1.2.1 Common Score Functions

There are roughly two types of score functions for learning BNs. The first one is the Bayesian score functions that measure how well the conditional independence constraints in a candidate structure fit the data under the posterior distribution. Examples include the BD and BDe score [HGC95], K2 score [CH92b], BDeu score [Bun91], etc.

Example 2. *(Bayesian score function)*

$$g(\mathcal{G}; X) := \log P(X \mid \mathcal{G}) + \log P(\mathcal{G}).$$

In Example 2, $P(\mathcal{X} \mid \mathcal{G})$ is the probability of data given DAG $\mathcal{G}$, and $P(\mathcal{G})$ is a prior probability on DAG structures and their parameters. With different choices of the prior distribution, we will get different score functions such as the BD (Bayesian Dirichlet) score function [HGC95] under Dirichlet prior and K2 score [CH92a] under a specially constructed prior.

The second type is information-theoretical score functions that measure how much data compression can be achieved via the conditional independence constraints in a candidate DAG. A typical example in this category is the minimal description length (MDL) score function [Ris86, Bou93]. Penalized likelihood criterion such as AIC [Aka74] and BIC [Sch78] also fall into this category.

Example 3. *(Minimum description length)*

$$g(\mathcal{G}; X) := -\log P(X \mid \mathcal{G}) + f(n) \cdot |\mathcal{G}|,$$

where $|\mathcal{G}|$ is a measure of network complexity (number of parameters) [Ris86] and $f(n)$ is a non-negative penalty function on the sample size n. $f(n) = 1$ corresponds to the AIC score function [Aka74] and $f(n) = \frac{1}{2}\log(n)$ corresponds to the BIC score function [E 78].

Since most score functions evaluate the fitness of a DAG based on its conditional independence constraints, it is preferable for a score function to assign the same score to DAGs within the same equivalence class as they all encode the same set of conditional independence constraints. This is a property called *score-equivalence*. Chickering [Chi95] showed that BD/BD3, AIC, BIC, and MDL score functions mentioned above indeed have this property. K2 [CH92a] is one example of those that are not score-equivalent.

1.2.2 Penalized Likelihood of Gaussian BNs

Recall the Gaussian BN model defined by the structural linear equations in (1.3), where we observe n i.i.d. samples from a Gaussian Bayesian network with p nodes parameterized by (B, Ω). The negative log-likelihood function of the data matrix $X \in \mathbb{R}^{n\times p}$ takes the following form:

$$L(B, \Omega \mid X) = \frac{n}{2} \log\det(\Omega) + \frac{p}{2} \operatorname{tr}(S(\Omega, B)),$$

where $S(\Omega, B)$ is an $n \times n$ sample covariance matrix of $(\varepsilon_1/\omega_1, \ldots, \varepsilon_p/\omega_p)$:

$$S(\Omega, B) = \frac{1}{p} \sum_{j=1}^{p} \frac{1}{\omega_j^2} (X_j - X\beta_j)(X_j - X\beta_j)^\top.$$

Applying the trace trick and expand the $\log\det(\cdot)$ into summation,

$$L(B, \Omega \mid X) = \frac{1}{2} \sum_{j=1}^{p} n \log(\omega_j^2) + \frac{1}{\omega_j^2} \|X_j - X\beta_j\|_2^2. \tag{1.4}$$

The negative log-likelihood (NLL) function in (1.4) naturally serves as a score function over the parameter space of (B, Ω) and we can define the estimators as

$$(\widehat{B}, \widehat{\Omega}) := \operatorname*{arg\,min}_{B\in\mathcal{D}, \Omega\in\operatorname{diag}\left(\mathbb{R}^p_{++}\right)} L(B, \Omega \mid X), \quad \mathcal{D} : \text{the space of WAM of DAGs}. \tag{1.5}$$

1.2.2.1 Natural Ordering of BNs

The issues with the optimization problem in (1.5) are that the score function $L(B, \Omega \mid X)$ is non-convex and the space of WAM of DAGs with p vertices scales super-exponentially. Many approaches have been proposed to modify the optimization task in (1.5) so that it can be solved in practice. Städler et al. [SBV10] proposed the following reparameterization:

$\rho_j = 1/\omega_j$ and $\phi_{ij} = \beta_{ij}/\omega_j$, to convert the NLL in (1.4) into

$$L(\Phi, R \mid X) = \frac{1}{2}\sum_{j=1}^{p} -n\log(\rho_j^2) + \|\rho_j X_j - X\phi_j\|_2^2. \tag{1.6}$$

where $\Phi = [\phi_1 \mid \ldots \mid \phi_p]$ is the column-normalized WAM and $R = \operatorname{diag}(\rho_i)$. The score function in (1.6) is jointly convex in (Φ, R). The support of matrix Φ can be used to identify the DAG structure in the same way as the WAM B and $\Phi \in \mathcal{D}$ if and only if the corresponding $B \in \mathcal{D}$.

Unfortunately, the constraint that $\Phi \in \mathcal{D}$ is still non-convex. In order to reduce the search space so that Φ is constrained to a smaller and convex set, we sometimes assume the true DAG possess a *natural ordering.* A natural ordering is a topological sort π of the vertices in the candidate DAGs that is known from prior knowledge. In many applications, the variables modeled by a Bayesian network exhibits a natural ordering. For example, when using Bayesian network to model gene transcription processes where each vertex represents a gene's expression level, the direction of information flow among the genes is usually known from biological studies. Sometimes besides the observational data, we also have access to additional experimental data that can be used to determine the ordering of the variables. Recall the definition of a permutation matrix P_π. Given a natural ordering π, we can define a set of WAMs $B(\pi)$:

$$B(\pi) := \{B \mid P_\pi B \text{ is strictly upper triangular}\}.$$

Conditioning on the natural ordering, the candidate set of Ψ in the minimization of the loss function in (1.6) can be restricted to the set of upper triangular matrices. This can be solved by a sequential regression method.

1.2.2.2 ℓ_1-Regularized Likelihood

We can always optimize over the parameters of a *complete* DAG to achieve a high likelihood score, but the estimated DAG will not provide any information on the conditional independence among the variables. Therefore, we always would like to estimate a DAG that is the most parsimonious representation of the independence constraints in the underlying distribution. In order to promote sparsity and avoid over-fitting, people have introduced complexity penalties to the likelihood score function, such as AIC, BIC, and MDL scores introduced before.

Many learning methods have adopted a concave penalty function together with the likelihood as the score function for BN learning. Examples include the Smoothly Clipped Absolute Deviation (SCAD) penalty by Fan and Li [FL01] and the minimax concave penalty (MCP) by Zhang ([Zha10]. Concave penalty functions have been a popular choice thanks to its theoretical properties such as unbiasness, sparsity, and continuity. With the recent advancement in Lasso theories [Tib96, GB09], ℓ_1-penalty function has been increasingly popular. ℓ_1-penalty is a convex relaxation of the ℓ_0-penalty proposed by Van de Geer et al [GB13] for learning sparse DAGs. Despite being biased, it can offer good theoretical guarantees on the estimation and prediction consistency. The score function with ℓ_1-penalty also can be optimized efficiently using computational methods such as block coordinate descent [WL08].

When the natural ordering π of the BN is given, the learning problem in (1.6) can be formulated into p classical Lasso regression problems after reordering the columns in X according to π. If we assume $\hat{\omega}_j$ is given for all $j = 1, \ldots, p$, we can define the estimator

$$\hat{\beta}_j := \arg\min_{\beta} \ \|X_j - X\beta\|_2^2 + \lambda\|\beta\|_1. \tag{1.7}$$

Since the columns of X are ordered, we can set $\hat{\beta}_{kj} = 0$ for $k \geq j$ and the estimated WAM $\widehat{B} = \left[\hat{\beta}_1 \mid \hat{\beta}_2 \mid \ldots \hat{\beta}_p\right]$ is a strictly upper triangular matrix. The estimators $\hat{\beta}_j$'s in (1.7) will share the theoretical proprieties of the Lasso estimator for regression problems [MB06] such as the finite sample deviation bound and the asymptotic consistency. For example, let β_j^* denote the true value of β_j and $s = \max_j \|\beta_j^*\|_0$ denote the sparsity constant. Suppose the random matrix X defined in (1.3) satisfies the following *restricted eigenvalue (RE) condition* [MB06] with high probability:

$$\frac{\|X\beta\|_2^2}{n} \geq c_1 \|\sqrt{\Psi}\beta\|_2^2 - c_2 \rho^2(\Psi) \frac{\log p}{n} \|\beta\|_1^2 \quad \text{for all } \beta \in \mathbb{R}^p, \tag{1.8}$$

where $\rho^2(\Psi)$ is the maximum diagonal entry of Ψ and c_1, c_2 are some positive constants. we can prove the following lemma.

Lemma 7 (Lasso oracle inequality [Wai19])**.** *Under the RE condition in* (1.8) *and pick* $\lambda \asymp \sqrt{\frac{\log p}{n}}$*, there exists a positive constant* $C > 0$ *such that the estimators* $\hat{\beta}_j$ *in* (1.7) *satisfy*

$$\sup_j \|\hat{\beta}_j - \beta_j^*\|_2^2 \leq C \cdot s \frac{\log p}{n}. \tag{1.9}$$

In this dissertation, we also focus on the ℓ_1-penalized log-likelihood as our choice of score function. However, because we assume the observations are dependent, the i.i.d. assumption that the Lasso estimator relies on will not hold. Therefore, we will investigate whether the theoretical properties of the Lasso estimators can be generalized to our DAG estimators and find out what extra conditions are required.

1.3 Gaussian Graphical Model

One way of modeling the sample dependency is using a Gaussian graphical model (GGM). In this section, we review some key concepts of GGM and briefly introduce the idea of modeling sample dependence using GGM under the Gaussian BN framework. Suppose we have n univariate variables $\{\varepsilon_1, \ldots, \varepsilon_n\}$ representing n observations from another model and they jointly follow a zero-mean multivariate Gaussian distribution $\mathcal{N}_n(0, \Sigma)$. A Gaussian graphical model on $\{\varepsilon_1, \ldots, \varepsilon_n\}$ is specified by a conditional independence graph (CIG)

$G(V,E)$ with n vertices and a multivariate Gaussian distribution $\mathcal{N}_n(0,\Sigma)$. A CIG is defined as follows:

Definition 8. *(Conditional independence graph) A CIG is a graphical model with an undirected graph $G(V,E)$ and a distribution $\mathbb{P}$ such that*

$$(i,j) \notin E \implies i \perp j \mid V \setminus \{i,j\}. \tag{1.10}$$

Similar to Bayesian network model, if the vertices in the CIG G are identified with the components in $\{\varepsilon_1,\dots,\varepsilon_n\}$, then these random variables obey the conditional independence constraints imposed by the CIG in the GMM. We summarize this property in Lemma 9.

Lemma 9. *Suppose $\{\varepsilon_1,\dots,\varepsilon_n\} \sim \mathcal{N}_n(0,\Sigma)$ with $\Sigma \succ 0$ and let $\Theta = (\theta_{ij})_{p\times p} = \Sigma^{-1}$ be the precision matrix. Then*

$$\theta_{ij} = 0 \Longleftrightarrow \varepsilon_i \perp \varepsilon_j \mid \varepsilon_{-\{i,j\}}.$$

In other words, the support of the inverse covariance matrix Σ^{-1} of a GGM $(G,\mathcal{N}(0,\Sigma))$ directly correspond to the set of edges in G.

1.3.1 Sparse Inverse Covariance Estimation

Suppose we observed p i.i.d samples $E = [\varepsilon_1 \mid \varepsilon_2 \mid \dots \mid \varepsilon_p]$ from a GMM $(G,\mathcal{N}_n(0,\Sigma))$ defined on variables $\{Y_1,\dots Y_n\}$. The problem of estimating the structure of G is equivalent to estimating Σ^{-1} according to Lemma 9. In order to avoid over-fitting and find an efficient representation of the conditional independence relations in $\mathcal{N}_n(0,\Sigma)$, we prefer estimating a sparse Σ^{-1}. This task is called *sparse inverse covariance estimation.* Tackling the DAG learning problem we consider in this dissertation will involve solving a version of the sparse inverse covariance estimation problem.

The classical method for model selection in GMM is to use a greedy step-wise selection procedure that selects and deletes each edge using hypothesis testing. But it fails to consider the effect of multiple-testing and has significant computational complexity. In 2001, Meinshausen and Buhlmann [MB06] proposed a neighborhood selection method using Lasso and showed that the estimated graph structure is consistent for large sparse graphs under the high-dimensional setting. Yuan and Lin [YL07] later proposed to directly minimize the ℓ_1-penalized negative log-likelihood function as in (1.11) over the space of inverse covariance matrices. This allows estimating the edge set and the parameters of the GMM at the same time. Let $\Theta = \Sigma^{-1}$ be the inverse covariance matrix. The penalized loss function proposed in [YL07] is the following:

$$\widehat{\Theta} := \underset{\Theta \succ 0}{\arg\min} \; -\log\det(\Theta) + \operatorname{tr}(S\Theta) + \lambda\|\Theta\|_1, \tag{1.11}$$

where $\|\Theta\|_1$ denotes the sum of absolute values of the off-diagonal entries in Θ, λ is a tuning parameter for the penalty term, and S is the sample covariance matrix defined as

$$S := \frac{1}{p}\sum_{k=1}^{p} \varepsilon_k \varepsilon_k^\top .$$

They showed that the loss function in the neighborhood selection method by [MB06] is similar to a quadratic approximation of the objective in (1.11) and the optimization in [MB06] is conducted over the space of symmetric matrices instead of positive definite matrices and is less efficient. Other methods such as [BEd08, DVR08] are also based on exact minimization of the ℓ_1-penalized likelihood loss.

In 2007, Friedman et al. [FHT08] proposed the *graphical Lasso* (GLasso) algorithm to minimize the objective function in (1.11). The graphical Lasso method creatively converts the optimization problem in (1.11) into a sequence of Lasso regression problems which can be solved efficiently. Graphical Lasso uses a coordinate descent method to solve the "normal equation" below which a solution to (1.11) must satisfy:

$$-\Theta^{-1} + S + \lambda\Gamma = 0, \tag{1.12}$$

where Γ is the matrix of sub-gradients:

$$\begin{aligned}
&\gamma_{jk} = \text{sign}(\theta_{jk}) \quad \text{if } \theta_{jk} \neq 0, \\
&\gamma_{jk} \in [-1, 1] \quad\; if\ \theta_{jk} = 0.
\end{aligned}$$

Following the notations in [MH12], let $W = \Theta^{-1}$ and consider a partition of Θ and Γ:

$$\Theta = \begin{pmatrix} \Theta_{11} & \boldsymbol{\theta}_{12} \\ \boldsymbol{\theta}_{21} & \theta_{22} \end{pmatrix}, \qquad \Gamma = \begin{pmatrix} \Gamma_{11} & \boldsymbol{\gamma}_{12} \\ \boldsymbol{\gamma}_{21} & \gamma_{22} \end{pmatrix},$$

where $\Theta_{11} \in \mathbb{S}^{(n-1)\times(n-1)}$, $\boldsymbol{(}\theta_{12}\boldsymbol{)} \in \mathbb{R}^{n-1}$ and $\theta_{22} \in \mathbb{R}$ is a scalar. If W and S are partitioned in the same way, [MH12] showed that (1.12) implies

$$W_{11}\frac{\boldsymbol{\theta}_{12}}{\theta_{22}} + s_{12} + \lambda\gamma_{12} = 0. \tag{1.13}$$

The graphical Lasso algorithm solves (1.13) with $\beta = \boldsymbol{\theta}_{12}/\theta_{22}$ which is equivalent to solving the following Lasso regression problem:

$$\hat{\beta} = \underset{\beta\in\mathbb{R}^{n-1}}{\arg\min}\ \frac{1}{2}\beta^\top W_{11}\beta + \beta^\top s_{12} + \lambda\|\beta\|_1. \tag{1.14}$$

The algorithm iterates through all n columns/rows of Θ and after each iteration, one column/row of the estimated precision matrix $\widehat{\Theta}$ is updated. The algorithm cycles through the blocks until convergence. The outline of the graphical Lasso is in Algorithm 1.

For Gaussian random vectors $\{Y_1, \ldots, Y_n\}$ and the observations $E \in \mathbb{R}^{n\times p}$, Ravikumar et al. [RWR11] proved the consistency of the estimator $\widehat{\Theta}$ in (1.11) under spectral norm if

Algorithm 1: Graphical Lasso Algorithm [FHT08]

```
Input  : $S, \lambda$
  1. Initialize $W = S + \lambda I_p$
  2. Cycle through the columns and perform the following steps until convergence:
      (a) Rearrange the rows/columns so that the target column is last.
      (b) Solve the Lasso problem in (1.14).
      (c) Update the row/column of the covariance $W$ using $\hat{\beta}$.
      (d) Save $\hat{\beta}$ for the current column into a matrix $B$.
  3. Finally, convert $B$ into $\widehat{\Theta}$.
Output: $\widehat{\Theta}$
```

$Y_j's$ satisfy an *mutual incoherence condition* (MIC) (defined in Assumption 1 in [RWR11]) with parameter α.

Theorem 10 (Corollary 1 in [RWR11])**.** *Suppose* $\{\varepsilon_1, \ldots, \varepsilon_p\}$ *satisfies the incoherence condition in Assumption 1 of [RWR11] with parameter* $\alpha \in (0, 1]$. *If* $\lambda_p = (8/\alpha)\sqrt{\log n/p}$. *Then if the sample size p is large enough, with probability* $1 - 1/p^{\tau-2} \to 1$, *we have:*

$$\|\widehat{\Theta} - \Theta^*\|_2 \lesssim m\sqrt{\frac{\log n}{p}}. \tag{1.15}$$

The mutual incoherence condition is analogous to the incoherence condition for Lasso [GB09] except it is imposed on the edge-based terms $Y_jY_k - \mathbb{E}[Y_jY_k]$ instead of on the node variables $Y_j's$. If the true edge set (supp(Θ)) of the GMM is known, the MIC will not be required for consistency.

1.3.2 Sparse Matrix Graphical Model

The Gaussian graphical model can be used to model the dependencies between samples generated from a Gaussian Bayesian network and the data in this case will follow a matrix-variate normal distribution. In this section, we discuss *matrix graphical model* and how the covariance estimation problem can be generalized to this model. We defer the details of modeling Gaussian BN with dependent data and its connection to the matrix normal distribution to Chapter 2.

With two covariance matrices $\Psi \in \mathbb{S}_{++}^{p\times p}$ and $\Sigma \in \mathbb{S}_{++}^{n\times n}$, the random matrix $X \in \mathbb{R}^{n\times p}$ follows a matrix normal distribution with parameters Ψ and Σ, denoted as

$$X \sim \mathcal{N}_{n,p}(0, \Sigma, \Psi), \tag{1.16}$$

if

$$\operatorname{vec}(X) \sim \mathcal{N}_{np}\left(0, \Psi \otimes \Sigma\right).$$

where $\operatorname{vec}(\cdot)$ is the vectorization operator that stacks the p columns in X into an $np \times 1$ column vector, and $\otimes$ is the Kronecker product.

Definition 11. *(Kronecker product of matrices) The Kronecker product of $A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{k\times l}$ is denoted by $A \otimes B$. The result is an $(nk) \times (ml)$ matrix defined as*

$$A \otimes B = \begin{bmatrix} A_{11}B & A_{12}B & \dots & A_{1m}B \\ A_{21}B & A_{22}B & \dots & A_{2m}B \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1}B & A_{n2}B & \dots & A_{nm}B \end{bmatrix}.$$

The matrix normal distribution is useful for representing the conditional independencies in a random matrix variable. Let $\Theta = \Sigma^{-1}$ and $\Phi = \Psi^{-1}$ be the inverse covariance matrices for the columns and rows of X, respectively. Similar to Gaussian graphical model for vector-valued variables, the conditional independencies among the columns and rows in X are coded by the zeros of Θ and Φ, respectively. In fact, the zeros in $\Phi \otimes \Theta$ define the pairwise conditional independence of the corresponding variables conditioning on all other variables

[Lau96]. Therefore, the model in (1.16) can be seen as a generalization of GMM to random matrix variables. We call it the *matrix graphical model* (MGM). It is also called tensor graphical model [LSW20], Kronecker product model [TH13], etc.

Similar to GMM, one of the main research direction for matrix graphical model is estimating the covariances Σ and Ψ upon observing i.i.d. samples from the model (1.16). Dutilleul [Dut99] first presented an MLE estimate of the variance components for the Kronecker covariance matrix. He showed that if we observe m samples $\{X_1, \dots, X_m\}$ from (1.16), then the MLE of Ψ and Σ exist if

$$m \geq \max\left\{\frac{p}{n}, \frac{n}{p}\right\} + 1.$$

However, the MLEs $\widehat{\Psi}$ and $\widehat{\Sigma}$ are not unique and are defined up to a positive multiplicative constant since for any constant $a > 0$, $\widehat{\Psi} \otimes \widehat{\Sigma} = a\widehat{\Psi} \otimes 1/a\widehat{\Sigma}$. Only the Kronecker product $\widehat{\Psi} \otimes \widehat{\Sigma}$ is uniquely defined.

1.3.2.1 Kronecker Graphical Lasso

Applications of the matrix graphical models often come with the following challenges:

- The dimensions n and p of the data are usually very large. As a result, the number of parameters scale up fast and the computational complexity is high.
- The underlying CIGs of the rows and columns are sparse.
- The number of samples from the matrix normal distribution is only one.

MGM is naturally a good fit for high-dimensional matrix data, because by assuming that the $np \times np$ variance structure *decomposes* into a Kronecker product of two covariances matrices, the number of parameters needed is reduced to $n^2 + p^2$. But developing computationally efficient algorithms for learning sparse covariances in MGM, especially, when the sample size is 1 is still challenging. The *Kronecker Graphical Lasso* (KGLasso) method proposed by

Allen and Tibshirani [AT10] is one of the learning methods that aim to address all of these challenges. The method was originally developed to solve the Netflix movie rating challenge [BLN07]. In this challenge, researchers were asked predict user ratings for films based on existing film ratings. There were hundreds of thousands of viewers (n) and movies (p) and one viewer's rating of particular movie is correlated with the viewer's ratings of other movies as well as the ratings of other similar viewers. Matrix graphical models are well suited for this type of problems. Allen and Tibshirani [AT10] formulated this problem into a missing value problem and used matrix graphical model to impute the missing ratings by estimating the correlations among users and the ratings simultaneously. They proposed a *transposable regularized covariance model* (TRCM) and defined the covariance estimators $(\widehat{\Theta}, \widehat{\Phi})$ as the maximizer of the ℓ_1-penalized log-likelihood. Given $X \sim \mathcal{N}_{n,p}(0, \Sigma, \Psi)$, $\Theta = \Sigma^{-1}, \Phi = \Psi^{-1}$, the ℓ_1-penalized log-likelihood defined in [AT10] is

$$l(\Theta, \Phi) = \frac{p}{2} \log\det(\Theta) + \frac{n}{2} \log\det\Phi - \frac{1}{2} \operatorname{tr}\left(\Theta X \Phi X^\top\right) - \rho_r \|\Theta\|_1 - \rho_c \|\Phi\|_1. \tag{1.17}$$

In this problem, only one ratings matrix X was observed. The authors adopted the graphical Lasso framework and proposed an iterated algorithm based on graphical Lasso to estimate the two covariances. The objective in (1.17) is not jointly convex in (Θ, Φ) but it is biconvex; therefore, the flip-flop algorithm in [Dut99] can be used. Keeping one covariance fixed and maximizing (1.17) w.r.t. the other covariance yields the following estimators:

$$\widehat{\Theta}(\Phi) = \arg\min_{\Theta} \; -\log\det(\Theta) + \operatorname{tr}\left(S_1(\Phi)\Theta\right) + \rho_r \|\Theta\|_1, \tag{1.18}$$

$$\widehat{\Phi}(\Theta) = \arg\min_{\Phi} \; -\log\det(\Phi) + \operatorname{tr}\left(S_2(\Theta)\Phi\right) + \rho_c \|\Phi\|_1, \tag{1.19}$$

where $S_1(\Phi) := X\Phi X^\top/p$ and $S_2(\Theta) := X^\top \Theta X/n$ can be interpreted as sample covariance matrices. The KGLasso algorithm is detailed in Algorithm 2.

Algorithm 2: Kronecker Graphical Lasso Algorithm [AT10]

```
Input  : $X, \rho_r, \rho_c$
    1. Initialize $\widehat{\Theta} = I_n$.
    2. Repeat until convergence:
        (a) $S_2 \leftarrow X^\top \widehat{\Theta} X / n$.
        (b) Compute $\widehat{\Phi}$ by solving (1.19) with graphical Lasso.
        (c) $S_1 \leftarrow X \widehat{\Phi} X^\top / p$.
        (d) Compute $\widehat{\Theta}$ by solving (1.18) with graphical Lasso.
Output: $\widehat{\Theta} \otimes \widehat{\Phi}$
```

1.4 Outline of the Dissertation

The remainder of the dissertation is structured as follows. In Chapter 2, we propose a novel Gaussian DAG model for dependent data and discuss its connections with some existing models. We also develop a structural learning algorithm under the proposed model and discuss its properties. Chapter 3 reports numerical results of our method with detailed comparisons with some competing methods on simulated and synthetic data. Section 3.3 presents an application of our method on a real single-cell RNA sequencing data set. Chapter 4 is devoted to our main theoretical results. We develop finite-sample error bounds for the estimators and establish large-sample consistencies. In Chapter 5 we summarize our work and discuss some potential research directions.

CHAPTER 2

Model and Algorithm

2.1 Motivation

Because X is defined by the Gaussian noise vectors ε_j according to the structural equations in (1.3), dependence among the rows of X may be introduced by modeling the covariance structure among the variables $\varepsilon_{1j}, \ldots, \varepsilon_{nj}$ in ε_j. Based on this observation, we will use an undirected graph G^* to define the sparsity pattern in the precision matrix of ε_j. When G^* is an empty graph, the variables in ε_j are independent as in the classical Gaussian DAG model. However, when G^* is not empty, X follows a more complex matrix normal distribution, and the variance is defined by the product of two covariance matrices, one for the DAG $\mathcal{G}^*$ and the other for the undirected graph G^*. As a result, estimating the structure of the DAG $\mathcal{G}^*$ as well as other model parameters under the sparsity constraints in both graphs is a challenging task. We will start off by assuming that a topological ordering π^* of $\mathcal{G}^*$ is given so that the search space for DAGs can be largely reduced. However, due to the presence of the second graph for network data, the usual likelihood-based objective function used in traditional score-based methods is non-convex. The constraint-based methods do not naturally extend to network data either due to the dependence among the individuals in X, which complicates the conditional independence tests. In order to find a suitable objective function and develop an optimization algorithm, we exploit the biconvex nature of a regularized likelihood score function and develop an effective block-wise coordinate descent algorithm with a nice convergence property. If the topological ordering of the DAG is unknown, it is impossible to identify a unique DAG from data due to the Markov equivalence of DAGs [Chi03]. Moreover, due to the lack of independence, it is very difficult to estimate the equivalence class defined by $\mathcal{G}^*$. In this case, we take advantage of an invariance property of the matrix normal distribution. Under some sparsity constraint on $\mathcal{G}^*$, we show that even with a random ordering, we can still get a good estimate of the covariance of ε_j, which can be used to decorrelate X so that existing DAG learning algorithms can be applied to estimate an equivalence class of $\mathcal{G}^*$.

