Title
Sparse Causal Network Estimation with Experimental Intervention

Author
Fu, Fei

Publication Date
2012

Peer reviewed|Thesis/dissertation

University of California

Los Angeles

Sparse Causal Network Estimation with Experimental Intervention

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Statistics

by

Fei Fu

2012

Abstract of the Dissertation

Sparse Causal Network Estimation with Experimental Intervention

by

Fei Fu

Doctor of Philosophy in Statistics

University of California, Los Angeles, 2012

Professor Qing Zhou, Chair

Causal Bayesian networks are graphically represented by directed acyclic graphs (DAGs). Learning causal Bayesian networks from data is a challenging problem due to the size of the space of DAGs, the acyclic constraint placed on the graphical structures and the presence of equivalence classes. Most existing methods for learning Bayesian networks are either constraint-based or score-based. In this dissertation, we develop new techniques for learning sparse causal Bayesian networks via regularization.

In the first part of the dissertation, we develop an L_1-penalized likelihood approach with the adaptive lasso penalty to estimate the structure of causal Gaussian networks. An efficient blockwise coordinate descent algorithm, which takes advantage of the acyclic constraint, is proposed for seeking a local maximizer of the penalized likelihood. We establish that model selection consistency for causal network structures can be achieved with the adaptive lasso penalty and sufficient experimental interventions. Simulations are used to demonstrate the effectiveness of our method. In particular, our method shows satisfactory performance for DAGs with 200 nodes which have about 20,000 free parameters.

In the second part, we perform a principled generalization of the methodol-

ogy developed for Gaussian variables to discrete data types by replacing the linear model with the multi-logit model. The adaptive group lasso penalty is utilized that encourages sparsity pattern at the factor level. Another blockwise coordinate descent algorithm is proposed to solve the corresponding optimization problem and asymptotic theory parallel to the one developed for Gaussian Bayesian networks is established.

Finally, we illustrate a real-world application of our penalized likelihood framework using a flow cytometry data set generated from a signaling network in human immune system cells.

The dissertation of Fei Fu is approved.

Jason Ernst

Yingnian Wu

Hongquan Xu

Qing Zhou, Committee Chair

University of California, Los Angeles

2012

To my mother and father,

and to my wife, Jia,

for their love and support

Table of Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I would like to thank my advisor, Professor Qing Zhou, for his supervision and guidance during the journey of my Ph.D. studies. I benefit much from his invaluable advice, constructive comments and insightful ideas. I am extremely fortunate to have him as my advisor.

I am grateful to Professor Hongquan Xu, who as my first instructor of statistics at the graduate level introduced me to the fascinating world of statistics. I also appreciate his help in providing me with recommendation letters for various purposes. I would like to thank Professor Yingnian Wu and Professor Jason Ernst for being kind and supportive and for serving on my committee.

Special thanks to Glenda Jones, our student affairs officer, who has always been helpful in all the administrative matters I have.

Last but not least, I am grateful to my parents and my wife, Jia, for their unconditional love and support. I would not have gone this far without them in my life.

Chapter 2 and part of Chapter 4 of this dissertation are based on a version of the manuscript "Fei Fu and Qing Zhou. Learning sparse causal Gaussian networks with experimental intervention: Regularization and coordinate descent" which has been submitted for publication. The principal investigator of this manuscript is Dr. Qing Zhou.

Vita

2005	B.S., Biological Science Tsinghua University Beijing, China
2008	M.S., Molecular and Medical Pharmacology University of California, Los Angeles Los Angeles, CA
2009	C.Phil., Statistics University of California, Los Angeles Los Angeles, CA
2008–2012	Teaching Assistant, Reader and Graduate Student Researcher Department of Statistics University of California, Los Angeles Los Angeles, CA

CHAPTER 1

Introduction

1.1 Bayesian Networks

Conditional independence structures among random variables are often visualized as graphical models, where the nodes represent the variables and the edges encode the relationships among them. Depending on whether the edges are directional or not, graphical models can be classified as either directed or undirected. Bayesian networks are a special type of graphical models, whose structures are represented by directed acyclic graphs (DAGs). They have become popular probabilistic models in many research areas, including computational biology, medical sciences, image processing, speech recognition, et cetera.

In a Bayesian network, the joint probability distribution P of a set of random variables $X_1, \ldots, X_p$ can be factorized as

$$P(X_1, \ldots, X_p) = \prod_{i=1}^{p} P(X_i | \Pi_i^{\mathcal{G}}), \tag{1.1}$$

where $\Pi_i^{\mathcal{G}} \subseteq \{X_1, \ldots, X_p\} \setminus \{X_i\}$ is called the set of parents of X_i. If X_i does not have any parents, then $\Pi_i^{\mathcal{G}} = \varnothing$. We can construct a DAG $\mathcal{G} = (V, E)$ to represent the structure of a Bayesian network. Here, $V = \{1, \ldots, p\}$ denotes the set of nodes in the graph, where the i^{th} node in V corresponds to X_i. For simplicity, we use X_i and i interchangeably to represent the i^{th} node and the i^{th} variable. The set of edges $E = \{(i, j) : X_i \in \Pi_j^{\mathcal{G}}\} \subseteq V \times V$ and an edge $(i, j) \in E$ is written as $i \to j$.

The structure of $\mathcal{G}$ must be acyclic so that (1.1) is a well-defined joint distribution. By convention, if a probability distribution P satisfies (1.1) with respect to a DAG $\mathcal{G}$, P is said to factorize according to $\mathcal{G}$. In a DAG $\mathcal{G}$, descendants of a node X_i is the set of nodes to which there is a directed path originating from X_i. Most of the time in this dissertation, $\mathcal{G} = (V, E)$ is used to refer to a Bayesian network (i.e., a DAG). However, we occasionally use $\mathcal{G}$ for an arbitrary graph, which may contain undirected edges. An edge $(i, j) \in E$ is called undirected (or bidirected) if $(j, i) \in E$ as well and is written as $i - j$.

If a node i in a graph $\mathcal{G}$ is connected to a node j by either a directed or undirected edge, i is said to be adjacent to j (or is a neighbor of j) and the variable X_i is said to be adjacent to the variable X_j (or is a neighbor of X_j). The adjacency set of the variable X_i consists of all variables adjacent to X_i in $\mathcal{G}$ and is denoted by $\boldsymbol{A}(\mathcal{G}, X_i)$.

A mathematically convenient representation of the structure of a graph $\mathcal{G} = (V, E)$ is the adjacency matrix, which is a $p \times p$ matrix $\mathbf{A} = (a_{ij})_{p \times p}$ such that

$$a_{ij} = \begin{cases} 1, & \text{if } (i, j) \in E, \\ 0, & \text{otherwise.} \end{cases}$$

The skeleton of a graph $\mathcal{G}$ is the undirected graph obtained from removing the directions of all directed edges in $\mathcal{G}$, which is represented by a symmetric adjacency matrix.

Due to its directed acyclic nature, every Bayesian network $\mathcal{G}$ has at least one linear ordering $\sqsubset$ of all its nodes, known as a topological sort of $\mathcal{G}$, such that $i \prec j$ in $\sqsubset$ if $i \in \Pi_j^{\mathcal{G}}$. Efficient algorithm exists to topologically sort a DAG (Cormen et al. 2009).

1.2 Conditional Independence and d-Separation

If $\mathcal{G}$ is a Bayesian network, it is easy to see from (1.1) that the joint probability distribution P and the DAG $\mathcal{G}$ satisfy the Markov condition: each variable X_i is conditionally independent of all its non-descendants with respect to P given its parent set $\Pi_i^{\mathcal{G}}$ (Pearl 1988; Spirtes et al. 1993). This can be used as an equivalent definition for Bayesian networks.

Given a Bayesian network $\mathcal{G}$, the Markov condition directly encodes some independence relations of the probability distribution P, which in turn may entail others in the sense that every probability distribution satisfying the Markov condition also has these further independencies (Spirtes et al. 1993). A graphical test for these independence relations is d-separation (Pearl 1988).

Definition 1.1. In a DAG $\mathcal{G}$, two sets of nodes $\boldsymbol{A}$ and $\boldsymbol{B}$ are said to be d-separated by a set of nodes $\boldsymbol{Z}$, denoted by $\boldsymbol{D}_{\mathcal{G}}(\boldsymbol{A}, \boldsymbol{B}|\boldsymbol{Z})$, if all paths between each node in $\boldsymbol{A}$ and $\boldsymbol{B}$ are blocked by $\boldsymbol{Z}$. A path is blocked by $\boldsymbol{Z}$ if and only if there are nodes x, y, z on it such that:

1. they form a chain $x \to z \to y$ and $z \in \boldsymbol{Z}$,
2. they form a fork $x \leftarrow z \to y$ and $z \in \boldsymbol{Z}$,
3. they form a collider $x \to z \leftarrow y$ and neither z nor its descendants are in $\boldsymbol{Z}$.

The set of nodes that d-separates $\boldsymbol{A}$ and $\boldsymbol{B}$ is denoted by $\boldsymbol{S}(\boldsymbol{A}, \boldsymbol{B})$.

Verma and Pearl (1988) showed that d-separation implies conditional independence as stated in the following theorem.

Theorem 1.1. *If two sets of nodes $\boldsymbol{A}$ and $\boldsymbol{B}$ are d-separated by $\boldsymbol{Z}$ in a Bayesian network $\mathcal{G}$, $\boldsymbol{A}$ and $\boldsymbol{B}$ are conditionally independent in P given $\boldsymbol{Z}$, which is denoted by $\boldsymbol{I}_P(\boldsymbol{A}, \boldsymbol{B}|\boldsymbol{Z})$.*

However, the converse of Theorem 1.1 is not always true. In general, a probability distribution P on a Bayesian network $\mathcal{G}$ may contain other conditional independence relations beyond those identifiable by the d-separation criterion. Hence, the following definition of faithfulness was proposed (Spirtes et al. 1993). This faithfulness assumption is needed for the PC algorithm outlined in Algorithm 1.1, which we will introduce in Section 1.4.

Definition 1.2. A distribution P and a DAG $\mathcal{G}$ are said to be faithful to each other if all and only the conditional independence relations true in P are entailed by the Markov condition applied to $\mathcal{G}$.

It follows immediately from Definition 1.2 that if the joint probability distribution P is faithful to the Bayesian network $\mathcal{G}$,

$$\boldsymbol{D}_{\mathcal{G}}(\boldsymbol{A}, \boldsymbol{B}|\boldsymbol{Z}) \Leftrightarrow \boldsymbol{I}_P(\boldsymbol{A}, \boldsymbol{B}|\boldsymbol{Z}).$$

At first, the faithfulness assumption might seem to be quite restrictive. However, Meek (1995b) provided some arguments for the justification of the faithfulness assumption. He proved that the following claim holds for the multivariate normal family of distributions as well as the multinomial family of distributions: the set of distributions that are not faithful to a DAG $\mathcal{G}$ has Lebesgue measure zero in the space of distributions associated with $\mathcal{G}$.

1.3 Observational Equivalence

Theorem 1.1 in the last section states that the structure $\mathcal{G}$ of a Bayesian network itself represents a set of independence relations that must be satisfied by any distribution P that factorizes according to $\mathcal{G}$. These independence relations are encoded by the set of d-separation statements that hold in $\mathcal{G}$. Two DAGs $\mathcal{G}$ and $\mathcal{G}'$ are called *independence equivalent* if they represent identical independence as-

sertions (Chickering 2002). On the other hand, $\mathcal{G}$ and $\mathcal{G}'$ are called *distributionally equivalent* if every distribution representable by $\mathcal{G}$ via (1.1) can be represented by $\mathcal{G}'$ as well, and vice versa (Chickering 2002). For the multivariate normal family of distributions and the multinomial family of distributions usually assumed in the literature, independence equivalence and distributional equivalence coincide. We therefore do not distinguish these two notions of equivalence in this dissertation.

Verma and Pearl (1990) provided a graphical characterization of equivalent DAGs. Before we state the theorem, the concept of v-structure needs to be defined. A v-structure in a DAG $\mathcal{G}$ is an ordered triple of nodes $\langle i, j, k\rangle$ such that $i \to j \leftarrow k$ and i, k are not adjacent in $\mathcal{G}$. The following theorem was proved by Verma and Pearl (1990).

Theorem 1.2. *Two DAGs $\mathcal{G}$ and $\mathcal{G}'$ are equivalent if and only if they have the same skeletons and the same v-structures.*

We use $\mathcal{G} \sim \mathcal{G}'$ to denote that two DAGs $\mathcal{G}$ and $\mathcal{G}'$ are equivalent. The relation $\sim$ defines a set of equivalence classes in the space of DAGs on the same set of nodes. It follows from the definition of equivalence that equivalent DAGs cannot be distinguished even with an infinite amount of observational data, a phenomenon known as 'observational equivalence'. It is one of the reasons why estimating Bayesian networks from data is challenging.

1.4 Structure Learning of Bayesian Networks

The primary contribution of this dissertation is the development of new techniques for estimating Bayesian networks from data. In statistics, learning the structure of a Bayesian network from data is an important and difficult problem. The challenges are three folds. First, the number of DAGs grows super-exponentially in the number of nodes (Robinson 1973). The exact number of DAGs can be

calculated using the following recursive equation (Robinson 1973):

$$\begin{cases} a_p = \sum_{i=1}^{p} (-1)^{i-1} \binom{p}{i} 2^{i(p-i)} a_{p-i}, \\ a_0 = 1. \end{cases}$$

Second, as mentioned in the last section, it may not be possible to identify the structure $\mathcal{G}$ of the true Bayesian network from observational data generated by $\mathcal{G}$, regardless of the sample size. Restricting ourselves only to the space of equivalence classes may not be promising either as the results of Gillispie and Perlman (2001) suggest that the number of equivalence classes also grows super-exponentially in the number of nodes. Third, the acyclic nature of DAGs may increase computational burdens depending on the specific method chosen for estimating Bayesian networks. Despite these difficulties, substantial amount of research has been devoted to the structure learning problem of Bayesian networks and many methods have been proposed. These methods can be roughly classified into two primary approaches.

The constraint-based approach relies on a set of conditional independence tests. We have shown in Section 1.2 the equivalence of d-separation and conditional independencies using the faithfulness assumption. Then, it is not difficult to show the following theorem in Spirtes et al. (1993), which is exploited by many constraint-based algorithms.

Theorem 1.3. *If a distribution P is faithful to a DAG $\mathcal{G}$, there is an edge between node i and j if and only if X_i and X_j are conditionally dependent in P given every $\boldsymbol{Z} \subseteq V \backslash \{X_i, X_j\}$.*

A well-known example in this category is the PC algorithm proposed by Spirtes et al. (1993). It works by first estimating the skeleton of a DAG using a set of conditional independence tests, then identifying v-structures in the skeleton and finally trying to orient the remaining edges such that no new conditional indepen-

dencies and no cycles are introduced. A complete outline of the PC algorithm is given in Algorithm 1.1 (Spirtes et al. 1993; Meek 1995a; Pearl 2000; Kalisch and Bühlmann 2007). Recently, Kalisch and Bühlmann (2007) considered the problem of estimating Bayesian networks with the PC algorithm and proposed an efficient implementation suitable for estimating sparse high-dimensional DAGs. They also proved the asymptotic consistency of the PC algorithm in that setting. We chose to use the PC algorithm to benchmark our algorithms developed in this dissertation due to its capability to handle both continuous and discrete data types, the availability of an efficient implementation of the PC algorithm and its asymptotic properties.

The second approach to learning Bayesian networks is score-based, which attempts to find a DAG that maximizes some scoring function through a certain search strategy (Cooper and Herskovits 1992; Lam and Bacchus 1994; Heckerman et al. 1995) or sample DAGs from a Bayesian posterior distribution (Madigan and York 1995; Friedman and Koller 2003; Ellis and Wong 2008). It was shown by Chickering (1996) and Chickering et al. (2004) under certain conditions that searching a high-scoring Bayesian network is NP-hard. The scoring functions that have been employed include several Bayesian Dirichlet metrics (Buntine 1991; Cooper and Herskovits 1992; Heckerman et al. 1995), Bayesian Information Criterion (Chickering and Heckerman 1997), Minimum Description Length (Bouckaert 1993; Suzuki 1993; Bouckaert 1994; Lam and Bacchus 1994), entropy (Herskovits and Cooper 1990), et cetera. The optimization procedures usually rely on heuristic search strategies while sampling is often done with Markov chain Monte Carlo. Many algorithms in this category work well for graphs that do not have a large number of nodes. However, due to the size of the space of DAGs, they become computationally impractical for large networks.

1.5 Causal Bayesian Networks and Interventions

Compared to undirected graphs, Bayesian networks have an attractive property: they can be used to represent causal relationships among random variables. As mentioned in Section 1.3, we cannot distinguish equivalent DAGs from observational data. However, equivalent DAGs do not have the same causal interpretations. Although some authors discussed the possibility of causal inference from observational data (Spirtes et al. 1993; Pearl 2000), most researchers agree that causal relations can only be reliably inferred using experimental data. Experimental interventions reveal causality among a set of variables by breaking down various connections in the underlying causal network. In this dissertation, we only consider using DAGs for causal inference, following Pearl's formulation of causal Bayesian networks (Pearl 2000). In this setting, experimental interventions can help us distinguish equivalent DAGs. For instance, consider the causal interpretations of two equivalent DAGs $\mathcal{G}_1$: $X_1 \to X_2$ and $\mathcal{G}_2$: $X_1 \leftarrow X_2$. Suppose that X_2 is fixed experimentally at x_2 (the fixed value itself might be drawn from some distribution independent of the DAG). If $\mathcal{G}_1$ is the true causal model, fixing X_2 eliminates any dependency of X_2 on X_1, in effect removing the directed edge from X_1 to X_2. Thus data generated in this manner follow the joint distribution $P(X_1, X_2) = P(X_1|\varnothing)P(X_2|\varnothing)$, where $P(X_1|\varnothing)$ is the marginal distribution of X_1 and $P(X_2|\varnothing)$ is the distribution from which experimental data on X_2 are drawn. On the other hand, if the true causal model is $\mathcal{G}_2$, interventions on X_2 leave the dependency between X_1 and X_2 intact. Hence, experimental data can be used to infer causal relationships among random variables. As this example demonstrates, if X_i ($i \in \mathcal{M}$) are under intervention, then the joint distribution in (1.1) becomes

$$P(X_1, \ldots, X_p) = \prod_{i \notin \mathcal{M}} P(X_i|\Pi_i^{\mathcal{G}}) \prod_{i \in \mathcal{M}} P(X_i|\varnothing), \tag{1.2}$$

where $P(X_i|\varnothing)$ denotes the distribution of X_i under intervention. In other words, we can view experimental data from $\mathcal{G}$ as generated from the DAG $\mathcal{G}'$ obtained by removing all directed edges pointing to the nodes under intervention in $\mathcal{G}$. When we make causal inference using the likelihood function (1.2), the term $\prod_{i\in\mathcal{M}} P(X_i|\varnothing)$ can be ignored, since they depend only on external parameters that are not relevant to the estimation of DAGs.

1.6 Outline of the Dissertation

In this dissertation, we present new methods for learning the structure of sparse causal Bayesian networks via regularization. The remaining part of the dissertation is organized as follows. In Chapter 2 we propose a penalized likelihood framework to learn causal Gaussian Bayesian networks with the adaptive lasso penalty. A coordinate descent algorithm is developed to solve the corresponding optimization problem. Chapter 3 extends the work in Chapter 2 to causal discrete Bayesian networks using the adaptive group lasso penalized likelihood based on the multi-logit model. A flow cytometry data set is used in Chapter 4 to demonstrate the usage of our methods in a typical real-world application. The dissertation is concluded with a summary in Chapter 5.

Algorithm 1.1 PC algorithm

Start with the complete undirected graph $\mathcal{G}$ on the set of nodes V
$\ell = 0$
repeat
  repeat
    Select an ordered pair of variables $\langle X_i, X_j \rangle$ that are adjacent in $\mathcal{G}$ such that $|\boldsymbol{A}(\mathcal{G}, X_i) \setminus \{X_j\}| \geq \ell$
    repeat
      choose $\boldsymbol{U} \subseteq A(\mathcal{G}, X_i) \setminus \{X_j\}$ with $|\boldsymbol{U}| = \ell$
      if X_i and X_j are conditionally independent given $\boldsymbol{U}$ **then**
        Delete edge $i - j$ in $\mathcal{G}$
        Record $\boldsymbol{U}$ in $\boldsymbol{S}(X_i, X_j)$ and $\boldsymbol{S}(X_j, X_i)$
      end if
    until edge $i - j$ is deleted or all $\boldsymbol{U} \subseteq A(\mathcal{G}, X_i) \setminus \{X_j\}$ with $|\boldsymbol{U}| = \ell$ have been chosen
  until all ordered pairs of adjacent variables X_i and X_j such that $|\boldsymbol{A}(\mathcal{G}, X_i) \setminus \{X_j\}| \geq \ell$ and $\boldsymbol{U} \subseteq A(\mathcal{G}, X_i) \setminus \{X_j\}$ with $|\boldsymbol{U}| = \ell$ have been tested for conditional independence
  $\ell = \ell + 1$
until for each ordered pair of variables $\langle X_i, X_j \rangle$ that are adjacent in $\mathcal{G}$: $|\boldsymbol{A}(\mathcal{G}, X_i) \setminus \{X_j\}| < \ell$
for all pairs of nonadjacent nodes X_i and X_j with common neighbor X_k **do**
  if $X_k \notin \boldsymbol{S}(X_i, X_j)$ **then**
    Orient $i - k - j$ as $i \to k \leftarrow j$
  end if
end for
repeat
  if $k \to i - j$ and j, k are nonadjacent **then**
    Orient $i - j$ as $i \to j$
  end if
  if $i - j$ and there is a directed path from i to j **then**
    Orient $i - j$ as $i \to j$
  end if
  if $i - j$ and there are two chains $i - k \to j$ and $i - \ell \to j$ such that k, ℓ are nonadjacent **then**
    Orient $i - j$ as $i \to j$
  end if
  if $i - j$ and there is a chain $i - k \to \ell \to j$ such that k, j are nonadjacent **then**
    Orient $i - j$ as $i \to j$
  end if
until no more edges can be oriented

CHAPTER 2

Learning Sparse Causal Gaussian Networks from Experimental Data

2.1 Introduction

In recent years, a number of researchers proposed to estimate the structures of graphical models through L_1-regularized likelihood approaches (lasso-type penalties). The L_1 penalty becomes popular because of the parsimonious solution it leads to as well as its computational tractability. Much of the research has focused on estimating undirected graphs with the L_1 penalty. Yuan and Lin (2007) proposed to maximize an L_1-penalized log-likelihood based on the "max-det" problem considered by Vandenberghe et al. (1998), while Banerjee et al. (2008) employed a blockwise coordinate descent algorithm to solve the optimization problem. Friedman et al. (2008) built on the method of Banerjee et al. (2008) a remarkably efficient algorithm called the graphical lasso. Another computationally attractive method was developed by Meinshausen and Bühlmann (2006), where an undirected graph is constructed by fitting a lasso regression on each node separately.

As for undirected graphs, sparsity in the structure of a causal Bayesian network is desired, which often gives more interpretable results. A natural generalization is to use the L_1 penalty in structure learning of causal Bayesian networks with experimental data. However, there are a number of difficulties for this seemingly natural generalization. First, existing theories on L_1-regularized estimation and penalized likelihood may not be directly applicable to structure learning of DAGs

with interventional data. Different interventions effectively change the structure of a DAG as shown in Section 1.5. Second, it is expected that the computation for estimating the structures of DAGs is much more challenging than that for undirected graphs because of the acyclic constraint. Indeed, recent work of Shojaie and Michailidis (2010) assumed a known ordering of the variables to simplify the computation for the structure learning problem of DAGs, which eliminates the need for estimating the directions of causality among random variables.

In this chapter, we develop an L_1-penalized likelihood approach to learn the structures of causal Gaussian Bayesian networks using experimental data. We consider this problem in the general setting where the ordering of the variables is unknown. To the best of our knowledge, this is the first method that estimates the structures of DAGs based on L_1-penalized likelihood without assuming a known ordering. In Section 2.2 we formulate the problem of learning causal Gaussian DAGs with experimental data. We develop a coordinate descent algorithm in Section 2.3 to search a locally optimal solution to this optimization problem and establish in Section 2.4 theoretical properties of the corresponding estimator. In Section 2.5 we present results of a simulation study. This chapter is concluded with discussion in Section 2.6. All mathematical proofs and certain supplementary materials are given in Section 2.7.

2.2 Problem Formulation

In a causal Gaussian Bayesian network $\mathcal{G}$, causal relationships among random variables are modeled as

$$X_j = \sum_{i \in \Pi_j^{\mathcal{G}}} \beta_{ij} X_i + \varepsilon_j, \quad j = 1, \ldots, p, \tag{2.1}$$

where β_{ij} is the coefficient representing the influence of X_i on X_j, and ε_j's are independent Gaussian noises with mean 0 and variance σ_j^2. We assume, throughout this chapter, that all X_j have mean 0. Then the joint distribution of $(X_1, \ldots, X_p)$ defined by (2.1) is $\mathcal{N}_p(\mathbf{0}, \mathbf{\Sigma})$, where the covariance matrix $\mathbf{\Sigma}$ depends on β_{ij} $(i, j = 1, \ldots, p$, and $i \neq j)$ and σ_j^2 $(j = 1, \ldots, p)$. The set of equations in (2.1) can be regarded as the mechanism for generating these random variables.

Consider an $n \times p$ data matrix $\mathbf{X}$ generated from $\mathcal{G}$. The data matrix $\mathbf{X}$ consists of p blocks with the j^{th} block $\mathbf{X}^j$ having size $n_j \times p$, where $n = \sum_{j=1}^{p} n_j$. Each row within $\mathbf{X}^j$ is generated by imposing an intervention on the node X_j, while the values for all other nodes X_k $(k \neq j)$ are observational. The experimental data on X_j generated by intervention are assumed to follow $\mathcal{N}(0, \widetilde{\sigma}_j^2)$ for $j = 1, \ldots, p$.

Let $\mathbf{B} = (\beta_{ij})_{p \times p}$ be the coefficient matrix, where $\beta_{ij} = 0$ if $i \notin \Pi_j^{\mathcal{G}}$. Let $\boldsymbol{\sigma}^2 = (\sigma_j^2)_{1 \times p}$ and $\widetilde{\boldsymbol{\sigma}}^2 = (\widetilde{\sigma}_j^2)_{1 \times p}$ be vectors of variances. Apparently, we can learn the structure of $\mathcal{G}$ by estimating the coefficient matrix $\mathbf{B}$. In the rest of the chapter, we will call $\mathcal{G}$ the graph induced by $\mathbf{B}$.

Now let $\mathcal{I}_j$ denote the collection of indices of samples in $\mathbf{X}^j$ and $\mathcal{O}_j = \{1, \ldots, n\} \setminus \mathcal{I}_j$ denote the collection of indices of samples in which X_j is not fixed experimentally, $j = 1, \ldots, p$. According to the factorization (1.2), the joint density of the data matrix $\mathbf{X}$ can be written as

$$f(\mathbf{X}) \propto \prod_{k=1}^{p} \prod_{h \in \mathcal{I}_k} \prod_{j \neq k} f(x_{hj} | \pi_{hj}) = \prod_{j=1}^{p} \prod_{h \in \mathcal{O}_j} f(x_{hj} | \pi_{hj}), \tag{2.2}$$

where x_{hj} is the value of X_j in the h^{th} sample (the $(h, j)^{\text{th}}$ element of the data matrix $\mathbf{X}$), π_{hj} are the values of the parents of X_j in the h^{th} sample, and $f(x_{hj} | \pi_{hj})$ is the conditional density of x_{hj} given π_{hj}. Note that, as mentioned in Section 1.5, the likelihood term $f(x_{hj} | \varnothing)$ is ignored if the value x_{hj} is fixed experimentally. Let $n_{-j} = |\mathcal{O}_j| = n - n_j$. Then using the relationship in (2.1), we can easily derive

that the negative log-likelihood of $\mathbf{B}$ and $\boldsymbol{\sigma}^2$ is

$$\sum_{j=1}^{p}\left[\frac{n_{-j}\log(\sigma_j^2)}{2}+\frac{\|\mathbf{X}_{[\mathcal{O}_j,j]}-\mathbf{X}_{[\mathcal{O}_j,-j]}\mathbf{B}_{[-j,j]}\|^2}{2\sigma_j^2}\right], \tag{2.3}$$

where $\mathbf{M}_{[I_r,I_c]}$ denotes the submatrix of $\mathbf{M}$ with rows in I_r and columns in I_c.

For many real-world applications, it is often the case that the underlying network structure is sparse. It is therefore important to find a sparse structure for the coefficient matrix $\mathbf{B}$. We propose here a penalized likelihood approach with the adaptive lasso penalty to learn the structure of $\mathbf{B}$. Specifically, given a weight matrix $\mathbf{W}=(w_{ij})_{p\times p}$, we seek the minimizer $(\widehat{\mathbf{B}},\widehat{\boldsymbol{\sigma}}^2)$ of

$$\begin{aligned}\sum_{j=1}^{p}\left[\frac{n_{-j}\log(\sigma_j^2)}{2}+\frac{\|\mathbf{X}_{[\mathcal{O}_j,j]}-\mathbf{X}_{[\mathcal{O}_j,-j]}\mathbf{B}_{[-j,j]}\|^2}{2\sigma_j^2}+\lambda\sum_{i\neq j}w_{ij}|\beta_{ij}|\right], \\ \text{subject to}\quad \mathcal{G}_{\mathbf{B}}\text{ is acyclic,}\end{aligned} \tag{2.4}$$

where $\mathcal{G}_{\mathbf{B}}$ denotes the graph induced by $\mathbf{B}$, and $\lambda>0$ is the penalty parameter.

Remark 2.1. Due to the acyclic constraint, one cannot transform (2.4) into an equivalent penalized least squares problem. Hence, σ_j^2 cannot be ignored in our formulation, which makes the minimization problem considerably harder than a penalized least squares problem.

The adaptive lasso was proposed by Zou (2006) as an alternative to the lasso technique (Tibshirani 1996) for regression problems. The adaptive lasso enjoys the oracle properties considered by Fan and Li (2001). In particular, it is consistent for variable selection. In our setting, the weights are defined as $w_{ij}=|\hat{\beta}_{ij}^{(\dagger)}|^{-\gamma}$ for some $\gamma>0$, where $\hat{\beta}_{ij}^{(\dagger)}$ is a consistent estimate of β_{ij}. Zou (2006) suggests using the ordinary least squares (OLS) estimates to define the weights in the regression setting. However, because of the existence of equivalent DAGs, the OLS estimates may not be consistent in our case. To obtain the initial consistent estimates $\hat{\beta}_{ij}^{(\dagger)}$'s, we let $\widetilde{w}_{ij}=\min(|\hat{\beta}_{ij}^{(\mathrm{OLS})}|^{-\gamma},M^{\gamma})$, where M is a large positive

constant (e.g. $M = 10^4$) and $\hat{\beta}_{ij}^{(\mathrm{OLS})}$'s are OLS estimates obtained by regressing X_j on other nodes using samples $h \in \mathcal{O}_j$. As will be shown in Section 2.4, there exists a $\sqrt{n}$-consistent local minimizer $\widehat{\mathbf{B}}$ of (2.4) with weights $\widetilde{w}_{ij}$, which can be used as $\hat{\beta}_{ij}^{(\dagger)}$'s. Then consistent estimate of the graph structure can be obtained with weights $w_{ij} = |\hat{\beta}_{ij}^{(\dagger)}|^{-\gamma}$.

After minimizing with respect to $\boldsymbol{\sigma}^2$, problem (2.4) becomes

$$\min_{\mathbf{B}} \; V(\mathbf{B}; \mathbf{W}) = \sum_{j=1}^{p} \left[\frac{n_{-j}}{2} \log \left(\|\mathbf{X}_{[\mathcal{O}_j, j]} - \mathbf{X}_{[\mathcal{O}_j, -j]} \mathbf{B}_{[-j,j]}\|^2 \right) + \lambda \sum_{i \neq j} w_{ij} |\beta_{ij}| \right], \quad \text{subject to} \quad \mathcal{G}_{\mathbf{B}} \text{ is acyclic}, \tag{2.5}$$

which is the problem we aim to solve.

2.3 Coordinate Descent Algorithm

Both the objective function V in (2.5) and the constraint set are nonconvex. Searching for the global optimal solution of (2.5) may be impractical. Moreover, the theoretical results in Section 2.4 only establish consistency of a local minimizer (see Theorem 2.3 and 2.4). Therefore, we develop in this section a coordinate descent (CD) algorithm in order to find a local minimum to the constrained optimization problem (2.5). A local minimizer $\widehat{\mathbf{B}}$ is defined as follows: (i) any local change in the structure of $\mathcal{G}_{\widehat{\mathbf{B}}}$, i.e., addition, removal or reversal of a single edge, increases the value of V; (ii) given the structure of $\mathcal{G}_{\widehat{\mathbf{B}}}$, $\widehat{\mathbf{B}}$ is a local minimizer of V. Coordinate descent methods have been successfully used to solve lasso-type problems (Fu 1998; Friedman et al. 2007; Wu and Lange 2008). They are attractive since minimizing the objective function in one coordinate at a time is computationally simple and gradient-free. As a result, these methods are easy to implement and are usually scalable.

2.3.1 Single-parameter update

Before detailing the coordinate descent algorithm, let us first consider minimizing V in (2.5) w.r.t. a single parameter β_{kj} ($k \neq j$) without the acyclic constraint. In particular, we seek the minimizer $\hat{\beta}_{kj}$ of

$$\begin{aligned} V_j &= \frac{n_{-j}}{2} \log\left(\|\mathbf{X}_{[\mathcal{O}_j,j]} - \mathbf{X}_{[\mathcal{O}_j,-j]}\mathbf{B}_{[-j,j]}\|^2\right) + \lambda \sum_{i\neq j} w_{ij}|\beta_{ij}| \\ &= \frac{n_{-j}}{2} \log\left[\sum_{h\in\mathcal{O}_j}\left(x_{hj} - \sum_{i\neq j,k} x_{hi}\beta_{ij} - x_{hk}\beta_{kj}\right)^2\right] \\ &\quad + \lambda \sum_{i\neq j,k} w_{ij}|\beta_{ij}| + \lambda w_{kj}|\beta_{kj}|, \end{aligned} \tag{2.6}$$

assuming all β_{ij}'s ($i \neq j, k$) are fixed. We can transform the weighted lasso penalty in (2.6) into an ordinary lasso penalty:

$$\begin{aligned} \min_{\tilde{\beta}_{kj}} \widetilde{V}_j &= \frac{n_{-j}}{2} \log\left[\sum_{h\in\mathcal{O}_j}\left(x_{hj} - \sum_{i\neq j,k} \tilde{x}_{hi}\tilde{\beta}_{ij} - \tilde{x}_{hk}\tilde{\beta}_{kj}\right)^2\right] \\ &\quad + \lambda \sum_{i\neq j,k} |\tilde{\beta}_{ij}| + \lambda|\tilde{\beta}_{kj}|, \end{aligned} \tag{2.7}$$

by letting $\tilde{\beta}_{ij} = w_{ij}\beta_{ij}$ and $\tilde{x}_{hi} = x_{hi}/w_{ij}$ for $i \neq j$. We further define $y_{hj}^{(k)} = x_{hj} - \sum_{i\neq j,k} \tilde{x}_{hi}\tilde{\beta}_{ij}$, $\xi_{kj} = \sum_{h\in\mathcal{O}_j} \tilde{x}_{hk} y_{hj}^{(k)} / \sum_{h\in\mathcal{O}_j} \tilde{x}_{hk}^2$, $c_{kj} = \sum_{h\in\mathcal{O}_j} \left(y_{hj}^{(k)}\right)^2 / \sum_{h\in\mathcal{O}_j} \tilde{x}_{hk}^2$ and $\gamma = \lambda/n_{-j}$. Note that according to Cauchy-Schwarz inequality, $c_{kj} - \xi_{kj}^2 \geq 0$. Then equivalently, (2.7) can be simplified to the problem

$$\min_{\tilde{\beta}_{kj}} g(\tilde{\beta}_{kj}) = \frac{1}{2} \log\left[\left(\tilde{\beta}_{kj} - \xi_{kj}\right)^2 + \left(c_{kj} - \xi_{kj}^2\right)\right] + \gamma|\tilde{\beta}_{kj}|. \tag{2.8}$$

The form of g is reminiscent of the lasso problem with a single predictor. However, minimizing g with respect to $\tilde{\beta}_{kj}$ is not as easy as the corresponding lasso problem, since g is not a convex function. The function g might have two local minima for some values of ξ_{kj}, c_{kj} and γ. It is therefore necessary to compare the values of the

two local minima under certain conditions. We summarize the solution to (2.8) in the following proposition and provide its proof in Section 2.7.1.