Notations For the convenience of the reader, we now summarize some notations to be used throughout the paper. We write $\mathcal{G}^*$ and G^* for the true DAG and the true undirected graph, respectively. Let $\Omega^* := \mathrm{diag}(\omega_j^{*2})$ be a $p \times p$ diagonal matrix of error variances, B^* denote the true WAM of $\mathcal{G}^*$, and $s := \sup_j \|\beta_j^*\|_0$ denote the maximum number of parents of any node in $\mathcal{G}^*$. Furthermore, X_j denotes the jth column of X for $j = 1, \dots, p$, and x_i denotes the ith row of X for $i = 1, \dots, n$. Given two sequences f_n and g_n, we write $f_n \lesssim g_n$ if $f_n = O(g_n)$, and $f_n \asymp g_n$ if $f_n \lesssim g_n$ and $g_n \lesssim f_n$. Denote by $[p]$ the index set $\{1, \dots, p\}$. For $x \in \mathbb{R}^n$, we denote by $\|x\|_q$ its ℓ_q norm for $q \in [0, \infty]$. For $A \in \mathbb{R}^{n \times m}$, $\|A\|_2 = \sup_v \{\|Av\|_2 : \|v\|_2 \leq 1, v \in \mathbb{R}^m\}$ is the operator norm of A, $\|A\|_f$ is the Frobenius norm of A, $\|A\|_\infty = \max_{i,j} |a_{ij}|$ is the element-wise maximum norm of A, and $|||A|||_\infty = \max_{i \in [n]} \sum_{j=1}^m |a_{ij}|$ is the maximum row-wise ℓ_1 norm of A. Denote by $\sigma_{\min}(A)$ and $\sigma_{\max}(A)$, respectively, the smallest and the largest singular values of a matrix A. Let $|S|$ be the size of a set S.

2.2 A Novel Model for Network Data

We model sample dependency through an undirected graph G^* on n vertices, with each vertex representing an observation $x_i, i \in [n]$, and the edges representing the conditional dependence relations among them. More explicitly, let $A(G^*)$ be the adjacency matrix of G^* so that

$$A(G^*)_{ij} = 0 \Rightarrow x_i \perp\!\!\!\perp x_j | x_{\setminus\{i,j\}}, \quad \forall\, i \neq j.$$

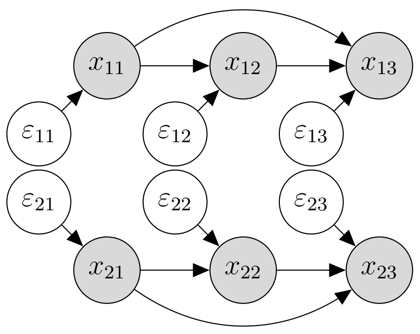


Figure 2.1: Markovian DAG model in (1.3)

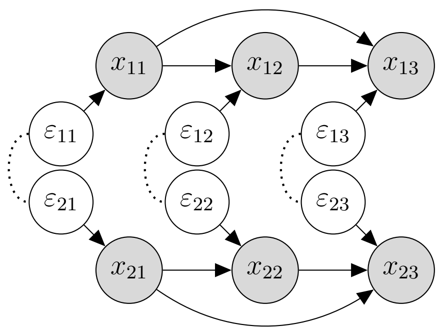


Figure 2.2: Semi-Markovian DAG model in (2.1)

Suppose we observe not only the dependent samples $\{x_i\}_{i=1}^n$ but also the graph (network) G^*. We generalize the structural equation model (SEM) in (1.3) to

$$X_j = \sum_{k \in \Pi_j} \beta^*_{kj} X_k + \varepsilon_j, \quad \varepsilon_j = (\varepsilon_{1j}, \ldots, \varepsilon_{nj}) \sim \mathcal{N}_n \left(0, \omega_j^{*2} \Sigma^*\right), \tag{2.1}$$

where $\Sigma^* \in \mathbb{R}^{n \times n}$ is positive definite. The support of the precision matrix $\Theta^* = (\Sigma^*)^{-1}$ is restricted by $\mathrm{supp}(\Theta^*) \subseteq A(G^*)$. We fix $\omega_1^* = 1$ so that the model is identifiable. Note that when $\Sigma^* = I_n$, the SEM (2.1) reduces to (1.3). Hence, the classical Gaussian DAG model in (1.3) is a special case of our proposed model (2.1). Under the more general model (2.1), we are facing a more challenging structural learning problem: Given dependent data X generated from a DAG $\mathcal{G}^*$ and the undirected graph G^* that encodes the sample dependencies, we want to estimate the DAG coefficients B^*, the noise variance $\Omega^* = \mathrm{diag}(\omega_j^{*2})$, and the precision matrix Θ^* of the samples. Before introducing our method, let us look at some useful properties of model (2.1) first.

2.2.1 Semi-Markovian Model

The distinction between (1.3) and (2.1) becomes more clear when we regard (2.1) as a semi-Markovian causal model [TKP06]. Following its causal reading [Pea95], we can represent each variable z_i in a DAG $\mathcal{G}$ on vertices $\{z_1, \ldots, z_p\}$ using a deterministic function:

$$z_i = f_i(\Pi_i, u_i), \quad i \in [p], \tag{2.2}$$

where Π_i is the set of parents of node z_i in G and u_i are noises, sometimes also referred to as background variables. The model (2.2) is *Markovian* if the noise variables u_i are jointly independent, and it is *semi-Markovian* if they are dependent. Now for a data matrix X with $n = 2$ and $p = 3$, consider the DAG models defined, respectively, by (1.3) and (2.1) over all six random variables $x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}$. Under SEM (1.3) we model $x_1 = (x_{11}, x_{12}, x_{13})$ and $x_2 = (x_{21}, x_{22}, x_{23})$ using the same SEM and assume they are independent, as shown in Figure 2.1.[1] In contrast, the model proposed in (2.1) allows observations to be dependent by relaxing the independence assumption between ε_{1k} and $\varepsilon_{2k}, k = 1, 2, 3$. If we use dashed edges to link correlated background variables, then we arrive at a semi-Markovian DAG model as shown in Figure 2.2. In general, the variables $x_{i1}, \ldots, x_{ip}$ in each individual under the semi-Markovian model satisfy the same conditional independence constraints defined by a DAG, while the background variables $\varepsilon_{1j}, \ldots, \varepsilon_{nj}$ across the n individuals are dependent. When estimating the DAG structure with such data, the correlations among individuals will reduce the effective sample size. Therefore, we need to take the distribution of the correlated ε_i into account.

2.2.2 Matrix Normal Distribution

Our model (2.1) defines a matrix normal distribution for X. To see this, note that $\varepsilon = (\varepsilon_{ij})_{n\times p}$ in (2.1) follows a matrix normal distribution:

$$\varepsilon \sim \mathcal{N}_{n,p}\left(0, \Sigma^*, \Omega^*\right) \Leftrightarrow \text{vec}(\varepsilon) \sim \mathcal{N}_{np}(0, \Omega^* \otimes \Sigma^*).$$

[1]Independent background variables are often omitted in the graph, but we include them here to better illustrate the differences between the two models.

Then, the random matrix X satisfies

$$X \sim \mathcal{N}_{n,p}(0, \Sigma^*, \Psi^*), \tag{2.3}$$

where $\Psi^* = (I - B^*)^{-\top} \Omega^* (I - B^*)^{-1}$. From the properties of a matrix normal distribution, we can easily prove the following lemma which will come in handy when estimating the row covariance matrix Σ^* from different orderings of nodes. Given a permutation π of the set $[p]$, define P_π as the permutation matrix such that $hP_\pi = (h_{\pi^{-1}(1)}, \ldots, h_{\pi^{-1}(p)})$ for any row vector $h = (h_1, \ldots, h_p)$.

Lemma 12. *If X follows the model* (2.1)*, then for any permutation π of $[p]$ we have*

$$XP_\pi \sim \mathcal{N}_{n,p}(0, \Sigma^*, P_\pi^\top \Psi^* P_\pi).$$

Although matrix normal distributions have been studied extensively in the past, the structural learning problem we consider here is quite unique. First of all, previous studies on matrix normal model usually assume we observe m copies of X and the MLE exists when $m \geq \max\{p/n, n/p\} + 1$ [Dut99]. In our case, we only observe one copy of X and thus the MLE does not exist without additional sparsity constraints. [AT10] proposed to use ℓ_1 regularization to estimate the covariance matrices when $m = 1$, but the estimation relies on the assumption that the model is transposable, meaning that the two components (Σ, Ψ) of the covariance are symmetric and can be estimated in a symmetric fashion. In model (2.1), however, the two covariance components have different structural constraints and cannot be estimated in the same way. Lastly, practitioners are often interested in estimating large Bayesian networks with hundreds or more nodes under certain sparsity assumptions on the WAM B. For example, for methods that minimize a score function to estimate the covariances, adding a sparsity regularization term on $\Psi = (I - B)^{-\top} \Omega (I - B)^{-1}$ to the score function does not necessarily lead to a sparse estimate of B. In this paper, we propose a new DAG estimation method under the assumption that both the underlying undirected network among individuals and the Bayesian network are sparse. We are not interested in estimating Ψ but a sparse factorization of Ψ represented by the WAM B. This would require imposing sparsity constraints on B itself instead of on Ψ. This is different from the recent work by [THZ13], [AT10], and [Zho14] on the Kronecker graphical lasso.

2.2.3 Score-equivalence

The likelihood function of the proposed model (2.1) also satisfies the desired score-equivalence property. To see this, let $\beta_j = (\beta_{1j}, \ldots, \beta_{pj})^\top$ be the jth column of the WAM B. Define an $n \times n$ sample covariance matrix of $\varepsilon_1/\omega_1, \ldots, \varepsilon_p/\omega_p$ from X as

$$S(\Omega, B) = \frac{1}{p}\sum_{j=1}^{p} \frac{1}{\omega_j^2} (X_j - X\beta_j)(X_j - X\beta_j)^\top. \tag{2.4}$$

Then the negative log-likelihood $L(B, \Omega, \Theta \mid X)$ from (2.1) is given by

$$2L(B, \Omega, \Theta \mid X) = n \log\det\Omega - p\log\det\Theta + p\,\mathrm{tr}(\Theta S(\Omega, B)). \tag{2.5}$$

Due to the dependence among observations, it is unclear whether the well-known score-equivalence property for Gaussian DAGs [Chi03] still holds for our model. Let $(\widehat{B}(\mathcal{G}), \widehat{\Omega}(\mathcal{G}), \widehat{\Theta}(\mathcal{G}))$ denote the MLE of (B, Ω, Θ) given a DAG $\mathcal{G}$ and the support restriction on Θ. Then, the following theorem confirms the score-equivalence property for our DAG model.

Theorem 13. *(Score equivalence) Suppose $\mathcal{G}_1$ and $\mathcal{G}_2$ are two Markov equivalent DAGs on the same set of p nodes. If the MLEs $(\widehat{B}(\mathcal{G}_m), \widehat{\Omega}(\mathcal{G}_m), \widehat{\Theta}(\mathcal{G}_m))$, $m = 1, 2$, exist for the matrix $X = (x_{ij})_{n\times p}$, then*

$$L(\widehat{B}(\mathcal{G}_1), \widehat{\Omega}(\mathcal{G}_1), \widehat{\Theta}(\mathcal{G}_1) \mid X) = L(\widehat{B}(\mathcal{G}_2), \widehat{\Omega}(\mathcal{G}_2), \widehat{\Theta}(\mathcal{G}_2) \mid X).$$

This property justifies the evaluation of estimated DAGs using common model selection criterion such as AIC and BIC. For examples, we show in Section 3.1 that one can use BIC scores to select the optimal penalty level for our proposed DAG estimation algorithm.

2.3 Methods

We have discussed the properties of our novel DAG model for dependent data and the unique challenges faced by the structural learning task. In this section, we develop a new method to estimate the parameters in model (2.1). Our estimator is defined by the minimizer of a score function that derives from a penalized log-likelihood. In order to explain our method, let us start from the penalized negative log-likelihood function:

$$f(B, \Omega, \Theta) := 2L(B, \Omega, \Theta \mid X) + \rho_1(B) + \rho_2(\Theta), \quad B \in \mathcal{D}, \tag{2.6}$$

where $\mathcal{D}$ is the space of WAMs for DAGs and ρ_1 and ρ_2 are some penalty functions. This loss function is difficult to minimize due to the non-convexity of L and the exponentially large search space of DAGs. One way to reduce the search space is to assume a given topological ordering. Recall that a WAM B is defined as $(\beta_{kj})_{p\times p}$ such that $\beta_{kj} \neq 0$ if and only if $(k, j) \in E(\mathcal{G})$; therefore, given a topological ordering π, we can define a set $\mathcal{D}(\pi)$ of WAMs compatible to π such that all $B \in \mathcal{D}(\pi)$ are strictly upper triangular after permuting its rows and columns according to π. Given a topological ordering π, the loss function (2.6) becomes

$$\begin{aligned} f(B, \Omega, \Theta) = &- p \log\det\Theta + n \log\det\Omega + \sum_{j=1}^{p} \frac{1}{\omega_j^2} \left\|LX_j - LX\beta_j\right\|_2^2 \\ &+ \rho_1(B) + \rho_2(\Theta), \quad B \in \mathcal{D}(\pi), \end{aligned} \tag{2.7}$$

where L is the Cholesky factor of Θ (i.e. $\Theta = L^\top L$). If $\rho_1(\cdot)$ and $\rho_2(\cdot)$ are convex loss functions and the noise covariance matrix $\Omega = \operatorname{diag}(\omega_j^2)$ is known, (2.7) will be a bi-convex function in (B, Θ), which can be minimized using iterative methods such as coordinate descent. [Tse01] showed that the coordinate descent algorithm in bi-convex problems converges to a stationary point. Inspired by this observation, we propose the following two-step algorithm:

Step 1: Pre-estimate Ω^* to get $\widehat{\Omega} = \mathrm{diag}(\hat{\omega}_j^2)$.

Step 2: Estimate $\widehat{B}$ and $\widehat{\Theta}$ by minimizing a biconvex score function derived from the penalized negative log-likelihood conditioning on $\hat{\omega}_j$.

Many existing noise estimation methods for high-dimensional linear models can be used to estimate $\widehat{\Omega}$ in Step 1 such as scaled lasso/MCP [SZ12], natural lasso [YB19], and refitted cross-validation [FGH12]. We will present the natural estimator of Ω and discuss a few other alternatives in Section 2.3.2. Importantly, the statistical properties of the chosen estimator $\widehat{\Omega}$ in Step 1 will affect the properties of the $\widehat{\Theta}$ and $\widehat{B}$ we get in Step 2, and thus we must choose the estimator carefully. We leave the detailed discussion of the theoretical properties of $\widehat{\Omega}$ and their implications to Chapter 4. Suppose $\widehat{\Omega}$ is given, we propose the following estimator for Step 2:

$$\left(\widehat{\Theta}, \widehat{B}(\pi)\right) = \underset{\Theta \succ 0, B \in \mathcal{D}(\pi)}{\arg\min} \left\{ -p \log\det\Theta + \sum_{j=1}^{p} \frac{1}{\hat{\omega}_j^2} \|LX_j - LX\beta_j\|_2^2 + \frac{\lambda_1}{\hat{\omega}_j^2} \|\beta_j\|_1 + \lambda_2 \|\Theta\|_1 \right\}. \tag{2.8}$$

The ℓ_1 regularization on $\beta_j/\hat{\omega}_j^2$ not only helps promote sparsity in the estimated DAG but also prevents the model from over-fitting variables that have small variances. The ℓ_1 regularization on Θ ensures that the estimator is unique and can improve the accuracy of $\widehat{\Theta}$ by controlling the error carried from the previous step. We will discuss how to control the estimation errors in more detail in Chapter 4.

In Section 2.3.1, we assume a topological ordering π^* of the true DAG $\mathcal{G}^*$ is known. In this case, we will order the columns of X according to π^* so that for each j, only the first $j-1$ entries in β_j can be nonzero. When minimizing (2.8), we fix $\beta_{jk} = 0$ for $k \geq j$ and the resulting $\widehat{B}$ is guaranteed to be upper-triangular. If π^* is unknown, we show in Section 2.3.3 how the score function in (2.8) is still useful for estimating Θ^* and describe a method of de-correlation so that standard DAG learning methods can be applied on the de-correlated data.

2.3.1 Block Coordinate Descent

We denote an estimate of the true precision matrix Θ^* at iteration t by $\widehat{\Theta}^{(t)}$. We also write $\widehat{L}^{(t)}$ and L^* for the Cholesky factors of the $\widehat{\Theta}^{(t)}$ and Θ^*, respectively. Since (2.8) is biconvex, it can be solved by iteratively minimizing over Θ and B, i.e., using block coordinate descent. Consider the tth iteration of block coordinate descent. Fixing $\widehat{\Theta}^{(t)}$, the optimization problem in (2.8) becomes the standard Lasso problem [Tib96] for each j:

$$\hat{\beta}_j^{(t+1)} = \arg\min_{\beta_j} \frac{1}{2n} \|\widehat{L}^{(t)} X_j - \widehat{L}^{(t)} X \beta_j\|_2^2 + \lambda_n \|\beta_j\|_1, \quad \lambda_n = \lambda_1/(2n), \tag{2.9}$$

where $\widehat{\Theta}^{(t)} = \widehat{L}^{(t)\top} \widehat{L}^{(t)}$ is the Cholesky decomposition. Since the columns of X are ordered according to π, we can set $\hat{\beta}_{ij}^{(t+1)} = 0$ for $i = j, j+1, \ldots, p$ and reduce the dimension of feasible β_j in (2.9) to $j-1$. In particular, $\hat{\beta}_1^{(t+1)}$ is always a zero vector. Fixing $\widehat{B}^{(t+1)}$, solving for $\widehat{\Theta}^{(t+1)}$ is equivalent to a graphical Lasso problem with fixed support [RWR11]

$$\widehat{\Theta}^{(t+1)} = \arg\min_{\Theta \succ 0,\ \mathrm{supp}(\Theta) \subseteq A(G^*)} -\log\det\Theta + \mathrm{tr}(\widehat{S}^{(t+1)}\Theta) + \lambda_p \|\Theta\|_1, \tag{2.10}$$

where $\widehat{S}^{(t+1)} = S(\widehat{\Omega}, \widehat{B}^{(t+1)})$ and $\lambda_p = \lambda_2/p$. The details of the method are given in Algorithm 3.

As shown in Proposition 14, Algorithm 3 will converge to a stationary point of the objective function (2.8). The stationary point here is defined as a point where all directional directives are nonnegative [Tse01].

Proposition 14. *Let* $\{(\widehat{B}^{(t)}, \widehat{\Theta}^{(t)}) : t = 1, 2, \ldots\}$ *be a sequence generated by the block coor-*

Algorithm 3: Block coordinate descent (BCD) algorithm

```
Input  : $X$, $\Theta^{(0)}$, $\widehat{\Omega}$, $\rho, A(G^*), T$
while $\max\left\{\|\widehat{\Theta}^{(t+1)} - \widehat{\Theta}^{(t)}\|_f, \|\widehat{B}^{(t+1)} - \widehat{B}^{(t)}\|_f\right\} > \rho$ and $t < T$ do
    for $j = 1, \ldots, p$ do
        $\hat{\beta}_j^{(t+1)} \leftarrow$ Lasso regression (2.9)
    $\widehat{\Theta}^{(t+1)} \leftarrow$ graphical Lasso with support restriction (2.10)
    $t \leftarrow t + 1$
Output: $\widehat{B} \leftarrow \widehat{B}^{(t)}, \widehat{\Theta} \leftarrow \widehat{\Theta}^{(t)}$
```

dinate descent Algorithm 3 for any $\lambda_1, \lambda_2 > 0$. Then for almost all $X \in \mathbb{R}^{n\times p}$, every cluster point of $\{(\widehat{B}^{(t)}, \widehat{\Theta}^{(t)})\}$ is a stationary point of the objective function in (2.8).

2.3.2 A Natural Estimator of Ω

We restrict our attention mostly to sparse undirected graphs G consisting of N connected components, which implies that the row precision matrix Θ is block-diagonal:

$$\Theta = \begin{pmatrix} \Theta_1 & & & \\ & \Theta_2 & & \\ & & \ddots & \\ & & & \Theta_N \end{pmatrix}. \tag{2.11}$$

The support of Θ inside each diagonal block Θ_i could be dense. This type of network is often seen in applications where individuals in the network form clusters: nodes in the same cluster are densely connected and those from different clusters tend to be more independent from each other. The underlying network G will be sparse if the individuals are from a large number of small clusters. In other words, the sparsity of G depends primarily on the number of diagonal blocks in Θ. More general network structures also are considered in the numerical experiments in Chapter 3.

Given the block-diagonal structure of Θ in (2.11), there are a few ways to estimate Ω. We use the *natural estimator* introduced by [YB19]. We estimate $\hat{\omega}_j^2$ using independent samples in X according to the block structure of Θ^*. Let $B \subseteq [n]$ be a row index set and A^B denote the submatrix formed by selecting rows from a matrix $A_{n\times m}$ with row index $i \in B$. We draw one sample from each block and form a smaller $N \times p$ design matrix X^B. It is not difficult to see that $X_j^B = X^B \beta_j^* + \varepsilon_j^B$. Next define the natural estimator of ω_j^{*2} for $j \in [p]$ as in [YB19]:

$$\hat{\omega}_j^2 = \min_{\beta_j} \left\{ \frac{1}{N} \|X_j^B - X^B \beta_j\|_2^2 + 2\lambda_N \|\beta_j\|_1 \right\}, \tag{2.12}$$

where $\lambda_N > 0$ is a tuning parameter. In Chapter 4, we discuss the estimation error rate of $\widehat{\Omega}$. Alternative methods, such as scaled lasso [SZ12] and the Stein's estimator [BEM13], can also be used to estimate ω_j^2.

2.3.3 Estimating DAGs with Unknown Ordering

Given any permutation π of $[p]$, there exists a DAG $\mathcal{G}_\pi$ such that (i) π is a topological sort of $\mathcal{G}_\pi$ and (ii) the joint distribution $\mathbb{P}$ of the p random variables factorizes according to $\mathcal{G}_\pi$. Under the assumption that the true DAG $\mathcal{G}^*$ is sparse, i.e., the number of nonzero entries in β_j^* is at most s for all j, for any random ordering π' we choose, the corresponding DAG $\mathcal{G}_{\pi'}$ is also likely to be sparse where the number of parents for each node is less than some positive constant s'. Following this intuition, we can randomly pick a permutation π' for the nodes and apply Algorithm 3 on $X_{\pi'} := XP_{\pi'}$, where $(XP_{\pi'})_{ij} = X_{i\pi'(j)}$. If the sparsity s' is small compared to the sample size n, the estimate $\hat{\beta}'_{ij}$ we get from solving the Lasso problem (2.9) will be consistent as well (we discuss the error bound on $\hat{\beta}_{ij}$ in details in Chapter 4). Moreover, since the covariance Θ^* is invariant to permutations by Lemma 12, the resulting estimate $\widehat{\Theta}$ under the random ordering π' will still be a good estimate of Θ^*. With the Cholesky factor $\widehat{L}$ of $\widehat{\Theta}$, we de-correlate the rows of X and treat

$$\widehat{X} = \widehat{L}X, \tag{2.13}$$

as the new data. Because the row correlations in $\widehat{X}$ vanish, we can apply existing structure learning methods which require independent observations to learn the underlying DAG. We find that this de-correlation step is able to substantially improve the accuracy of structure learning by well-known state-of-the-art methods, such as the greedy equivalence search (GES) [Chi03] and the PC algorithm [SGS00]. See Chapter 3 for more details.

CHAPTER 3

Numerical Results

Under the assumption that observations generated from a DAG model are dependent, we will evaluate the performance of the block coordinate descent (BCD) algorithm, i.e., Algorithm 3, in recovering the DAG compared to traditional methods that treat data as independent. We expect that the BCD method would give more accurate structural estimation than the baselines by taking the dependence information into account. When a topological ordering of the true DAG is known, we can identify a DAG from data using BCD. When the ordering is unknown, the BCD algorithm may still give an accurate estimate of the row correlations that are invariant to node-wise permutations according to Lemma 12. The estimated row correlation matrix can then be used to de-correlate the data so that traditional DAG learning algorithms would be applicable. We will demonstrate this idea of de-correlation with numerical results as well.

3.1 Simulated Networks

We first perform experiments on simulated networks for both ordered and unordered cases. To apply the BCD algorithm, we need to set values for λ_1 and λ_2 in (2.8). Since the support of Θ^* is restricted to G^*, we simply fixed λ_2 to a small value ($\lambda_2 = 0.01$) in all the experiments. For each data set, we computed a solution path from the largest $\lambda_{1\,\max}$, for which we get an empty DAG, to $\lambda_{1\,\min} = \lambda_{1\,\max}/100$. The optimal λ_1 was then chosen by minimizing the BIC score over the DAGs on the solution path.

We generated random DAGs with p nodes and fixed the total number of edges s_0 in each DAG to $2p$. The entries in the weighted adjacency matrix B^* of each DAG were drawn uniformly from $[-1, -0.1] \cup [0.1, 1]$, and ω_j^*'s were sampled uniformly from $[0.1, 2]$. In our simulations of Θ^*, we first considered networks with a clustering structure, i.e., Θ^* was block-diagonal as in (2.11). We fixed the size of the clusters to 20 or 30, and within each cluster, the individuals were correlated according to the following four covariance structures.

- Toeplitz: $\Sigma^*_{ij} = 0.3^{|i-j|/5}$.
- Equal correlation: $\Sigma^*_{ij} = 0.7$ if $i \neq j$, and $\Sigma^*_{ii} = 1$.
- Star-shaped: $\Theta^*_{1j} = \Theta^*_{i1} = a, i, j \geq 2, a \in (0, 1)$, and $\Theta^*_{ii} = 1$.
- Autoregressive (AR): $\Theta^*_{ij} = 0.7^{|i-j|}$ if $|i-j| \leq \lceil b/4 \rceil$; $\Theta^*_{ij} = 0$ otherwise, where b is the cluster size.

Toeplitz covariance structure implies that the observations are correlated as in a Markov chain. Equal correlation structure represents the cases when all observations are fully connected in a cluster. Star-shaped and AR structures capture intermediate dependence levels. Besides these block-diagonal covariances, we also considered a more general covariance structure defined through *stochastic block models* (SBM), in which G^* consists of several clusters and nodes within a cluster have a higher probability to be connected than those from different clusters. More explicitly, we generated Θ^* as follows:

1. Let $\mathcal{B}_1, \ldots, \mathcal{B}_L$ be L clusters with varying sizes that form a partition of $\{1, \ldots, n\}$, where the number of clusters L ranges from 5 to 10 in our experiments. Define a probability matrix $P \in \mathbb{R}^{n \times n}$ where $P_{ij} = 0.5$ if $i, j \in \mathcal{B}_l, l \in \{1, \ldots, L\}$; otherwise, $P_{ij} = 0.1$.

2. Construct the adjacency matrix A of G^*:

$$A_{ij} \sim \text{Bern}(P_{ij}).$$

3. Sample $\Theta'_{ij} \sim \text{Unif}[-5, 5]$ if $A_{ij} = 1$. Otherwise, $\Theta'_{ij} = 0$. To ensure a positive-definite Θ^*, we then perform the following transformations to get Θ^*:

$$\begin{aligned}\widetilde{\Theta} &= (\Theta' + \Theta'^{\top})/2 \\ \Theta^* &= \widetilde{\Theta} - \left(\sigma_{\min}(\widetilde{\Theta}) - 0.01\right) \cdot I_n\end{aligned} \tag{3.1}$$

Under the stochastic block model, two nodes from different clusters in G^* are connected with probability 0.1, so Θ^* is not block-diagonal in general. As explained in Section 4.2, our proposed BCD algorithm does not require Θ^* to be block-diagonal in practice to produce accurate estimates of B^* and Θ^*. Our numerical experiments will confirm this theory and demonstrate the robustness of the BCD method.

We compared the BCD algorithm with its competitors under both high-dimensional ($p > n$) and low-dimensional ($p < n$) settings with respect to DAG learning. For each (n, p) and each type of covariances, we simulated 10 random DAGs and then generated one data set following equation (2.1) for each DAG. Thus, we had 10 results for each of the $2 \times 5 = 10$ simulation settings. In the end, we averaged the results over the 10 simulations under each setting for comparison.