Proposition 2.1. *Let* $\Delta = 1 - 4(c_{kj} - \xi_{kj}^2)\gamma^2$ *and* $\beta_1^* = \operatorname{sgn}(\xi_{kj})\left(|\xi_{kj}| - \dfrac{1-\sqrt{\Delta}}{2\gamma}\right)$. *Then the solution to the optimization problem (2.8) is given by*

$$\arg\min_{\tilde{\beta}_{kj}} g = \begin{cases} \beta_1^*, & \text{if } 0 < \gamma < |\xi_{kj}|/c_{kj}, \\ \beta_1^*, & \text{if } |\xi_{kj}|/c_{kj} \le \gamma < \left(2\sqrt{c_{kj} - \xi_{kj}^2}\right)^{-1}, \gamma > (2|\xi_{kj}|)^{-1} \\ & \quad \text{and } g(\beta_1^*) < g(0), \\ 0, & \text{otherwise.} \end{cases}$$

Remark 2.2. The form of β_1^* suggests that $\arg\min_{\tilde{\beta}_{kj}} g$ is similar to a soft thresholded version (Donoho and Johnstone 1995) of ξ_{kj} in nature. One difference, however, is that $\arg\min_{\tilde{\beta}_{kj}} g$ can be zero even when $|\xi_{kj}| - (1 - \sqrt{\Delta})(2\gamma)^{-1} > 0$ (see proof of Proposition 2.1 in Section 2.7.1). Note that if $4(c_{kj} - \xi_{kj}^2)\gamma^2 = o(1)$, by Taylor expansion $\sqrt{\Delta} \approx 1 - 2(c_{kj} - \xi_{kj}^2)\gamma^2$. Then $\beta_1^* \approx \operatorname{sgn}(\xi_{kj})\left(|\xi_{kj}| - (c_{kj} - \xi_{kj}^2)\gamma\right) = \operatorname{sgn}(\xi_{kj})\left(|\xi_{kj}| - c_{kj}(1-\zeta^2)\gamma\right)$, where ζ is the correlation coefficient between $\tilde{x}_{hk}$ and $y_{hj}^{(k)}$ for $h \in \mathcal{O}_j$.

Remark 2.3. In Proposition 2.1, we could find a more explicit condition on γ to determine when $g(\beta_1^*) < g(0)$, but the condition does not have an analytical solution. Thus, it seems more effective to compare $g(\beta_1^*)$ and $g(0)$ directly.

2.3.2 Description of the CD algorithm

The difficulty in minimizing V in (2.5) is due to the constraint that the graphical representation of Bayesian networks is acyclic. One immediate consequence of the constraint is that a pair of coefficients β_{ij} and β_{ji} cannot both be nonzero. We thus take advantage of this implication when designing the CD algorithm. Instead of minimizing V over a single parameter β_{ij} at each step, we perform minimization

over β_{ij} and β_{ji} simultaneously. Hence, our method can be naturally described as a blockwise coordinate descent method. For a p-node problem, the $p(p-1)$ coefficients are partitioned into $p(p-1)/2$ blocks. Each block consists of a pair of coefficients β_{ij} and β_{ji}. The algorithm starts with an initial estimate of the coefficient matrix $\mathbf{B}$ (for instance, the zero matrix) and assumes a predefined order to cycle through the $p(p-1)/2$ blocks. At each step, V is minimized over a certain block of β_{ij} and β_{ji} while all other blocks are held constant. Given the current estimates of other blocks, β_{ij} (or β_{ji}) is constrained to zero if a nonzero value introduces cycles in the resulting graph. In this case, V is only minimized over β_{ji} (or β_{ij}). Otherwise, the algorithm compares $\min_{\beta_{ij},\beta_{ji}=0} V$ with $\min_{\beta_{ij}=0,\beta_{ji}} V$ in order to update β_{ij} and β_{ji}. We repeat cycling through the $p(p-1)/2$ blocks until some stopping criterion is satisfied.

The major steps in the CD algorithm are summarized as follows, where we use $\tilde{\beta}_{ij} \Leftarrow 0$ to mean that $\tilde{\beta}_{ij}$ must be set to zero due to the acyclic constraint. In the following, different $\mathbf{X}_{[\mathcal{O}_j,\cdot]}$'s are treated as different entities so that operations on $\mathbf{X}_{[\mathcal{O}_j,\cdot]}$ will not affect $\mathbf{X}_{[\mathcal{O}_k,\cdot]}$ for $k \neq j$.

Algorithm 2.1

CD algorithm for estimating Gaussian Bayesian networks

1. Center and standardize the columns of $\mathbf{X}_{[\mathcal{O}_j,\cdot]}$ $(j = 1, \ldots, p)$ to have mean zero and unit L_2 norm. Transform the weighted lasso problem (2.5) to an ordinary lasso problem by defining $\widetilde{\mathbf{X}}_{[\mathcal{O}_j,i]} = \mathbf{X}_{[\mathcal{O}_j,i]}/w_{ij}$, $i \neq j$, for $j = 1, \ldots, p$. Choose $\mathbf{B}^0$ such that $\mathcal{G}_{\mathbf{B}^0}$ is acyclic.

2. Cycle through the $p(p-1)/2$ blocks of coefficients. Specifically, do one of the following for the pair of coefficients $\tilde{\beta}_{ij}$ and $\tilde{\beta}_{ji}$ $(i \neq j)$, given the current estimates of other coefficients.

   (a) If $\tilde{\beta}_{ji} \Leftarrow 0$, minimize $\widetilde{V}_j$ in (2.7) w.r.t. $\tilde{\beta}_{ij}$ according to Proposition 2.1 and find $\tilde{\beta}^*_{ij} = \arg\min_{\tilde{\beta}_{ij}} \widetilde{V}_j$. Then set $(\tilde{\beta}_{ij}, \tilde{\beta}_{ji}) = (\tilde{\beta}^*_{ij}, 0)$.

(b) If $\tilde{\beta}_{ij} \Leftarrow 0$, minimize $\widetilde{V}_i$ w.r.t. $\tilde{\beta}_{ji}$ according to Proposition 2.1 and find $\tilde{\beta}^*_{ji} = \arg\min_{\tilde{\beta}_{ji}} \widetilde{V}_i$. Then set $(\tilde{\beta}_{ij}, \tilde{\beta}_{ji}) = (0, \tilde{\beta}^*_{ji})$.

(c) If 2a and 2b do not apply, then compare the following two sums: $S_1 = \widetilde{V}_i|_{\tilde{\beta}_{ji}=0} + \widetilde{V}_j|_{\tilde{\beta}_{ij}=\tilde{\beta}^*_{ij}}$ and $S_2 = \widetilde{V}_i|_{\tilde{\beta}_{ji}=\tilde{\beta}^*_{ji}} + \widetilde{V}_j|_{\tilde{\beta}_{ij}=0}$. Set $(\tilde{\beta}_{ij}, \tilde{\beta}_{ji}) = (\tilde{\beta}^*_{ij}, 0)$ if $S_1 \leq S_2$. Otherwise, set $(\tilde{\beta}_{ij}, \tilde{\beta}_{ji}) = (0, \tilde{\beta}^*_{ji})$.

3. Repeat step 2 until the maximum absolute difference among all coefficients between successive cycles is below some threshold or until the maximum number of iterations is reached.

4. Output the estimates $\hat{\beta}_{ij} = \tilde{\beta}_{ij}/w_{ij}$ for $i, j = 1, \ldots, p$ and $i \neq j$.

To ensure the acyclic constraint by checking whether $\tilde{\beta}_{ij} \Leftarrow 0$, we employ a breadth-first search algorithm based on Algorithm 4 in Ellis (2006). A detailed description of this algorithm is given in Section 2.7.2.

2.3.3 Practical considerations

Since it is difficult in practice to predetermine the optimal value of λ, we compute solutions using a decreasing sequence of values for λ, following the practice of Friedman et al. (2010). The solution for the current λ is used as the initial estimate for the next value of λ in the sequence. Since large values of λ force many β_{ij} to be zero and make the optimization much easier, the solution for large λ is likely to agree well with some sub-graph of the true model. Therefore, employing warm starts may boost the performance of the CD algorithm. Indeed, our experience suggests that results obtained using a sequence of λ's are better than those computed using individual λ.

To speed up the CD algorithm, we utilize an active set method that is better suited for warm starts, as was done by Friedman et al. (2010). The algorithm first performs a complete cycling through all $p(p-1)/2$ blocks of coefficients to

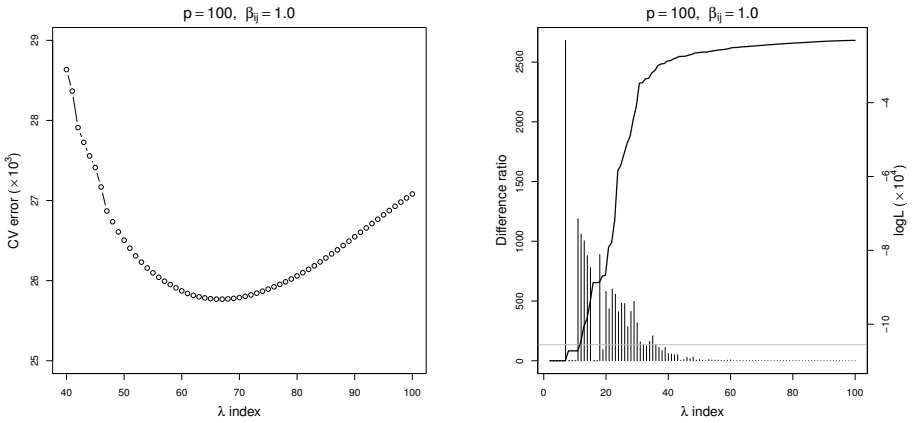
identify the active set—the set of nonzero coefficients. We then only iterate over the active set until the maximum coefficient difference falls below the threshold or the maximum number of iterations has been reached. The algorithm stops if another full cycle of all the blocks does not change the active set; otherwise the above process is repeated.

It should be noted that convergence of coordinate descent methods often requires the objective function to be strictly convex differentiable. For nondifferentiable functions, coordinate descent may get stuck at non-optimal points, although Tseng (2001) considered generalizations to nondifferentiable functions with certain separability and regularity properties. Because of the nonconvex nature of the objective function V in (2.5) and the constraint set, convergence of the CD algorithm deserves a rigorous investigation, which is beyond the scope of this study. We conjecture that the CD algorithm converges under certain conditions. In practice, we have never encountered any examples so far where the algorithm does not converge.

2.3.4 Choice of the tuning parameter

The graphical model learned by the CD algorithm depends on the choice of the penalty λ. Model selection is usually based on an estimate of prediction error, and commonly used model selection methods include Bayesian information criterion (BIC) and cross-validation among others. However, rather than selecting a model with a small prediction error, our goal is to estimate a graphical structure from data that is as close as possible to the true model. The model with the smallest prediction error on a test data of finite size is not necessarily the model that is the closest to the true network structure. According to our experience, models selected based on prediction errors are often too complex compared to the true models when the true models are sparse. Figure 2.1A plots the 5-fold cross-validation error for a sequence of graphs learned given a decreasing sequence of λ

from a simulated data set with $p = 100$, $n = 500$ and $\beta_{ij} = 1.0$. The CV error is minimized at the 67^{th} λ. The corresponding graph $\hat{\mathcal{G}}_{67}$ (obtained using λ_{67} as the tuning parameter on the whole data set) has a total of 993 predicted edges with an 82.6% false discovery rate, while the true graph only has 200 directed edges. Similar results are obtained if we use BIC or other scoring metrics such as the Bayesian score of a graph.



(A) (B)

Figure 2.1: Plots of (A) CV error, (B) difference ratio ("histogram-like" vertical lines) and log-likelihood (solid line) for graphs estimated using a decreasing sequence of λ.

In this chapter, we employ an empirical model selection criterion that works well in practice. Note that as we decrease λ and thus increase model complexity, the log-likelihood of the estimated graph will increase. Denote by $\widehat{\mathbf{B}}_{\lambda_i}$ a solution to (2.5) with the i^{th} penalty parameter λ_i. For the estimated graph $\hat{\mathcal{G}}_{\lambda_i}$ induced by $\widehat{\mathbf{B}}_{\lambda_i}$, we estimate the unpenalized coefficient matrix, denoted by $\widetilde{\mathbf{B}}_i$, by regressing X_k on $\Pi_k^{\hat{\mathcal{G}}_{\lambda_i}}$, $k = 1, \ldots, p$. Given two estimated graphs $\hat{\mathcal{G}}_{\lambda_i}$ and $\hat{\mathcal{G}}_{\lambda_j}$ ($\lambda_i > \lambda_j$), let $\Delta L_{ij} = L(\widetilde{\mathbf{B}}_j) - L(\widetilde{\mathbf{B}}_i)$ and $\Delta e_{ij} = e_{\lambda_j} - e_{\lambda_i}$, where $L(\widetilde{\mathbf{B}}) = -V(\widetilde{\mathbf{B}}; \mathbf{0})$ denotes the log-likelihood function and e denotes the total number of edges in an estimated graph. We then define the difference ratio of two estimated graphs as $dr_{(ij)} = \Delta L_{ij}/\Delta e_{ij}$. We reason that, an increase in model complexity, which is represented by an increase in the total number of predicted edges, is desirable only

if there is a substantial increase in the log-likelihood. Therefore, we compute successively the difference ratios between two adjacent graphs in the solution path, $\{dr_{(12)}, \ldots, dr_{(i,i+1)}, \ldots, dr_{(m-1,m)}\}$, where m is the number of λ in the sequence. The graph with the following index is selected:

$$K = \sup\{k : dr_{(k-1,k)} \geq \alpha \times \max(dr_{(12)}, \ldots, dr_{(m-1,m)}), k = 2, \ldots, m\}, \quad (2.9)$$

where α is a thresholding parameter. Essentially, this is the graph from which further increase in model complexity will not lead to substantial increase in the likelihood. We find that $\alpha \in [0.05, 0.1]$ works well in our simulation. Figure 2.1B plots the difference ratio as well as the log-likelihood for different graphs learned from the same data. The graph selected according to (2.9) with $\alpha = 0.05$ is $\hat{\mathcal{G}}_{36}$, which has 168 edges with a 77% true positive rate and an 8.3% false discovery rate, much less than 82.6%.

2.4 Asymptotic Properties

In this section, we develop asymptotic theories on the penalized likelihood estimator of DAGs. To simplify notations, we write $\mathbf{B}$ in a vector format as $\boldsymbol{\phi} = (\phi_j)_{1\times d} = \big((\mathbf{B}_{[-1,1]})^T, \ldots, (\mathbf{B}_{[-p,p]})^T\big)$, where $d = p(p-1)$ is the length of $\boldsymbol{\phi}$. Similarly, we write the weight matrix $\mathbf{W}$ in a vector format as $\mathcal{T} = (\tau_j)_{1\times d}$. We say $\boldsymbol{\phi}$ is acyclic if the graph $\mathcal{G}_{\boldsymbol{\phi}}$ induced by $\boldsymbol{\phi}$ (or the corresponding $\mathbf{B}$) is acyclic. Let $\boldsymbol{\theta} = (\boldsymbol{\phi}, \boldsymbol{\sigma}^2, \widetilde{\boldsymbol{\sigma}}^2)$ be the vector of parameters and $\boldsymbol{\Omega} = \{\boldsymbol{\theta} : \boldsymbol{\phi}$ is acyclic, $\boldsymbol{\sigma}^2 > 0, \widetilde{\boldsymbol{\sigma}}^2 > 0\}$ be the parameter space. Recall that $\boldsymbol{\sigma}^2 = (\sigma_j^2)_{1\times p}$ and $\widetilde{\boldsymbol{\sigma}}^2 = (\widetilde{\sigma}_j^2)_{1\times p}$ are vectors of variances defined in Section 2.2. Denote the true parameter value by $\boldsymbol{\theta}^* = (\boldsymbol{\phi}^*, (\boldsymbol{\sigma}^2)^*, (\widetilde{\boldsymbol{\sigma}}^2)^*) \in \boldsymbol{\Omega}$. Let $\mathcal{G}_{\boldsymbol{\phi}^*}$ denote the DAG induced by $\boldsymbol{\phi}^*$, i.e., the true DAG.

Let $\boldsymbol{\theta}_k = (\boldsymbol{\phi}_k, \boldsymbol{\sigma}^2_{[-k]}, \widetilde{\sigma}^2_k)$, where $\boldsymbol{\phi}_k$ is obtained from $\boldsymbol{\phi}$ by replacing $\mathbf{B}_{[-k,k]}$ with $\mathbf{0}$, i.e., by suppressing all edges pointing to the k^{th} node from its parents. Here $\boldsymbol{\nu}_{[I]}$

denotes the subvector of a vector $\boldsymbol{\nu}$ with indices in I. As mentioned in Section 1.5, $\mathbf{X}^k$, the k^{th} block of the data matrix, can be regarded as *i.i.d.* observations from a distribution factorized according to the DAG $\mathcal{G}_{\boldsymbol{\phi}_k}$, and we denote the corresponding density by $f(\mathbf{x}|\boldsymbol{\theta}_k)$, where $\mathbf{x} = (x_1, \ldots, x_p)$. For Gaussian random variables, f is the density function of $\mathcal{N}_p(\mathbf{0}, \boldsymbol{\Sigma}(\boldsymbol{\theta}_k))$. Here we emphasize the dependence of the variance-covariance matrix $\boldsymbol{\Sigma}$ on $\boldsymbol{\theta}_k$. Recall that $\mathcal{I}_k$ denotes the collection of indices of samples in $\mathbf{X}^k$. Then we define the penalized log-likelihood with the adaptive lasso penalty as

$$R(\boldsymbol{\theta}) = L(\boldsymbol{\theta}) - \lambda_n \sum_{j=1}^{d} \tau_j |\phi_j| = \sum_{k=1}^{p} L_k(\boldsymbol{\theta}_k) - \lambda_n \sum_{j=1}^{d} \tau_j |\phi_j|, \tag{2.10}$$

where $L_k(\boldsymbol{\theta}_k) = \sum_{h \in \mathcal{I}_k} \log f(\mathbf{X}_{[h,\cdot]}|\boldsymbol{\theta}_k)$. Our goal is to seek a maximizer of $R(\boldsymbol{\theta})$ in the parameter space $\boldsymbol{\Omega}$ to obtain an estimator $\hat{\boldsymbol{\theta}}$. Note that the log-likelihood function $L(\boldsymbol{\theta})$ is different from the one in (2.4) and (2.5), since here we also include in $L(\boldsymbol{\theta})$ terms depending on $\widetilde{\boldsymbol{\sigma}}^2$. It is easily seen that these two formulations of the likelihood function are equivalent for the purpose of estimating the coefficients and the structure of Bayesian networks.

Even with interventional data the coefficient matrix of a DAG may not be identifiable because of interventional Markov equivalence among DAGs (Hauser and Bühlmann 2011). We introduce below the notion of natural parameters to establish identifiability of DAGs for the case where each variable has interventional data. Suppose that X_i is an ancestor of X_j in a DAG $\mathcal{G}$, that is, there exists at least one path from X_i to X_j (see Lauritzen 1996, chapter 2 for terminology used in graphical models). Let

$$\Gamma(i,j) = \{(i_0, \ldots, i_m) : i_k \to i_{k+1} \text{ for } 0 \le k \le m-1, i_0 = i, i_m = j, m \ge 1\} \tag{2.11}$$

be the set of paths from X_i to X_j. Define the coefficient of influence of X_i on X_j

by $\beta_{i\to j} = \sum_{\Gamma(i,j)} \prod_{k=0}^{m-1} \beta_{i_k i_{k+1}}$, that is, the sum of products of coefficients along each path from X_i to X_j. Denote the set of ancestors of X_j by $\mathrm{an}(X_j)$.