3.1.1 Learning with Given Ordering

Assuming the nodes in the DAG are sorted according to a given topological ordering, we compared our BCD algorithm against a baseline setting which fixes $\Theta^* = I_n$. In other words, the baseline algorithm ignores the dependencies among observations when estimating the DAG with BCD. The block sizes in Θ^* were set to 20 in all cases except SBM whose block sizes ranged from 5 to 25. Among other estimates, both algorithms return an estimated weighted adjacency matrix $\widehat{B}$ for the optimal λ_1 selected by BIC. For the BCD algorithm, we use $\widehat{B}$ and $\widehat{\Theta}$ for $\widehat{B}^{(\infty)}$ and $\widehat{\Theta}^{(\infty)}$ after convergence (see Algorithm 3). Note that, since $\widehat{\Theta}^{(0)}$ is initialized to I_n by default in the BCD algorithm, the estimated $\widehat{B}$ from the baseline algorithm is the same as the estimate $\widehat{B}^{(1)}$ from BCD after one iteration.

We also included the Kronecker graphical Lasso (KGLasso) algorithm [AT10, THZ13] mentioned in Section 2.2 in our comparison, which estimates both $\widehat{\Psi}$ and $\widehat{\Theta}$ via graphical Lasso in an alternating fashion. When estimating Θ^*, KGLasso also makes use of its block-diagonal structure. After KGLasso converges, we perform Cholesky factorization on $\widehat{\Psi} = (I - \widehat{B})^{-\top}\widehat{\Omega}(I - \widehat{B})^{-1}$ according to the given ordering to obtain $\widehat{B}$ and $\widehat{\Omega}$. A distinction between BCD and KGLasso is that KGLasso imposes a sparsity regularization on Ψ instead of B, so the comparison between these two will highlight the importance of imposing sparsity directly on the Cholesky factor.

Given the estimate $\widehat{B}$ from a method, we hard-thresholded the entries in $\widehat{B}$ at a threshold value $\bar{\tau}$ to obtain an estimated DAG. To compare the three methods, we chose $\bar{\tau}$ such that they predicted roughly the same number of edges (E). Then we calculated the number of true positives (TP), false positives (FP) and false negatives (FN, missing edges), and two overall accuracy metrics: Jaccard index (TP / (FP + s_0)) and structural Hamming distances (SHD = FP+FN). Note that, there were no reserved edges (i.e., estimated edges whose orientation is incorrect) because the ordering of the nodes was given. Detailed comparisons are summarized in Table 3.1 and Table 3.2. In general, the BCD algorithm outperformed the competitors by having more true positives and less false positives in every case. Because the KGLasso method does not impose sparsity directly on the DAG structure, it suffered from having too many false negatives after thresholding when $p > n$. When $p < n$, the correlations between observations had a more significant impact on the estimation accuracy for DAGs. As a result, BCD and KGLasso which take this correlation into account performed better than the baseline. In particular, BCD substantially reduced the number of missing edges (FNs) and FDR, compared to the baseline. Both BCD and KGLasso yielded accurate estimates of $\widehat{\Theta}$ when $n < p$. When $n > p$, as the sample size p for estimating $\Theta^* \in \mathbb{R}^{n\times n}$ decreased relative to the dimension n, $\widehat{\Theta}$ became less accurate. The difference in the accuracy of $\widehat{\Theta} = \widehat{\Theta}^{(\infty)}$ and $\widehat{\Theta}^{(1)}$ was not significant.

Θ-Network	Method	(n, p, s_0)	E	FN	TP	FDR	JI	SHD	$\mathrm{err}(\widehat{\Theta})$ $(\mathrm{err}(\widehat{\Theta}^{(1)}))$
equi-cor	BCD	(150, 300, 600)	686.2	214.0	386.0	0.355	0.443	514.2	0.00034 (0.00032)
	Baseline	(150, 300, 600)	642.4	240.3	359.7	0.383	0.410	523.0	—
	KGLasso	(150, 300, 600)	756.2	504.9	95.1	0.822	0.080	1166.0	0.00019
toeplitz	BCD	(200, 400, 800)	535.0	306.4	493.6	0.077	0.586	347.8	0.00143 (0.00833)
	Baseline	(200, 400, 800)	549.5	425.2	374.8	0.317	0.384	599.9	—
	KGLasso	(200, 400, 800)	550.6	634.3	165.7	0.698	0.139	1019.2	0.01617
star	BCD	(200, 400, 800)	543.1	301.1	498.9	0.081	0.591	345.3	0.00006 (0.00051)
	Baseline	(200, 400, 800)	515.7	338.3	461.7	0.103	0.541	392.3	—
	KGLasso	(200, 400, 800)	495.8	504.3	295.7	0.403	0.295	704.4	0.00233
AR(5)	BCD	(100, 200, 400)	253.1	193.8	206.2	0.184	0.461	240.7	0.00247 (0.00219)
	Baseline	(100, 200, 400)	245.8	208.9	191.1	0.217	0.420	263.6	—
	KGLasso	(100, 200, 400)	270.4	271.1	128.9	0.521	0.237	412.6	0.02587
SBM	BCD	(100, 300, 600)	343.3	313.6	286.4	0.159	0.435	370.5	0.51266 (0.52297)
	Baseline	(100, 300, 600)	344.6	338.4	261.6	0.233	0.383	421.4	—
	KGLasso	(100, 300, 600)	301.3	510.7	89.3	0.696	0.110	722.7	0.46201

Table 3.1: Results for ordered DAGs on simulated data when $n < p$. The last column shows the ℓ_2-estimation errors of $\widehat{\Theta}$ and $\widehat{\Theta}^{(1)}$ normalized by the true support size. The numbers in the brackets are errors after one iteration of BCD. Each number corresponds to the average over 10 simulations.

Θ-Network	Method	(n, p, s_0)	E	FN	TP	FDR	JI	SHD	$\mathrm{err}(\widehat{\Theta})$ $(\mathrm{err}(\widehat{\Theta}^{(1)}))$
equi-cor	BCD	(200, 100, 200)	143.0	75.1	124.9	0.119	0.570	93.2	0.00211 (0.00199)
	Baseline	(200, 100, 200)	140.0	103.8	96.2	0.305	0.394	147.6	—
	KGLasso	(200, 100, 200)	149.0	129.5	70.5	0.501	0.255	208.0	0.00207
toeplitz	BCD	(200, 100, 200)	167.1	56.7	143.3	0.137	0.639	80.5	0.51735 (0.68703)
	Baseline	(200, 100, 200)	166.7	104.8	95.2	0.425	0.351	176.3	—
	KGLasso	(200, 100, 200)	158.6	70.6	129.4	0.176	0.564	99.8	0.85023
star	BCD	(200, 100, 200)	186.9	50.6	149.4	0.199	0.629	88.1	0.54769 (0.39316)
	Baseline	(200, 100, 200)	171.7	69.2	130.8	0.236	0.543	110.1	—
	KGLasso	(200, 100, 200)	183.0	66.9	133.1	0.271	0.532	116.8	0.18472
AR(5)	BCD	(200, 100, 200)	185.9	52.2	147.8	0.200	0.622	90.3	0.01644 (0.01184)
	Baseline	(200, 100, 200)	180.1	66.2	133.8	0.253	0.545	112.5	—
	KGLasso	(200, 100, 200)	177.0	62.7	137.3	0.215	0.574	102.4	0.01038
SBM	BCD	(300, 100, 200)	140.1	72.2	127.8	0.086	0.602	84.5	0.31189 (0.31448)
	Baseline	(300, 100, 200)	139.9	85.7	114.3	0.181	0.507	111.3	—
	KGLasso	(300, 100, 200)	128.8	161.1	38.9	0.689	0.134	251.0	0.32263

Table 3.2: Results for ordered DAGs on simulated data when $n > p$.

Figure 3.1 shows the ROC curves of all three methods over a sequence of $\bar{\tau}$ under the 10 settings. The $\bar{\tau}$ sequence contains 30 equally spaced values in $[0, 0.5]$. The BCD algorithm uniformly outperformed the others in terms of the area under the curve (AUC) with substantial margins when $n < p$. When $n > p$, the BCD still did better than the other two most of the time but its lead over KGLasso and baseline was not as significant in some cases. This was largely due to insufficient regularization on $\widehat{\Theta}$. Fixing $\lambda_2 = 0.01$ in this case implies $\lambda_p = 0.01/p = 0.0001$ in the graphical Lasso step (2.10) of BCD, resulting in severe overestimates of the magnitude of the entries in Θ^*. After we increased λ_p to 0.1 which is still quite small, the BCD indeed outperformed the other two methods by much larger margins. KGLasso also performed much better when $n > p$ as shown in Table 3.2 and Figure 3.1. This is expected because when n is large compared to p, the dependence among individuals will have a larger impact on the accuracy of the estimation of DAGs. Since KGLasso is designed to iteratively estimate Θ^* and Ψ^*, the more accurate estimates of Θ^* as reported in Table 3.2 compensated for the relatively inaccurate $\widehat{B}$.

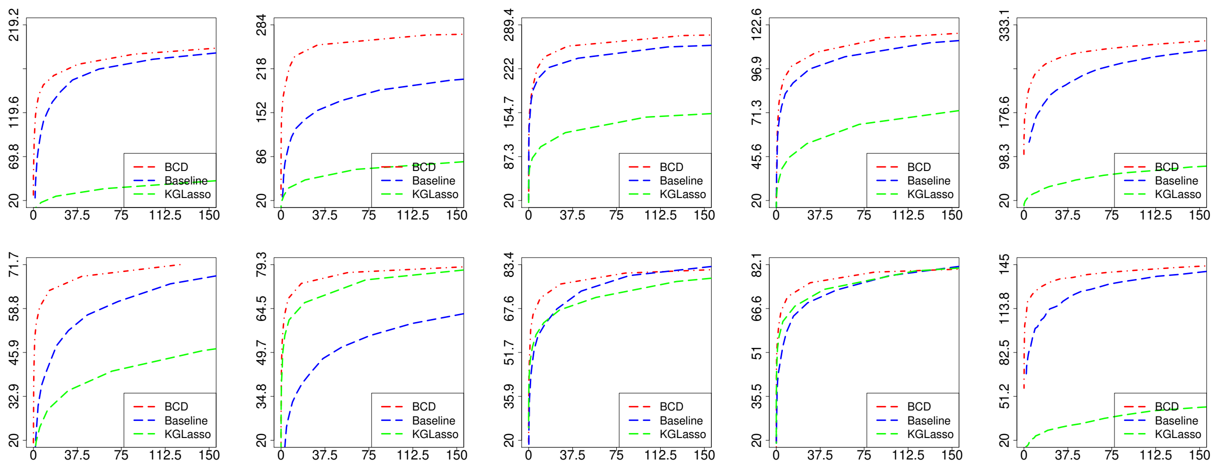


Figure 3.1: ROC curves of BCD, baseline, and KGLasso on simulated and sorted DAGs: x-axis reports the number of false positive edges and y-axis true positive edges. Structure of Θ^* from left to right: equal correlation, Toeplitz, AR, star, and SBM. Top row: $n < p$. Bottom row: $n > p$. Each data point in the ROC curves corresponds to the average over 10 simulations.

We also compared test data log-likelihood among the three methods. Specifically, under each setting, we generated a test sample matrix X_{test} from the true distribution for each of the 10 repeated simulations and computed $-L(\widehat{B}, \widehat{\Theta}, \widehat{\Omega} \mid X_{\text{test}})$ using the estimates from the three methods following equation (2.5). Figure 3.2 shows the boxplots of the test data log-likelihood, normalized by $\sqrt{np}$ after subtracting the median of the baseline method: $\mathcal{L}_{\text{plot}} = \left(\mathcal{L}_0 - \text{median}(\mathcal{L}_0^{\text{baseline}})\right) / \sqrt{np}$, where $\mathcal{L}_0$ is the original test data log-likelihood. The top row shows the test log-likelihood when $n < p$, where we did not include the data for KGLasso in four cases because its test data log-likelihood values were too small to fit in the same plot with the other two methods. The bottom row shows the results for $n > p$. For both cases, we see that the test data log-likelihood of the BCD method (in green) is consistently higher than that of the other methods.

3.1.2 Learning with De-correlation

When the natural ordering is unknown, we focus on estimating the row-wise covariance Σ^*. Given $\widehat{\Sigma}$ we can de-correlate the data by Equation (2.13) and apply existing structural

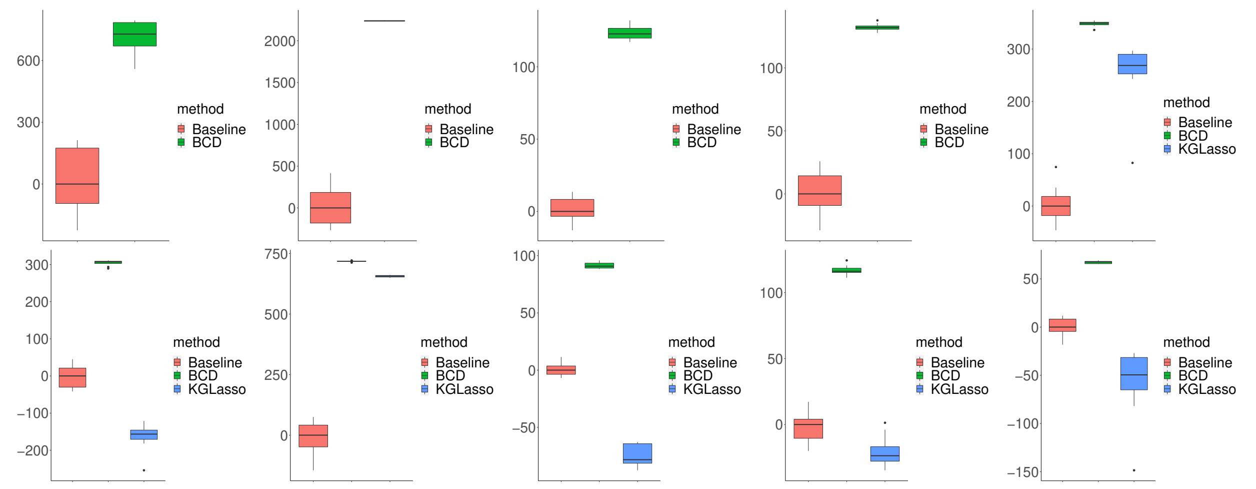


Figure 3.2: Normalized test data log-likelihood of BCD and baseline methods on simulated sorted DAGs. Top row: $n < p$. Bottom row: $n > p$. Each boxplot contains 10 data points from the 10 repeated experiments.

learning methods. In this study, we compared the performances of three structure learning methods before and after de-correlation: GES [Chi03] and sparsebn [AGZ19] which are score-based methods implemented respectively in the R packages `rcausal` [RGS17] and `sparsebn`, and PC [SGS00] which is a constraint-based method implemented in `pcalg` [KMC12]. All three methods rely on the independent data assumption, so we expect the de-correlation step to improve their performances significantly. Different from the previous comparison, the ordering of the nodes is unknown so GES and PC return an estimated CPDAG (completed acyclic partially directed graph) instead of a DAG. Thus, in the following comparisons, we converted both the estimated DAG from sparsebn and the true DAG to CPDAGs, so that all the reported metrics are computed with respect to CPDAGs.

As before, we divided the cases into $n < p$ and $n > p$. The block size for the four block-diagonal Θ was fixed to 30. The estimated Cholesky factor $\widehat{L}$ of $\widehat{\Theta}$ used for de-correlating X in (2.13) was calculated by our BCD algorithm with tuning parameter λ_1 selected by BIC. Figure 3.3 shows the decrease in SHD and increase in Jaccard index via de-correlation of GES, PC and sparsebn on 10 random DAGs, generated under each row-covariance structure and each sample size. For almost all types of covariances and (n, p)

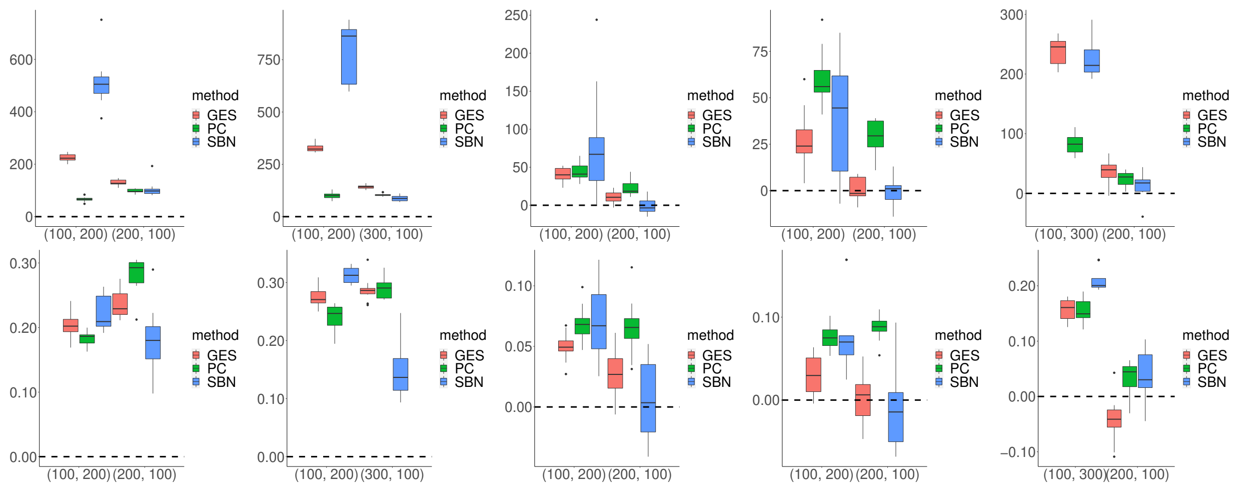


Figure 3.3: Decrease in SHD (top row) and increase in Jaccard index (bottom row) via de-correlation on simulated unsorted DAGs, with x-axis reporting the value of (n, p). In each panle, the three boxplots on the left and the three on the right correspond to the cases of $n < p$ and $n > p$, respectively. Each boxplot contains 10 data points from 10 simulations.

settings we considered, there is significant improvement of all three methods in estimating the CPDAG structures after de-correlation. Additional tables with detailed results can be found in the Supplementary Material. Before decorrelation, GES and sparsebn, both score-based methods, tend to significantly overestimate the number of edges, resulting in high false positives, so does PC in some of the cases. After decorrelation, both GES and sparsebn had significant improvements and outperformed PC, as long as $\widehat{\Theta}$ was accurately estimated. The test data log-likelihood (normalized by $\sqrt{np}$) of all three algorithms also increased significantly after decorrelation as shown in Figure 3.4.

3.2 Experiments on Real Network Structures

In this section, we look at the performance of the BCD algorithm on real network structures. We took four real DAGs from the bnlearn repository [Scu10]: `Andes`, `Hailfinder`, `Barley`, `Hepar2`, and two real undirected networks from `tnet` [Ops09]: `facebook` [OP09] and `celegans_n306` [WS98]. Only the structures (supports) of these real networks were used and the parameters of the edges were simulated as follows. Given a DAG structure, we sampled

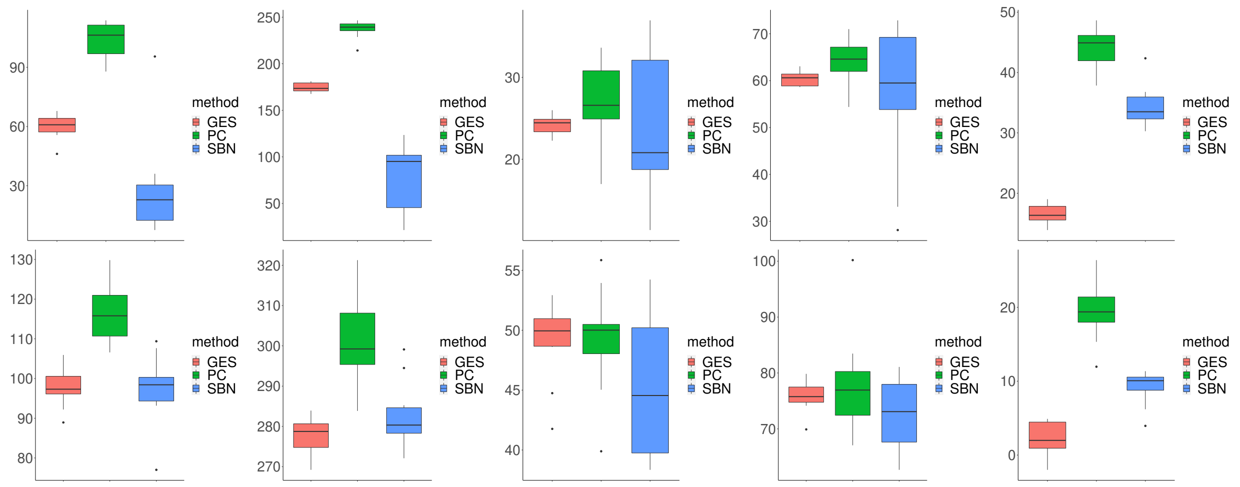


Figure 3.4: Increase in the normalized test data log-likelihood after decorrelation on simulated unsorted DAGs. Top row: $n < p$. Bottom row: $n > p$.

the coefficients β_j^* uniformly from $[-1, -0.1] \cup [0.1, 1]$. Given the support of Θ^*, we generated Θ'_{ij} uniformly from $[-5, 5]$. Then, we applied the transformations in (3.1) to get Θ^*. In order to increase the size of the underlying DAG and show the scalability of the algorithm, we duplicated the DAGs above to form larger networks. In Section 3.2.1 and 3.2.2, we again consider undirected networks consisting of several disconnected subgraphs, corresponding to a block-diagonal structure in Θ^*. Each of the subgraphs was sub-sampled from the original real network. In Section 3.2.3 we present experiments on more general Θ^* without a block-diagonal structure. The ω_j^* were uniformly sampled from $[0.1, 2]$ as before. With these parameters, we generated observational samples X following the structural equation (2.1).

3.2.1 Learning with Given Ordering

Similar to the previous section, we first looked at the results on ordered DAGs. We considered four different combinations of network structures as shown in Table 3.5, with both $n > p$ and $n < p$. Because KGLasso does not scale well with n, we did not include it in our comparisons. BCD continued to outperform the baseline method by modeling the sample correlation. This improvement was more prominent when $n > p$, where BCD significantly

			Before decorrelation							After decorrelation						
Method	Θ-Network	(n, p, s_0)	E	FN	TP	FDR	JI	R	SHD	E	FN	TP	FDR	JI	R	SHD
	equi-cor	(300, 100, 200)	222.4	100.2	99.8	0.550	0.310	30.6	253.4	123.2	85.9	114.1	0.070	0.550	29.1	124.1
	toeplitz	(300, 100, 200)	227.1	97.6	102.4	0.550	0.320	31.6	253.9	134.9	74.0	126.0	0.070	0.600	28.9	111.8
GES	star	(300, 100, 200)	136.8	77.9	122.1	0.110	0.570	29.1	121.7	132.5	76.0	124.0	0.060	0.590	26.5	111.0
	AR(5)	(300, 100, 200)	136.6	77.1	122.9	0.100	0.580	25.8	116.6	131.0	78.8	121.2	0.070	0.580	27.7	116.3
	SBM	(200, 100, 200)	169.3	110.2	89.8	0.469	0.322	34.8	224.5	76.8	139.6	60.4	0.187	0.279	31.6	187.6
	equi-cor	(300, 100, 200)	184.1	118.3	81.7	0.560	0.270	49.1	269.8	132.4	81.9	118.1	0.110	0.550	75.6	171.8
	toeplitz	(300, 100, 200)	192.5	115.9	84.1	0.560	0.270	50.9	275.2	139.0	78.0	122.0	0.120	0.560	76.6	171.6
PC	star	(300, 100, 200)	160.5	81.6	118.4	0.260	0.490	74.0	197.7	138.3	79.3	120.7	0.130	0.550	77.9	174.8
	AR(5)	(300, 100, 200)	167.5	82.6	117.4	0.300	0.470	71.0	203.7	138.5	79.0	121.0	0.130	0.560	78.1	174.6
	SBM	(200, 100, 200)	137.1	125.2	74.8	0.454	0.285	45.0	232.5	109.7	125.5	74.5	0.316	0.318	47.5	208.2
	equi-cor	(300, 100, 200)	202.5	113.7	86.3	0.570	0.280	36.0	265.9	126.4	97.8	102.2	0.180	0.450	38.2	160.2
	toeplitz	(300, 100, 200)	187.1	112.4	87.6	0.530	0.290	38.9	250.8	121.5	101.3	98.7	0.180	0.440	38.2	162.3
SBN	star	(300, 100, 200)	121.1	102.6	97.4	0.190	0.440	37.6	163.9	123.5	100.9	99.1	0.190	0.440	39.1	164.4
	AR(5)	(300, 100, 200)	118.8	103.7	96.3	0.190	0.430	35.6	161.8	111.1	108.0	92.0	0.170	0.420	35.4	162.5
	SBM	(200, 100, 200)	159.9	123.9	76.1	0.522	0.268	38.5	246.2	164.2	114.7	85.3	0.476	0.307	39.0	232.6

Table 3.3: Results for unordered DAGs on simulated data. The average number of predicted (P), true positive (TP), false positive (FP), reversed (R) edges, the average Jaccard index (JI), and Structural Hamming Distance (SHD) for CPDAGs learned by the two BCD algorithms. (Section 5.1.2)

reduced the number of false positive edges, achieving higher JI and lower SHD compared to the baseline as well as to itself in the $n < p$ case. Figure 3.5 compares the test data log-likelihood of the two methods across 10 simulations, and BCD scored significantly higher test data log-likelihood in all the cases. The ROC curves of the two methods is provided in a figure in the Supplementary Material. Both figures indicate the BCD method indeed gives better DAG estimates than the baseline method.