Definition 2.1 (Natural parameters)**.** We say that $\boldsymbol{\theta}$ is natural if the corresponding coefficient matrix $\mathbf{B}$ satisfies

$$\beta_{i\to j} \neq 0 \quad \text{for all } X_i \in \mathrm{an}(X_j), \quad 1 \leq j \leq p. \tag{2.12}$$

Note that if the underlying DAG is a polytree, the corresponding parameter is always natural. For more general DAGs, natural parameters imply that the causal effects along multiple paths connecting the same pair of nodes do not cancel, which is a reasonable assumption for many real-world problems. If the true parameter is natural, then with sufficient experimental data, the parameter $\boldsymbol{\theta}$ is identifiable as indicated by the following theorem. The proof of Theorem 2.2 is given in Section 2.7.1.

Theorem 2.2. *Suppose that samples in $\mathbf{X}^k$ are i.i.d. with probability density $f(\mathbf{x}|\boldsymbol{\theta}_k^*)$ of the normal distribution $\mathcal{N}_p(\mathbf{0}, \boldsymbol{\Sigma}(\boldsymbol{\theta}_k^*))$ for $k = 1, \ldots, p$. Assume that the true parameter $\boldsymbol{\theta}^*$ is natural. Then*

$$f(\mathbf{x}|\boldsymbol{\theta}_k) = f(\mathbf{x}|\boldsymbol{\theta}_k^*) \quad \textit{a.e. for all } k = 1, \ldots, p \qquad \Longrightarrow \qquad \boldsymbol{\theta} = \boldsymbol{\theta}^*. \tag{2.13}$$

If we further assume that $n_k/n \to \alpha_k > 0$ as $n \to \infty$, then

$$P_{\boldsymbol{\theta}^*}(L(\boldsymbol{\theta}^*) > L(\boldsymbol{\theta})) \to 1 \tag{2.14}$$

for any $\boldsymbol{\theta} \neq \boldsymbol{\theta}^*$.

Now we state the following theorems to establish the asymptotic properties of $\hat{\boldsymbol{\theta}}$. We follow arguments similar to those given by Fan et al. (2009) to prove Theorem 2.3 and Theorem 2.4. However, one cannot directly apply Fan et al.'s results here, because the parameters must satisfy the acyclic constraint, the data we have are not *i.i.d.* observations due to interventions, and the identifiability of a DAG is not always guaranteed.

Let $\hat{\phi}_k^{(\mathrm{OLS})}$ $(1 \leq k \leq d)$ be the estimate of ϕ_k when the corresponding β_{ij} $(i \neq j)$ is estimated by $\hat{\beta}_{ij}^{(\mathrm{OLS})}$. Let $\mathcal{A} = \{j : \phi_j^* = 0\}$ and $\boldsymbol{\phi}_{\mathcal{A}} = (\phi_j)_{j \in \mathcal{A}}$. It is assumed that $\boldsymbol{\theta}^*$ is natural in the following two theorems. We relegate the proofs of Theorem 2.3 and Theorem 2.4 to Section 2.7.1.

Theorem 2.3. *Assume the adaptive lasso penalty with weights* $\tau_j = \min(|\hat{\phi}_j^{(\mathrm{OLS})}|^{-\gamma}, M^\gamma)$ *for all* j, *where* $\gamma, M > 0$. *As* $n \to \infty$, *if* $\lambda_n/\sqrt{n} \to 0$ *and* $n_k/n \to \alpha_k > 0$ *for* $k = 1, \ldots, p$, *then there exists a local maximizer* $\hat{\boldsymbol{\theta}}$ *of* $R(\boldsymbol{\theta})$ *such that* $\|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*\| = O_p(n^{-1/2})$.

Theorem 2.4. *Assume the adaptive lasso penalty with weights* $\tau_j = |\tilde{\phi}_j|^{-\gamma}$ *for some* $\gamma > 0$ *and all* j, *where* $\tilde{\phi}_j$ *is* $\sqrt{n}$*-consistent for* ϕ_j^*. *As* $n \to \infty$, *if* $\lambda_n/\sqrt{n} \to 0$, $\lambda_n n^{(\gamma-1)/2} \to \infty$ *and* $n_k/n \to \alpha_k > 0$ *for* $k = 1, \ldots, p$, *then there exists a local maximizer* $\hat{\boldsymbol{\theta}}$ *of* $R(\boldsymbol{\theta})$ *such that* $\|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*\| = O_p(n^{-1/2})$. *Furthermore, with probability tending to one, the* $\sqrt{n}$*-consistent local maximizer* $\hat{\boldsymbol{\theta}}$ *must satisfy:* $\hat{\boldsymbol{\phi}}_{\mathcal{A}} = \mathbf{0}$.

Remark 2.4. To achieve consistency in model selection with the adaptive lasso penalty, we need some consistent estimate of the vector $\boldsymbol{\phi}$ to construct the weights. Theorem 2.3 suggests that we first use $\tau_j = \min(|\hat{\phi}_j^{(\mathrm{OLS})}|^{-\gamma}, M^\gamma)$ as weights to obtain an initial consistent estimate $\tilde{\boldsymbol{\phi}}$. Then with weights constructed from $\tilde{\boldsymbol{\phi}}$, Theorem 2.4 guarantees that model selection consistency can be achieved.

2.5 Simulation Study

2.5.1 Small sample sizes

To test the performance of the CD algorithm, we conducted a simulation study. We randomly generated graphs with p nodes ($p = 20, 50, 100, 200$) and $2p$ edges. To further control the sparsity of the graphs, we set the maximum number of parents for any given node to be 4. For each value of p, we simulated 10 different

random graphs, and for each graph, three data sets were generated according to equation (2.1) with $\beta_{ij} = 0.2, 0.5$ and 1.0, respectively. The variance σ_j^2 of the Gaussian noise ε_j $(j = 1, \ldots, p)$ was set to 1 in all our simulations. Each data set has $n = 5p$ samples. As described in Section 2.2, these samples are divided into p blocks such that the sample size of each block is $n_j = 5$, $j = 1, \ldots, p$. The j^{th} block $\mathbf{X}^j$ contains experimental data on the node X_j, which were drawn from the standard normal distribution $\mathcal{N}(0, 1)$. For each data set, we applied the CD algorithm to compute the solution path using a geometric sequence of λ's, starting from the largest value $\lambda_{\max}$ for which $\widehat{\mathbf{B}}_{\lambda_{\max}} = \mathbf{0}$ and decreasing to the smallest value $\lambda_{\min}$. The sequence typically contained 50 or 100 different values of λ's with the ratio $\lambda_{\min}/\lambda_{\max}$ set to some small value such as 0.001. Graphical models were then selected from the solution paths according to (2.9) with $\alpha = 0.1$. We used $\gamma = 0.15$ for all data sets, except for the two cases with $p \geq 100$ and $\beta_{ij} = 1.0$, where γ was set to 0.5.

Table 2.1 summarizes the average performance of the CD algorithm over 10 data sets for each combination of p and β_{ij}. For instance, when $p = 100$ and $\beta_{ij} = 0.5$, the estimated graphical model on average contains 220.9 directed edges, of which 156.5 edges are present in the true graph, 28.3 edges have directions reversed, and the rest 36.1 edges are not included in the true graph. On average, there are also 15.2 true edges missing in the estimated model. Results in Table 2.1 suggest that our method can estimate the structures of DAGs with reasonable accuracy even when the sample sizes are limited. All TPRs (defined in Table 2.1) are above 0.70 except for $\beta_{ij} = 0.2$, cases where signal-to-noise ratios are too small. Though for small networks (small p and thus small sample sizes) or small β_{ij} values (small signal-to-noise ratios) FDRs are relatively high, they become much lower for larger networks and higher values of β_{ij}. Note that, when $p = 200$, the number of parameters to be estimated is around 20,000, which is much larger than the sample size $n = 5p = 1000$. Even in this high-dimensional setting our

Table 2.1: The average number of predicted (P), expected (E), reversed (R), missed (M) and false positive (FP) edges and the average true positive rates (TPRs[1]) and false discovery rates (FDRs[2]) for DAGs learned by the CD algorithm (sample size n=5p)

p	β_{ij}	CD algorithm							KO method	
		P	E	R	M	FP	TPR	FDR	TPR	FDR
20	0.2	59.6	17.3	10.9	11.8	31.4	0.433 (0.069)	0.694 (0.080)	0.375	0.213
	0.5	48.6	29.2	6.1	4.7	13.3	0.730 (0.152)	0.399 (0.083)	0.908	0.086
	1.0	65.5	34.0	2.8	3.2	28.7	0.850 (0.092)	0.429 (0.138)	0.723	0.065
50	0.2	158.9	54.0	32.8	13.2	72.1	0.540 (0.048)	0.652 (0.061)	0.732	0.128
	0.5	114.9	74.5	17.4	8.1	23.0	0.745 (0.100)	0.351 (0.085)	0.992	0.045
	1.0	132.7	70.5	5.0	24.5	57.2	0.705 (0.113)	0.453 (0.090)	0.763	0.050
100	0.2	246.0	137.9	53.2	8.9	54.9	0.690 (0.027)	0.431 (0.075)	0.952	0.088
	0.5	220.9	156.5	28.3	15.2	36.1	0.783 (0.058)	0.290 (0.071)	0.993	0.032
	1.0	167.8	149.1	11.4	39.5	7.3	0.746 (0.087)	0.109 (0.074)	0.508	0.011
200	0.2	421.2	325.3	72.2	2.5	23.7	0.813 (0.054)	0.226 (0.061)	1.000	0.051
	0.5	430.2	341.8	41.9	16.3	46.5	0.855 (0.053)	0.203 (0.071)	1.000	0.016
	1.0	328.1	298.5	18.3	83.2	11.3	0.746 (0.100)	0.090 (0.049)	0.549	0.004

Note:
1. [1] TPR = E/T, where T = $2p$ is the total number of true edges. [2] FDR = $(\mathrm{R}+\mathrm{FP})/\mathrm{P}$.
2. The numbers in parentheses are the standard deviations across 10 data sets. As a comparison, the last two columns list the average TPRs and FDRs for DAGs estimated by the approach of Shojaie and Michailidis (2010) assuming the ordering of the variables is known.

CD algorithm was still able to estimate DAGs quite accurately.

Since the CD algorithm computes a set of solutions along the solution path of problem (2.5), another way to evaluate the performance is to investigate the relationship between TPR and false positive rate [$\mathrm{FPR} = (\mathrm{R}+\mathrm{FP})/(p(p-1)-\mathrm{T})$] as the penalty parameter λ varies, which is known as the receiver operating characteristic (ROC) analysis. However, since the sequence of λ's we used was data-dependent, we examined the TPR-FPR relationships as the number of predicted edges increases. Figure 2.2 presents the results of ROC analysis. Again, these

ROC curves suggest satisfactory performance of the CD algorithm except when signal-to-noise ratios are small ($\beta_{ij} = 0.2$). In particular, we note that for large networks ($p = 100, 200$), as we increase the number of predicted edges and the complexity of estimated graphs by adjusting the penalty λ, we will increase TPR without affecting FPR much until TPR reaches a plateau, a level at which the estimated DAGs are structurally similar to the true DAGs.

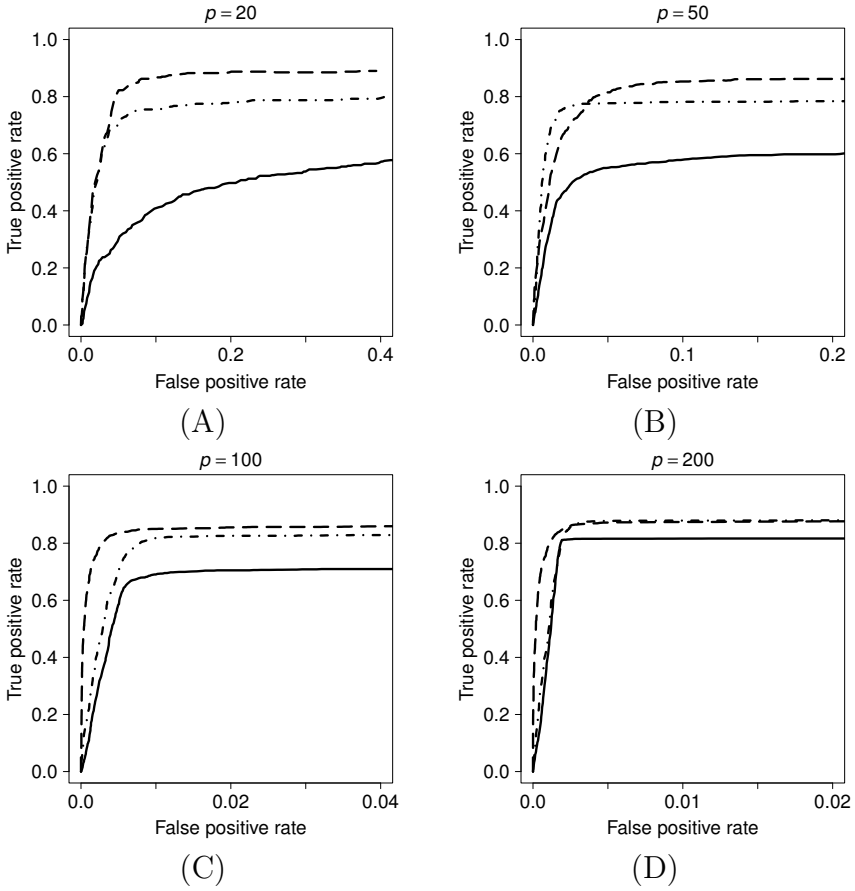


Figure 2.2: ROC curves for $\beta_{ij} = 0.2$ (solid lines), $\beta_{ij} = 0.5$ (dotdashed lines) and $\beta_{ij} = 1.0$ (long dashed lines), sample size n=5p.

To get a sense of the amount of information that interventional data can provide to resolve directionality of DAGs, we also applied the CD algorithm to simulated observational data with the same sample sizes as their interventional counterparts. The results are summarized in Table 2.7 in Section 2.7.3. We found that interventional data helped to increase TPRs and simultaneously reduce

FDRs, and the boost in TPRs ranges from 2% up to about 50%.

To benchmark the performance of the CD algorithm, we compared our method to a PC algorithm based approach. The PC algorithm is a classical constraint-based method that can estimate DAGs with hundreds of nodes. We did not compare with Monte Carlo approaches, as even the most recent developments, such as the order-graph sampler (Ellis and Wong 2008), have not shown convincing performance on graphs with more than 50 nodes.

The PC algorithm is designed to estimate from observational data a completed partially directed acyclic graph (CPDAG), which contains both directed and undirected edges. We therefore took a two-step approach to produce results favorable for the PC algorithm. We first used the PC algorithm to estimate CPDAGs from data. Then one may try to estimate the direction of an undirected edge using interventions and produce a DAG. In this comparison, however, we simply counted an undirected edge between two nodes in a CPDAG as an expected edge, provided that there is a corresponding directed edge between the two nodes in the true DAG. Thus, the reported result is the best (or an upper bound) one can obtain by a two-step PC algorithm based method (PC-based method). The performance of this PC-based method applied to our simulated data sets is shown in Table 2.2. Unlike graphs selected by criterion (2.9), graphs learned by the PC-based method generally have fewer edges than the true models. So to make fair comparisons, we selected graphs constructed by the CD algorithm to match the total number of edges of graphs learned by the PC-based method. The corresponding results are also presented in Table 2.2. It can be easily seen that the CD algorithm outperforms the PC-based method in all cases of our simulations. Graphs estimated using our method have both higher TPRs and lower FDRs. This result shows the advantage of using experimental data in an integrated penalized likelihood method. In addition, we compared the performance of the CD algorithm and the PC-based method on observational data (see Table 2.8 in Section 2.7.3). We found

that our method still outperforms the PC-based method except for $\beta_{ij} = 1.0$ or $p = 20$.

Table 2.2: The average performance of the two-step PC-based method and the average performance of the CD algorithm when models are selected to match the number of edges of those learned by the PC-based method (sample size n=5p)

p	β_{ij}	PC-based method			CD algorithm		
		P	TPR	FDR	P	TPR	FDR
20	0.2	7.6	0.103 (0.042)	0.443 (0.262)	8.3	0.115 (0.044)	0.442 (0.184)
	0.5	18.4	0.313 (0.049)	0.311 (0.134)	19.8	0.383 (0.095)	0.227 (0.148)
	1.0	15.7	0.290 (0.061)	0.254 (0.172)	15.7	0.318 (0.103)	0.183 (0.088)
50	0.2	53.3	0.221 (0.037)	0.585 (0.071)	52.8	0.299 (0.032)	0.430 (0.072)
	0.5	70.3	0.409 (0.081)	0.422 (0.082)	72.5	0.557 (0.117)	0.233 (0.109)
	1.0	54.7	0.313 (0.042)	0.427 (0.054)	50.5	0.355 (0.094)	0.296 (0.080)
100	0.2	173.8	0.399 (0.050)	0.542 (0.051)	173.5	0.610 (0.034)	0.297 (0.026)
	0.5	153.1	0.456 (0.053)	0.405 (0.048)	154.6	0.596 (0.077)	0.231 (0.066)
	1.0	107.8	0.328 (0.069)	0.396 (0.096)	107.2	0.513 (0.042)	0.041 (0.051)
200	0.2	429.1	0.506 (0.030)	0.528 (0.028)	431.8	0.815 (0.053)	0.245 (0.051)
	0.5	351.4	0.493 (0.075)	0.438 (0.085)	357.4	0.734 (0.093)	0.181 (0.068)
	1.0	235.3	0.335 (0.056)	0.433 (0.069)	234.8	0.561 (0.059)	0.043 (0.046)

Note: The numbers in parentheses are the standard deviations across 10 data sets.

We also compared the running time for both methods. Table 2.3 summarizes the CPU time for one run of each algorithm averaged over 10 data sets. Each run of the CD algorithm uses a sequence of 50 λ's with $\lambda_{\min}/\lambda_{\max} = 0.001$. The CD algorithm is implemented in R with the majority of its core computation executed in C. The PC algorithm we used was implemented by Kalisch et al. (2012) in the R package *pcalg*. The running time for the PC algorithm depends on the argument *u2pd*, which we assume to be *rand* (see online manuals for further details). According to Table 2.3, the average CPU time for the PC algorithm is faster than the CD algorithm. However, considering that the CD algorithm estimates 50 (or more generally a sequence of) graphical models in each run, it is

on average at least as fast as the PC algorithm for estimating a single graph.