3.2.2 Learning with De-correlation

When the ordering of the DAG nodes is not given, we compared the effect of decorrelation as in Section 3.1.2. All network parameters were generated in the same way as before but we randomly shuffled the columns of X. The decrease in the structural Hamming distance and increase in Jaccard index from decorrelation over 10 simulations are summarized as boxplots in Figure 3.7. PC performed uniformly better after decorrelation compared to before. GES and sparsebn also improved after decorrelation in most cases. The changes in the test data log-likelihood are shown in Figure 3.8, which are positive for almost all date sets, except two

Method	Θ-Network	(n, p, s_0)	Before decorrelation							After decorrelation						
			E	FN	TP	FDR	JI	R	SHD	E	FN	TP	FDR	JI	R	SHD
	equi-cor	(100, 200, 400)	322.5	271.9	128.1	0.6	0.22	49.6	515.9	181.9	228.1	171.9	0.05	0.42	52.80	290.9
	toeplitz	(100, 200, 400)	395.9	279.7	120.3	0.70	0.18	45.8	601.1	196.1	214.2	185.8	0.05	0.45	48.8	273.3
GES	star	(100, 200, 400)	219.2	221.2	178.8	0.18	0.41	50.6	312.2	196.4	213.3	186.7	0.05	0.46	48.8	271.8
	AR(5)	(100, 200, 400)	221.7	217.2	182.8	0.18	0.42	48.9	305.0	196.7	215.7	184.3	0.06	0.45	49.60	277.7
	SBM	(100, 300, 600)	407.7	379.0	221.0	0.458	0.281	90.2	655.9	277.0	332.9	267.1	0.036	0.438	75.0	417.8
	equi-cor	(100, 200, 400)	188.9	298.9	101.1	0.46	0.21	65.3	452.0	223.8	224.7	175.3	0.22	0.39	112.8	386.0
	toeplitz	(100, 200, 400)	193.1	316.1	83.9	0.57	0.16	54.3	479.6	226.3	219.8	180.2	0.2	0.4	114.3	380.2
PC	star	(100, 200, 400)	244.6	239.0	161.0	0.34	0.33	101.6	414.2	228.6	219.8	180.2	0.21	0.4	112.0	380.2
	AR(5)	(100, 200, 400)	264.2	240.7	159.3	0.40	0.32	100.9	446.5	224.8	224.2	175.8	0.22	0.39	112.4	385.6
	SBM	(100, 300, 600)	313.8	421.2	178.8	0.429	0.243	115.0	671.2	356.4	327.4	272.6	0.235	0.399	177.2	588.4
	equi-cor	(100, 200, 400)	747.4	221.5	178.5	0.76	0.19	75.2	865.6	265.4	207.6	192.4	0.26	0.41	70.1	350.7
	toeplitz	(100, 200, 400)	1182.2	243.6	156.4	0.86	0.11	69.4	1338.8	309.6	188.1	211.9	0.3	0.43	75.8	361.6
SBN	star	(100, 200, 400)	340.0	211.4	188.6	0.43	0.34	67.5	430.3	277.5	201.3	198.7	0.27	0.42	68.1	348.2
	AR(5)	(100, 200, 400)	355.8	214.1	185.9	0.44	0.33	68.0	452.0	301.8	194.6	205.4	0.3	0.41	67.4	358.4
	SBM	(100, 300, 600)	356.1	428.5	171.5	0.516	0.219	90.9	704.0	332.5	320.1	279.9	0.158	0.429	105.9	478.6

Table 3.4: Results for unordered DAGs on simulated data. Block size is equal to 20. The average number of predicted (P), true positive (TP), false positive (FP), reversed (R) edges, the average Jaccard index (JI), and Structural Hamming Distance (SHD) for CPDAGs learned by the two BCD algorithms. (Section 5.1.2)

DAG	Θ-Network	Method	(n, p, s_0)	E	FN	TP	FDR	JI	SHD	$\text{err}(\widehat{\Theta})$ $(\text{err}(\widehat{\Theta}^{(1)}))$
		BCD	(100 446, 676)	440.9	314.7	361.3	0.176	0.478	394.3	0.03458 (0.03564)
		Baseline	(100 446, 676)	436.6	334.2	341.8	0.211	0.444	429.0	—
Andes	facebook									
(2)		BCD	(500 446, 676)	500.0	197.3	478.7	0.043	0.686	218.6	0.07816 (0.07758)
		Baseline	(500 446, 676)	499.1	206.4	469.6	0.058	0.666	235.9	—
		BCD	(100, 224, 264)	154.5	124.2	139.8	0.092	0.502	138.9	0.03922 (0.02167)
		Baseline	(100, 224, 264)	154.8	126.8	137.2	0.110	0.487	144.4	—
Hailfinder	celegan_n306									
(4)		BCD	(500, 224, 264)	168.4	97.1	166.9	0.009	0.629	98.6	0.02010 (0.02138)
		Baseline	(500, 224, 264)	168.0	98.0	166.0	0.012	0.624	100.0	—
		BCD	(100, 192, 336)	211.0	150.5	185.5	0.119	0.513	176.0	0.01453 (0.00100)
		Baseline	(100, 192, 336)	211.9	156.2	179.8	0.150	0.489	188.3	—
Barley	facebook									
(4)		BCD	(500, 192, 336)	260.5	91.3	244.7	0.061	0.696	107.1	0.23144 (0.28172)
		Baseline	(500, 192, 336)	260.2	97.6	238.4	0.083	0.666	119.4	—
		BCD	(100, 280, 492)	366.7	234.5	257.5	0.295	0.428	343.7	0.05101 (0.03104)
		Baseline	(100, 280, 492)	373.9	238.0	254.0	0.314	0.416	357.9	—
Hepar2	celegan_n306									
(4)		BCD	(500, 280, 492)	417.5	122.2	369.8	0.114	0.685	169.9	0.00759 (0.00755)
		Baseline	(500, 280, 492)	417.1	126.0	366.0	0.122	0.674	177.1	—

Table 3.5: Results for ordered DAGs on real network data. Block size is 20 for $n < p$ and 50 for $n > p$. The number under each DAG reports the number of times it is duplicated to form a large DAG. All numbers represent the average over 10 simulations.

outliers (removed from plots) of sparsebn in the second and fourth panels in the top row.

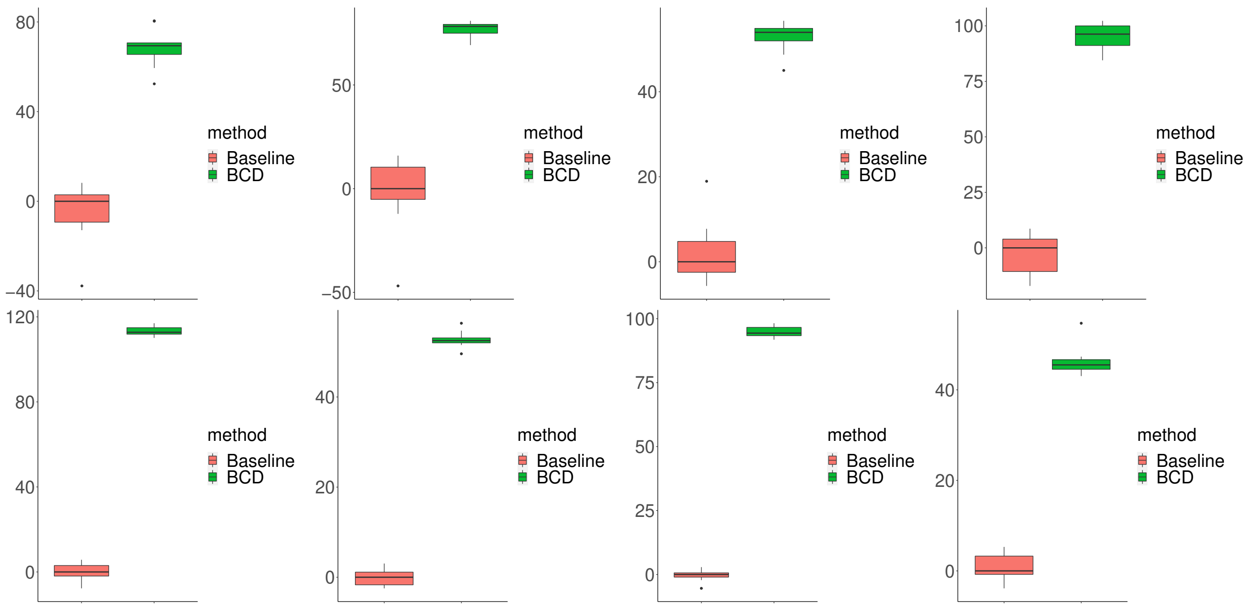


Figure 3.5: Test data log-likelihood normalized by $\sqrt{np}$ on real sorted DAGs. Top row: $n < p$. Bottom row: $n > p$.

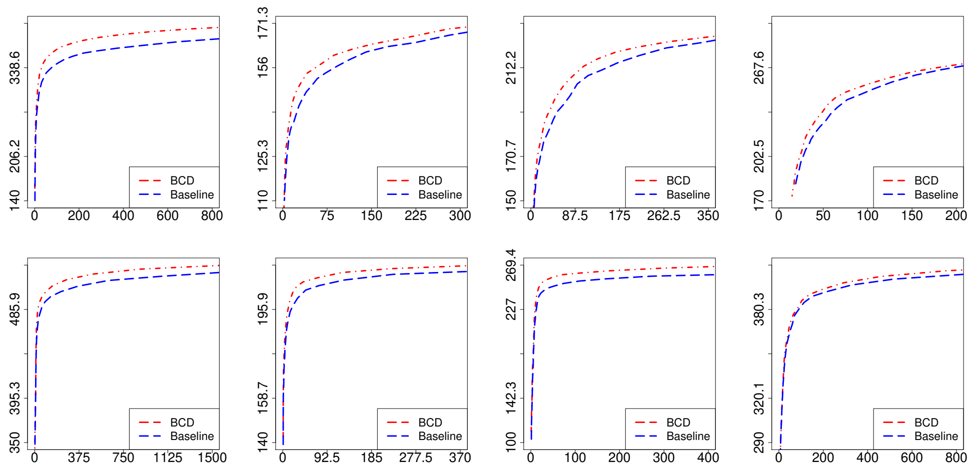


Figure 3.6: ROC curves of BCD and baseline methods on real networks across 50 threshold values ($\bar{\tau}$) equally spaced from $[0, 0.5]$. Top row: $n < p, n = 100$, block size is 20. Bottom row: $n > p, n = 500$, block size is 50. (Section 5.2.1)

3.2.3 Learning under General Covariance Structure

In the following experiments, we generated the support of Θ^* without the block-diagonal constraint by directly sampling the real undirected networks `facebook` and `celegans_n306`.

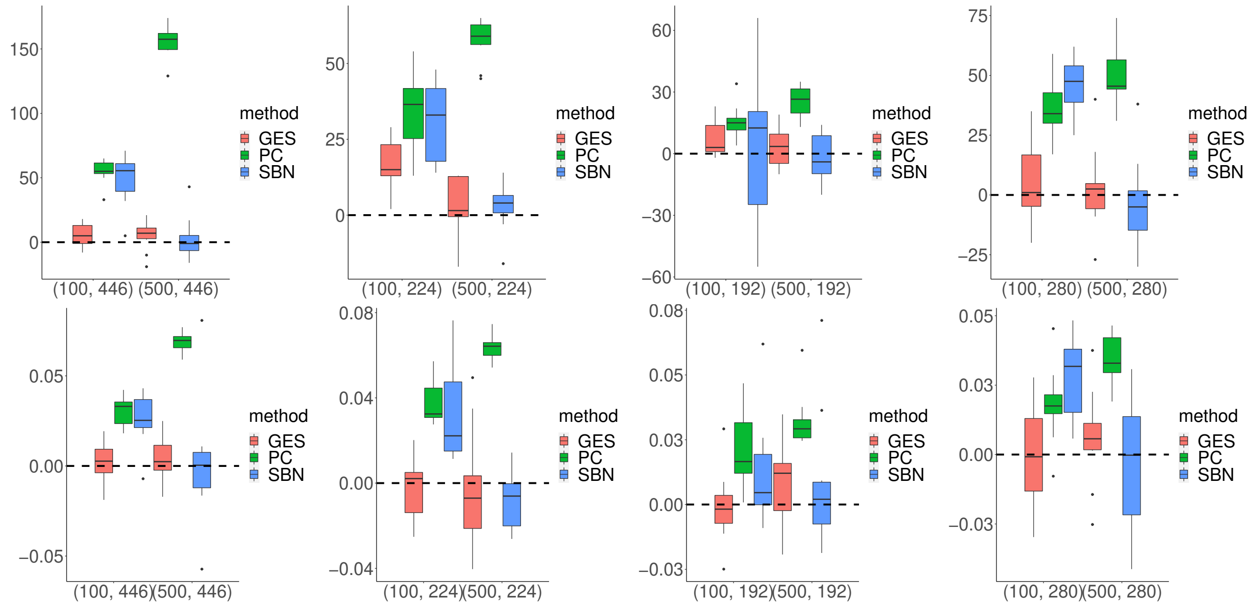


Figure 3.7: Experiments on real unsorted DAGs. Decrease in SHD (top row) and increase in JI (bottom row) via de-correlation for real networks, where the x-axis reports the value of (n, p). In each panel, the three boxplots on the left and the three on the right correspond to cases of $n < p$ and $n > p$, respectively.

In other words, the underlying undirected network may have only one connected subgraph where all individuals are dependent. This setup poses major challenge particularly for the estimation of Θ because its support becomes much larger, and as a result, we will need to impose stronger regularization in the graphical Lasso step when $n > p$. For simplicity, we focus on the setting $p > n$ so that we can still fix $\lambda_2 = 0.01$. Proceeding as before, we generated B^* from real DAGs with duplications: `Andes`, `Hailfinder`, `Barley`, and `Hepar2`.

First we assume that the DAG ordering is known and compare the BCD method against the baseline method. In three out of the four cases we considered, BCD gave better estimates of the DAG structure in terms of Jaccard index and structural Hamming distance as shown in Table 3.7. Next, without assuming a known DAG ordering, we compare the performance of GES, PC, and sparsebn before and after de-correlation. The first row in Figure 3.9 shows the increase in the normalized test data log-likelihood after de-correlation, and the increases are positive across all 10 simulations for each of the four scenarios. The second row shows

!ht

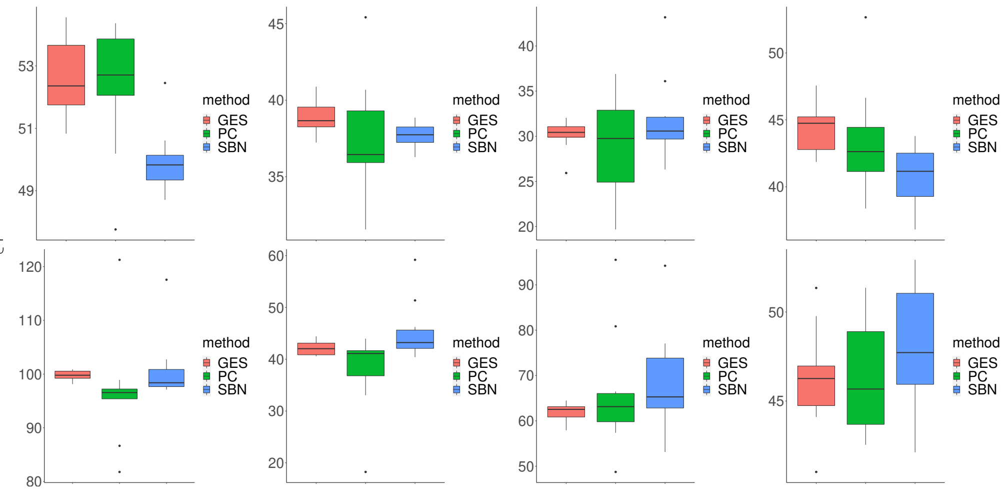


Figure 3.8: Increases in test data log-likelihood on real unsorted DAGs. Top row: $n < p$. Bottom row: $n > p$. Some outliers in the top panels are not shown for a better view of the boxplots.

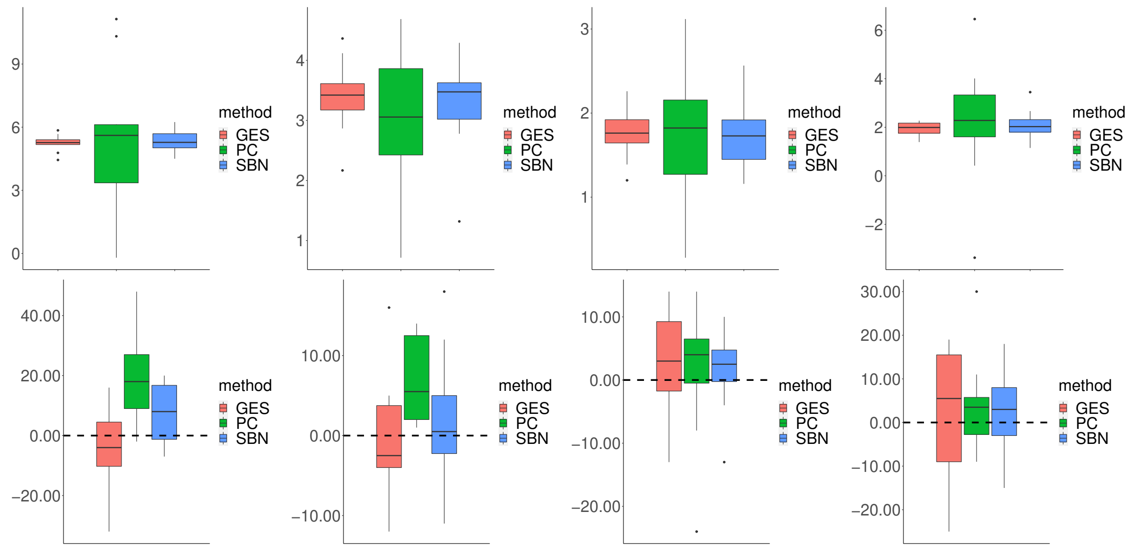


Figure 3.9: Results on real unsorted DAGs with general Θ. Top row: increase in normalized test data log-likelihood after de-correlation. Bottom row: Decrease in SHD after de-correlation.

				Before decorrelation							After decorrelation						
DAG	Θ-Network	(n, p, s_0)	Method	E	FN	TP	FDR	JI	R	SHD	E	FN	TP	FDR	JI	R	SHD
Andes (2)	facebook	(100 446, 676)	PC	602.8	371.6	304.4	0.490	0.310	183.4	853.4	551.8	362.4	313.6	0.430	0.340	197.9	798.5
			GES	359.6	355.1	320.9	0.110	0.450	92.8	486.6	339.0	361.0	315.0	0.070	0.450	96.3	481.3
			SBN	456.4	361.2	314.8	0.310	0.390	147.7	650.5	397.5	363.4	312.6	0.210	0.410	153.6	601.9
		(500 446, 676)	PC	1005.6	189.8	486.2	0.520	0.410	292.8	1002.0	846.8	185.4	490.6	0.420	0.480	304.5	846.1
			GES	508.5	200.9	475.1	0.070	0.670	74.0	308.3	504.8	201.2	474.8	0.060	0.670	72.3	303.5
			SBN	396.9	349.0	327.0	0.170	0.440	153.9	572.8	395.8	348.6	327.4	0.170	0.440	153.6	570.6
Hailfinder (4)	celegans_n306	(100, 224, 264)	PC	256.0	147.8	116.2	0.550	0.290	75.1	362.7	219.8	145.0	119.0	0.460	0.330	82.6	328.4
			GES	165.6	123.8	140.2	0.150	0.480	48.5	197.7	147.8	129.9	134.1	0.090	0.480	37.0	180.6
			SBN	200.2	137.0	127.0	0.360	0.380	66.0	276.2	169.4	138.1	125.9	0.260	0.410	63.2	244.8
		(500, 224, 264)	PC	366.2	82.9	181.1	0.510	0.400	122.1	390.1	299.3	84.6	179.4	0.400	0.470	128.0	332.5
			GES	207.1	68.0	196.0	0.050	0.710	36.8	115.9	202.0	70.7	193.3	0.040	0.710	33.9	113.3
			SBN	190.1	121.6	142.4	0.250	0.460	71.6	240.9	179.2	126.6	137.4	0.230	0.450	70.0	238.4
Barley (4)	facebook	(100, 192, 336)	PC	205.6	196.1	139.9	0.320	0.350	89.4	351.2	189.2	194.5	141.5	0.250	0.370	93.9	336.1
			GES	191.5	178.6	157.4	0.180	0.430	56.7	269.4	183.8	181.1	154.9	0.160	0.420	52.3	262.3
			SBN	242.1	178.2	157.8	0.340	0.380	70.0	332.5	248.3	172.4	163.6	0.330	0.390	72.5	329.6
		(500, 192, 336)	PC	280.1	134.9	201.1	0.280	0.480	131.7	345.6	255.0	134.6	201.4	0.210	0.520	132.1	320.3
			GES	268.3	114.4	221.6	0.170	0.580	53.7	214.8	267.4	112.7	223.3	0.160	0.590	55.1	211.9
			SBN	239.9	159.2	176.8	0.260	0.440	71.9	294.2	251.1	153.4	182.6	0.270	0.450	73.9	295.8
Hepar2 (4)	celegans_n306	(100, 280, 492)	PC	297.9	338.1	153.9	0.480	0.240	94.5	576.6	265.0	335.8	156.2	0.410	0.260	96.0	540.6
			GES	294.9	270.8	221.2	0.250	0.390	72.8	417.3	276.0	276.4	215.6	0.220	0.390	76.3	413.1
			SBN	608.0	280.5	211.5	0.480	0.290	95.3	772.3	565.5	277.2	214.8	0.410	0.320	98.3	726.2
		(500, 280, 492)	PC	395.4	254.2	237.8	0.400	0.370	145.8	557.6	351.0	252.4	239.6	0.320	0.400	148.4	512.2
			GES	398.0	155.4	336.6	0.150	0.610	77.6	294.4	391.2	157.4	334.6	0.140	0.610	82.4	296.4
			SBN	385.4	240.0	252.0	0.350	0.400	105.2	478.6	364.0	252.4	239.6	0.340	0.390	108.4	485.2

Table 3.6: Results for unordered DAGs on real network data. The number under each DAG reports the number of times it is duplicated to form a larger DAG. (Section 5.2.2)

the distribution of the decrease in SHD across 10 simulations after de-correlation. In most cases, all three methods gave much more accurate estimates after de-correlation. We defer the additional tables and figures containing more detailed results to the Supplementary Material. The above results confirm that our methods can indeed improve the accuracy in DAG estimation even Θ^* is not block-diagonal, as suggested by the theoretical results in Chapter 4.

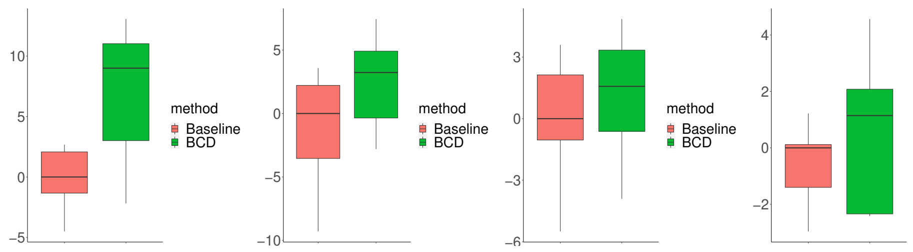


Figure 3.10: Normalized test data log-likelihood for ordered DAGs on real network data with general Θ^*. (Section 5.2.3)

DAG	Θ-Network	Method	(n, p, s_0)	E	FN	TP	FDR	JI	SHD	$\text{err}(\widehat{\Theta})$ $(\text{err}(\widehat{\Theta}^{(1)}))$
Andes	facebook	BCD	(100 446, 676)	424.1	319.9	356.1	0.155	0.478	387.9	0.01531 (0.00288)
(2)		Baseline	(100 446, 676)	422.2	326.2	349.8	0.165	0.467	398.6	—
Hailfinder	celegans_n306	BCD	(100,224,264)	122.0	163.0	101.0	0.172	0.354	184.0	0.08678 (0.08309)
(4)		Baseline	(100,224,264)	122.0	161.0	103.0	0.156	0.364	180.0	—
Barley	facebook	BCD	(100,192,336)	252.6	147.3	188.7	0.248	0.471	211.2	0.08021 (0.08555)
(4)		Baseline	(100,192,336)	249.4	155.1	180.9	0.268	0.448	223.6	—
Hepar2	celegans_n306	BCD	(100, 420, 738)	492.3	386.7	351.3	0.285	0.399	527.7	0.03725 (0.03614)
(6)		Baseline	(100, 420, 738)	490.8	397.8	340.2	0.303	0.384	548.4	—

Table 3.7: Results for ordered DAGs on real network data without block structure.

				Before decorrelation							After decorrelation						
DAG	Θ-Network	(n, p, s_0)	Method	E	FN	TP	FDR	JI	R	SHD	E	FN	TP	FDR	JI	R	SHD
			PC	566.8	339.2	336.8	0.406	0.372	205.9	775.1	552.5	334.2	341.8	0.381	0.386	211.2	756.1
Andes	facebook	(100 446, 676)	GES	379.7	321.9	354.1	0.067	0.505	101.1	448.6	375.8	322.2	353.8	0.059	0.507	108.5	452.7
(2)			SBN	457.4	322.0	354.0	0.226	0.454	154.9	580.3	443.6	324.8	351.2	0.208	0.457	155.6	572.8
			PC	354.7	393.2	221.8	0.374	0.297	130.8	656.9	349.2	394.4	220.6	0.367	0.297	130.0	653.0
Hepar2	celegans_n306	(100 420, 738)	GES	353.4	324.0	291.0	0.176	0.430	105.9	492.3	349.0	324.0	291.0	0.166	0.432	108.0	490.0
(6)			SBN	308.2	389.7	225.3	0.269	0.323	119.8	592.4	302.0	392.5	222.5	0.264	0.321	117.7	589.7
			PC	313.3	163.7	166.3	0.469	0.349	109.1	419.8	307.0	165.1	164.9	0.463	0.349	111.6	418.8
Hailfinder	celegans_n306	(100, 224, 264)	GES	187.2	159.2	170.8	0.088	0.493	70.9	246.5	183.4	161.4	168.6	0.081	0.489	68.0	244.2
(4)			SBN	215.0	169.9	160.1	0.255	0.416	86.4	311.2	207.1	171.0	159.0	0.232	0.421	90.3	309.4
			PC	275.2	229.5	190.5	0.308	0.378	122.3	436.5	268.1	229.0	191.0	0.287	0.384	123.4	429.5
Barley	facebook	(100, 192, 336)	GES	224.2	227.2	192.8	0.140	0.427	71.3	329.9	220.8	230.0	190.0	0.139	0.422	69.9	330.7
(4)			SBN	184.6	272.4	147.6	0.200	0.323	79.3	388.7	182.9	272.3	147.7	0.192	0.325	79.2	386.7

Table 3.8: Results for unordered DAGs on real network data with general Θ^*. (Section 5.2.3)

3.3 Application on Single Cell RNA Data

Gene regulatory networks (GRNs) enable biologists to examine the causal relations in gene expression during different biological processes, and are usually estimated from gene expression data. Recent advances in single-cell RNA sequencing technology has made it possible to trace cellular lineages during differentiation and to identify new cell types by measuring gene expression of thousands of individual cells. A key question arises now is whether we can discover the GRN that controls cellular differentiation and drives transitions from one cell type to another using this type of data. Such GRNs can be interpreted as causal networks among genes, where nodes correspond to different genes and a directed edge encodes a direct causal effect of one gene on another.

The RNA-seq data set used in this section can be found in NCBI's Gene Expression

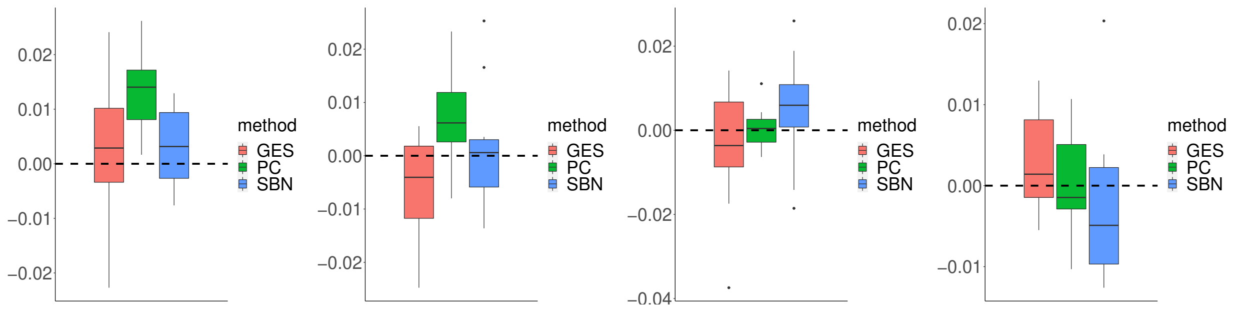


Figure 3.11: Increase in JI after de-correlation on unordered DAGs with real network structures and general Θ^*. (Section 5.2.3)

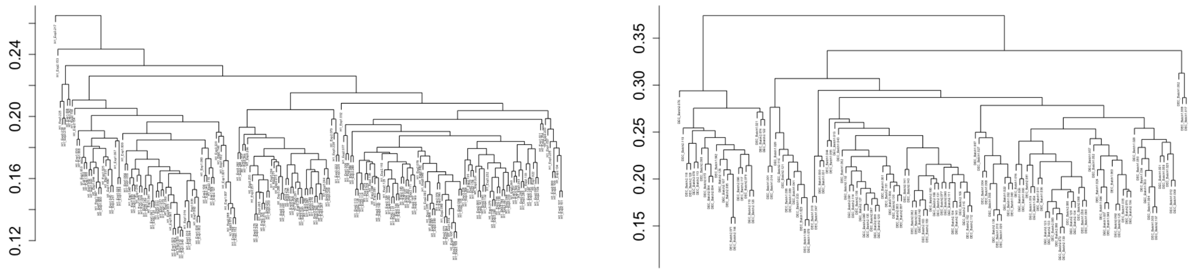


Figure 3.12: Cluster dendrograms of H1 (left) and DEC (right) cells from hierarchical clustering. The y-axis represents $1 - \rho$ and the leaf nodes are individual cells.