Table 2.3: Comparison of average CPU time (in seconds) between the CD algorithm and the PC algorithm (sample size n=5p)

β_{ij}	CD algorithm				PC algorithm			
	$p = 20$	$p = 50$	$p = 100$	$p = 200$	$p = 20$	$p = 50$	$p = 100$	$p = 200$
0.2	0.09	1.32	17.54	255.09	0.04	0.28	4.18	28.74
0.5	0.15	4.54	112.69	1938.23	0.09	1.23	9.67	76.94
1.0	0.32	10.05	193.17	4595.95	0.09	0.97	5.10	33.73
mean	0.19	5.30	107.80	2263.09	0.07	0.83	6.32	46.47

Recently, Shojaie and Michailidis (2010) developed an approach to estimate directed acyclic graphs assuming a known ordering of the variables, which we will refer to as the KO method. Knowing the ordering greatly simplifies the structure learning problem. Following their formulation, we can simply estimate the coefficient matrix $\mathbf{B}$ (and thus the structure of directed graphs) by regressing each variable on all preceding variables in a given ordering. Hence, the problem of estimating directed graphs becomes $p - 1$ separate lasso problems, which can be solved efficiently using either the LARS algorithm (Efron et al. 2004) or the pathwise coordinate descent algorithm (Friedman et al. 2007). To obtain an estimate of a directed graph, Shojaie and Michailidis (2010) proposed to use

$$\lambda_i(\alpha) = 2\tilde{n}^{-1/2} Z^*_{\alpha/[2p(i-1)]} \tag{2.15}$$

as the penalty for the i^{th} individual lasso problem, where $\tilde{n}$ is the sample size, Z^*_q is the $(1-q)^{\text{th}}$ quantile of the standard normal distribution, and α is a parameter controlling the probability of falsely joining two ancestral sets in a graph (see Shojaie and Michailidis 2010). We applied their method to our simulated data sets. Since the criterion (2.15) led to over-sparse solutions when applied to our data sets with a limited sample size ($n = 5p$), we thus scaled down the tuning

parameters $\lambda_i(\alpha)$ in (2.15) proportionately and the results are summarized in Table 2.1 (KO method). The α level was chosen to be 0.1 as suggested by Shojaie and Michailidis (2010). As anticipated, most of the results obtained by assuming a known ordering are clearly better than the results of the CD algorithm. However, almost all TPRs from the CD algorithm are above 75% of those from the KO method. Furthermore, the CD algorithm seemed to outperform the KO method when $p = 200$ and $\beta_{ij} = 1.0$.

2.5.2 Large sample sizes

We also carried out a simulation study using a large sample size $n = 6000$. The setup for this study is the same as the one used for sample size $n = 5p$.

Table 2.4 summarizes the average performance of the CD algorithm over 10 data sets using 6000 samples. A comparison between Table 2.1 and Table 2.4 reveals that the accuracy of estimation can be greatly improved if large sample size is available. Given sufficient amount of data, the percentage of true positive edges that we can detect is close to or well above 90% for most cases. On the other hand, all FDRs in Table 2.4 are less than 20%. Cases where the signal-to-noise ratios are small ($\beta_{ij} = 0.2$) can be estimated much more accurately compared to Table 2.1. The only scenario where TPR falls below 80% is when $p = 100$ and $\beta_{ij} = 1.0$. However, even in this case, we were able to increase TPR to 0.86, at the expense of 4% increase in FDR, by reducing the thresholding parameter α of model selection. If we ignore the directionality, almost all the edges in the true graphs are included in our estimated models. The last two columns of Table 2.4 report the average TPRs and FDRs for DAGs estimated by the KO method of Shojaie and Michailidis (2010). Their model selection criterion (2.15) works very well with large sample sizes. Surprisingly, there are cases where the estimation accuracy of the CD algorithm is better than that of the KO method, such as $p = 20$ and $\beta_{ij} = 1.0$. One consequence of knowing the ordering is that no

Table 2.4: The average number of predicted (P), expected (E), reversed (R), missed (M) and false positive (FP) edges and the average TPRs and FDRs for DAGs learned by the CD algorithm (sample size n=6000)

p	β_{ij}	CD algorithm							KO method	
		P	E	R	M	FP	TPR	FDR	TPR	FDR
20	0.2	40.0	38.5	1.4	0.1	0.1	0.963 (0.036)	0.038 (0.032)	1.000	0.000
	0.5	39.8	36.8	2.0	1.2	1.0	0.920 (0.101)	0.075 (0.083)	1.000	0.007
	1.0	39.9	38.8	0.8	0.4	0.3	0.970 (0.037)	0.027 (0.031)	0.883	0.023
50	0.2	100.7	89.5	10.5	0.0	0.7	0.895 (0.036)	0.111 (0.037)	1.000	0.000
	0.5	102.2	84.7	10.7	4.6	6.8	0.847 (0.062)	0.167 (0.085)	0.999	0.008
	1.0	101.3	89.3	5.7	5.0	6.3	0.893 (0.028)	0.115 (0.062)	0.795	0.026
100	0.2	200.3	178.5	20.8	0.7	1.0	0.892 (0.033)	0.109 (0.034)	0.989	0.000
	0.5	214.1	170.8	21.5	7.7	21.8	0.854 (0.064)	0.199 (0.085)	0.982	0.008
	1.0	170.7	157.0	8.6	34.4	5.1	0.785 (0.108)	0.078 (0.048)	0.459	0.022
200	0.2	399.5	375.3	23.9	0.8	0.3	0.938 (0.020)	0.061 (0.018)	0.995	0.000
	0.5	419.9	358.3	31.9	9.8	29.7	0.896 (0.041)	0.145 (0.063)	0.992	0.003
	1.0	349.4	320.5	16.8	62.7	12.1	0.801 (0.078)	0.081 (0.053)	0.464	0.021

Note:
The numbers in parentheses are the standard deviations across 10 data sets. As a comparison, the last two columns report the average TPRs and FDRs for DAGs estimated by the approach of Shojaie and Michailidis (2010) assuming the ordering of the variables is known.

reversed edges will be predicted by the KO method. However, if we ignore the directions of predicted edges, the coordinate descent results are close to or even better than the results of the KO method, which in some sense can be regarded as the best possible results we might expect.

Figure 2.3 presents the results of ROC analysis for the CD algorithm. According to the ROC curves, we are confident that graphs learned by the CD algorithm using large sample sizes do not have many false positives as long as they are not too complex compared to the true graphs.

Results of the comparison between the PC-based method and the CD algorithm are reported in Table 2.5. With 6000 samples, the CD algorithm out-

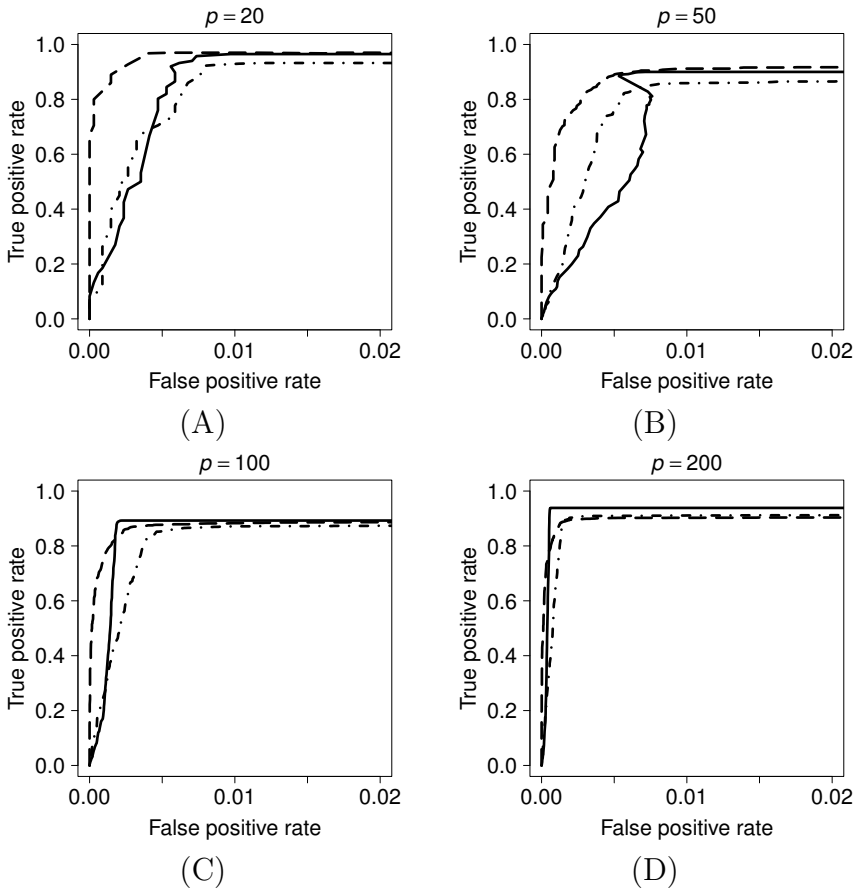


Figure 2.3: ROC curves for $\beta_{ij} = 0.2$ (solid lines), $\beta_{ij} = 0.5$ (dotdashed lines) and $\beta_{ij} = 1.0$ (long dashed lines), sample size n=6000. The TPRs stabilize for FPRs > 0.02.

Table 2.5: The average performance of the two-step PC-based method and the average performance of the CD algorithm when models are selected to match the number of edges of those learned by the PC-based method (sample size n=6000)

p	β_{ij}	PC-based method			CD algorithm		
		P	TPR	FDR	P	TPR	FDR
20	0.2	40.2	0.555 (0.110)	0.447 (0.111)	40.3	0.963 (0.032)	0.045 (0.026)
	0.5	34.5	0.537 (0.072)	0.374 (0.092)	35.1	0.823 (0.117)	0.064 (0.071)
	1.0	31.2	0.490 (0.097)	0.373 (0.106)	31.9	0.795 (0.090)	0.003 (0.010)
50	0.2	101.7	0.527 (0.045)	0.482 (0.046)	101.7	0.895 (0.035)	0.120 (0.042)
	0.5	90.5	0.567 (0.095)	0.374 (0.094)	90.4	0.787 (0.098)	0.132 (0.053)
	1.0	75.8	0.447 (0.061)	0.409 (0.077)	73.4	0.698 (0.068)	0.047 (0.059)
100	0.2	207.5	0.520 (0.059)	0.499 (0.053)	207.6	0.892 (0.033)	0.140 (0.034)
	0.5	176.8	0.479 (0.077)	0.460 (0.065)	178.8	0.737 (0.113)	0.178 (0.091)
	1.0	135.2	0.376 (0.060)	0.447 (0.057)	133.6	0.644 (0.066)	0.033 (0.044)
200	0.2	436.9	0.551 (0.040)	0.495 (0.035)	435.6	0.938 (0.020)	0.138 (0.030)
	0.5	371.0	0.474 (0.028)	0.489 (0.035)	372.9	0.809 (0.080)	0.133 (0.060)
	1.0	275.7	0.340 (0.037)	0.504 (0.064)	274.1	0.662 (0.051)	0.032 (0.046)

Note:
The numbers in parentheses are the standard deviations across 10 data sets.

performs the PC-based method by a large margin. Almost all TPRs of graphs estimated by the CD algorithm are 25% above those of graphs estimated by the PC-based method and their FDRs are much lower. Table 2.6 summarizes the running time comparison for one run of each algorithm using 6000 samples. Each run of the CD algorithm uses a sequence of 50 λ's with $\lambda_{\min}/\lambda_{\max} = 0.001$. Interestingly, for large networks ($p = 100, 200$), the CD algorithm runs faster on average compared to the running time using small sample size (see Table 2.3), which may imply faster convergence. These results illustrate the efficiency of our CD algorithm: Using 6000 samples, it can estimate 50 DAGs for $p = 100$ in about a minute, with a high accuracy (Table 2.4).

Table 2.6: Comparison of average CPU time (in seconds) between the CD algorithm and the PC algorithm (sample size n=6000)

β_{ij}	CD algorithm				PC algorithm			
	$p = 20$	$p = 50$	$p = 100$	$p = 200$	$p = 20$	$p = 50$	$p = 100$	$p = 200$
0.2	1.12	6.27	35.56	303.49	1.25	10.31	45.64	104.29
0.5	1.17	7.33	66.37	1017.54	1.28	13.68	77.19	555.58
1.0	1.26	7.93	86.52	1728.39	1.01	8.70	41.31	188.44
mean	1.18	7.18	62.82	1016.47	1.18	10.90	54.71	282.77

2.6 Conclusions

We have developed a method to estimate the structure of causal Gaussian networks using a penalized likelihood approach with the adaptive lasso penalty. Without knowing the ordering of the variables, we rely on experimental data to retrieve information about the directionality of the edges in a graph. The acyclic constraint on the structure of Bayesian networks presents a challenge to the maximization of the penalized log-likelihood function. A blockwise coordinate descent algorithm has been developed for this optimization problem. The algorithm runs reasonably fast and can be applied to large networks. Simulations have been conducted to demonstrate the performance of our method for various sizes of Bayesian networks.

We have also established asymptotic properties for the penalized likelihood estimator of the coefficient matrix of Gaussian Bayesian networks, assuming that the number of variables p is fixed. Asymptotic theory for the estimator if p is allowed to grow as a function of the sample size remains to be established in the future. This type of asymptotic problems has been studied in various settings of undirected graph and precision matrix estimation (e.g., Meinshausen and Bühlmann 2006; Lam and Fan 2009), where $p(n) = O(n^c)$ for some $c > 0$ or is of an even higher order. Following our current setup, however, we may need

to restrict our attention to the case where $0 < c < 1$ so that every variable will have sufficient interventional samples in the data matrix $\mathbf{X}$ as $n \to \infty$. The satisfactory results in our simulation for $p \geq 100$ and $n = 5p$ seem to suggest that our CD algorithm is effective even for $p > \sqrt{n}$. It will also be interesting to study the theoretical properties of this penalized likelihood approach when not all variables have experimental data, for which the concept of interventional Markov equivalence (Hauser and Bühlmann 2011) will be relevant.

Throughout this chapter, variables are assumed to be Gaussian. In the next chapter, we extend our penalized likelihood approach to discrete data types. Though the principal idea remains the same, a different algorithm needs to be developed to handle modeling details specific to discrete variables.

2.7 Appendix

2.7.1 Proofs

2.7.1.1 Proof of Proposition 2.1

We want to minimize $g = \frac{1}{2}\log\left[\left(\tilde{\beta}_{kj} - \xi_{kj}\right)^2 + \left(c_{kj} - \xi_{kj}^2\right)\right] + \gamma|\tilde{\beta}_{kj}|$ over $\tilde{\beta}_{kj}$. After differentiating g with respect to $\tilde{\beta}_{kj}$ and setting the derivative to zero, we obtain for $\tilde{\beta}_{kj} > 0$,

$$\gamma\tilde{\beta}_{kj}^2 - (2\gamma\xi_{kj} - 1)\tilde{\beta}_{kj} + (c_{kj}\gamma - \xi_{kj}) = 0, \tag{2.16}$$

and for $\tilde{\beta}_{kj} < 0$,

$$\gamma\tilde{\beta}_{kj}^2 - (2\gamma\xi_{kj} + 1)\tilde{\beta}_{kj} + (c_{kj}\gamma + \xi_{kj}) = 0. \tag{2.17}$$

Apparently, both (2.16) and (2.17) have the same discriminant $\Delta = 1 - 4(c_{kj} - \xi_{kj}^2)\gamma^2$. The only possible minimizers of g are 0, positive real roots of (2.16) or negative real roots of (2.17).

In the rest of the proof, we will only show that Proposition 2.1 holds when $\xi_{kj} \geq 0$. The proof for $\xi_{kj} < 0$ is analogous. First, consider $\xi_{kj} = 0$. It is easily seen that g is minimized at $\tilde{\beta}_{kj} = 0$, which is included in the third case of the proposition.

Now consider the case when $\xi_{kj} > 0$. In this case, let $\beta_1^* = \frac{\left(2\gamma\xi_{kj}-1\right)+\sqrt{\Delta}}{2\gamma}$ and $\beta_2^* = \frac{\left(2\gamma\xi_{kj}-1\right)-\sqrt{\Delta}}{2\gamma}$ be the two possible real roots of (2.16). If (2.16) has two real roots, β_2^* is a local maximum. Also note that if $\xi_{kj} > 0$, (2.17) can only have positive real roots. Thus, g can only be minimized at 0 or β_1^* if it is real. Now we only need to find out when 0 or β_1^* minimizes g. There are four cases:

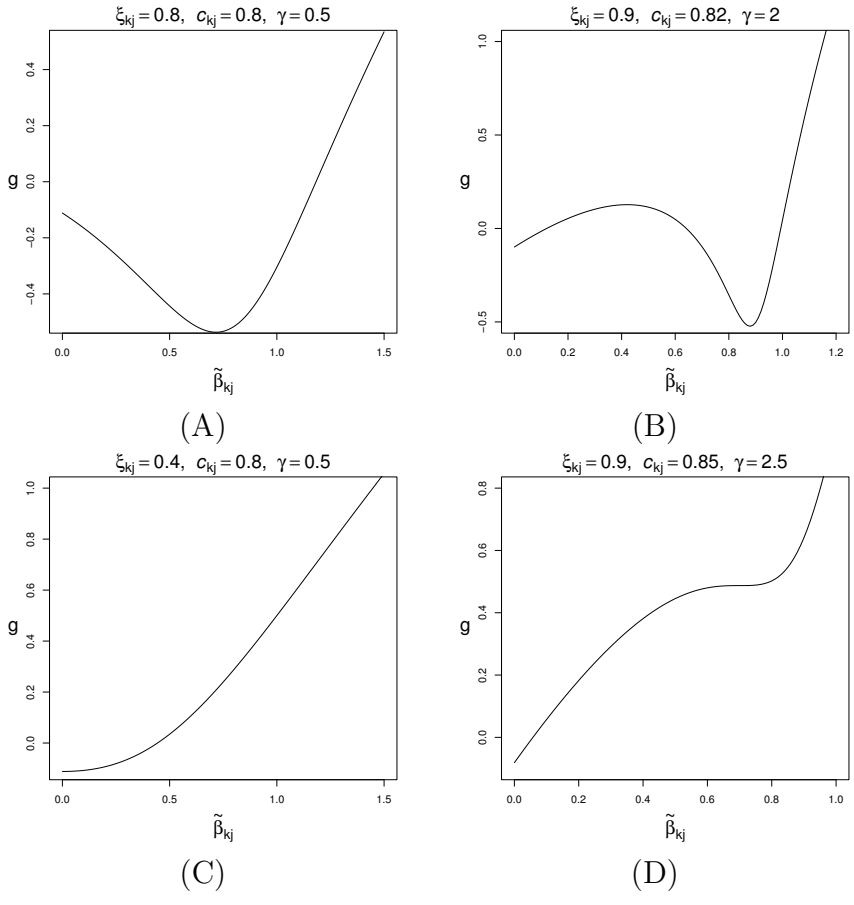


Figure 2.4: Examples illustrating different scenarios for minimizing g over $\tilde{\beta}_{kj}$ when $\xi_{kj} > 0$.