Omnibus and is accessible through GEO series accession number GSE75748. The data set contains gene expression measurements of around 20,000 genes from $n = 1018$ cells. Before conducting the experiment, we processed the data according to [LL18] by imputing missing values and applying log transformation. In this study, we focus on estimating a GRN among $p = 51$ target genes selected by [CLZ16], while the rest of the genes, which we call background genes, are used to estimate an undirected network for all 1018 cells.

3.3.1 Pre-estimate the Undirected Network

An essential input to Algorithm 3 is $A(G^*)$, the adjacency matrix of a known undirected network of observations (cells in this case). The 1018 cells in our data come from 7 distinct cell types (H1, H9, HFF, TB, NPC, DEC, and EC), and it is reasonable to assume that the similarity between cells of different types is minimal. Therefore, we posit that the network of the 1018 cell consists of at least 7 connected components, i.e. $N \geq 7$, where N denotes the number of diagonal blocks in Θ^* as in Section 4.1. Since it is unlikely that all cells of the same type are strongly associated with one another, we further divided each type of cells into smaller clusters by applying classical clustering algorithm on the background genes. More specifically, we randomly selected 8000 genes from the background genes and applied hierarchical clustering on each type of cells. In this experiment, we used hierarchical clustering with complete-linkage and a distance metric between two cells defined as $1 - \rho$, where ρ is the correlation between their observed gene expression levels. We verified our choice of clustering algorithm by applying it on the entire data set, and it clearly grouped the cells into 7 groups, coinciding with the 7 cell types. At the end of the hierarchical clustering step, we needed to pick cutoff levels in order to finish clustering. Because the levels of dependence among cells are quite different across cell types as shown in Figure 3.12, we picked cutoff points separately for each cell type. Generally, we chose the cutoff thresholds such that the largest cluster is smaller than $p = 51$, so λ_2 can be set to 0.01. By shifting the cutoff levels, we also obtained different number N of blocks in Θ^*. In the end, this clustering process returns an adjacency matrix A of the estimated network defined by the N clusters. In our experiments, we compared results from three choices of $N \in \{383, 519, 698\}$. The cluster size varied from 1 (singleton clusters) to 43 across the three cases.

3.3.2 Model Evaluation

In this experiment, the input to the BCD algorithm is a data matrix $X_{1018\times 51}$ and $\text{supp}(\Theta^*) \subseteq A$ estimated by the above hierarchical clustering. The matrix of error variances $\widehat{\Omega}$ was obtained following the method described in Section 2.3.2. The output is a solution path of $(\widehat{B}, \widehat{\Theta})$ for a range of λ_1's. We computed the corresponding MLEs $(\widehat{B}^{MLE}, \widehat{\Omega}^{MLE})$ given the support of each $\widehat{B}$ on the solution path. We picked the $(\widehat{B}^{MLE}, \widehat{\Omega}^{MLE})$ with the smallest BIC from the solution path as in Section 3.1 and used the corresponding $\widehat{\Theta}$ to de-correlate X.

Table 3.9 shows the results of GES, PC, and sparsebn before and after de-correlation. In each case, we computed the BIC of the estimated GRN and we used the Likelihood Ratio (LR) test to determine whether the increase in log-likelihood from de-correlation is significant or not. The LR test statistic is defined as follows:

$$\mathrm{LR} = 2(\log p(X \mid \widehat{\Theta}, \widehat{B}^{MLE}_{decor}, \widehat{\Omega}^{MLE}_{decor}) - \log p(X \mid I_n, \widehat{B}^{MLE}_{baseline}, \widehat{\Omega}^{MLE}_{baseline})) \sim \chi^2_{df},$$

where $(\widehat{B}^{MLE}_{baseline}, \widehat{\Omega}^{MLE}_{baseline})$ and $(\widehat{B}^{MLE}_{decor}, \widehat{\Omega}^{MLE}_{decor})$ denote the estimates from GES, PC, and sparsebn before and after de-correlation, respectively. The degrees of freedom of the LR statistic is

$$df = \frac{|\operatorname{supp}(\widehat{\Theta})| - n}{2} + |\operatorname{supp}(\widehat{B}^{MLE}_{decor})| - |\operatorname{supp}(\widehat{B}^{MLE}_{baseline})|.$$

In most cases, we saw significant improvements, in terms of both the BIC and the χ^2 statistic, in all three DAG estimation methods by de-correlating X using the estimated $\widehat{\Theta}$ from the BCD algorithm. This confirms the dependence among individual cells and implies that our proposed network model fits this real-world data better. Figure 3.13 shows the estimated CPDAGs after de-correlation for the case $N = 383$, which corresponds to the minimum BIC for all three methods in our experiments. It is interesting to note that a directed edge NANOG$\rightarrow$POU5F1, between the two master regulators in embryonic stem cells, appears in all three estimated CPDAGs, consistent with previously reported gene regulatory networks [CXY08, ZCM07].

model_type	N	BIC(baseline)	BIC(decor)	ll(baseline)	ll(decor)	LR χ^2	df	p-value
GES		-35036.60	-41799.39	17593.30	22667.70	10148.79	3386	0
PC	383	-14102.81	-32274.01	7102.91	17901.01	21596.19	3425	0
sparsebn		-28553.26	-37894.85	14336.13	20697.42	12722.59	3381	0
GES		-35036.60	-33312.95	17593.30	17597.47	8.34	1732	1
PC	519	-14102.81	-18738.62	7102.91	10297.31	6388.81	1753	0
sparsebn		-28553.26	-28797.87	14336.13	15324.43	1976.61	1732	0
GES		-35036.60	-34921.63	17593.30	17879.82	573.03	688	1
PC	698	-14102.81	-16959.60	7102.91	8879.30	3552.79	696	0
sparsebn		-28553.26	-29751.59	14336.13	15273.80	1875.33	677	0

Table 3.9: BIC scores and log-likelihood(ll) values from GES, PC, and sparsebn before and after de-correlation. N denotes the number of clusters among cells.

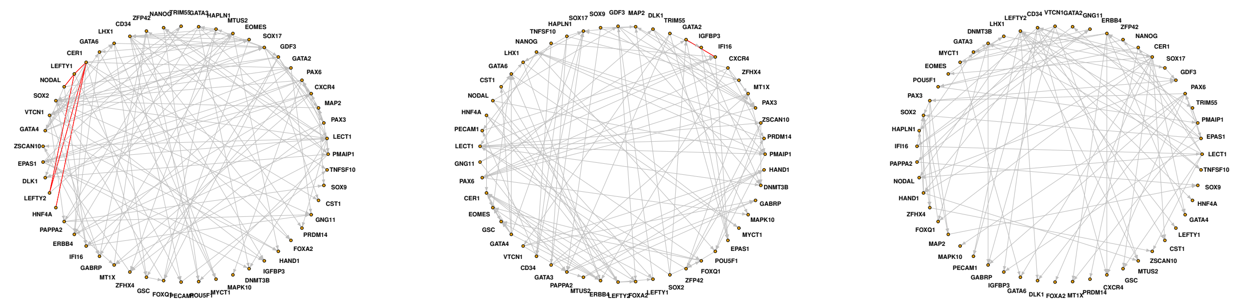


Figure 3.13: Estimated gene regulatory networks (CPDAGs) after de-correlation, with E edges and U undirected edges colored in red. Left: GES ($E = 131, U = 5$), middle: PC ($E = 119, U = 1$), right: sparsebn ($E = 90, U = 0$).

CHAPTER 4

Theory

In this section, we present our main theoretical results for Algorithm 3 assuming a true ordering is given. Section 4.1 is devoted to the error bounds of $\widehat{\Omega}$ using the natural estimator. We state our main theorem, Theorem 17, in Section 4.2, along with some important corollaries. Finally, in Section 4.3, we compare the error rates of our estimators with those in related problems. Before we start, let us introduce some additional notations used in this section.

Notations Let the errors of $\widehat{L}$ and $\widehat{\Theta}$ be defined as $\widehat{\Delta}_{chol} := \widehat{L} - L^*$ and $\widehat{\Delta}_{prec} := \widehat{\Theta} - \Theta^*$. Let $\widehat{\Delta}_j := \hat{\beta}_j - \beta_j^* \in \mathbb{R}^p$ denote the estimation error of the jth column of B^*. Let

$$\bar{\beta} = \sup_{1 \le i,j \le p} |\beta_{ij}^*|, \quad \bar{\omega} = \sup_{1 \le j \le p} \omega_j^*, \quad \bar{\psi}^2 = \sup_{1 \le j \le p} \Psi_{jj}^*,$$

where $\Psi^* = (I - B^*)^{-\top} \Omega^* (I - B^*)^{-1}$. In the proofs, we also use $X_{i\cdot}$ and $X_{\cdot j}$ to denote the ith row and jth column of X, respectively. Let $\widetilde{X} = L^* X \sim \mathcal{N}(0, \Psi^* \otimes I_n)$ and $\tilde{\varepsilon} = L^* \varepsilon$. Then the rows of $\widetilde{X}$, i.e. $\tilde{x}_i, i \in [n]$, are i.i.d from $\mathcal{N}(0, \Psi^*)$. Let m denote the maximum degree of the undirected graph G^*, which is allowed to grow with n. Following the setup in [RWR11], the set of non-zero entries in the precision matrix is denoted as $\operatorname{supp}(\Theta^*) := \{(i,j) \mid \Theta_{ij}^* \neq 0\}$. Let us use the shorthand S and S^c to denote the support and its complement in the set

$[n] \times [n]$, respectively. Define the following constants:

$$\begin{aligned} \kappa_{\Sigma^*} &:= |||\Sigma^*|||_\infty = \max_{1 \le i \le n} \sum_{j=1}^{n} |\Sigma^*_{ij}|, \\ \Gamma^*_{SS} &:= \left[\Theta^{*-1} \otimes \Theta^{*-1}\right]_{SS} \in \mathbb{R}^{(|S|+n)\times(|S|+n)}, \\ \kappa_{\Gamma^*} &:= \left|\left|\left|(\Gamma^*_{SS})^{-1}\right|\right|\right|_\infty. \end{aligned} \tag{4.1}$$

4.1 Error Bound on $\widehat{\Omega}$ under Block-diagonal Θ^*

Recall from Section 2.3 that $\widehat{\Omega}$ is pre-estimated at Step 1 in our two-step learning procedure. As we will discuss in more detail in Section 4.2, the accuracy of $\widehat{\Theta}$ obtained in Step 2 using Algorithm 3 depends on the accuracy of $\widehat{\Omega}$. Existing methods for estimating the error variance in linear models, such as the scaled Lasso [SZ12], square-root lasso [BCW11], and natural lasso [YB19], often assume independence among samples, which is not necessarily true under our network setting. However, if we assume the network of the samples is block diagonal and the samples form many small clusters, we would be able to collect independent samples from different clusters. This intuition suggests that existing methods are readily applicable in our setting to get consistent estimates of Ω^*, as long as there are enough independent clusters in the undirected network G^*.

Formally, suppose Θ^* has a block-diagonal structure defined in (2.11). Let N be the number of blocks. If we use the *natural estimator* from [YB19] (described in Section 2.3) to get $\hat{\omega}_j$, then we have the following error bound:

Lemma 15. *Let X be generated from* (2.1) *and assume Θ^* is block-diagonal with N blocks. Recall that $s = \sup_j \|\beta^*_j\|_0$. Let $\widehat{\Omega}$ be the natural estimator defined in* (2.12) *with*

$$\lambda_N = 12\bar{\psi}\bar{\omega}\left(\sqrt{\frac{2\log p}{N}} + \sqrt{\frac{2\log 2 + 4\log p}{N}}\right),$$

then with probability at least $1 + 1/p^2 - 3/p$*,*

$$\sup_{1\le j\le p}\left|\hat{\omega}_j^2 - \omega_j^{*2}\right| \le \lambda_N s\bar{\beta} + 4\bar{\omega}\sqrt{\frac{\log 2 + \log p}{N}}.$$

Lemma 15 shows that the maximum error of $\hat{\omega}_j$ is upper bounded by $C\cdot s\sqrt{\log p/N}$ where $C > 0$ is constant, and thus is consistent as long as $s\sqrt{\log p/N} \to 0$. When p is fixed and N increases, there are more diagonal blocks in Θ^*, indicating more independent samples, and thus, $\hat{\omega}_j^2$ will be more accurate. When $N = n$, Θ^* becomes a diagonal matrix and the rows of X become i.i.d. Another useful quantity for our subsequent discussion is the estimation error of $1/\hat{\omega}_j^2$. Let $r(\widehat{\Omega})$ be defined as:

$$r(\widehat{\Omega}) := \sup_{1\le j\le p}\left|\frac{1}{\hat{\omega}_j^2} - \frac{1}{\omega_j^{*2}}\right|. \tag{4.2}$$

We can easily show that the following lemma holds:

Lemma 16. *Suppose* $\frac{1}{2}\inf_{1\le j\le p}\omega_j^{*2} - \sup_{1\le j\le p}\left|\hat{\omega}_j^2 - \omega_j^{*2}\right| > b > 0$*. Then*

$$r(\widehat{\Omega}) \le \frac{1}{b^4}\sup_{1\le j\le p}\left|\hat{\omega}_j^2 - \omega_j^{*2}\right|.$$

When $s(\log p/N)^{1/2}$ is small enough, Lemma 15 implies that b can be taken as a positive constant with high probability.

4.2 Error Bounds and Consistency of $\widehat{B}^{(1)}$ and $\widehat{\Theta}^{(1)}$

Applying Algorithm 3 with a pre-estimated $\widehat{\Omega}$ as input gives us $\widehat{B}^{(t)}$ and $\widehat{\Theta}^{(t)}$. In this section, we study the error bounds and consistency of $\widehat{B}^{(t)}$ and $\widehat{\Theta}^{(t)}$. Although Algorithm 3 is computational efficient and has a desirable convergence behavior as described in Proposition 14, there are technical difficulties in establishing the consistency of $\widehat{B}^{(\infty)}$ and $\widehat{\Theta}^{(\infty)}$ after convergence, due to the dependence between $(\widehat{B}^{(t)}, \widehat{\Theta}^{(t)})$ across iterations. However, we show that as long as we have a suitable initial estimate satisfying Assumption 1, Algorithm 3 can produce consistent estimators after one iteration, i.e., $(\widehat{B}^{(1)}, \widehat{\Theta}^{(1)})$ is consistent.

Assumption 1. *There exists a constant* $0 < M \leq \sigma^2_{\min}(L^*)$*, such that the initial estimate* $\widehat{\Theta}^{(0)}$ *satisfies*

$$\|\widehat{\Theta}^{(0)} - \Theta^*\|_2 \leq M.$$

Assumption 1 states that the initial estimate $\widehat{\Theta}^{(0)}$ is inside an operator norm ball centered at the true parameter Θ^* with a radius smaller than a constant M. The constant M is less than or equal to the smallest eigenvalue of Θ^*. Assumption 1 may not be easy to verify in practice, but since it only requires $\widehat{\Theta}^{(0)}$ to be within an ℓ_2-ball of constant radius around Θ^*, it is not difficult for Assumption 1 to be met if Θ^* is sparse and normalized. Under Assumption 1 we can establish finite-sample error bounds for $(\widehat{B}^{(1)}, \widehat{\Theta}^{(1)})$. Recall that m denotes the maximum degree of the undirected graph G^* and $s = \sup_{1\leq j\leq p} \|\beta^*_j\|_0$. Define

$$\bar{R}(s,p,n) := \max\left\{6\bar{\omega}r(\widehat{\Omega}), \frac{72\bar{\omega}\bar{\psi}s}{b}\sqrt{\frac{\log p \log(\max\{n,p\})^2}{n}}\right\}, \tag{4.3}$$

which depends on $r(\widehat{\Omega})$, the error of $\widehat{\Omega}$ defined in (4.2).

Theorem 17. *Consider a sample matrix X from model* (2.1)*. Let* $\widehat{\Theta}^{(1)}, \widehat{B}^{(1)}$ *be the estimates after one iteration of the Algorithm 3, given initial estimator* $\widehat{\Theta}^{(0)}$ *satisfying Assumption 1. Suppose* $b > 0$ *as defined in Lemma 16. Pick the regularization parameters in* (2.9) *and* (2.10) *such that*

$$\lambda_n \geq 12\bar{\psi}\bar{\omega}\left(\sqrt{\frac{2\log p}{n}} + \sqrt{\frac{2\log 2 + 4\log p}{n}}\right),$$

$$\lambda_p \geq 40\sqrt{2}\sqrt{\frac{\tau \log n + \log 4}{p}} + \bar{R}(s,p,n),$$

where $\tau > 2$ *and* $r(\widehat{\Omega})$ *is defined in* (4.2). *Let* $\bar{\kappa} = \sigma_{\min}(\Psi^*)$. *Then for some positive constant* c_1, *we have*

$$\sup_j \|\hat{\beta}_j^{(1)} - \beta_j^*\|_2 \leq \frac{\sqrt{s}}{c_1 \bar{\kappa}} \lambda_n,$$

with probability at least $(1-2/p)^2 - \{1/(\exp\{n/32\}-1) + 1/n^{\tau-2} + 5n^2/\max\{n,p\}^4\}$. *If in addition* n, p *satisfy*

$$\begin{gathered} 3200 \log(4n^\tau) \max\{160, 24mC\} \leq p, \\ \max\left\{ r(\widehat{\Omega}), \frac{4s\bar{\omega}^3}{b} \sqrt{\frac{\log p \log^2 \max\{n,p\}}{n}} \right\} \leq 1/(24C), \end{gathered} \tag{4.4}$$

where $C = \max\{\kappa_{\Sigma^*}\kappa_{\Gamma^*}, \kappa_{\Sigma^*}^3 \kappa_{\Gamma^*}^2\}$, *we also have*

$$\|\widehat{\Theta}^{(1)} - \Theta^*\|_2 \leq 4\kappa_{\Gamma^*} m \lambda_p,$$

with the same probability.

We leave the detailed proof for Theorem 17 to the Appendix. The quantities κ_{Γ^*} and κ_{Σ^*} defined in (4.1) measure, respectively, the scale of the entries in Σ^* and the inverse Hessian matrix Γ_{SS}^{*-1} of the graphical Lasso log-likelihood function (2.10), and they may scale with n and p in Theorem 17. To simplify the following asymptotic results, we assume they are bounded by a constant as $n, p \to \infty$; see [RWR11] for a related discussion. In addition, assume $\bar{\kappa}, \bar{\psi}, \bar{\omega}$ stay bounded as well. Then, under the conditions in Theorem 17, we have for fixed positive constants c_2, c_3, c_4 that

$$\begin{gathered} \sup_j \|\hat{\beta}_j^{(1)} - \beta_j^*\|_2^2 \leq c_2 s \frac{\log p}{n}, \\ \|\widehat{\Theta}^{(1)} - \Theta^*\|_2 \leq c_3 m \left(\sqrt{\frac{\tau \log n}{p}} + \max\left\{ r(\widehat{\Omega}), c_4 s \sqrt{\frac{\log p \log^2 \max\{n,p\}}{n}} \right\} \right), \end{gathered} \tag{4.5}$$

with high probability. For simplicity, we assume that Θ^* consists of N blocks as in (2.11) hereafter. If $\widehat{\Omega}$ satisfies the convergence rate specified in Lemmas 15 and 16, i.e., $r(\widehat{\Omega}) \lesssim s\sqrt{\log p/N}$, then the sample constraints in (4.4) are satisfied as long as

$$m \log n \lesssim p, \quad s^2 \log p \lesssim N, \quad s^2 \log^3 \max\{n, p\} \lesssim n. \tag{4.6}$$

As a result, we have the following two asymptotic results. The first one considers the scaling $p \gg n$ under which DAG estimation is high-dimensional. The second one considers the case $n \gg p$ so that the estimation of Θ^* is a high-dimensional problem.

Corollary 18. *Suppose the sample size and the number of blocks satisfy*

$$p \gg N \log^2 p \gtrsim n \gtrsim N \gg \log p \to \infty.$$

Assume $\bar{\beta}, \bar{\omega}, \bar{\psi}, \kappa_{\Gamma^*}, \kappa_{\Sigma^*} < \infty$ as $n, p \to \infty$ and $r(\widehat{\Omega}) \lesssim s\sqrt{\log p/N}$. Then under the same assumptions as Theorem 17, we have

$$\sup_j \|\hat{\beta}_j^{(1)} - \beta_j^*\|_2^2 = O_p\left(s\frac{\log p}{n}\right),$$
$$\|\widehat{\Theta}^{(1)} - \Theta^*\|_2 = O_p\left(ms\sqrt{\frac{\log^3 p}{n}}\right).$$

Corollary 19. *Suppose the sample size and block numbers satisfy*

$$n \gg s^2 p \log p \log n \gtrsim N \gtrsim s^2 p \to \infty.$$

Assume $\bar{\beta}, \bar{\omega}, \bar{\psi}, \kappa_{\Gamma^*}, \kappa_{\Sigma^*} < \infty$ as $n, p \to \infty$ and $r(\widehat{\Omega}) \lesssim s\sqrt{\log p/N}$. Then under the same *assumptions as Theorem 17, we have*

$$\sup_j \|\hat{\beta}_j^{(1)} - \beta_j^*\|_2^2 = O_p\left(s\frac{\log p}{n}\right),$$
$$\|\widehat{\Theta}^{(1)} - \Theta^*\|_2 = O_p\left(m\sqrt{\frac{\log n}{p}}\right).$$

Remark 20. *Although we derived the consistency of $\widehat{\Theta}^{(1)}$ and $\widehat{B}^{(1)}$ in the above two corollaries under the setting where Θ^* is block-diagonal, these consistency properties still hold even when Θ^* is not block-diagonal. The main purpose of the block-diagonal setting is to provide an example where we can conveniently control the error of $\widehat{\Omega}$. But in practice Θ^* does not have to be block-diagonal. In particular, we did not assume any block-diagonal structure of Θ^* for the error bounds in Theorem 17. It can be seen that the error bound of $\widehat{B}^{(1)}$ does not depend on the error of $\widehat{\Omega}$ at all. Hence, the accuracy of $\widehat{\Omega}$ has no impact on the accuracy of $\widehat{B}^{(1)}$. The error bound of $\widehat{\Theta}^{(1)}$ in* (4.5) *is determined by the trade-off among three terms, one of which is the error rate $r(\widehat{\Omega})$ as in* (4.2)*. This is supported by our numerical results as well. In Chapter 3, we demonstrate, with both simulated and real networks where Θ^* is not block-diagonal, that our proposed BCD method can still accurately estimate Θ^* and B^* whenever a relatively accurate $\widehat{\Omega}$ is provided.*

4.3 Comparison to Other Results

If the data matrix X consists of i.i.d. samples generated from the Gaussian linear SEM (1.3), assuming the topological sort of the vertices is known, the DAG estimation problem in (2.8) is reduced to solving the standard Lasso regression in (2.9) with $\widehat{L}^{(t)} = I_n$, and thus independent of the initial $\widehat{\Theta}^{(0)}$ estimator. Under the *restricted eigenvalue condition* and letting $\lambda_n \asymp \sqrt{\log p/n}$, it is known the Lasso estimator has the following optimal rate for ℓ_2

error [GB09]:

$$\sup_j \|\hat{\beta}_j - \beta_j^*\|_2^2 = O_p\left(s\frac{\log p}{n}\right).$$

When the data are dependent, Theorem 17 shows that the estimator from Algorithm 3 can achieve the same optimal rate if we make the extra assumptions above. In particular, what we need is a reasonably good initial $\widehat{\Theta}^{(0)}$ estimate such that $\|\widehat{\Theta}^{(0)} - \Theta^*\|_2 \leq M$ for a small positive constant M.

On the other hand, if the underlying DAG is an empty graph and $\Omega^* = I_p$, the problem of estimating Θ^* can be solved using graphical Lasso in (2.10) because the data (columns in X) are i.i.d. The sample variance $\widehat{S}$ would also be an unbiased estimator of Σ^*. In this case, [RWR11] showed that

$$\|\widehat{\Theta} - \Theta^*\|_2 = O_p\left(m\sqrt{\frac{\log n}{p}}\right).$$

This results does not require knowing $\mathrm{supp}(\Theta^*)$ but assumes a *mutual incoherence* condition on the Hessian of the log likelihood function. In our case, $\widehat{S}^{(1)}$ is biased due to the accumulated errors from the previous Lasso estimation as well as $\widehat{\Omega}$. As a result, there is an extra bias term $\bar{R}(s,n,p)$ in $\|\widehat{S}^{(1)} - \Sigma^*\|_\infty$ (see Lemma 32 in Appendix 4.4.1.2) compared to the i.i.d. setting:

$$\|\widehat{S}^{(1)} - \Sigma^*\|_\infty = \bar{\delta}_f(p, n^\tau) + \bar{R}(s,n,p),$$

where $\bar{\delta}_f(p, n^\tau) \asymp \sqrt{\log n/p}$ is the classical graphical Lasso error rate, and

$$\bar{R}(n,p,s) \asymp \max\left\{r(\widehat{\Omega}), s\sqrt{\log p \log \max\{n,p\}/n}\right\}$$

depends on the estimation errors of $\widehat{B}^{(1)}$ and $\widehat{\Omega}$. When $n \gg p$ and $r(\widehat{\Omega})$ is dominated by

$\sqrt{\log(n)/p}$, we get the same rate for the ℓ_2 consistency of $\widehat{\Theta}$ (Corollary 19) under slightly more strict constraint on the sample size (4.6). If $n \ll p$, then the ℓ_2 error rate is determined by $\max\{r(\widehat{\Omega}), s(\log^3 p/n)^{1/2}\}$. Suppose Θ^* is block-diagonal. If the number of blocks N is much smaller than n, then the ℓ_2 rate will be dominated by $r(\widehat{\Omega}) \asymp s\sqrt{\log p/N}$, which could be slower than the optimal graphical Lasso rate. But that is expected due to the error introduced in the DAG estimates $\widehat{B}^{(1)}$ and $\widehat{\Omega}$.

4.4 Proofs

4.4.1 Some Auxiliary Results

Here we introduce four lemmas that we use to establish the error bounds of $\widehat{\Theta}^{(1)}$ and $\widehat{B}^{(1)}$. Let us start by deriving an upper bound on the ℓ_2 deviation of $\widehat{L}^{(0)}$ from L^* under Assumption 1.

Lemma 21. *Suppose Assumption 1 holds and let $L^*, \widehat{L}^{(t)}$ be the Cholesky factors of Θ^* and $\widehat{\Theta}^{(t)}$, respectively. Let $\widehat{\Delta}^{(t)}_{chol} = \widehat{L}^{(t)} - L^*$. Then,*

$$\|\widehat{\Delta}^{(0)}_{chol}\|_2 \leq \frac{M}{2\sigma_{\min}(L^*)},$$

where $\sigma_{\min}(L^*)$ is the smallest singular value of L^*, and M is from Assumption 1.

To generalize the basic bound on $\|\hat{\beta}^{lasso} - \beta^*_j\|_2$ from [BG11] to dependent data, we need to control the ℓ_∞-norm of an empirical process component $2X^\top\widehat{\Theta}^{(0)}\epsilon_j/n$. Let us start with the case when the data are independent. Define the following events, where $\widetilde{X} = L^*X$ represents the independent data as explained in Chapter 4:

$$\mathscr{E} := \bigcap_{k=1}^{p} \left\{ \|\widetilde{X}_k\|_2 \leq 6\bar{\psi}\sqrt{n} \right\}. \tag{4.7}$$

Then the following lemma follows from $\widetilde{X}$ being sub-Gaussian.