Case 1. $\Delta > 0$ and $\beta_1^* > 0 > \beta_2^*$: This is equivalent to $0 < \gamma < \xi_{kj}/c_{kj}$. In this

case, we have $g(\beta_1^*) < g(0)$ (see Figure 2.4A).

Case 2. $\Delta > 0$ and $\beta_1^* > \beta_2^* \geq 0$: This is equivalent to $\xi_{kj}/c_{kj} \leq \gamma < \left(2\sqrt{c_{kj} - \xi_{kj}^2}\right)^{-1}$ and $\gamma > (2\xi_{kj})^{-1}$. In this case, β_1^* is a local minimum and β_2^* is a local maximum (see Figure 2.4B). Thus, we need to compare $g(\beta_1^*)$ with $g(0)$ to determine $\arg\min_{\tilde{\beta}_{kj}} g$.

Case 3. $\Delta > 0$ and $0 \geq \beta_1^* > \beta_2^*$: This is equivalent to $\xi_{kj}/c_{kj} \leq \gamma < \left(2\sqrt{c_{kj} - \xi_{kj}^2}\right)^{-1}$ and $\gamma \leq (2\xi_{kj})^{-1}$. In this case, neither β_1^* nor β_2^* is positive. So $\arg\min_{\tilde{\beta}_{kj}} g = 0$ (see Figure 2.4C).

Case 4. $\Delta \leq 0$: This is equivalent to $\gamma \geq \left(2\sqrt{c_{kj} - \xi_{kj}^2}\right)^{-1}$. If $\Delta < 0$, clearly $\arg\min_{\tilde{\beta}_{kj}} g = 0$. If $\Delta = 0$, $\beta_1^* = \beta_2^*$ is an inflection point if they are positive (see Figure 2.4D). So it is also true that $\arg\min_{\tilde{\beta}_{kj}} g = 0$.

Therefore, we have shown that Proposition 2.1 holds.

2.7.1.2 Proof of Theorem 2.2

We prove the first claim (2.13) by contradiction. Suppose $\boldsymbol{\theta} \neq \boldsymbol{\theta}^*$ and $f(\mathbf{x}|\boldsymbol{\theta}_k) = f(\mathbf{x}|\boldsymbol{\theta}_k^*)$ a.e. for $k = 1, \ldots, p$. Let $\mathcal{S}(\mathcal{G})$ denote the set of topological sorts of a DAG $\mathcal{G}$. Recall that we denote by $\mathcal{G}_{\boldsymbol{\phi}}$ and $\mathcal{G}_{\boldsymbol{\phi}^*}$ the DAGs induced by $\boldsymbol{\phi}$ and $\boldsymbol{\phi}^*$, respectively. There are two cases for $\mathcal{G}_{\boldsymbol{\phi}}$ and $\mathcal{G}_{\boldsymbol{\phi}^*}$ if $\boldsymbol{\theta}$ is different from $\boldsymbol{\theta}^*$:

Case 1: $\mathcal{S}(\mathcal{G}_{\boldsymbol{\phi}}) \cap \mathcal{S}(\mathcal{G}_{\boldsymbol{\phi}^*}) \neq \varnothing$. Let $\sqsubset \in \mathcal{S}(\mathcal{G}_{\boldsymbol{\phi}}) \cap \mathcal{S}(\mathcal{G}_{\boldsymbol{\phi}^*})$, i.e., an ordering compatible with both $\mathcal{G}_{\boldsymbol{\phi}}$ and $\mathcal{G}_{\boldsymbol{\phi}^*}$. Assume without loss of generality that in this ordering $i \prec j$ if $i < j$. Apparently, $\sqsubset$ is also compatible with $\mathcal{G}_{\boldsymbol{\phi}_k}$ and $\mathcal{G}_{\boldsymbol{\phi}_k^*}$ for $k = 1, \ldots, p$. Then we can write $f(\mathbf{x}|\boldsymbol{\theta}_k) = \prod_{i=1}^p f(x_i|x_1, \ldots, x_{i-1}, \boldsymbol{\theta}_k) = \prod_{i=1}^p f(x_i|\Pi_i^{\mathcal{G}_{\boldsymbol{\phi}_k}}, \boldsymbol{\theta}_k)$ and $f(\mathbf{x}|\boldsymbol{\theta}_k^*) = \prod_{i=1}^p f(x_i|x_1, \ldots, x_{i-1}, \boldsymbol{\theta}_k^*) = \prod_{i=1}^p f(x_i|\Pi_i^{\mathcal{G}_{\boldsymbol{\phi}_k^*}}, \boldsymbol{\theta}_k^*)$. Since $f(\mathbf{x}|\boldsymbol{\theta}_k) = f(\mathbf{x}|\boldsymbol{\theta}_k^*)$, it follows that $\Pi_i^{\mathcal{G}_{\boldsymbol{\phi}_k}} = \Pi_i^{\mathcal{G}_{\boldsymbol{\phi}_k^*}}$ for all i and thus $\mathcal{G}_{\boldsymbol{\phi}_k} = \mathcal{G}_{\boldsymbol{\phi}_k^*}$ for all k. However, since $\boldsymbol{\theta} \neq \boldsymbol{\theta}^*$, there exists some k such that $\boldsymbol{\theta}_k \neq \boldsymbol{\theta}_k^*$. Therefore, there exists a k such that, the common multivariate normal density $f(\mathbf{x}|\boldsymbol{\theta}_k) = f(\mathbf{x}|\boldsymbol{\theta}_k^*)$, factorized according to a common structure $\mathcal{G}_{\boldsymbol{\phi}_k} = \mathcal{G}_{\boldsymbol{\phi}_k^*}$, can be parameterized by two

different parameters $\boldsymbol{\theta}_k$ and $\boldsymbol{\theta}_k^*$. This is apparently impossible.

Case 2: $\mathcal{S}(\mathcal{G}_{\boldsymbol{\phi}}) \cap \mathcal{S}(\mathcal{G}_{\boldsymbol{\phi}^*}) = \varnothing$, that is, none of the orderings of $\mathcal{G}_{\boldsymbol{\phi}^*}$ is compatible with $\mathcal{G}_{\boldsymbol{\phi}}$. In this case, there must exist a pair of indices (i, j) such that in $\mathcal{G}_{\boldsymbol{\phi}^*}$ $X_i \in \text{an}(X_j)$, but in $\mathcal{G}_{\boldsymbol{\phi}}$ X_j is a non-descendant of X_i. Then X_j is independent of X_i in $f(\mathbf{x}|\boldsymbol{\theta}_i)$, since in $\mathcal{G}_{\boldsymbol{\phi}_i}$ X_i has no parents and X_j is a non-descendant of X_i. So $\text{Cov}(X_i, X_j) = 0$ in $f(\mathbf{x}|\boldsymbol{\theta}_i)$. However, in $\mathcal{G}_{\boldsymbol{\phi}_i^*}$ we still have $X_i \in \text{an}(X_j)$. It is easy to show that $\text{Cov}(X_i, X_j) = \beta_{i\to j}^* \text{Var}(X_i) \neq 0$ in $f(\mathbf{x}|\boldsymbol{\theta}_i^*)$ since $\boldsymbol{\theta}^*$ is natural. Therefore, there exists $1 \leq i \leq p$ such that $f(\mathbf{x}|\boldsymbol{\theta}_i) \neq f(\mathbf{x}|\boldsymbol{\theta}_i^*)$, which contradicts our assumption.

So in both *case 1* and *case 2* we have a contradiction. Thus, the first claim holds.

For the second claim (2.14), first note that by the law of large numbers,

$$\frac{1}{n}(L(\boldsymbol{\theta}) - L(\boldsymbol{\theta}^*)) = \sum_{k=1}^{p} \frac{n_k}{n} \frac{1}{n_k} \sum_{h \in \mathcal{I}_k} \log \frac{f(\mathbf{X}_{[h,\cdot]}|\boldsymbol{\theta}_k)}{f(\mathbf{X}_{[h,\cdot]}|\boldsymbol{\theta}_k^*)} \longrightarrow_p \sum_{k=1}^{p} \alpha_k \mathbf{E}_{\boldsymbol{\theta}_k^*} \left[\log \frac{f(\mathbf{Y}|\boldsymbol{\theta}_k)}{f(\mathbf{Y}|\boldsymbol{\theta}_k^*)} \right], \tag{2.18}$$

where $\mathbf{Y}$ is a random vector with probability density $f(\mathbf{x}|\boldsymbol{\theta}_k^*)$. Then the desired result follows immediately using Jensen's inequality and (2.13).

2.7.1.3 Proof of Theorem 2.3

Define $a_n = 1/\sqrt{n}$ and $\mathcal{B} = \{j : \phi_j^* \neq 0\}$. Let

$$\mathbf{I}(\boldsymbol{\theta}_k) = \mathbf{E}_{\boldsymbol{\theta}_k} \left\{ \left[\frac{\partial}{\partial \boldsymbol{\theta}_k} \log f(\mathbf{x}|\boldsymbol{\theta}_k) \right] \left[\frac{\partial}{\partial \boldsymbol{\theta}_k} \log f(\mathbf{x}|\boldsymbol{\theta}_k) \right]^T \right\}$$

be the Fisher information matrix.

Consider $\boldsymbol{\theta} = (\boldsymbol{\phi}, \boldsymbol{\sigma}^2, \widetilde{\boldsymbol{\sigma}}^2) \in \text{nb}(\boldsymbol{\theta}^*)$, where $\text{nb}(\boldsymbol{\theta}^*)$ is an arbitrarily small neighborhood of $\boldsymbol{\theta}^*$. The components of $\boldsymbol{\phi}$ must satisfy $\phi_i \phi_i^* > 0$ if $\phi_i^* \neq 0$ $(i = 1, \ldots, d)$, since otherwise $\|\boldsymbol{\theta} - \boldsymbol{\theta}^*\| \geq \min_{j:\phi_j^* \neq 0} |\phi_j^*|$. In particular, this implies that if

$\boldsymbol{\theta} \in \text{nb}(\boldsymbol{\theta}^*)$, $i \to j$ in $\mathcal{G}_{\boldsymbol{\phi}}$ for all $i \to j$ in $\mathcal{G}_{\boldsymbol{\phi}^*}$ and thus $\mathcal{G}_{\boldsymbol{\phi}}$ and $\mathcal{G}_{\boldsymbol{\phi}^*}$ have compatible orderings. If we restrict to the lower dimensional space $\boldsymbol{\Omega}_k = \{\boldsymbol{\theta}_k : \boldsymbol{\theta} \in \boldsymbol{\Omega}\}$, the same arguments apply to an arbitrarily small neighborhood of $\boldsymbol{\theta}_k^*$ in this space, that is, $\mathcal{G}_{\boldsymbol{\phi}_k}$ and $\mathcal{G}_{\boldsymbol{\phi}_k^*}$ have compatible orderings. Then it follows from the arguments used in *Case 1* in the proof of Theorem 2.2 that $f(\mathbf{x}|\boldsymbol{\theta}_k) \neq f(\mathbf{x}|\boldsymbol{\theta}_k^*)$ for $\boldsymbol{\theta}_k \in \text{nb}(\boldsymbol{\theta}_k^*) \backslash \{\boldsymbol{\theta}_k^*\}$. Since f is a Gaussian density, it follows that $\mathbf{I}(\boldsymbol{\theta}_k^*)$ is positive definite for all k.

Let $\mathbf{u} \in \{\mathbf{u} : \boldsymbol{\theta}^* + a_n \mathbf{u} \in \boldsymbol{\Omega}\}$ and denote its components by u_j. Let $\mathbf{u}_k$ be the vector defined in the same way as $\boldsymbol{\theta}_k$, $k = 1, \ldots, p$. Note that $\sum_{k=1}^p \|\mathbf{u}_k\|^2 \geq \|\mathbf{u}\|^2$. Let $\delta_{\min}^k > 0$ be the minimal eigenvalue of $\mathbf{I}(\boldsymbol{\theta}_k^*)$ and $\rho = \min_k(\alpha_k \delta_{\min}^k/2)$. Then

$$\sum_{k=1}^{p} \frac{\alpha_k}{2} \mathbf{u}_k^T \mathbf{I}(\boldsymbol{\theta}_k^*) \mathbf{u}_k \geq \sum_{k=1}^{p} \frac{\alpha_k}{2} \delta_{\min}^k \|\mathbf{u}_k\|^2 \geq \rho \sum_{k=1}^{p} \|\mathbf{u}_k\|^2 \geq \rho \|\mathbf{u}\|^2. \tag{2.19}$$

Now we study the behavior of $R(\boldsymbol{\theta})$ in a small neighborhood of the true value

$\boldsymbol{\theta}^*$ by expanding $L(\boldsymbol{\theta})$ around $\boldsymbol{\theta}^*$. We have, as $n \to \infty$,

$$
\begin{aligned}
& R(\boldsymbol{\theta}^* + a_n \mathbf{u}) - R(\boldsymbol{\theta}^*) \\
\leq\ & L(\boldsymbol{\theta}^* + a_n \mathbf{u}) - L(\boldsymbol{\theta}^*) - \lambda_n \sum_{j \in \mathcal{B}} \tau_j (|\phi_j^* + a_n u_j| - |\phi_j^*|) \\
=\ & \sum_{k=1}^{p} [L_k(\boldsymbol{\theta}_k^* + a_n \mathbf{u}_k) - L_k(\boldsymbol{\theta}_k^*)] - \lambda_n a_n \sum_{j \in \mathcal{B}} \tau_j u_j \mathrm{sgn}(\phi_j^*) \\
=\ & \sum_{k=1}^{p} \left[a_n L_k'(\boldsymbol{\theta}_k^*)^T \mathbf{u}_k - \frac{1}{2} n_k a_n^2 \mathbf{u}_k^T \mathbf{I}(\boldsymbol{\theta}_k^*) \mathbf{u}_k \left\{1 + o_p(1)\right\} \right] \\
& - \lambda_n a_n \sum_{j \in \mathcal{B}} \tau_j u_j \mathrm{sgn}(\phi_j^*) \\
=\ & \sum_{k=1}^{p} \left[\sqrt{\alpha_k} \frac{L_k'(\boldsymbol{\theta}_k^*)^T}{\sqrt{n_k}} \mathbf{u}_k \left\{1 + o_p(1)\right\} - \frac{\alpha_k}{2} \mathbf{u}_k^T \mathbf{I}(\boldsymbol{\theta}_k^*) \mathbf{u}_k \left\{1 + o_p(1)\right\} \right] \\
& - \frac{\lambda_n}{\sqrt{n}} \sum_{j \in \mathcal{B}} \tau_j u_j \mathrm{sgn}(\phi_j^*) \\
\leq\ & \sum_{k=1}^{p} \left[\sqrt{\alpha_k} \frac{L_k'(\boldsymbol{\theta}_k^*)^T}{\sqrt{n_k}} \mathbf{u}_k \left\{1 + o_p(1)\right\} \right] - \rho \|\mathbf{u}\|^2 \left\{1 + o_p(1)\right\} \\
& - \frac{\lambda_n}{\sqrt{n}} \sum_{j \in \mathcal{B}} \tau_j u_j \mathrm{sgn}(\phi_j^*). \qquad (2.20)
\end{aligned}
$$

The last inequality is due to (2.19). From the central limit theorem, $n_k^{-1/2} L_k'(\boldsymbol{\theta}_k^*) = O_p(1)$ for all k. By assumption, $\tau_j = O_p(1)$ for $j = 1, \ldots, d$ and $\lambda_n / \sqrt{n} = o_p(1)$. Therefore, for a sufficiently large C, the second order term in the last line of (2.20) dominates the first and third terms uniformly in $\{\mathbf{u} : \|\mathbf{u}\| = C, \boldsymbol{\theta}^* + a_n \mathbf{u} \in \boldsymbol{\Omega}\}$. Hence, for any given $\varepsilon > 0$, there exists a sufficiently large C such that

$$
P\left(\sup_{\|\mathbf{u}\| = C} R(\boldsymbol{\theta}^* + a_n \mathbf{u}) < R(\boldsymbol{\theta}^*) \right) \geq 1 - \varepsilon, \qquad (2.21)
$$

which implies that with probability at least $1 - \varepsilon$, there exists a local maximizer $\hat{\boldsymbol{\theta}}$ of $R(\boldsymbol{\theta})$ in the ball $\{\boldsymbol{\theta}^* + a_n \mathbf{u} \in \boldsymbol{\Omega} : \|\mathbf{u}\| \leq C\}$. Thus, there exists a local maximizer $\hat{\boldsymbol{\theta}}$ of $R(\boldsymbol{\theta})$ such that $\|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*\| = O_p(n^{-1/2})$.

2.7.1.4 Proof of Theorem 2.4

We omit the proof of the first part of Theorem 2.4, since it is similar to that of Theorem 2.3. To prove the second part, let $\mathcal{B} = \{j : \phi_j^* \neq 0\}$. For notational ease, let us first, by permuting the indices, rewrite the parameter $\boldsymbol{\theta}$ as $\boldsymbol{\theta} = (\boldsymbol{\theta}_a, \boldsymbol{\theta}_b) = (\boldsymbol{\phi}_{\mathcal{A}}, \boldsymbol{\phi}_{\mathcal{B}}, \boldsymbol{\sigma}^2, \widetilde{\boldsymbol{\sigma}}^2)$, where $\boldsymbol{\theta}_a = \boldsymbol{\phi}_{\mathcal{A}}$ and $\boldsymbol{\theta}_b = (\boldsymbol{\phi}_{\mathcal{B}}, \boldsymbol{\sigma}^2, \widetilde{\boldsymbol{\sigma}}^2)$. Let $r = |\mathcal{A}|$ be the number of zero elements of $\boldsymbol{\phi}^*$.