Lemma 22. *Let $\alpha > 2$ be an integer. If $n > 2\sqrt{2\alpha}\log p$, then the event $\mathscr{E}$ defined in* (4.7) *holds with probability at least $1 - 1/p^{\alpha-1}$.*

Next, let $\widetilde{X}_{[j-1]}$ denote the first $j-1$ columns in $\widetilde{X}$ and define the following events that depend on λ_n

$$\begin{aligned} \mathscr{T}_j &:= \left\{2\|\widetilde{X}_{[j-1]}^\top \tilde{\varepsilon}_j\|_\infty / n \leq \lambda_n\right\}, \quad j = 1, \ldots, p \\ \mathscr{T} &:= \bigcap_{j=1}^{p} \mathscr{T}_j. \end{aligned} \tag{4.8}$$

Lemma 23. *Let $\widetilde{X}_k$ consist of n i.i.d sub-Gaussian random variables with parameter $\bar{\psi}^2$ for $k = 1, \ldots, p$. If*

$$\lambda_n = 12\bar{\psi}\bar{\omega}\left(\sqrt{\frac{2\log p}{n}} + \sqrt{\frac{2\log 2 + 4\log p}{n}}\right),$$

then the probability of $\mathscr{T}$ satisfies

$$\mathbb{P}(\mathscr{T}) \geq \left(1 - \frac{1}{p}\right)^2.$$

Lemma 23 implies that if $\lambda_n \asymp \sqrt{\frac{\log p}{n}}$, then the error terms will be uniformly under control with high probability, especially when both n and p are large.

Lemma 24 (Maximal inequality)**.** *Let $x_i = (x_{i1}, \ldots, x_{ip})$ be a random vector where each element x_{ij} is sub-Gaussian with parameter $\bar{\psi}^2$, then*

$$\mathbb{P}\left(\|x_i\|_\infty \geq 2\bar{\psi}\sqrt{\log p}\right) \leq 2/p.$$

From model (2.3), it is clear that each row in $\widetilde{X}$ is sub-Gaussian with parameter $\bar{\psi}^2$. By Lemma 24, we have $\|\tilde{x}_i\|_\infty \lesssim \sqrt{\log p}$ w.h.p.

4.4.1.1 Error Bound of $\widehat{B}^{(1)}$

The estimation error bound for the classical Lasso problem where samples are i.i.d. was established by choosing a penalty coefficient that dominates the measurement error term. Specifically, as shown in Lemma 23, this can be achieved with high probability by setting $\lambda_n \asymp \sqrt{\frac{\log p}{n}}$. In order to prove the consistency of $\widehat{B}^{(1)}$ from Algorithm 3, we need to control a similar error term which depends on $\widehat{\Theta}^{(0)}$. Notably, such error can be controlled under the same rate as λ_n, see Theorem 25.

Theorem 25 (Control the empirical process)**.** *Let λ_n be the same as in Lemma 23. Suppose the initial estimator $\widehat{\Theta}^{(0)}$ satisfies Assumption 1. Then*

$$\mathbb{P}\left(\sup_{j\in[p]} 2\|X^\top \widehat{\Theta}^{(0)}\varepsilon_j\|_\infty/n \le \lambda_n\right) \ge \left(1-\frac{1}{p}\right)\left(1-\frac{2}{p}\right). \tag{4.9}$$

Next, we show that the random matrix $\widehat{L}^{(0)}X$ satisfies the Restricted Eigenvalue (RE) condition [Wai19] w.h.p. Towards that end, we define the event $\mathscr{K}$ as in Theorem 7.16 from [Wai19] given as

$$\mathscr{K} := \left\{\|\widetilde{X}\beta\|_2^2/n \ge \tilde{c}_1\|\sqrt{\Psi^*}\beta\|_2^2 - \tilde{c}_2\rho^2(\Psi^*)\frac{\log p}{n}\|\beta\|_1^2\right\}, \tag{4.10}$$

where $\rho^2(\Psi^*)$ is the maximum diagonal entry of Ψ^* and $\widetilde{X} = L^*X$ is the de-correlated data.

Lemma 26 (Restricted eigenvalue condition)**.** *Consider a random matrix $X \in \mathbb{R}^{n\times p}$, which is drawn from a $\mathcal{N}_{n\times p}(0, \Sigma^*, \Psi^*)$ distribution. Let $\widehat{\Theta}^{(0)}$ be the initial estimate of $\Theta^* = \Sigma^{*-1}$ satisfying Assumption 1, $\widehat{L}^{(0)}$ be the Cholesky factor of $\widehat{\Theta}^{(0)}$, and $\rho^2(\Psi^*)$ be the maximum diagonal entry of Ψ^*. Then under event $\mathscr{K}$ defined in* (4.10)*, there are universal positive constants $c_1 < 1 < c_2$ such that*

$$\frac{\|\widehat{L}^{(0)}X\beta\|_2^2}{n} \ge c_1\|\sqrt{\Psi^*}\beta\|_2^2 - c_2\rho^2(\Psi^*)\frac{\log p}{n}\|\beta\|_1^2, \tag{4.11}$$

for all $\beta \in \mathbb{R}^p$.

The probability of event $\mathscr{K}$ can be found in Theorem 7.16 from [Wai19] . This event is a restriction on the design matrix $\widetilde{X}$ and it holds with high probability for a variety of matrix ensembles. Note that the restricted eigenvalue (RE) condition proved in Lemma 26 implies that all sub-design matrices $[X_1], [X_1 \mid X_2], [X_1 \mid X_2 \mid X_3], \ldots, [X_1 \mid \ldots, \mid X_p]$ involved in the Lasso iteration in Algorithm 3 also satisfy the required RE condition. With Theorem 25 and Lemma 26, it is possible to prove an *oracle inequality* for the dependent Lasso problem, which yields a family of upper bounds on the estimation error.

Theorem 27 (Lasso oracle inequality)**.** *Consider the Lasso problem in* (2.9) *for* $t = 0$. *Suppose the inequality* (4.11) *and the event in* (4.9) *hold. Let* $\bar{\kappa} = \sigma_{\min}(\Psi^*)$. *For* $j \in [p]$ *and any* $\beta_j^* \in \mathbb{R}^p$, *if*

$$\lambda_n \geq 12\bar{\psi}\bar{\omega}\left(\sqrt{\frac{2\log p}{n}} + \sqrt{\frac{2\log 2 + 4\log p}{n}}\right),$$

then any optimal solution $\hat{\beta}_j^{(1)}$ *satisfies:*

$$\|\hat{\beta}_j^{(1)} - \beta_j^*\|_2^2 \leq \frac{768\lambda_n^2}{c_1^2\bar{\kappa}^2}|S| + \frac{64\lambda_n}{4c_1\bar{\kappa}}\|\beta_{j,S^c}^*\|_1 + \frac{128c_2}{c_1}\frac{\rho^2(\Psi^*)}{\bar{\kappa}}\frac{\log p}{n}\|\beta_{j,S^c}^*\|_1^2, \tag{4.12}$$

for any subset S *with cardinality* $|S| \leq \frac{c_1}{64c_2}\frac{\bar{\kappa}}{\rho^2(\Psi^*)}\frac{n}{\log p}$. *Let* $\widehat{L}^{(0)}$ *be the Cholesky factor of* $\widehat{\Theta}^{(0)}$. *Then,*

$$\|\widehat{L}^{(0)}X\left(\hat{\beta}_j^{(1)} - \beta_j^*\right)\|_2^2/n \leq 6\lambda_n\|\beta_j^*\|_1. \tag{4.13}$$

Theorem 27 implies $\sup_{j\in[p]}\|\hat{\beta}_j^{(1)} - \beta_j^*\|_2^2 \leq \frac{768\lambda_n^2}{c_1^2\bar{\kappa}^2}s \asymp s\frac{\log p}{n}$, where s is the maximum in-degree of the true DAG.

4.4.1.2 Error Bounds of $\widehat{\Theta}^{(1)}$

Recall that s denotes the maximum number of nonzero entries in β_j^* for $j \in [p]$. In order to control $\|\widehat{\Theta}^{(1)} - \Theta^*\|_2$, we need to rely on certain type of error bound on $\widehat{S}^{(1)} - \Sigma^*$, where $\widehat{S}^{(1)}$ is the sample covariance defined in (2.10) when $t = 0$. Therefore, we adopt the definition of tail condition on the sample covariance from [RWR11].

Definition 28. *(Tail conditions) We say the $n \times p$ random matrix X from model* (2.1) *satisfies tail condition $\mathcal{T}(f, v_*)$ if there exists a constant $v_* \in (0, \infty]$ and a function $f : \mathbb{N} \times (0, \infty) \to (0, \infty)$ such that for any $(i, j) \in [n] \times [n]$,*

$$\mathbb{P}\left[|\widehat{S}_{ij}^{(1)} - \Sigma_{ij}^*| \geq \delta\right] \leq 1/f(p, \delta) \quad \forall \delta \in (0, 1/v_*].$$

We require $f(p, \delta)$ to be monotonically increasing in p, so for a fixed $\delta > 0$, define the inverse function

$$\bar{p}_f(\delta; r) := \arg\max \{p \mid f(p, \delta) \leq r\}.$$

Similarly, f should be increasing in δ for each fixed p, so we define an inverse function in the second argument:

$$\bar{\delta}_f(p; r) := \arg\max \{\delta \mid f(p, \delta) \leq r\}. \tag{4.14}$$

Under the setting of a Gaussian DAG model, we can derive a sub-Gaussian tail bound.

Lemma 29. *Let X be a sample from our Gaussian DAG model* (2.1). *The sample covariance matrix*

$$\widehat{\Sigma} = \frac{1}{p} \sum_{j=1}^{p} \frac{1}{\omega_j^{2*}} \left(X_j - X\beta_j^*\right) \left(X_j - X\beta_j^*\right)^\top, \tag{4.15}$$

satisfies the tail bound

$$\mathbb{P}\left(\sup_{i,j} |\widehat{\Sigma}_{ij} - \Sigma^*_{ij}| > \delta\right) \leq 4\exp\left\{-\frac{p\delta^2}{3200}\right\},$$

for all $\delta \in (0, 40)$.

Corollary 30. *If $f(p,\delta) = 4\exp\left\{\frac{p\delta^2}{3200}\right\}$, then the inverse function $\bar{\delta}_f(p; n^\tau)$ takes the following form,*

$$\bar{\delta}_f(p; n^\tau) = 40\sqrt{2}\sqrt{\frac{\tau \log n + \log 4}{p}}.$$

Based on the tail bound in Corollary 30, we can control the sampling noise $\widehat{\Sigma} - \Sigma^*$ as in Lemma 31.

Lemma 31 (Lemma 8 in [RWR11])**.** *Define event*

$$\mathcal{A} = \left\{\|\widehat{\Sigma} - \Sigma^*\|_\infty \leq \bar{\delta}_f(p; n^\tau)\right\}, \tag{4.16}$$

where $\bar{\delta}_f(p; n^\tau) = 40\sqrt{2}\sqrt{\frac{\tau \log n + \log 4}{p}}$. For any $\tau > 2$ and (n, p) such that $\bar{\delta}_f(p; n^\tau) \leq 1/40$, we have

$$\mathbb{P}\left[\mathcal{A}^c\right] \leq \frac{1}{n^{\tau-2}} \to 0.$$

Recall $r(\widehat{\Omega})$ defined in (4.2) and the constant b defined in Lemma 16.

Lemma 32. *Suppose $b > 0$ and*

$$\sup_j \|\hat{\beta}^{(1)}_j - \beta^*_j\|_2^2 \leq c \cdot s\frac{\log p}{n}, \tag{4.17}$$

for a fixed positive constant c. Let $\widehat{W}^{(1)} := \widehat{S}^{(1)} - \Sigma^*$*. Then, we have*

$$\|\widehat{W}^{(1)}\|_\infty \le 40\sqrt{2}\sqrt{\frac{\tau \log n + \log 4}{p}} + \max\left\{6\bar{\omega} r(\widehat{\Omega}), \frac{72\bar{\omega}\bar{\psi} s}{b}\sqrt{\frac{\log p \log^2 \max\{n,p\}}{n}}\right\}$$

with probability at least $1 - 1/n^{\tau-2} - 5n^2/\max\{n,p\}^4$*.*

Theorem 33. *Assume* $\widehat{B}^{(1)}$ *satisfies* (4.17) *and* $b > 0$*. Let*

$$\bar{R}(s,p,n) = \max\left\{6\bar{\omega} r(\widehat{\Omega}), \frac{72\bar{\omega}\bar{\psi} s}{b}\sqrt{\frac{\log p \log^2 \max\{n,p\}}{n}}\right\},$$
$$\bar{\delta}_f(p; n^\tau) = 40\sqrt{2}\sqrt{\frac{\tau \log n + \log 4}{p}}.$$

Consider the graphical Lasso estimate $\widehat{\Theta}^{(1)}$ *from Algorithm 3 with* $\lambda_p = \bar{\delta}_f(p; n^\tau) + \bar{R}$ *for* $\tau > 2$*. Assume*

$$\bar{p}_f\left(1/\max\{160, 24mC\}, n^\tau\right) \le p \quad \text{and} \quad \bar{R} \le \frac{1}{24C}, \tag{4.18}$$

where $C = \max\{\kappa_{\Sigma^*}\kappa_{\Gamma^*}, \kappa_{\Sigma^*}^3\kappa_{\Gamma^*}^2\}$*. Then, with probability at least* $1 - 1/n^{\tau-2} - 5n^2/\max\{n,p\}^4$*, we have*

$$\|\widehat{\Theta}^{(1)} - \Theta^*\|_\infty \le 4\kappa_{\Gamma^*}\left(\bar{\delta}_f(p; n^\tau) + \bar{R}\right),$$
$$\|\widehat{\Theta}^{(1)} - \Theta^*\|_2 \le 4\kappa_{\Gamma^*}(m+1)\left(\bar{\delta}_f(p; n^\tau) + \bar{R}\right).$$

where m is the maximum degree of the undirected network G^**.*

4.4.2 Proofs of Main Results

We collect the proofs of our main theoretical results here, including the proofs of Theorem 13 in Section 2.2, Proposition 14 in Section 2.3, and Theorem 17, Corollary 18 and Corollary 19 in Section 4.2.

Proof of Proposition 14

Proof. First, notice that, with probability one, X has full column rank. Also, because the Cholesky factor $\widehat{L}^{(t)}$ is always positive definite for each iteration, $\widehat{L}^{(t)}X$ is in *general position* a.s. Note that the first three terms of (2.8) are differentiable (regular) and the whole function is continuous. Furthermore, solving (2.8) with respect to each variable gives a unique coordinate-wise minimum. Therefore, by Theorem 4.1 (c) in [Tse01], the block coordinate descent converges to a stationary point. $\square$

Proof of Theorem 13

Proof. By Theorem 3 from [Chi03], there exists a finite sequence of covered edge reversals in $\mathcal{G}_1$ such that at each step $\mathcal{G}_1 \backsimeq \mathcal{G}_2$ and eventually $\mathcal{G}_1 = \mathcal{G}_2$. Hence it suffices to show the result for those $\mathcal{G}_1$ and $\mathcal{G}_2$ that differ only by one edge reversal.

Before we show the main result, let us first prove that given the same initial Θ_0, the BCD algorithm will generate the same limiting estimate $(\widehat{\Theta}, \widehat{B}, \widehat{\Omega})$. This limit point is a stationary and partial optimal point for (2.5) by proposition 14. To do this, it suffices to show that the sample covariance matrices calculated from (2.4) (appear in the trace term $\mathrm{tr}(S\Theta)$) for the two networks are equal: $S_{\mathcal{G}_1} = S_{\mathcal{G}_2}$.

Suppose X_i and X_j are two nodes in the DAGs and the edges between them have opposite directions in $\mathcal{G}_1$ and $\mathcal{G}_2$. We assume the nodes follow a topological order, and let Z denote the common parents of X_i and X_j. The sample covariance matrix $S(\widehat{B}, \widehat{\Omega})$ is defined for a DAG $\mathcal{G}$ as the following:

$$S_{\mathcal{G}} = \sum_{k=1}^{p} \frac{1}{(\widehat{\omega}_k^{\mathcal{G}})^2} \varepsilon_k^{\mathcal{G}} \varepsilon_k^{\mathcal{G}\top},$$

where ε_k is the residual after projecting X_k onto its parents (given by a DAG). Also, $\widehat{\omega}_k^2 =$

$\|\varepsilon_k^{\mathcal{G}}\|_2^2/n$. In our case, we simply need to show that

$$\frac{1}{(\widehat{\omega}_i^{\mathcal{G}_1})^2}\varepsilon_i^{\mathcal{G}_1}\varepsilon_i^{\mathcal{G}_1\top} + \frac{1}{(\widehat{\omega}_j^{\mathcal{G}_1})^2}\varepsilon_j^{\mathcal{G}_1}\varepsilon_j^{\mathcal{G}_1\top} = \frac{1}{(\widehat{\omega}_i^{\mathcal{G}_2})^2}\varepsilon_i^{\mathcal{G}_2}\varepsilon_i^{\mathcal{G}_2\top} + \frac{1}{(\widehat{\omega}_j^{\mathcal{G}_2})^2}\varepsilon_j^{\mathcal{G}_2}\varepsilon_j^{\mathcal{G}_2\top}, \tag{4.19}$$

because all other terms in the summation are equal for both DAGs.

Let $X_i^\perp, X_j^\perp$ be the respective residuals in the projection of X_i and X_j to the span(Z), and $\tilde{X}_i^\perp, \tilde{X}_j^\perp$ be the normalized $X_i^\perp$ and $X_j^\perp$. Then if we let $X_i \to X_j$ in $\mathcal{G}_1$ and $X_j \to X_i$ in $\mathcal{G}_2$, we have

$$\text{LHS of (4.19)} = \tilde{X}_i^\perp \tilde{X}_i^{\perp\top} + \left(\frac{X_j^\perp - \langle X_j^\perp, \tilde{X}_i^\perp\rangle \tilde{X}_i^\perp}{\|X_j^\perp - \langle X_j^\perp, \tilde{X}_i^\perp\rangle \tilde{X}_i^\perp\|}\right)\cdot\left(\frac{X_j^\perp - \langle X_j^\perp, \tilde{X}_i^\perp\rangle \tilde{X}_i^\perp}{\|X_j^\perp - \langle X_j^\perp, \tilde{X}_i^\perp\rangle \tilde{X}_i^\perp\|}\right)^\top. \tag{4.20}$$

Denote $\langle \tilde{X}_i^\perp, \tilde{X}_j^\perp\rangle = \cos\theta$, and notice that

$$\begin{aligned}
(4.20) &= \tilde{X}_i^\perp \tilde{X}_i^{\perp\top} + \frac{\left(X_j^\perp - \langle X_j^\perp, \tilde{X}_i^\perp\rangle \tilde{X}_i^\perp\right)\cdot\left(X_j^\perp - \langle X_j^\perp, \tilde{X}_i^\perp\rangle \tilde{X}_i^\perp\right)^\top}{\|X_j^\perp\|^2 \sin^2\theta} \\
&= \tilde{X}_i^\perp \tilde{X}_i^{\perp\top} + \frac{\tilde{X}_j^\perp \tilde{X}_j^{\perp\top}}{\sin^2\theta} + \frac{\tilde{X}_i^\perp \tilde{X}_i^{\perp\top}\cos^2\theta}{\sin^2\theta} + \text{A shared term} \\
&= \frac{\tilde{X}_j^\perp \tilde{X}_j^{\perp\top} + \tilde{X}_i^\perp \tilde{X}_i^{\perp\top}}{\sin^2\theta} + \text{A shared term.}
\end{aligned} \tag{4.21}$$

Since (4.21) is symmetric in i and j, we have that LHS of (4.19) = RHS of (4.19). In other words, given the same initial $\widehat{\Theta}_0$, the iterative algorithm will generate the same sequence of $(\widehat{B}, \widehat{\Omega}, \widehat{\Theta})$, thus the same limiting point.

Note that the MLE estimates for $\mathcal{G}_1$ and $\mathcal{G}_2$ are also limiting points given some initial Θ. Let us suppose the MLE exists and $(\widehat{\Theta}_1, \widehat{B}_1), (\widehat{\Theta}_2, \widehat{B}_2)$ are the MLEs for $\mathcal{G}_1, \mathcal{G}_2$, respectively, then according to the results above we have

$$L_{\mathcal{G}_1}(\widehat{\Theta}_1, \widehat{B}_1(\mathcal{G}_1)) = L_{\mathcal{G}_2}(\widehat{\Theta}_1, \widehat{B}_1(\mathcal{G}_2)) \le L_{\mathcal{G}_2}(\widehat{\Theta}_2, \widehat{B}_2(\mathcal{G}_2)) = L_{\mathcal{G}_1}(\widehat{\Theta}_2, \widehat{B}_2(\mathcal{G}_1)).$$

Therefore,

$$L_{\mathcal{G}_1}(\widehat{\Theta}_1, \widehat{B}_1(\mathcal{G}_1)) = L_{\mathcal{G}_2}(\widehat{\Theta}_2, \widehat{B}_2(\mathcal{G}_2)).$$

Thus, all MLEs for $\mathcal{G}_1$ yield the same likelihood value which is equal to the likelihood value of any MLE for $\mathcal{G}_2$. □

Proof of Theorem 17

Proof. We first prove the consistency in $\widehat{B}^{(1)}$. Under Assumption 1, Theorem 25 shows that for the given λ_n, the empirical process term of the noises can be uniformly bounded with high probability. Therefore, in order to obtain the conclusion in Theorem 27, we only need the inequality (4.11) in Lemma 26 to hold. Since the event $\mathscr{K}$ in (4.10) holds with high probability by Theorem 7.16 in [Wai19], (4.11) holds by Lemma 26. Next, we show $\widehat{\Theta}^{(1)}$ is consistent by invoking Theorem 33. For the chosen λ_p and under the constraint on (n, p) specified in (4.4), the sample size requirement in (4.18) is satisfied. Therefore, the results follow from Theorem 33. Combining Theorem 27 and 33, we get the desired results. □

Proof of Corollary 18

Proof. The rate of $\sup_j \|\hat{\beta}_j^{(1)} - \beta_j^*\|_2^2$ follows directly from the choice of $\lambda_n \asymp \sqrt{\frac{\log p}{n}}$. Since $r(\widehat{\Omega}) = O_p\left(s\sqrt{\frac{\log p}{N}}\right)$ and $p \gg n$,

$$\begin{aligned}
\|\widehat{\Theta}^{(1)} - \Theta^*\|_2 &= O_p\left(m\left(\sqrt{\frac{\log n}{p}} + s\max\left\{\sqrt{\frac{\log p}{N}}, \sqrt{\frac{\log^3 p}{n}}\right\}\right)\right) \\
&= O_p\left(ms\max\left\{\sqrt{\frac{\log p}{N}}, \sqrt{\frac{\log^3 p}{n}}\right\}\right) \quad (n \gtrsim N) \\
&= O_p\left(ms\sqrt{\frac{\log^3 p}{n}}\right) \quad (N\log^2 p \gtrsim n).
\end{aligned}$$

□

Proof of Corollary 19

Proof. The rate of $\sup_j \|\hat{\beta}_j^{(1)} - \beta_j^*\|_2^2$ can be derived in the same way as in the proof of Corollary 18. Since $r(\widehat{\Omega}) = O_p\left(s\sqrt{\frac{\log p}{N}}\right)$ and $n \gg p$,

$$\begin{aligned}\|\widehat{\Theta}^{(1)} - \Theta^*\|_2 &= O_p\left(m\left(\sqrt{\frac{\log n}{p}} + s\max\left\{\sqrt{\frac{\log p}{N}}, \sqrt{\frac{\log p \log^2 n}{n}}\right\}\right)\right) \\ &= O_p\left(m\left(\sqrt{\frac{\log n}{p}} + s\max\left\{\sqrt{\frac{\log p}{N}}, \sqrt{\frac{\log p \log^2 n}{n}}\right\}\right)\right)\end{aligned}$$

$N \gtrsim s^2 p \implies \sqrt{\frac{\log n}{p}} \gtrsim \sqrt{\frac{s^2 \log p}{N}}$ and $n \gg s^2 p \log p \log n \implies \sqrt{\frac{\log n}{p}} \gtrsim \sqrt{\frac{s^2 \log p \log^2 n}{n}}$. Thus,

$$\|\widehat{\Theta}^{(1)} - \Theta^*\|_2 = O_p\left(m\sqrt{\frac{\log n}{p}}\right).$$

□

4.4.3 Proofs of Intermediate Results for Theorem 17

We include the proofs for all the intermediates results that lead to Theorem 17 in this section.