Now we only need to show that with probability tending to 1, for any $\boldsymbol{\theta}_b$ satisfying $\|\boldsymbol{\theta}_b - \boldsymbol{\theta}_b^*\| = O_p(n^{-1/2})$ and any constant $C > 0$,

$$(\mathbf{0}, \boldsymbol{\theta}_b) = \underset{\|\boldsymbol{\theta}_a\| \leq C/\sqrt{n}}{\arg\max}\, R\big((\boldsymbol{\theta}_a, \boldsymbol{\theta}_b)\big). \tag{2.22}$$

To establish (2.22), we again study the behavior of $R(\boldsymbol{\theta})$ around the point $(\mathbf{0}, \boldsymbol{\theta}_b)$ by expanding $L(\boldsymbol{\theta})$ around $(\mathbf{0}, \boldsymbol{\theta}_b)$. Let $a_n = 1/\sqrt{n}$, $\boldsymbol{\theta}^o = (\mathbf{0}, \boldsymbol{\theta}_b)$, and $\boldsymbol{\theta} = \boldsymbol{\theta}^o + a_n \mathbf{u} \in \boldsymbol{\Omega}$, where $\mathbf{u} = (\mathbf{u}_a, \mathbf{u}_b)$, $\|\mathbf{u}\| \leq C$ and $\mathbf{u}_b = \mathbf{0}$. Then we have the following result similar to that in Theorem 2.3:

$$\begin{aligned}
& R(\boldsymbol{\theta}^o + a_n \mathbf{u}) - R(\boldsymbol{\theta}^o) \\
= & \sum_{k=1}^{p} \left[\sqrt{\alpha_k} \frac{L_k'(\boldsymbol{\theta}_k^o)^T}{\sqrt{n_k}} \mathbf{u}_k \left\{1 + o_p(1)\right\} - \frac{\alpha_k}{2} \mathbf{u}_k^T \mathbf{I}(\boldsymbol{\theta}_k^o) \mathbf{u}_k \left\{1 + o_p(1)\right\} \right] \\
& - \frac{\lambda_n}{\sqrt{n}} \sum_{j=1}^{r} \tau_j |u_j| \\
= & \sum_{k=1}^{p} \left[\sqrt{\alpha_k} \frac{L_k'(\boldsymbol{\theta}_k^o)^T}{\sqrt{n_k}} \mathbf{u}_k \left\{1 + o_p(1)\right\} - \frac{\alpha_k}{2} \mathbf{u}_k^T \mathbf{I}(\boldsymbol{\theta}_k^o) \mathbf{u}_k \left\{1 + o_p(1)\right\} \right] \\
& - \frac{\lambda_n}{\sqrt{n}} n^{\gamma/2} \sum_{j=1}^{r} |\sqrt{n}\tilde{\phi}_j|^{-\gamma} |u_j|.
\end{aligned} \tag{2.23}$$

Note that both the first and second terms in the last line of (2.23) are on the order of $O_p(1)$ for any fixed constant C. Since $\tilde{\phi}_j$ is $\sqrt{n}$-consistent, we have $|\sqrt{n}\tilde{\phi}_j| = O_p(1)$, for $j = 1, \ldots, r$. Then the third term in the last line of (2.23)

is on the order of $\lambda_n n^{(\gamma-1)/2} \to \infty$. Therefore, (2.22) holds, and the proof is complete.

2.7.2 Supplementary algorithm

The following algorithm is used in the second step of the CD algorithm to impose the acyclic constraint. The time complexity is $O(V + E)$.

Algorithm 2.2 Check whether a DAG $\mathcal{G}$ remains acyclic if an edge $i \to j$ is added.

```
function CYCLE(G, i, j)
    for v ∈ V\{i} do
        C_v ← 0
    end for
    C_i ← 1
    Q ← ∅
    ENQUEUE(Q, i)
    while Q ≠ ∅ do
        u ← DEQUEUE(Q)
        for v ∈ Π_u^G do
            if v = j then
                return true
            else
                if C_v = 0 then
                    C_v ← 1
                    ENQUEUE(Q, v)
                end if
            end if
        end for
    end while
    return false
end function
```

2.7.3 Supplementary tables

The following two tables summarize the results of the CD algorithm on observational data as well as a comparison with the PC-based method on observational data.

Table 2.7: The average number of predicted (P), expected (E), reversed (R), missed (M) and false positive (FP) edges and the average TPRs and FDRs for DAGs learned by applying the CD algorithm to observational data (sample size n=5p)

p	β_{ij}	CD algorithm						
		P	E	R	M	FP	TPR	FDR
20	0.2	59.5	15.0	14.4	10.6	30.1	0.375 (0.068)	0.742 (0.056)
	0.5	37.8	16.9	11.6	11.5	9.3	0.423 (0.106)	0.556 (0.066)
	1.0	99.4	24.1	9.2	6.7	66.1	0.603 (0.130)	0.727 (0.091)
50	0.2	140.0	51.2	36.9	11.9	51.9	0.512 (0.064)	0.626 (0.075)
	0.5	103.9	62.7	23.8	13.5	17.4	0.627 (0.141)	0.397 (0.115)
	1.0	155.4	55.6	19.3	25.1	80.5	0.556 (0.089)	0.624 (0.091)
100	0.2	257.4	124.0	67.8	8.2	65.6	0.620 (0.059)	0.513 (0.054)
	0.5	223.2	143.8	35.9	20.3	43.5	0.719 (0.071)	0.349 (0.097)
	1.0	123.9	55.6	43.4	101.0	24.9	0.278 (0.096)	0.547 (0.091)
200	0.2	422.0	303.8	93.8	2.4	24.4	0.760 (0.029)	0.279 (0.047)
	0.5	435.7	334.1	46.5	19.4	55.1	0.835 (0.053)	0.231 (0.059)
	1.0	271.9	135.1	78.0	186.9	58.8	0.338 (0.080)	0.496 (0.095)

Note: The numbers in parentheses are the standard deviations across 10 data sets.

Table 2.8: Comparison of the average performance of the two-step PC-based method and that of the CD algorithm on observational data when models are selected to match the number of edges of those learned by the PC-based method (sample size n=5p)

p	β_{ij}	PC-based method			CD algorithm		
		P	TPR	FDR	P	TPR	FDR
20	0.2	8.9	0.130 (0.037)	0.403 (0.187)	8.8	0.118 (0.029)	0.456 (0.137)
	0.5	18.0	0.315 (0.057)	0.295 (0.126)	18.6	0.213 (0.060)	0.542 (0.099)
	1.0	16.4	0.255 (0.033)	0.367 (0.118)	16.2	0.228 (0.081)	0.429 (0.169)
50	0.2	54.3	0.238 (0.057)	0.561 (0.108)	55.4	0.308 (0.062)	0.447 (0.075)
	0.5	70.9	0.411 (0.082)	0.424 (0.077)	73.6	0.495 (0.143)	0.330 (0.144)
	1.0	52.0	0.303 (0.058)	0.421 (0.079)	47.3	0.255 (0.073)	0.462 (0.084)
100	0.2	174.4	0.403 (0.050)	0.538 (0.056)	175.4	0.557 (0.051)	0.365 (0.048)
	0.5	153.6	0.443 (0.071)	0.424 (0.079)	153.0	0.530 (0.099)	0.311 (0.090)
	1.0	104.1	0.273 (0.047)	0.477 (0.068)	104.8	0.245 (0.049)	0.531 (0.094)
200	0.2	430.9	0.506 (0.031)	0.530 (0.030)	433.3	0.761 (0.029)	0.298 (0.028)
	0.5	348.6	0.462 (0.046)	0.468 (0.066)	350.8	0.707 (0.083)	0.196 (0.053)
	1.0	234.5	0.305 (0.059)	0.482 (0.066)	237.1	0.301 (0.064)	0.493 (0.086)

Note: The numbers in parentheses are the standard deviations across 10 data sets.

CHAPTER 3

Learning Sparse Causal Discrete Networks from Experimental Data

In this chapter, we focus on estimating causal discrete Bayesian networks from experimental data. We present a principled generalization of the penalized likelihood methodology developed for Gaussian Bayesian networks to discrete data types. This chapter is organized as follows. Section 3.1 gives a brief introduction to the multinomial model commonly employed for modeling discrete Bayesian networks. We introduce in Section 3.2 the multi-logit model based on which we formulate the problem of estimating discrete Bayesian networks with the adaptive group lasso penalty. A blockwise coordinate descent algorithm is proposed in Section 3.3 and we solve each coordinate descent step by applying a quadratic approximation iteratively. Asymptotic theory parallel to the one developed for Gaussian Bayesian networks is established in Section 3.4. We report in Section 3.5 results of numerical evaluation of the discrete CD algorithm on three types of networks. The chapter is concluded with Section 3.6. Several technical proofs are relegated to Section 3.7.

3.1 Introduction to Discrete Bayesian Networks

In a discrete Bayesian network $\mathcal{G}$, the variable X_i is a factor with r_i levels denoted by $1, \ldots, r_i$, $i = 1, \ldots, p$. The set of parents of X_i, denoted by $\Pi_i^{\mathcal{G}}$, has a total of $q_i = \prod_{X_j \in \Pi_i^{\mathcal{G}}} r_j$ possible joint states. The conditional distribution $P(X_i|\Pi_i^{\mathcal{G}})$ in

(1.1) is given by

$$P\left(X_i \,|\, \Pi_i^{\mathcal{G}} = \boldsymbol{\pi}_k\right) = \prod_{j=1}^{r_i} \Theta_{ijk}^{I(X_i=j)}, \tag{3.1}$$

where $\boldsymbol{\pi}_k$ is the k^{th} joint state of X_i's parents, Θ_{ijk} denotes the probability for $X_i = j$ given $\Pi_i^{\mathcal{G}} = \boldsymbol{\pi}_k$, and $I(X_i = j)$ is the indicator variable. A discrete Bayesian network $\mathcal{G}$ is therefore parameterized by $\boldsymbol{\Theta} = \{\Theta_{ijk} \geq 0 : \sum_j \Theta_{ijk} = 1\}$. It is not difficult to show that given an $n \times p$ data set $\mathbb{X}$ with *i.i.d.* observations, the likelihood of $\boldsymbol{\Theta}$ is a product multinomial

$$P\left(\mathbb{X} \,|\, \boldsymbol{\Theta}, \mathcal{G}\right) = \prod_{i=1}^{p} \prod_{k=1}^{q_i} \prod_{j=1}^{r_i} \Theta_{ijk}^{N_{ijk}}, \tag{3.2}$$

where N_{ijk} is the number of observations in which X_i is found at level j when its parents are found in the k^{th} joint state. Given the structure of $\mathcal{G}$, the number of parameters in the multinomial model (3.2) equals $|\boldsymbol{\Theta}| = \sum_{i=1}^{p} r_i q_i = \sum_{i=1}^{p} r_i \prod_{X_j \in \Pi_i^{\mathcal{G}}} r_j$. If we assume that every variable has the same number of levels, i.e., $r_i = r$ for all i, then

$$|\boldsymbol{\Theta}| = r \sum_{i=1}^{p} r^{|\Pi_i^{\mathcal{G}}|}, \tag{3.3}$$

which grows exponentially as the size of the parent set $|\Pi_i^{\mathcal{G}}|$ increases. In the next section, we present the multi-logit model for discrete Bayesian networks for which development of a penalized likelihood framework is much more straightforward. Furthermore, given the structure of $\mathcal{G}$, the number of parameters can be much smaller. Hence, we regard the multi-logit model as an approximation to the full multinomial model (3.2).

3.2 The Multi-logit Model and the Adaptive Group Lasso Regularized Log-likelihood

For a discrete random variable X_i, let d_i be the degrees of freedom of X_i, $i = 1, \ldots, p$. In this chapter, we only consider the main effect of X_i and therefore $d_i = r_i - 1$. Denote by $\mathbf{x}_{h,i} \in \mathbb{R}^{d_i}$ the group of encoded variables corresponding to the i^{th} factor X_i in the h^{th} sample and write $\mathbf{x}_h = (1, \mathbf{x}_{h,1}^T, \ldots, \mathbf{x}_{h,p}^T)^T \in \mathbb{R}^d$, where $d = 1 + \sum_{i=1}^p d_i$.

In a causal discrete Bayesian network $\mathcal{G}$, we model the conditional distribution $X_j | \Pi_j^{\mathcal{G}}$ $(j = 1, \ldots, p)$ using the multi-logit model

$$\begin{aligned} p_{j\ell}(\mathbf{x}_h) &= P(X_j = \ell | \mathbf{x}_h) \\ &= \frac{\exp(\beta_{j\ell 0} + \sum_{i=1}^p \mathbf{x}_{h,i}^T \boldsymbol{\beta}_{j\ell i})}{\sum_{m=1}^{r_j} \exp(\beta_{jm0} + \sum_{i=1}^p \mathbf{x}_{h,i}^T \boldsymbol{\beta}_{jmi})} \\ &= \frac{\exp(\mathbf{x}_h^T \boldsymbol{\beta}_{j\ell\cdot})}{\sum_{m=1}^{r_j} \exp(\mathbf{x}_h^T \boldsymbol{\beta}_{jm\cdot})}, \quad \ell = 1, \ldots, r_j, \end{aligned} \tag{3.4}$$

where $\beta_{j\ell 0}$ is the intercept, $\boldsymbol{\beta}_{j\ell i} \in \mathbb{R}^{d_i}$ is the coefficient vector corresponding to X_i for predicting the ℓ^{th} level of X_j, and $\boldsymbol{\beta}_{j\ell\cdot} = (\beta_{j\ell 0}, \boldsymbol{\beta}_{j\ell 1}^T, \ldots, \boldsymbol{\beta}_{j\ell p}^T)^T \in \mathbb{R}^d$. Note that in the multi-logit model (3.4) above, we set $\boldsymbol{\beta}_{j\ell i} = \mathbf{0}$ for all ℓ if $i \notin \Pi_j^{\mathcal{G}}$. We choose to use a symmetric form of the multi-logit model here, as was done in Zhu and Hastie (2004) and Friedman et al. (2010). Since this model is unidentifiable without constraints, we impose the following constraint on the intercepts

$$\beta_{j10} = 0, \quad \forall j. \tag{3.5}$$

The unidentifiability of other parameters can be resolved via regularization as demonstrated by Friedman et al. (2010). The particular form of regularization we

use in this chapter leads to the constraints

$$\sum_{\ell=1}^{r_j} \boldsymbol{\beta}_{j\ell i} = \mathbf{0}, \quad \forall i, j. \tag{3.6}$$

For $j = 1, \ldots, p$, we further define $\boldsymbol{\beta}_{j\cdot i} = (\boldsymbol{\beta}_{j1i}^T, \ldots, \boldsymbol{\beta}_{jr_ji}^T)^T \in \mathbb{R}^{d_i r_j}$ $(i = 1, \ldots, p)$ to be the vector of coefficients representing the influence of X_i on X_j and $\boldsymbol{\beta}_{j\cdot 0} = (\beta_{j10}, \ldots, \beta_{jr_j0})^T \in \mathbb{R}^{r_j}$ to be the vector of intercepts for predicting X_j. Let $\boldsymbol{\beta} = (\boldsymbol{\beta}_{1\cdot 0}^T, \boldsymbol{\beta}_{1\cdot 1}^T, \ \ldots \ , \boldsymbol{\beta}_{1\cdot p}^T, \ \ldots\ldots \ , \boldsymbol{\beta}_{p\cdot 0}^T, \boldsymbol{\beta}_{p\cdot 1}^T, \ \ldots \ , \boldsymbol{\beta}_{p\cdot p}^T)^T \in \mathbb{R}^\nu$ be the vector of parameters, where $\nu = d \times \sum_{i=1}^p r_i$. Given the structure of $\mathcal{G}$, the number of nonzero elements of $\boldsymbol{\beta}$ is given by $\sum_{j=1}^p r_j(1 + \sum_{i \in \Pi_j^{\mathcal{G}}} d_i) - p$. If we further assume that $r_i = r$ for all i,

$$|\{\beta \in \boldsymbol{\beta} : \beta \neq 0\}| = r \sum_{i=1}^{p} \left(r|\Pi_i^{\mathcal{G}}| - |\Pi_i^{\mathcal{G}}| + 1 \right) - p, \tag{3.7}$$

which only grows linearly as the size of the parent set $|\Pi_i^{\mathcal{G}}|$ increases. Given the structure of $\mathcal{G}$, this rate of growth is much slower compared to the multinomial model.

In this chapter, the data set we consider is an $n \times p$ data matrix $\mathcal{X}$ consisting of p blocks of data. The dimension of the j^{th} block $\mathcal{X}^j$ is $n_j \times p$, where $n = \sum_{j=1}^p n_j$. For each row of $\mathcal{X}^j$, X_j is experimentally fixed to level $\ell \in \{1, \ldots, r_j\}$. Let $\mathcal{I}_j$ be the collection of indices of samples in $\mathcal{X}^j$ and $\mathcal{O}_j = \{1, \ldots, n\} \setminus \mathcal{I}_j$ be the collection of indices of samples in which X_j is only observed. Denote by $n_{-j} = |\mathcal{O}_j| = n - n_j$ the number of samples in $\mathcal{O}_j$. Let $\mathbf{X}_{ji}$ be the j^{th} design matrix corresponding to X_i, which is of dimension $n_{-j} \times d_i$ and whose rows are composed of $\mathbf{x}_{h,i}^T$ for $h \in \mathcal{O}_j$. We assume in this chapter that $\mathbf{X}_{ji}$ satisfies $\mathbf{X}_{ji}^T \mathbf{X}_{ji} = I_{d_i}$ for $i, j = 1, \ldots, p$, which can be achieved by performing Gram-Schmidt orthonormalization if matrices $\mathbf{X}_{ji}$ are of full column rank. If we adopt the multi-logit model (3.4), it can be shown using the factorization (1.2) that the log-likelihood function $\ell(\boldsymbol{\beta})$ can be written

as

$$\begin{aligned}
\ell(\boldsymbol{\beta}) &\propto \sum_{k=1}^{p} \sum_{h \in \mathcal{I}_k} \sum_{j \neq k} \log \left(P(\mathcal{X}_{hj} | \mathbf{x}_{h,i}, i \in \Pi_j^{\mathcal{G}}) \right) \\
&\propto \sum_{j=1}^{p} \sum_{h \in \mathcal{O}_j} \log \left(P(\mathcal{X}_{hj} | \mathbf{x}_{h,i}, i \in \Pi_j^{\mathcal{G}}) \right) \\
&\propto \sum_{j=1}^{p} \sum_{h \in \mathcal{O}_j} \left[\sum_{\ell=1}^{r_j} y_{hj\ell} \mathbf{x}_h^T \boldsymbol{\beta}_{j\ell \cdot} - \log \left(\sum_{\ell=1}^{r_j} \exp(\mathbf{x}_h^T \boldsymbol{\beta}_{j\ell \cdot}) \right) \right], \qquad (3.8)
\end{aligned}$$

where $\mathcal{X}_{hj}$ is the level of X_j in the h^{th} sample, and $y_{hj\ell} = I(\mathcal{X}_{hj} = \ell)$ are indicator variables.