Proof of Theorem 25

Proof. For any $j = 1, \ldots, p$,

$$\begin{aligned}\frac{2}{n}\|X^\top \widehat{\Theta}^{(0)} \varepsilon_j\|_\infty = \frac{2}{n}\|X^\top (\Theta^* + \widehat{\Delta}_{prec}^{(0)})\varepsilon_j\|_\infty &\leq \frac{2}{n}\|\widetilde{X}^\top \tilde{\varepsilon}_j\|_\infty + \frac{2}{n}\|X^\top \widehat{\Delta}_{prec}^{(0)} \varepsilon_j\|_\infty \\ &\leq \frac{2}{n}\|\widetilde{X}^\top \tilde{\varepsilon}_j\|_\infty + \frac{2}{n}\|\widetilde{X}^\top L^{*-\top} \widehat{\Delta}_{prec}^{(0)} L^{*-1} \tilde{\varepsilon}_j\|_\infty.\end{aligned}$$

Let $\widehat{K}^{(0)} = L^{*-\top}\widehat{\Delta}_{prec}^{(0)}L^{*-1}$. Then following Assumption 1,

$$\|\widehat{K}^{(0)}\|_2 \le \|\widehat{\Delta}_{prec}^{(0)}\|_2/\sigma_{\min}^2(L^*) \le M/\sigma_{\min}^2(L^*).$$

Under event $\mathscr{E}$ defined in (4.7), for $j \in [p]$,

$$\|\widehat{K}^{(0)}\widetilde{X}_j\|_2 \le 6\bar{\psi}M\sqrt{n}/\sigma_{\min}^2(L^*) \le 6\bar{\psi}\sqrt{n}.$$

For $j \in [p]$, define the event $\overline{\mathscr{T}}_j$ as the following

$$\overline{\mathscr{T}}_j := \left\{2\|\widetilde{X}^\top \widehat{K}^{(0)}\tilde{\varepsilon}_j\|_\infty/n < \lambda_n/2\right\}.$$

Similar to the proof of Lemma 23, we can show

$$\mathbb{P}\left(\bigcup_{j=1}^{p}\overline{\mathscr{T}}_j^c \mid \mathscr{E}\right) \le \frac{1}{p},$$

and

$$\mathbb{P}\left(\overline{\mathscr{T}}\bigcap\mathscr{T}\right) \ge \mathbb{P}\left(\overline{\mathscr{T}}\bigcap\mathscr{T} \mid \mathscr{E}\right)\mathbb{P}\left(\mathscr{E}\right) \ge (1-\frac{1}{p})(1-\frac{2}{p}) \ge (1-\frac{2}{p})^2,$$

where $\mathscr{T}$ is defined in Lemma 23. □

Proof of Lemma 26

Proof. We observe that

$$\begin{aligned}
\|\widehat{L}^{(0)}X\theta\|_2/\sqrt{n} &\geq \|L^*X\theta\|_2\sqrt{n} - \|\widehat{\Delta}^{(0)}_{chol}X\theta\|_2/\sqrt{n} \\
&= \|L^*X\theta\|_2/\sqrt{n} - \|\widehat{\Delta}^{(0)}_{chol}L^{*-1}L^*X\theta\|_2/\sqrt{n} \\
&\geq \left(1 - \|\widehat{\Delta}^{(0)}_{chol}\|_2/\sigma_{\min}\left(L^*\right)\right)\|L^*X\theta\|_2/\sqrt{n} \\
&\geq \left(1 - \frac{M}{2\sigma^2_{\min}\left(L^*\right)}\right)\|\widetilde{X}\theta\|_2/\sqrt{n} \quad \text{(By Assumption 1 and Lemma 21)} \\
&\geq \frac{1}{2\sqrt{n}}\|\widetilde{X}\theta\|_2,
\end{aligned}$$

when n is sufficiently large. Since event $\mathscr{K}$ defined in (4.10) holds, by Theorem 7.16 in [Wai19], we have

$$\begin{aligned}
\frac{\|\widehat{L}^{(0)}X\theta\|_2^2}{n} &\geq \frac{1}{4}\left(\tilde{c}_1\|\sqrt{\Psi^*}\theta\|_2^2 - \tilde{c}_2\rho^2(\Psi^*)\frac{\log p}{n}\|\theta\|_1^2\right) \\
&\geq c_1\|\sqrt{\Psi^*}\theta\|_2^2 - c_2\rho^2(\Psi^*)\frac{\log p}{n}\|\theta\|_1^2.
\end{aligned}$$

□

Proof of Theorem 27

Proof. Consider the penalized negative likelihood function from (2.9):

$$\mathcal{L}(\beta_j, \lambda_n) = \frac{1}{2n}\|\widehat{L}^{(0)}X_j - \widehat{L}^{(0)}X\beta_j\|_2^2 + \lambda_n\|\beta_j\|_1.$$

For simplicity, we drop the superscript (t) in $\hat{\beta}^{(1)}$ and $\widehat{L}^{(0)}$. Let ρ stand for $\rho(\Psi^*)$, $\beta_j^* \in \mathbb{R}^p$, and $\widehat{\Delta}_j = \hat{\beta}_j - \beta_j^*$. We start from the *basic inequality* [Wai19]:

$$\mathcal{L}(\hat{\beta}_j, \lambda_n) \leq \mathcal{L}(\beta_j^*, \lambda_n) = \frac{1}{2n}\|\widehat{L}\varepsilon_j\|_2^2 + \lambda_n\|\beta_j^*\|_1.$$

After rearranging some terms,

$$0 \leq \frac{1}{2n}\|\widehat{L}X\widehat{\Delta}_j\|_2^2 \leq \frac{\varepsilon_j^\top \widehat{\Theta} X \widehat{\Delta}_j}{n} + \lambda_n \left(\|\beta_j^*\|_1 - \|\hat{\beta}_j\|_1 \right). \tag{4.22}$$

Next, for any subset $S \subseteq [p]$, we have

$$\|\beta_j^*\|_1 - \|\hat{\beta}_j\|_1 = \|\beta_{j,S}^*\|_1 + \|\beta_{j,S^c}^*\|_1 - \|\beta_{j,S}^* + \widehat{\Delta}_{j,S}\|_1 - \|\widehat{\Delta}_{j,S^c} + \beta_{j,S^c}^*\|_1. \tag{4.23}$$

Combined (4.22) with (4.23), and apply triangle and Hölder's inequalities,

$$\begin{aligned}
0 \leq \frac{1}{2n}\|\widehat{L}X\widehat{\Delta}_j\|_2^2 &\leq \frac{1}{n}\varepsilon_j^\top \widehat{\Theta} X \widehat{\Delta}_j + \lambda_n \left(\|\widehat{\Delta}_{j,S}\|_1 - \|\widehat{\Delta}_{j,S^c}\|_1 + 2\|\beta_{j,S^c}^*\|_1 \right) \\
&\leq \|X^\top \widehat{\Theta}\varepsilon_j\|_\infty / n \|\widehat{\Delta}_j\|_1 + \lambda_n \left(\|\widehat{\Delta}_{j,S}\|_1 - \|\widehat{\Delta}_{j,S^c}\|_1 + 2\|\beta_{j,S^c}^*\|_1 \right) \\
&\leq \frac{\lambda_n}{2} \left(\|\widehat{\Delta}_j\|_1 + 2\|\widehat{\Delta}_{j,S}\|_1 - 2\|\widehat{\Delta}_{j,S^c}\|_1 + 4\|\beta_{j,S^c}^*\|_1 \right) \\
&\leq \frac{\lambda_n}{2} \left[3\|\widehat{\Delta}_{j,S}\|_1 - \|\widehat{\Delta}_{j,S^c}\|_1 + 4\|\beta_{j,S^c}^*\|_1 \right], \qquad (4.24) \\
\|\widehat{\Delta}_j\|_1 &\leq 4 \left(\|\widehat{\Delta}_{j,S}\|_1 + \|\beta_{j,S^c}^*\|_1 \right).
\end{aligned}$$

This inequality implies (apply Cauchy-Schwarz inequality)

$$\|\widehat{\Delta}_j\|_1^2 \leq \left(4\|\widehat{\Delta}_{j,S}\|_1 + 4\|\beta_{j,S^c}^*\|_1 \right)^2 \leq 32 \left(|S| \ \|\widehat{\Delta}_j\|_2^2 + \|\beta_{j,S^c}^*\|_1^2 \right). \tag{4.25}$$

Next, from (4.11) and (4.25), we know,

$$\begin{aligned}
\frac{\|\widehat{L}X\widehat{\Delta}_j\|_2^2}{n} &\geq \left(c_1\bar{\kappa} - 32c_2\rho^2|S|\frac{\log p}{n} \right) \|\widehat{\Delta}_j\|_2^2 - 32c_2\rho^2 \frac{\log p}{n}\|\beta_{j,S^c}^*\|_1^2 \\
&\geq \frac{c_1\bar{\kappa}}{2}\|\widehat{\Delta}_j\|_2^2 - 32c_2\rho^2\frac{\log p}{n}\|\beta_{j,S^c}^*\|_1^2, \qquad (4.26)
\end{aligned}$$

where the last inequality comes from the condition $|S| \leq \frac{c_1}{64c_2}\frac{\bar{\kappa}}{\rho^2(\Psi^*)}\frac{n}{\log p}$. Now let's analyze the following two cases regarding (4.26):

Case 1. *Suppose that* $\frac{c_1\bar{\kappa}}{4}\|\widehat{\Delta}_j\|_2^2 \geq 32c_2\rho^2\frac{\log p}{n}\|\beta^*_{j,S^c}\|_1^2$*, then from* (4.24) *we can get*

$$\frac{c_1\bar{\kappa}}{4}\|\widehat{\Delta}_j\|_2^2 \leq \lambda_n\left(3\sqrt{|S|}\|\widehat{\Delta}_j\|_2 + 4\|\beta^*_{j,S^c}\|_1\right).$$

Solving for the zeros of this quadratic form in $\|\widehat{\Delta}_j\|_2$ *yields*

$$\|\widehat{\Delta}_j\|_2^2 \leq \frac{48\lambda_n^2}{c_1^2\bar{\kappa}^2}|S| + \frac{16\lambda_n\|\beta^*_{j,S^c}\|_1}{c_1\bar{\kappa}}.$$

Case 2. *Otherwise, we have* $\frac{c_1\bar{\kappa}}{4}\|\widehat{\Delta}_j\|_2^2 \leq 32c_2\rho^2\frac{\log p}{n}\|\beta^*_{j,S^c}\|_1^2$.

After combining the two cases, we obtain the claimed bound in (4.12). To get the prediction bound in (4.13), we first show $\|\widehat{\Delta}_j\|_1 \leq 4\|\beta^*_j\|_1$. From basic inequality, we have

$$0 \leq \frac{1}{2n}\|\widehat{L}X\widehat{\Delta}_j\|_2^2 \leq \frac{\varepsilon_j^\top\widehat{\Theta}X\widehat{\Delta}_j}{n} + \lambda_n\left(\|\beta^*_j\|_1 - \|\hat{\beta}_j\|_1\right).$$

By Hölder's inequality and Theorem 25,

$$\left|\frac{\varepsilon_j^\top\widehat{\Theta}X\widehat{\Delta}_j}{n}\right| \leq \left\|\frac{X^\top\widehat{\Theta}\varepsilon_j}{n}\right\|_\infty \|\widehat{\Delta}_j\|_1 \leq \frac{\lambda_n}{2}\left(\|\beta^*_j\|_1 + \|\hat{\beta}_j\|_1\right).$$

Combine the two inequalities above, we get

$$0 \leq \frac{3\lambda_n}{2}\|\beta^*_j\|_1 - \frac{\lambda_n}{2}\|\hat{\beta}_j\|_1,$$

which implies $\|\hat{\beta}_j\|_1 \leq 3\|\beta^*_j\|_1$. Consequently, we have

$$\|\widehat{\Delta}_j\|_1 \leq \|\hat{\beta}_j\|_1 + \|\beta^*_j\|_1 \leq 4\|\beta^*_j\|_1.$$

Return to the basic inequality, we have

$$\begin{aligned}\frac{\|\widehat{L}X\widehat{\Delta}_j\|_2^2}{2n} &\le \frac{\lambda_n}{2}\|\widehat{\Delta}_j\|_1 + \lambda_n\left(\|\beta_j^*\|_1 - \|\beta_j^* + \widehat{\Delta}_j\|_1\right)\\ &\le \frac{3}{2}\lambda_n\|\widehat{\Delta}_j\|_1 \le 6\lambda_n\|\beta_j^*\|_1.\end{aligned}$$

□

Proof of Lemma 29

Proof. The proof follows a similar approach as the proof for Lemma 1 in [RWR11]. □

Proof of Corollary 30

Proof. A little calculation using Lemma 29 and Definition 28 shows that the corresponding inverse functions for data from the Gaussian DAG model (2.1) are:

$$\bar{\delta}_f(p;r) = 40\sqrt{\frac{2\log(4r)}{p}}, \quad \text{and} \quad \bar{p}_f(\delta;r) = \frac{3200\log(4r)}{\delta^2}.$$

Setting $r = n^\tau$ yields the desired result. □

Proof of Lemma 32

Proof. Let $\widehat{S}_{ij}^{(t)}$ denote the (i,j) entry of the sample variance matrix $\widehat{S}^{(t)}$ defined in (2.10). Let $X_{i\cdot}$ and $X_{\cdot j}$ denote the ith row and jth column of X, respectively. Let $\varepsilon_{ik}^* := X_{ik} - X_{i\cdot}\beta_k^*$ where β_k^* is the kth column of B^*, $\rho_k^* = 1/\omega_k^{*2}$, $\hat{\rho}_k = 1/\hat{\omega}_k^2$, $\widehat{\Delta}_k^{(t)} := \hat{\beta}_k^{(t)} - \beta_k^*$, and $\hat{\delta}_k = \hat{\rho}_k - \rho_k^*$.

Then,

$$\begin{aligned}
\widehat{S}_{ij}^{(1)} &= \frac{1}{p}\sum_{k=1}^{p}\hat{\rho}_k\left(X_{ik}-X_{i\cdot}\hat{\beta}_k^{(1)}\right)\left(X_{jk}-X_{j\cdot}\hat{\beta}_k^{(1)}\right) \\
&= \frac{1}{p}\sum_{k=1}^{p}\hat{\rho}_k\left(\varepsilon_{ik}^{*}-X_{i\cdot}\widehat{\Delta}_k^{(1)}\right)\left(\varepsilon_{jk}^{*}-X_{j\cdot}\widehat{\Delta}_k^{(1)}\right) \\
&= \widehat{\Sigma}_{ij}+\frac{1}{p}\sum_{k=1}^{p}\rho_k^{*}\left(-\varepsilon_{ik}^{*}X_{j\cdot}\widehat{\Delta}_k^{(1)}-\varepsilon_{jk}^{*}X_{i\cdot}\widehat{\Delta}_k^{(1)}+X_{i\cdot}\widehat{\Delta}_k^{(1)}X_{j\cdot}\widehat{\Delta}_k^{(1)}\right) \\
&\quad+\frac{1}{p}\sum_{k=1}^{p}\hat{\delta}_k\left(\varepsilon_{ik}^{*}-X_{i\cdot}\widehat{\Delta}_k^{(1)}\right)\left(\varepsilon_{jk}^{*}-X_{j\cdot}\widehat{\Delta}_k^{(1)}\right) \\
&= \widehat{\Sigma}_{ij}+\frac{1}{p}\sum_{k=1}^{p}\hat{\rho}_k\left(-\varepsilon_{ik}^{*}X_{j\cdot}\widehat{\Delta}_k^{(1)}-\varepsilon_{jk}^{*}X_{i\cdot}\widehat{\Delta}_k^{(1)}+X_{i\cdot}\widehat{\Delta}_k^{(1)}X_{j\cdot}\widehat{\Delta}_k^{(1)}\right)+\frac{1}{p}\sum_{k=1}^{p}\hat{\delta}_k\varepsilon_{ik}^{*}\varepsilon_{jk}^{*}.
\end{aligned}$$

If we let $R_{ij}=\frac{1}{p}\sum_{k=1}^{p}\hat{\rho}_k\left(-\varepsilon_{ik}^{*}X_{j\cdot}\widehat{\Delta}_k^{(1)}-\varepsilon_{jk}^{*}X_{i\cdot}\widehat{\Delta}_k^{(1)}+X_{i\cdot}\widehat{\Delta}_k^{(1)}X_{j\cdot}\widehat{\Delta}_k^{(1)}\right)+\frac{1}{p}\sum_{k=1}^{p}\hat{\delta}_k\varepsilon_{ik}^{*}\varepsilon_{jk}^{*}$, we can upper bound $|R_{ij}|$ by dividing it into three terms and controlling each term separately.

Part 1.

We observe that

$$\begin{aligned}
\left|\frac{1}{p}\sum_{k=1}^{p}\hat{\rho}_k\varepsilon_{ik}^{*}X_{j\cdot}\widehat{\Delta}_k^{(1)}\right| &= \left|\frac{1}{p}\sum_{k=1}^{p}\left(\rho_k^{*}+\hat{\delta}_k\right)\varepsilon_{ik}^{*}X_{j\cdot}\widehat{\Delta}_k^{(1)}\right| \\
&\le \left|\frac{1}{p}\sum_{k=1}^{p}\rho_k^{*}\varepsilon_{ik}^{*}X_{j\cdot}\widehat{\Delta}_k^{(1)}\right|+\left|\frac{1}{p}\sum_{k=1}^{p}\hat{\delta}_k\varepsilon_{ik}^{*}X_{j\cdot}\widehat{\Delta}_k^{(1)}\right|.
\end{aligned}$$

If $\widehat{\Delta}_k^{(1)}$ and $\hat{\delta}_k$ satisfy (4.17), both ρ_k^{*} and $\hat{\delta}_k$ can be bounded by positive constants ($r(\widehat{\Omega})\ll 1$).

Define the following events:

$$\begin{aligned}
\mathcal{B}_1 &= \bigcup_{i=1}^{n} \left\{ \|\varepsilon_{i\cdot}\|_\infty \geq 6\bar{\omega}\sqrt{\log \max\{n,p\}} \right\}, \\
\mathcal{B}_2 &= \bigcup_{k=1}^{n} \left\{ \|\varepsilon^*_{k\cdot}\|_2 \geq 6\bar{\omega}\sqrt{p} \right\}, \\
\mathcal{B}_3 &= \bigcup_{k=1}^{n} \left\{ \|X_{k\cdot}\|_\infty \geq 6\bar{\psi}\sqrt{\log \max\{n,p\}} \right\}.
\end{aligned}$$

Under event $\mathcal{B}_1$, $\mathcal{B}_2$, and $\mathcal{B}_3$,

$$\begin{aligned}
\left| \frac{1}{p}\sum_{k=1}^{p} \hat{\rho}_k \varepsilon^*_{ik} X_{j\cdot}\widehat{\Delta}^{(1)}_k \right| &\leq \frac{2}{b}\sup_k |\varepsilon^*_{ik} X_{j\cdot}\widehat{\Delta}^{(1)}_k| \\
&\leq \frac{12\bar{\omega}}{b}\sqrt{\log\max\{n,p\}}\sup_k \|X_{j\cdot}\|_\infty \|\widehat{\Delta}^{(1)}_k\|_1 \quad \text{(By Hölder's Inequality and } \mathcal{B}_1\text{)} \\
&\leq \frac{12\bar{\omega}s\sqrt{\log p \log\max\{n,p\}}}{b\sqrt{n}}\|X_{j\cdot}\|_\infty \quad \text{(From } \|\widehat{\Delta}^{(1)}_k\|_1 \leq 4\sqrt{2s}\|\widehat{\Delta}^{(1)}_k\|_2\text{)} \\
&\leq \frac{72\bar{\omega}\bar{\psi}s}{b}\sqrt{\frac{\log p \log^2\max\{n,p\}}{n}} \quad \text{(By event } \mathcal{B}_3\text{)}.
\end{aligned}$$

The second last inequality comes from $\|\widehat{\Delta}^{(1)}\|_1 \leq 4\sqrt{2s}\|\widehat{\Delta}^{(1)}\|_2$ in the proof of Theorem 27.

Part 2

Notice that

$$\begin{aligned}
\left| \frac{1}{p}\sum_{k=1}^{p} \hat{\rho}_k X_{i\cdot}\widehat{\Delta}^{(1)}_k X_{j\cdot}\widehat{\Delta}^{(1)}_k \right| &\leq \frac{2}{b}\sup_k |X_{j\cdot}\widehat{\Delta}^{(1)}_k||X_{i\cdot}\widehat{\Delta}^{(1)}_k| \\
&\leq \frac{2s^2\log p}{bn}\|X_{j\cdot}\|_\infty\|X_{i\cdot}\|_\infty \\
&\leq \frac{12\bar{\psi}s^2}{b}\sqrt{\frac{\log^2 p \log^2\max\{n,p\}}{n^2}}.
\end{aligned}$$

Part 3

By Lemma 16, $\|\hat{\delta}\|_\infty = \sup_k |\hat{\rho}_k - \rho_k^*| = \sup_k |1/\hat{\omega}_k^2 - 1/\omega_k^{*2}| = r(\widehat{\Omega})$. Combining with $\mathcal{B}_2$, we have

$$\left|\frac{1}{p}\sum_{k=1}^{p}\hat{\delta}_k \varepsilon_{ik}^*\varepsilon_{jk}^*\right| \leq \frac{1}{p}\|\hat{\delta}\|_\infty \sum_{k=1}^{p}|\varepsilon_{ik}^*\varepsilon_{jk}^*| \leq \frac{1}{p}\|\delta\|_\infty \|\varepsilon_{i\cdot}^*\|_2 \|\varepsilon_{j\cdot}^*\|_2 \leq 6\bar{\omega}\|\hat{\delta}\|_\infty = 6\bar{\omega}r(\widehat{\Omega}).$$

Combine all three parts, we have

$$|R_{ij}| \leq \max\left\{6\bar{\omega}r(\widehat{\Omega}), \frac{72\bar{\omega}\bar{\psi}s}{b}\sqrt{\frac{\log p \log^2 \max\{n,p\}}{n}}\right\}.$$

Using Lemma 24 and Lemma 22, we can derive the upper bound for the probabilities of $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$:

$$\begin{aligned} \mathbb{P}(\mathcal{B}_1) &\leq 2/\max\{n,p\}^4, \\ \mathbb{P}(\mathcal{B}_2) &\leq 1/\max\{n,p\}^4 \quad \text{if } n > 2\sqrt{10}\log\max\{n,p\}, \\ \mathbb{P}(\mathcal{B}_2) &\leq 2/\max\{n,p\}^4, \\ \mathbb{P}\left(\bigcup_{l=1}^{3}\mathcal{B}_i\right) &\leq 5/\max\{n,p\}^4. \end{aligned}$$

Applying union bound,

$$\|R\|_\infty \leq \max\left\{6\bar{\omega}r(\widehat{\Omega}), \frac{72\bar{\omega}\bar{\psi}s}{b}\sqrt{\frac{\log p \log^2 \max\{n,p\}}{n}}\right\},$$

with probability at least $1 - \frac{5n^2}{\max\{n,p\}^4}$. Take event $\mathcal{A}$ from Lemma 31 into account and apply union bound one more time, we arrive at the desired conclusion. □

Proof of Theorem 33

Proof. Let $\bar{R}(s,p,n)$ and $\bar{\delta}_f(p;n^\tau)$ be defined as stated, then the monotonicity of the inverse tail function (4.14) and condition (4.18) on (n, p) implies that $\bar{\delta}_f(p; n^\tau) \leq 1/40$. Lemma 31 and Lemma 32 imply that the event $\mathcal{A}$ defined in (4.16) and the events $\mathcal{B}_1$,$\mathcal{B}_2$,$\mathcal{B}_3$ defined in the proof of Lemma 32 hold with high probability. Conditioning on these events, we have

$$\|\widehat{W}^{(1)}\|_\infty \leq \bar{\delta}_f(p; n^\tau) + \bar{R}(s, p, n).$$

Choose $\lambda_p = \bar{\delta}_f(p; n^\tau) + \bar{R}$. By Lemma 32 and condition (4.18) we have that

$$2\kappa_{\Gamma^*}\left(\|\widehat{W}^{(1)}\|_\infty + \lambda_p\right) \leq 4\kappa_{\Gamma^*}\left(\bar{\delta}_f(p; n^\tau) + \bar{R}\right) \leq \min\left\{\frac{1}{3\kappa_{\Sigma^*}m}, \frac{1}{3\kappa_{\Sigma^*}^3 2\kappa_{\Gamma^*}m}\right\}.$$

Applying Lemma 6 in [RWR11] we obtain

$$\begin{aligned}\|\widehat{\Theta}^{(1)} - \Theta^*\|_\infty &\leq 4\kappa_{\Gamma^*}\left(\bar{\delta}_f(p; n^\tau) + \bar{R}\right), \\ \|\widehat{\Theta}^{(1)} - \Theta^*\|_2 &\leq \|A\|_2\|\widehat{\Theta}^{(1)} - \Theta^*\|_\infty \leq (m+1)\|\widehat{\Theta}^{(1)} - \Theta^*\|_\infty.\end{aligned}$$

□

4.4.4 Proofs of Other Auxiliary Results

This section includes the proofs for Lemma 15 and 16 as well as the four lemmas introduced in Section 4.4.1.

Proof of Lemma 15

Proof. From Lemma 1 in [YB19] we know that if $\lambda_N \geq N^{-1}\|X^{(B)\top}\varepsilon_j^{(B)}\|_\infty$, then

$$\left|\hat{\omega}_j^2 - N^{-1}\|\varepsilon_j^{(B)}\|_2^2\right| \leq 2\lambda_N\|\beta_j^*\|_1 \leq \lambda_N s\bar{\beta}.$$

Pick $\lambda_N = 12\bar{\psi}\bar{\omega}\left(\sqrt{\frac{2\log p}{N}} + \sqrt{\frac{2\log 2 + 6\log p}{N}}\right)$, then applying the Chernoff bound for sub-

Gaussian random variables (e.g., see proof of Lemma 23) we can show that

$$\lambda_N \geq \sup_j N^{-1} \|X^{(B)\top} \varepsilon_j^{(B)}\|_\infty.$$

holds with probability at least $(1-\frac{1}{p})^2$. This proves the first inequality. To prove the second inequality, notice that

$$\begin{aligned}\sup_j \left|\hat{\omega}_j^2 - \omega_j^{*2}\right| &= \sup_j \left|\hat{\omega}_j^2 - \frac{\|\varepsilon_j^{(B)}\|_2^2}{N} + \frac{\|\varepsilon_j^{(B)}\|_2^2}{N} - \omega_j^{*2}\right| \\ &\leq \sup_j \left|\hat{\omega}_j^2 - \frac{\|\varepsilon_j^{(B)}\|_2^2}{N}\right| + \sup_j \left|\frac{\|\varepsilon_j^{(B)}\|_2^2}{N} - \omega_j^{*2}\right|.\end{aligned}$$

From χ^2 concentration inequality, (e.g. [Wai19] Example 2.11)

$$\begin{aligned}\left|\omega_j^{*2} - \frac{1}{N}\|\varepsilon_j^{(B)}\|_2^2\right| &\geq 2\sqrt{2}\bar{\omega}\sqrt{\frac{\log 2 + 3\log p}{N}} \quad \text{with probability at most } 1/p^3, \\ \sup_j \left|\omega_j^{*2} - \frac{1}{N}\|\varepsilon_j^{(B)}\|_2^2\right| &\geq 2\sqrt{2}\bar{\omega}\sqrt{\frac{\log 2 + 3\log p}{N}} \quad \text{with probability at most } 1/p^2.\end{aligned}$$

Combining all the inequalities, we can show that the second inequality holds with probability at least $(1-1/p)^2 - 1/p$. □

Proof of Lemma 16

Proof. Simply notice that

$$\sup_{1\leq j\leq p}\left|\frac{1}{\hat{\omega}_j^2} - \frac{1}{\omega_j^{*2}}\right| = \sup_{1\leq j\leq p}\left|\frac{1}{\hat{\omega}_j^2\omega_j^{*2}}\right| \sup_{1\leq j\leq p}\left|\omega_j^{*2} - \hat{\omega}_j^2\right| \leq \frac{1}{b^4}\sup_{1\leq j\leq p}|\omega_j^{*2} - \hat{\omega}_j^2|.$$

□

Proof of Lemma 21

Proof. The claim follows since

$$\begin{aligned}
\|\widehat{\Delta}_{prec}\|_2 &= \|\widehat{\Theta} - \Theta^*\|_2 \\
&= \| \left(L^* + \widehat{\Delta}_{chol}\right)^\top \left(L^* + \widehat{\Delta}_{chol}\right) - L^{*\top} L^* \|_2 \\
&= \|L^{*\top}\widehat{\Delta}_{chol} + \widehat{\Delta}_{chol}^\top L^* + \widehat{\Delta}_{chol}^\top \widehat{\Delta}_{chol}\|_2 \\
&\geq \max_{x \in \mathbb{S}^{n-1}} x^\top \left(L^{*\top}\widehat{\Delta}_{chol} + \widehat{\Delta}_{chol}^\top L^* + \widehat{\Delta}_{chol}^\top \widehat{\Delta}_{chol}\right) x \\
&\geq \max_{x \in \mathbb{S}^{n-1}} x^\top \left(L^{*\top}\widehat{\Delta}_{chol} + \widehat{\Delta}_{chol}^\top L^*\right) x \quad (\text{as } \widehat{\Delta}_{chol}^\top \widehat{\Delta}_{chol} \succcurlyeq 0.) \\
&= \|L^{*\top}\widehat{\Delta}_{chol} + \widehat{\Delta}_{chol}^\top L^*\|_2 \\
&\geq 2\sigma_{\min}\left(L^*\right) \|\widehat{\Delta}_{chol}\|_2.
\end{aligned}$$

□

Proof of Lemma 22

Proof. Notice that $\widetilde{X}_k \in \mathbb{R}^n$ is a sub-Gaussian random vector with variance smaller than $\bar{\psi}$. By Theorem 1.19 in [Rig15a], we have that

$$\mathbb{P}\left(\|\widetilde{X}_k\|_2 > 4\bar{\psi}\sqrt{n} + 2\bar{\psi}\sqrt{2\log(1/\delta)}\right) \leq \delta.$$

Setting $\delta = 1/p^\alpha$ and using union bound we obtain the desired conclusion. □

Proof of Lemma 23

Proof. Lemma 22 implies that with probability at least $1 - 1/p$,

$$\|\widetilde{X}_k\|_2 \leq 4\bar{\psi}\sqrt{n} + 2\bar{\psi}\sqrt{2\log p} \leq 6\bar{\psi}\sqrt{n}, \tag{4.27}$$

for all k. Under event the $\mathscr{E}$ defined in (4.7), $\|\widetilde{X}_{[j-1]}^\top \tilde{\varepsilon}_j\|_\infty / n$ corresponds to the absolute maximum of $j-1$ zero-mean Gaussian variables, each with variance at most $36\bar{\psi}^2\bar{\omega}^2/n$.

Next, we calculate the probability of the event $\mathscr{T} \cap \mathscr{E}$, where $\delta = 1/p^2$. We also let

$$t = \sqrt{\frac{2\log 2 + 4\log p}{n}},$$
$$\lambda_n = 12\bar{\psi}\bar{\omega}\left(\sqrt{\frac{2\log p}{n}} + t\right).$$

Because both $\widetilde{X}$ and $\tilde{\varepsilon}$ are random, we use the equivalence: $p(y) = \mathbb{E}_{p(x)}\left[p(y \mid x)\right]$ to apply the properties of fixed-design Lasso: Let $X_{[j-1]}$ denote the first $j-1$ columns in X,

$$\begin{aligned}
1 - \mathbb{P}(\mathscr{T}_j \mid \mathscr{E}) &= \mathbb{E}_{X_{[j-1]}}\mathbb{P}\left\{2\|\widetilde{X}_{[j-1]}^{\top}\tilde{\varepsilon}_j\|_\infty/n > \lambda_n \mid X_{[j-1]}, \mathscr{E}\right\} \\
&= \mathbb{E}_{X_{[j-1]}}\mathbb{P}\left\{2\|\widetilde{X}_{[j-1]}^{\top}\tilde{\varepsilon}_j\|_\infty/n > 6\bar{\psi}\bar{\omega}\left(\sqrt{\frac{2\log p}{n}} + t\right) \mid X_{[j-1]}, \mathscr{E}\right\} \\
&\leq 2\exp\left\{-\frac{nt^2}{2}\right\} = 1/p^2.
\end{aligned}$$

where in the last inequality we apply the Chernoff standard Gaussian tail bound. Hence,

$$1 - \mathbb{P}\left(\mathscr{T} \mid \mathscr{E}\right) = 1 - \mathbb{P}\left(\bigcap_{j=1}^{p} \mathscr{T}_j \mid \mathscr{E}\right) = \mathbb{P}\left(\bigcup_{j=1}^{p} \mathscr{T}_j^c \mid \mathscr{E}\right) \leq \frac{1}{p}. \tag{4.28}$$

Finally, by Lemma 22 with $\alpha = 2$ we get

$$\mathbb{P}(\mathscr{T}) \geq \mathbb{P}(\mathscr{E})\mathbb{P}\left(\mathscr{T} \mid \mathscr{E}\right) \geq \left(1 - \frac{1}{p}\right)^2. \tag{4.29}$$

□

Proof of Lemma 24

Proof. By the sub-Gaussian maximal inequality (e.g., Theorem 1.14 in [Rig15b]), we know that if $X_1, \ldots X_N$ are random variables such that $X_i \sim$ sub-Gaussian with parameter σ^2, then for any $t > 0$,

$$\mathbb{P}\left(\max_{1\leq i\leq N} |X_i| \geq t\right) \leq 2N \exp\left(-\frac{t^2}{2\sigma^2}\right).$$

Letting $t = \sqrt{4\bar{\psi}^2 \log p}$ and taking $\sigma^2 = \bar{\psi}^2$, we arrive at the desired result. $\square$

CHAPTER 5

Summary and Discussion

5.1 Summary of Contributions

In this paper our main goal is to generalize the existing Gaussian DAG model to dependent data. We proposed to model the covariance between observations by assuming a non-diagonal covariance structure of the noise vectors. This generalization is related to the semi-Markovian assumption in causal DAG models. Our main contributions include the development of a consistent structural learning method for the DAG and the sample network under sparsity assumptions and finite-sample guarantees for the estimators.

Our proposed BCD algorithm is built upon existing Lasso regression and graphical Lasso covariance estimation methods. When a topological ordering of the true DAG is known, it estimates the covariance between the observations Σ and the WAM of the DAG B in an iterative way. The method is fast and often converges in a few iterations. Our theoretical analysis shows that the estimates after one iteration are ℓ_2-consistent under various asymptotic frameworks including both $n \ll p$ and $n \gg p$, assuming a proper initialization of the precision matrix $\widehat{\Theta}^{(0)}$. The estimate of the DAG WAM $\widehat{B}^{(1)}$ achieves the optimal rate as Lasso estimators. The estimate of the precision matrix $\widehat{\Theta}^{(1)}$ achieves the same optimal rate as the graphical Lasso method when $n \gg p$ and there are sufficiently many independent subgroups within the data. Otherwise, it has a slightly worse rate due to the bias of the sample covariance matrix. When the DAG ordering is unknown, we showed the covariance Σ is invariant under permutations of the DAG nodes. Therefore, if the true DAG is sparse, our BCD algorithm can still give a good estimate of Θ which can be used to decorrelate the data. In addition to the theoretical analysis, we conducted extensive experiments on both synthetic and real data to compare our method with existing methods. When a true ordering of the DAG was given, the BCD algorithm significantly improved the structural estimation accuracy compared to the baseline method which ignored the sample dependency. When the ordering was unknown, classical DAG learning methods, such as GES, PC, and sparsebn, can all be greatly improved with respect to structural learning of CPDAGs by using our proposed de-correlation method based on the BCD algorithm. In all cases, our estimation methods under the proposed network Gaussian DAG model yielded significantly higher test data log-likelihood compared to other competing methods, indicating better predictive modeling performance.

5.2 Future Directions

There are several unexplored directions from our research and we discuss some of them here.

5.2.1 Algorithm and Optimization

In this project we primarily focused on the cases when a natural ordering of the DAG is given and the error bounds and consistency results we had are also based on this assumption. In practice, however, it can be hard to obtain the ordering in advance. The main reason that we needed a natural ordering is to reduce the search space of possible DAGs ($O(p!2^{\binom{p}{2}})$) to a much smaller space. To avoid searching through the DAG space, people have proposed approximate algorithms to search through the $p!$ orderings instead [SNM07b, TK05]. But these methods usually do not scale up to more than a hundred nodes. Recently, Niinimäki et al. proposed an MCMC method to sample partial orderings of DAGs. It would be interesting to see if we can combine partial order sampling with our method and extend the theoretical results.

Another way of bypassing the search space barrier for score-based methods is formulating the acyclicity constraint into a smooth function so that classical optimization algorithms can be directly applied. Recently, a breakthrough method called NOTEARS [ZAR18] converts the acyclicity constraint into a continuous function based on the trace of the adjacency matrix B of the candidate DAGs:

$$h(B) := \operatorname{tr}\left[(I + \alpha B \circ B)^p\right] - p = 0, \forall \alpha > 0.$$

It then minimizes a least-square loss to get $\widehat{B}$ subject to the constraint. The method is fast and the estimates, which are stationary points of the objective function, are close to the global optimal solution based on empirical studies. It might be possible to apply this idea to our DAG learning problem with dependent data.

5.2.2 Nonlinear and Discrete Data

In this dissertation, we focused on Bayesian networks whose nodes are linear functions of other nodes and the noise terms follow Gaussian distribution. In the next step, we can generalize our model to other functional and distributional assumptions under the network data framework. For example, we can start by considering nonlinear structural equations. Our model in (2.1) assumes X satisfies the following linear structural equation:

$$X := (I_n - B^\top)^{-1}E, \quad E = [\varepsilon_1 \mid \dots \mid \varepsilon_p], \quad \varepsilon_j \sim \mathcal{N}(0, \omega_j^2 \Sigma).$$

We can generalize it to

$$X := f\left((I_n - B^\top)^{-1}E\right), \quad E = [\varepsilon_1 \mid \dots \mid \varepsilon_p], \quad \varepsilon_j \sim \mathcal{N}(0, \omega_j^2 \Sigma). \tag{5.1}$$

where f is an unknown non-linear function. If f is invertible, model (5.1) is equivalent to $f^{-1}(X) = B^\top f^{-1}(X) + E$. The two functions f and its inverse f^{-1} can be any nonlinear function such as two deep neural networks. In this case, we can estimate the parameters using variational auto-encoder such as in [YCG19].

Another direction is to consider discrete or binary data. The variational method mentioned above can be directly applied after assuming a factored categorical distribution $p(X \mid E)$ with probabilities equal to $\operatorname{softmax}\left(f\left((I_n - B^\top)^{-1}E\right)\right)$. For binary data, we can also use a two-layer model with a truncation layer. Suppose the data $Y = [Y_1 \mid \dots \mid Y_p] \in \{0,1\}^{n\times p}$ are binary. Let $\eta(x;\tau) = I(x < \tau)$ be the hard thresholding function. Then we can model Y as

$$Y_j = \eta(X;\tau_j), \quad j = 1,\dots,p, \tag{5.2}$$

where X is generated by the original model (2.1). The model parameters can be estimated using the hierarchical Bayes method.

5.2.3 Alternative Ways of Estimating Ω^*

The proposed BCD algorithm minimizes the objective function in (2.8) which is different from the penalized negative log-likelihood in (2.7). Ideally, we would like to estimate (Ω, B, Θ) jointly by directly minimizing (2.7). One of the ways we explored was to reparameterize the model by letting $\rho_j = \frac{1}{\omega_j}$ and $\phi_j = \frac{\beta_j}{\omega_j}$ as in [SBV10], and optimize a reparameterized log-likelihood function:

$$\begin{aligned} L(\Phi, P, \Theta) = &- p\log\det\Theta - n\log\det P + \sum_{j=1}^{p} \|\rho_j L X_j - L X\phi_j\|_2^2 \\ &+ \rho_1(\Phi) + \rho_2(\Theta), \quad \Phi \in \mathcal{D}(\pi), \end{aligned} \tag{5.3}$$

where π is a permutation of the DAG nodes, $P = \operatorname{diag}(\rho_j)$, and $\Phi = (\phi_{ij})$. The objective function (5.3) is not exactly equivalent to (2.7) since $\rho(\Phi)$ is different from $\rho(B)$. $L(\Phi, P, \Theta)$ in (5.3) is also a biconvex function in all three estimators and thus can be minimized using block coordinate descent. The numerical results proved that by minimizing (5.3), we can also get more accurate estimators for the DAG and row correlations compared to the benchmark methods. However, we found it difficult to establish the consistency of the estimators, especially the consistency of $\widehat{P} = \widehat{\Omega}^{-1}$. In the future, it is worth investigating the conditions under which $\widehat{P}$ would be consistent when jointly estimating all three parameters.

5.2.4 Theory

Our current theoretical results are only for estimators obtained after one iteration of the BCD algorithm. However, our numerical results showed that the estimators $(\widehat{B}^{(\infty)}, \widehat{\Theta}^{(\infty)})$ obtained after convergence are also very accurate. Therefore, it would be interesting to investigate the theoretical properties of those estimators and compare them with the global optimal value. Finally, the theoretical properties we obtained for $\widehat{B}^{(1)}$ and $\widehat{\Theta}^{(1)}$ replies on a consistent $\widehat{\Omega}$ estimator. In order to control the error of $\widehat{\Omega}$, we currently assume Θ^* has a block-diagonal structure with N blocks. It would be interesting to explore other ways of deriving consistent $\widehat{\Omega}$.

REFERENCES

[AFD18] Rupesh Agrahari, Amir Foroushani, T. Roderick Docking, Linda Chang, Gerben Duns, Monika Hudoba, Aly Karsan, and Habil Zare. "Applications of Bayesian network models in predicting types of hematological malignancies." *Scientific Reports*, **8**(1):6951, 2018.

[AGZ19] Bryon Aragam, Jiaying Gu, and Qing Zhou. "Learning Large-Scale Bayesian Networks with the sparsebn Package." *Journal of Statistical Software, Articles*, **91**(11):1–38, 2019.

[Aka74] Hirotugu Akaike. "A new look at the statistical model identification." *IEEE transactions on automatic control*, **19**(6):716–723, 1974.

[AT10] Genevera I Allen and Robert Tibshirani. "Transposable regularized covariance models with an application to missing data imputation." *The Annals of Applied Statistics*, **4**(2):764, 2010.

[AZ15] Bryon Aragam and Qing Zhou. "Concave Penalized Estimation of Sparse Gaussian Bayesian Networks." *Journal of Machine Learning Research*, **16**:2273–2328, 2015.

[BCW11] A. Belloni, V. Chernozhukov, and L. Wang. "Square-root lasso: pivotal recovery of sparse signals via conic programming." *Biometrika*, **98**(4):791–806, 2011.

[BEd08] Onureena Banerjee, Laurent El Ghaoui, and Alexandre d'Aspremont. "Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data." *The Journal of Machine Learning Research*, **9**:485–516, 2008.

[BEM13] Mohsen Bayati, Murat A Erdogdu, and Andrea Montanari. "Estimating LASSO Risk and Noise Level." In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[BG11] Peter Bhlmann and Sara van de Geer. *Statistics for High-Dimensional Data: Methods, Theory and Applications.* Springer Publishing Company, Incorporated, 1st edition, 2011.

[BLN07] James Bennett, Stan Lanning, and Netflix Netflix. "The Netflix Prize." In *In KDD Cup and Workshop in conjunction with KDD*, 2007.

[Bou93] Remco R. Bouckaert. "Probabilistic network construction using the minimum description length principle." In Michael Clarke, Rudolf Kruse, and Serafín Moral, editors, *Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pp. 41–48, Berlin, Heidelberg, 1993. Springer Berlin Heidelberg.

[Bun91] Wray Buntine. “Theory refinement on Bayesian networks.” In *Uncertainty proceedings 1991*, pp. 52–60. Elsevier, 1991.

[But08] Carter T Butts et al. “Social network analysis with sna.” *Journal of statistical software*, **24**(6):1–51, 2008.

[CH92a] Gregory F. Cooper and Edward Herskovits. “A Bayesian method for the induction of probabilistic networks from data.” *Machine Learning*, **9**(4):309–347, Oct 1992.

[CH92b] Gregory F Cooper and Edward Herskovits. “A Bayesian method for the induction of probabilistic networks from data.” *Machine learning*, **9**(4):309–347, 1992.

[Chi95] David Maxwell Chickering. “A Transformational Characterization of Equivalent Bayesian Network Structures.” In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, UAI’95, p. 87–98, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[Chi03] David Maxwell Chickering. “Optimal Structure Identification with Greedy Search.” *Journal of Machine Learning Research*, **3**:507–554, March 2003.

[CLZ16] Li-Fang Chu, Ning Leng, Jue Zhang, Zhonggang Hou, Daniel Mamott, David T. Vereide, Jeea Choi, Christina Kendziorski, Ron Stewart, and James A. Thomson. “Single-cell RNA-seq reveals novel regulators of human embryonic stem cell differentiation to definitive endoderm.” *Genome Biology*, **17**(1):173, 2016.

[CXY08] Xi Chen, Han Xu, Ping Yuan, Fang Fang, Mikael Huss, Vinsensius B Vega, Eleanor Wong, Yuriy L Orlov, Weiwei Zhang, Jianming Jiang, et al. “Integration of external signaling pathways with the core transcriptional network in embryonic stem cells.” *Cell*, **133**(6):1106–1117, 2008.

[DS93] Marek J Druzdzel and Herbert A Simon. “Causality in Bayesian belief networks.” In *Uncertainty in Artificial Intelligence*, pp. 3–11. Elsevier, 1993.

[Dut99] Pierre Dutilleul. “The MLE algorithm for the matrix normal distribution.” *Journal of Statistical Computation and Simulation*, **64**(2):105–123, 1999.

[DVR08] Joachim Dahl, Lieven Vandenberghe, and Vwani Roychowdhury. “Covariance selection for nonchordal graphs via chordal embedding.” *Optimization Methods & Software*, **23**(4):501–520, 2008.

[E 78] Gideon E. Schwarz. “Estimating the Dimension of a Model.” *The Annals of Statistics*, **6**, 03 1978.

[FGH12] Jianqing Fan, Shaojun Guo, and Ning Hao. “Variance estimation using refitted cross-validation in ultrahigh dimensional regression.” *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, **74**(1):37–65, 2012.

[FHT08] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. "Sparse inverse covariance estimation with the graphical lasso." *Biostatistics*, **9**(3):432–441, 2008.

[FL01] Jianqing Fan and Runze Li. "Variable Selection via Nonconcave Penalized Likelihood and its Oracle Properties." *Journal of the American Statistical Association*, **96**(456):1348–1360, 2001.

[FZ13] Fei Fu and Qing Zhou. "Learning sparse causal Gaussian networks with experimental intervention: regularization and coordinate descent." *Journal of the American Statistical Association*, **108**(501):288–300, 2013.

[GAE14] Maxime Gasse, Alex Aussem, and Haytham Elghazel. "A Hybrid Algorithm for Bayesian Network Structure Learning with Application to Multi-Label Learning." *Expert Syst. Appl.*, **41**(15):6755–6772, November 2014.

[GB09] Sara A. van de Geer and Peter Bühlmann. "On the conditions used to prove oracle results for the Lasso." *Electronic Journal of Statistics*, **3**(none):1360 – 1392, 2009.

[GB13] Sara van de Geer and Peter Bühlmann. "l_0-PENALIZED MAXIMUM LIKELIHOOD FOR SPARSE DIRECTED ACYCLIC GRAPHS." *The Annals of Statistics*, **41**(2):536–567, 2013.

[GFZ19] Jiaying Gu, Fei Fu, and Qing Zhou. "Penalized estimation of directed acyclic graphs from discrete data." *Statistics and Computing*, **29**(1):161–176, 2019.

[HGC95] David Heckerman, Dan Geiger, and David M. Chickering. "Learning Bayesian Networks: The Combination of Knowledge and Statistical Data." *Machine Learning*, **20**(3):197–243, Sep 1995.

[HS95] David Heckerman and Ross Shachter. "Decision-theoretic foundations for causal reasoning." *Journal of Artificial Intelligence Research*, **3**:405–430, 1995.

[Hus03] Dirk Husmeier. "Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic Bayesian networks." *Bioinformatics*, **19**(17):2271–2282, 2003.

[KMC12] Markus Kalisch, Martin Mächler, Diego Colombo, Marloes H. Maathuis, and Peter Bühlmann. "Causal Inference Using Graphical Models with the R Package pcalg." *Journal of Statistical Software*, **47**(11):1–26, 2012.

[KR20] Suprateek Kundu and Benjamin B. Risk. "Scalable Bayesian matrix normal graphical models for brain functional networks." *Biometrics*, **n/a**(n/a), 2020.

[Lau96] Steffen L Lauritzen. *Graphical models*, volume 17. Clarendon Press, 1996.

[LKM96] Pedro Larranaga, Cindy MH Kuijpers, Roberto H Murga, and Yosu Yurramendi. "Learning Bayesian network structures by searching for the best ordering with genetic algorithms." *IEEE transactions on systems, man, and cybernetics-part A: systems and humans*, **26**(4):487–493, 1996.

[LL18] Wei Vivian Li and Jingyi Jessica Li. "An accurate and robust imputation method scImpute for single-cell RNA-seq data." *Nature Communications*, **9**(1):997, 2018.

[LSW20] Xiang Lyu, Will Wei Sun, Zhaoran Wang, Han Liu, Jian Yang, and Guang Cheng. "Tensor Graphical Model: Non-Convex Optimization and Statistical Inference." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **42**(8):2024–2037, 2020.

[MB06] Nicolai Meinshausen and Peter Bühlmann. "High-dimensional graphs and variable selection with the Lasso." *The Annals of Statistics*, **34**(3):1436 – 1462, 2006.

[MH12] Rahul Mazumder and Trevor Hastie. "The graphical lasso: New insights and alternatives." *Electronic journal of statistics*, **6**:2125, 2012.

[MM97] Christopher Meek and C. Meek. "Graphical models: selecting causal and statistical models." 1997.

[OP09] Tore Opsahl and Pietro Panzarasa. "Clustering in weighted networks." *Social Networks*, **31**(2):155 – 163, 2009.

[Ops09] Tore Opsahl. *Structure and Evolution of Weighted Networks.* University of London (Queen Mary College), London, UK, 2009.

[PC10] Han-Saem Park and Sung-Bae Cho. "Building mobile social network with semantic relation using Bayesian network-based life-log mining." In *2010 IEEE Second International Conference on Social Computing*, pp. 401–406. IEEE, 2010.

[Pea95] Judea Pearl. "Causal diagrams for empirical research." *Biometrika*, **82**(4):669–688, 12 1995.

[RGS17] Joseph Ramsey, Madelyn Glymour, Ruben Sanchez-Romero, and Clark Glymour. "A million variables and more: the Fast Greedy Equivalence Search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images." *International Journal of Data Science and Analytics*, **3**(2):121–129, Mar 2017.

[Rig15a] Philippe Rigollet. "18.S997: High Dimensional Statistics Lecture Notes." `https://ocw.mit.edu/courses/mathematics/18-s997-high-dimensional-statistics-spring-2015/lecture-notes/MIT18_S997S15_CourseNotes.pdf`, 2015. [Online; accessed 10-December-2020].

[Rig15b] Philippe Rigollet. "High-Dimensional Statistics - Lecture Notes.", 2015. `https://ocw.mit.edu/courses/mathematics/18-s997-high-dimensional-statistics-spring-2015/lecture-notes/MIT18_S997S15_CourseNotes.pdf`.

[Ris86] J Rissanen. "The MDL principle, universal coding, and modeling." In *1986 25th IEEE Conference on Decision and Control*, pp. 1491–1494. IEEE, 1986.

[Roo17] Teemu Roos. *Minimum Description Length Principle*, pp. 823–827. Springer US, Boston, MA, 2017.

[RWR11] Pradeep Ravikumar, Martin J. Wainwright, Garvesh Raskutti, and Bin Yu. "High-dimensional covariance estimation by minimizing l1-penalized log-determinant divergence." *Electronic Journal of Statistics*, **5**(none):935 – 980, 2011.

[SBV10] Nicolas Städler, Peter Bühlmann, and Sara Van De Geer. "l1-penalization for mixture regression models." *Test*, **19**(2):209–256, 2010.

[SCC16] Mauro Scanagatta, Giorgio Corani, Cassio P de Campos, and Marco Zaffalon. "Learning Treewidth-Bounded Bayesian Networks with Thousands of Variables." In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pp. 1462–1470. Curran Associates, Inc., 2016.

[Sch78] Gideon Schwarz. "Estimating the Dimension of a Model." *The Annals of Statistics*, **6**(2):461 – 464, 1978.

[Scu10] Marco Scutari. "Learning Bayesian Networks with the bnlearn R Package." *Journal of Statistical Software, Articles*, **35**(3):1–22, 2010.

[SGS00] Peter Spirtes, Clark N Glymour, Richard Scheines, David Heckerman, Christopher Meek, Gregory Cooper, and Thomas Richardson. *Causation, prediction, and search.* MIT press, 2000.

[SNB13] M Berkan Sesen, Ann E Nicholson, Rene Banares-Alcantara, Timor Kadir, and Michael Brady. "Bayesian networks for clinical decision support in lung cancer care." *PLoS One*, **8**(12):e82349, 2013.

[SNM07a] Mark Schmidt, Alexandru Niculescu-Mizil, and Kevin Murphy. "Learning Graphical Model Structure Using L1-regularization Paths." In *Proceedings of the 22Nd National Conference on Artificial Intelligence - Volume 2*, AAAI'07, pp. 1278–1283. AAAI Press, 2007.

[SNM07b] Mark Schmidt, Alexandru Niculescu-Mizil, Kevin Murphy, et al. "Learning graphical model structure using L1-regularization paths." In *AAAI*, volume 7, pp. 1278–1283, 2007.

[SZ12] Tingni Sun and Cun-Hui Zhang. "Scaled sparse linear regression." *Biometrika*, **99**(4):879–898, 2012.

[TBA06] Ioannis Tsamardinos, Laura E. Brown, and Constantin F. Aliferis. "The Max-Min Hill-Climbing Bayesian Network Structure Learning Algorithm." *Mach. Learn.*, **65**(1):31–78, October 2006.

[TH13] Theodoros Tsiligkaridis and Alfred O. Hero. "Covariance Estimation in High Dimensions Via Kronecker Product Expansions." *IEEE Transactions on Signal Processing*, **61**(21):5347–5360, 2013.

[THZ13] Theodoros Tsiligkaridis, Alfred O Hero III, and Shuheng Zhou. "On convergence of kronecker graphical lasso algorithms." *IEEE transactions on signal processing*, **61**(7):1743–1755, 2013.

[Tib96] Robert Tibshirani. "Regression shrinkage and selection via the lasso." *Journal of the Royal Statistical Society: Series B (Methodological)*, **58**(1):267–288, 1996.

[TK05] Marc Teyssier and Daphne Koller. "Ordering-Based Search: A Simple and Effective Algorithm for Learning Bayesian Networks." In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, UAI'05, p. 584–590, Arlington, Virginia, USA, 2005. AUAI Press.

[TKP06] Jin Tian, Changsung Kang, and Judea Pearl. "A characterization of interventional distributions in semi-Markovian causal models." In *Proceedings of The National Conference on Artificial Intelligence*, volume 21, p. 1239. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.

[TP02] Jin Tian and Judea Pearl. "A general identification condition for causal effects." In *Aaai/iaai*, pp. 567–573, 2002.

[Tse01] P. Tseng. "Convergence of a Block Coordinate Descent Method for Nondifferentiable Minimization." *Journal of Optimization Theory and Applications*, **109**(3):475–494, Jun 2001.

[VCC12] Nguyen Xuan Vinh, Madhu Chetty, Ross Coppel, and Pramod P Wangikar. "Gene regulatory network modeling via global optimization of high-order dynamic bayesian network." *BMC bioinformatics*, **13**(1):1–16, 2012.

[VP90] Thomas Verma and Judea Pearl. "Causal networks: Semantics and expressiveness." In *Machine intelligence and pattern recognition*, volume 9, pp. 69–76. Elsevier, 1990.

[Wai19] Martin J. Wainwright. *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2019.

[WL08] Tong Tong Wu and Kenneth Lange. “Coordinate descent algorithms for lasso penalized regression.” *The Annals of Applied Statistics*, **2**(1):224–244, 2008.

[WS98] Duncan J Watts and Steven H Strogatz. “Collective dynamics of ‘small-world’networks.” *nature*, **393**(6684):440, 1998.

[YAZ19] Qiaoling Ye, Arash A. Amini, and Qing Zhou. “Optimizing regularized Cholesky score for order-based learning of Bayesian networks.”, 2019. arXiv:1904.12360.

[YB19] Guo Yu and Jacob Bien. “Estimating the error variance in a high-dimensional linear model.” *Biometrika*, **106**(3):533–546, 05 2019.

[YCG19] Yue Yu, Jie Chen, Tian Gao, and Mo Yu. “DAG-GNN: DAG Structure Learning with Graph Neural Networks.” In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7154–7163. PMLR, 09–15 Jun 2019.

[YL07] Ming Yuan and Yi Lin. “Model selection and estimation in the Gaussian graphical model.” *Biometrika*, **94**(1):19–35, 2007.

[YSW04] Jing Yu, V Anne Smith, Paul P Wang, Alexander J Hartemink, and Erich D Jarvis. “Advances to Bayesian network inference for generating causal networks from observational biological data.” *Bioinformatics*, **20**(18):3594–3603, 2004.

[ZAR18] Xun Zheng, Bryon Aragam, Pradeep Ravikumar, and Eric P. Xing. “DAGs with NO TEARS: Continuous Optimization for Structure Learning.” In *Advances in Neural Information Processing Systems*, 2018.

[ZBC13] Jesse D Ziebarth, Anindya Bhattacharya, and Yan Cui. “Bayesian Network Webserver: a comprehensive tool for biological network modeling.” *Bioinformatics*, **29**(21):2801–2803, 2013.

[ZCM07] Qing Zhou, Hiram Chipperfield, Douglas A. Melton, and Wing Hung Wong. “A gene regulatory network in mouse embryonic stem cells.” *Proceedings of the National Academy of Sciences*, **104**(42):16438–16443, 2007.

[Zha10] Cun-Hui Zhang. “Nearly unbiased variable selection under minimax concave penalty.” *The Annals of Statistics*, **38**(2):894 – 942, 2010.

[Zho14] Shuheng Zhou. “Gemini: Graph estimation with matrix variate normal instances.” *The Annals of Statistics*, **42**(2):532–562, 2014.