Estimating the structure of a discrete Bayesian network $\mathcal{G}$ from data is equivalent to estimating the sparsity pattern of the parameter $\boldsymbol{\beta}$ using the following equivalence

$$\boldsymbol{\beta}_{j \cdot i} = \mathbf{0} \qquad \Longleftrightarrow \qquad i \notin \Pi_j^{\mathcal{G}}. \qquad (3.9)$$

In order to learn a sparse DAG from data, we propose to use a penalized likelihood approach to estimate $\boldsymbol{\beta}$. It can be seen from (3.9) that, for discrete Bayesian networks, the set of parents of X_j is determined by the magnitude of $\boldsymbol{\beta}_{j \cdot i}$. The regular lasso penalty on $\boldsymbol{\beta}$ is inappropriate for this purpose since it penalizes each component of $\boldsymbol{\beta}$ separately. We instead wish to penalize $\boldsymbol{\beta}_{j \cdot i}$ as a whole to obtain a sparse DAG. Bakin (1999) and Yuan and Lin (2006) proposed the group lasso penalty to select grouped variables (factors) in linear regression. Subsequently, Kim et al. (2006) extended the group lasso to general loss functions and Meier et al. (2008) presented an alternative algorithm for group lasso penalized logistic regression. However, similar to the regular lasso penalty, the group lasso penalty suffers certain drawbacks such as inconsistent variable selection. Hence, the adaptive group lasso penalty was developed to overcome the limitation of the group lasso (Wang and Leng 2008; Bach 2008; Wei and Huang 2010). Here, we propose the following adaptive group lasso estimator for learning the structures of discrete

Bayesian networks:

$$\begin{aligned}\hat{\boldsymbol{\beta}}_\lambda &= \underset{\boldsymbol{\beta}:\mathcal{G}_{\boldsymbol{\beta}} \text{ is acyclic}}{\arg\min} R_\lambda(\boldsymbol{\beta}) \\ &= \underset{\boldsymbol{\beta}:\mathcal{G}_{\boldsymbol{\beta}} \text{ is acyclic}}{\arg\min} \left[-\ell(\boldsymbol{\beta}) + \lambda \sum_{j=1}^{p} \sum_{i=1}^{p} w_{ji} \|\boldsymbol{\beta}_{j\cdot i}\|_2 \right], \end{aligned} \tag{3.10}$$

where $\mathbf{W} = (w_{ij})_{p\times p}$ is a given weight matrix used for the adaptive group lasso and $\mathcal{G}_{\boldsymbol{\beta}}$ denotes the graph induced by $\boldsymbol{\beta}$.

3.3 Coordinate Descent Algorithm

Parameter estimation for discrete Bayesian networks is computationally much more demanding than for Gaussian networks because of the nonlinear nature of the multi-logit model (3.4) used in Section 3.2. However, the idea of using coordinate descent type of algorithms to solve (2.5) can be readily generalized to solve (3.10). In this section, we develop a blockwise coordinate descent algorithm to estimate DAGs from discrete data.

3.3.1 One coordinate descent step

We first consider solving (3.10) with respect to $\boldsymbol{\beta}_{j\cdot i}$ given the current estimates of all the other parameters. We define

$$\begin{aligned} R_{\lambda,j}(\boldsymbol{\beta}_{j\cdot\cdot}) &= -\ell_j(\boldsymbol{\beta}_{j\cdot\cdot}) + \lambda \sum_{i=1}^{p} w_{ji} \|\boldsymbol{\beta}_{j\cdot i}\|_2 \\ &= -\sum_{h\in\mathcal{O}_j} \left[\sum_{\ell=1}^{r_j} y_{hj\ell} \mathbf{x}_h^T \boldsymbol{\beta}_{j\ell\cdot} - \log\left(\sum_{\ell=1}^{r_j} \exp(\mathbf{x}_h^T \boldsymbol{\beta}_{j\ell\cdot}) \right) \right] \\ &\quad + \lambda \sum_{i=1}^{p} w_{ji} \|\boldsymbol{\beta}_{j\cdot i}\|_2, \end{aligned} \tag{3.11}$$

where $\boldsymbol{\beta}_{j\cdot\cdot} = (\boldsymbol{\beta}_{j\cdot 0}^T, \boldsymbol{\beta}_{j\cdot 1}^T, \ \ldots \ , \boldsymbol{\beta}_{j\cdot p}^T)^T$. For simpler notations, we suppress the dependence of $R_{\lambda,j}$ on $\boldsymbol{\beta}_{j\cdot k}$ for $k \neq i$ when considering the problem of $\min_{\boldsymbol{\beta}_{j\cdot i}} R_{\lambda,j}(\cdot)$, and thus write it as $R_{\lambda,j}(\boldsymbol{\beta}_{j\cdot i})$.

To minimize $R_{\lambda,j}(\cdot)$ with respect to $\boldsymbol{\beta}_{j\cdot i}$, we follow the approach of Tseng and Yun (2009) and Meier et al. (2008). We form a partial quadratic approximation to $\ell_j(\cdot)$ using a second-order Taylor expansion at $\boldsymbol{\beta}_{j\cdot\cdot}^{(t)}$, allowing only $\boldsymbol{\beta}_{j\cdot i}$ to vary. The quadratic approximation is defined as

$$\begin{aligned} Q_{\lambda,j}^{(t)}(\boldsymbol{\beta}_{j\cdot i}) &= -\{\ell_j(\boldsymbol{\beta}_{j\cdot i}^{(t)}) + (\boldsymbol{\beta}_{j\cdot i} - \boldsymbol{\beta}_{j\cdot i}^{(t)})^T \nabla \ell_j(\boldsymbol{\beta}_{j\cdot i}^{(t)}) \\ &\quad + \frac{1}{2}(\boldsymbol{\beta}_{j\cdot i} - \boldsymbol{\beta}_{j\cdot i}^{(t)})^T H_{ji}^{(t)} (\boldsymbol{\beta}_{j\cdot i} - \boldsymbol{\beta}_{j\cdot i}^{(t)})\} + \lambda w_{ji} \|\boldsymbol{\beta}_{j\cdot i}\|_2. \end{aligned} \tag{3.12}$$

The gradient of the log-likelihood function $\ell_j(\cdot)$ is

$$\nabla \ell_j(\boldsymbol{\beta}_{j\cdot i}^{(t)}) = \sum_{h \in \mathcal{O}_j} \begin{pmatrix} \left(y_{hj1} - p_{j1}^{(t)}(\mathbf{x}_h)\right) \mathbf{x}_{h,i} \\ \vdots \\ \left(y_{hjr_j} - p_{jr_j}^{(t)}(\mathbf{x}_h)\right) \mathbf{x}_{h,i} \end{pmatrix}, \tag{3.13}$$

where $p_{j\ell}^{(t)}(\mathbf{x}_h)$ $(1 \leq \ell \leq r_j)$ are evaluated at the current parameter estimates. In equation (3.12), $H_{ji}^{(t)}$ is some matrix approximating the Hessian $H_{\ell_j}(\boldsymbol{\beta}_{j\cdot i}^{(t)})$ of the log-likelihood function $\ell_j(\cdot)$

$$H_{\ell_j}(\boldsymbol{\beta}_{j\cdot i}^{(t)}) = -\sum_{h \in \mathcal{O}_j} \begin{pmatrix} w_{j11}^{(t)}(\mathbf{x}_h) \mathbf{x}_{h,i} \mathbf{x}_{h,i}^T & \cdots & \cdots & w_{j1r_j}^{(t)}(\mathbf{x}_h) \mathbf{x}_{h,i} \mathbf{x}_{h,i}^T \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ w_{jr_j1}^{(t)}(\mathbf{x}_h) \mathbf{x}_{h,i} \mathbf{x}_{h,i}^T & \cdots & \cdots & w_{jr_jr_j}^{(t)}(\mathbf{x}_h) \mathbf{x}_{h,i} \mathbf{x}_{h,i}^T \end{pmatrix}, \tag{3.14}$$

where

$$w_{jmn}^{(t)}(\mathbf{x}_h) = \begin{cases} p_{jm}^{(t)}(\mathbf{x}_h)(1 - p_{jn}^{(t)}(\mathbf{x}_h)) & \text{if } 1 \leq m = n \leq r_j \\ -p_{jm}^{(t)}(\mathbf{x}_h) p_{jn}^{(t)}(\mathbf{x}_h) & \text{if } 1 \leq m \neq n \leq r_j \end{cases}. \tag{3.15}$$

In the current implementation, we assume that $H_{ji}^{(t)} = h_{ji}^{(t)} I_{d_i r_j}$ for some scalar $h_{ji}^{(t)} \in \mathbb{R}^-$, where $I_{d_i r_j}$ is the identity matrix of size $d_i r_j$.

It is not difficult to show the following proposition.

Proposition 3.1. *Let* $\boldsymbol{d}_{ji}^{(t)} = \nabla \ell_j(\boldsymbol{\beta}_{j\cdot i}^{(t)}) - h_{ji}^{(t)} \boldsymbol{\beta}_{j\cdot i}^{(t)}$. *If* $\|\boldsymbol{d}_{ji}^{(t)}\|_2 \leq \lambda w_{ji}$, *the minimizer of* $Q_{\lambda,j}^{(t)}(\boldsymbol{\beta}_{j\cdot i})$ *is*

$$\boldsymbol{\beta}_{j\cdot i}^{(t+1)} = \mathbf{0}. \tag{3.16}$$

Otherwise,

$$\boldsymbol{\beta}_{j\cdot i}^{(t+1)} = -\frac{1}{h_{ji}^{(t)}} \left[\boldsymbol{d}_{ji}^{(t)} - \lambda w_{ji} \frac{\boldsymbol{d}_{ji}^{(t)}}{\|\boldsymbol{d}_{ji}^{(t)}\|_2} \right]. \tag{3.17}$$

Remark 3.1. It follows from Proposition 3.1 that for the unpenalized intercepts, the solution to $\min_{\boldsymbol{\beta}_{j\cdot 0}} Q_{\lambda,j}^{(t)}(\cdot)$ is

$$\boldsymbol{\beta}_{j\cdot 0}^{(t+1)} = -\frac{1}{h_{j0}^{(t)}} \boldsymbol{d}_{j0}^{(t)}. \tag{3.18}$$

Remark 3.2. To achieve sufficient descent, a line search should be performed if $\boldsymbol{\beta}_{j\cdot i}^{(t+1)} \neq \boldsymbol{\beta}_{j\cdot i}^{(t)}$. See Tseng and Yun (2009) and Meier et al. (2008) for details.

Remark 3.3. Some of the parameters are always constrained to zero, e.g., $\boldsymbol{\beta}_{j\cdot j}$ and β_{j10} for all j. They should not be updated.

Therefore, in order to minimize $R_{\lambda,j}(\cdot)$ with respect to $\boldsymbol{\beta}_{j\cdot i}$, we apply the quadratic approximation iteratively until some stopping criterion is met.

3.3.2 Blockwise coordinate descent

We outline below the complete blockwise coordinate descent algorithm (CD algorithm) for discrete Bayesian networks. The algorithm works in the same spirit as its counterpart for Gaussian Bayesian networks. In the following algorithm, $\boldsymbol{\beta}_{j\cdot i} \Leftarrow 0$ is used to imply that given current estimates of other parameters, $\boldsymbol{\beta}_{j\cdot i}$ must be set to zero due to the acyclic constraint. Minimization of $R_{\lambda,j}(\cdot)$ with

respect to $\boldsymbol{\beta}_{j\cdot i}$ is done with the method presented in Section 3.3.1.

Algorithm 3.1 CD algorithm for estimating discrete Bayesian networks

1: Choose $\boldsymbol{\beta}$ as the initial vector of parameters such that $\mathcal{G}_{\boldsymbol{\beta}}$ is acyclic
2: **for** $i = 1, \ldots, p-1$ **do**
3: **for** $j = i+1, \ldots, p$ **do**
4: **if** $\boldsymbol{\beta}_{j\cdot i} \Leftarrow \mathbf{0}$ **then**
5: $\boldsymbol{\beta}_{i\cdot j} \leftarrow \arg\min_{\boldsymbol{\beta}_{i\cdot j}} R_{\lambda,i}(\cdot), \quad \boldsymbol{\beta}_{j\cdot i} \leftarrow \mathbf{0}$
6: **else if** $\boldsymbol{\beta}_{i\cdot j} \Leftarrow \mathbf{0}$ **then**
7: $\boldsymbol{\beta}_{i\cdot j} \leftarrow \mathbf{0}, \quad \boldsymbol{\beta}_{j\cdot i} \leftarrow \arg\min_{\boldsymbol{\beta}_{j\cdot i}} R_{\lambda,j}(\cdot)$
8: **else**
9: $S_1 \leftarrow \min_{\boldsymbol{\beta}_{i\cdot j}} R_{\lambda,i}(\cdot) + R_{\lambda,j}(\cdot)|_{\boldsymbol{\beta}_{j\cdot i}=\mathbf{0}}$
10: $S_2 \leftarrow R_{\lambda,i}(\cdot)|_{\boldsymbol{\beta}_{i\cdot j}=\mathbf{0}} + \min_{\boldsymbol{\beta}_{j\cdot i}} R_{\lambda,j}(\cdot)$
11: **if** $S_1 \leq S_2$ **then**
12: $\boldsymbol{\beta}_{i\cdot j} \leftarrow \arg\min_{\boldsymbol{\beta}_{i\cdot j}} R_{\lambda,i}(\cdot), \quad \boldsymbol{\beta}_{j\cdot i} \leftarrow \mathbf{0}$
13: **else**
14: $\boldsymbol{\beta}_{i\cdot j} \leftarrow \mathbf{0}, \quad \boldsymbol{\beta}_{j\cdot i} \leftarrow \arg\min_{\boldsymbol{\beta}_{j\cdot i}} R_{\lambda,j}(\cdot)$
15: **end if**
16: **end if**
17: **end for**
18: **end for**
19: Update all the intercepts: $\boldsymbol{\beta}_{j\cdot 0} \leftarrow \arg\min_{\boldsymbol{\beta}_{j\cdot 0}} R_{\lambda,j}(\cdot), \quad \forall j$
20: Repeat step 2 to 19 until some stopping criterion is met

We use Algorithm 3.1 to compute the solutions $\hat{\boldsymbol{\beta}}_\lambda$ of (3.10) over a grid of J penalty values $\lambda_1 > \ldots > \lambda_J \geq 0$, where at λ_1 every parameter other than the intercepts is penalized to zero. It follows from (3.10) that

$$\begin{aligned} \lambda_1 &\geq \frac{1}{w_{ji}} \|\nabla \ell_j(\boldsymbol{\beta}_{j\cdot i})|_{\boldsymbol{\beta}_{j\cdot i}=\mathbf{0}, 1\leq i\leq p}\|_2 \\ &= \frac{1}{w_{ji}} \left\| \begin{pmatrix} \mathbf{X}_{ji}^T(\mathbf{y}_{j1} - \bar{\mathbf{y}}_{j1}) \\ \vdots \\ \mathbf{X}_{ji}^T(\mathbf{y}_{jr_j} - \bar{\mathbf{y}}_{jr_j}) \end{pmatrix} \right\|_2 \end{aligned} \tag{3.19}$$

for all $1 \leq i, j \leq p$, where $\mathbf{y}_{j\ell} = (y_{hj\ell})_{h\in\mathcal{O}_j}$ and $\bar{\mathbf{y}}_{j\ell} = \bar{y}_{j\ell}\mathbf{1}_{n_{-j}}$, $\ell = 1, \ldots, r_j$. Here, $\bar{y}_{j\ell}$ is the mean of $\mathbf{y}_{j\ell}$ and $\mathbf{1}_{n_{-j}}$ is a vector of ones of length n_{-j}. The solution $\hat{\boldsymbol{\beta}}_{\lambda_t}$ is used as a warm start for calculating $\hat{\boldsymbol{\beta}}_{\lambda_{t+1}}$, $t = 1, \ldots, J-1$.

Similar to the Gaussian case, we do not cycle through all blocks of parameters

in each coordinate descent cycle. Instead, we only iterate over the current active set until the stopping criterion is reached. The algorithm stops if another full coordinate descent cycle does not alter the active set. Otherwise, the process is repeated.

3.3.3 Tuning parameter selection

As we reasoned in Section 2.3.4, traditional model selection criteria such as cross-validation and BIC do not work well for the purpose of estimating DAGs from data. In order to select a suitable penalty parameter λ_t, we use the same idea of empirical model selection outlined in Section 2.3.4 to do parameter tuning. Let $\hat{\mathcal{G}}_{\lambda_t}$ be the DAG induced by $\hat{\boldsymbol{\beta}}_{\lambda_t}$ and e_{λ_t} be the number of directed edges in $\hat{\mathcal{G}}_{\lambda_t}$. We calculate the unpenalized vector of parameters $\widetilde{\boldsymbol{\beta}}_{\lambda_t}$ by estimating a multi-logit model for predicting X_j with $\Pi_j^{\hat{\mathcal{G}}_{\lambda_t}}$ using samples $h \in \mathcal{O}_j$, $j = 1, \ldots, p$. This is done using the R package *mlogit* (Croissant 2011). We use $\widetilde{\boldsymbol{\beta}}_{\lambda_t}$ to compute the log-likelihood $\ell(\widetilde{\boldsymbol{\beta}}_{\lambda_t})$. The difference ratio of two estimated DAGs $\hat{\mathcal{G}}_{\lambda_t}$ and $\hat{\mathcal{G}}_{\lambda_{t+1}}$ is defined to be $dr_{(t,t+1)} = \Delta\ell_{(t,t+1)}/\Delta e_{(t,t+1)}$, where $\Delta\ell_{(t,t+1)} = \ell(\widetilde{\boldsymbol{\beta}}_{\lambda_{t+1}}) - \ell(\widetilde{\boldsymbol{\beta}}_{\lambda_t})$ and $\Delta e_{(t,t+1)} = e_{\lambda_{t+1}} - e_{\lambda_t}$. The selected penalty parameter is indexed by

$$T = \sup\left\{t : dr_{(t-1,t)} \geq \alpha \times \max(dr_{(1,2)}, \ldots, dr_{(J-1,J)}), t = 2, \ldots, J\right\}. \tag{3.20}$$

The typical value of the thresholding parameter α that we used is 0.1.

3.4 Asymptotic Properties

In this section, we establish the asymptotic theory for the adaptive group lasso penalized likelihood estimator of discrete Bayesian networks. Our exposition is parallel to the one in Section 2.4. By rearranging and relabeling individual components, we rewrite $\boldsymbol{\beta}$ as $\boldsymbol{\phi} = (\boldsymbol{\phi}_{(1)}^T, \boldsymbol{\phi}_{(2)}^T)^T$, where $\boldsymbol{\phi}_{(1)} =$

$(\boldsymbol{\beta}_{1\cdot 1}^T, \ \ldots \ , \boldsymbol{\beta}_{1\cdot p}^T, \ \ldots\ldots \ , \boldsymbol{\beta}_{p\cdot 1}^T, \ \ldots \ , \boldsymbol{\beta}_{p\cdot p}^T)^T$ is the parameter vector of interest and $\boldsymbol{\phi}_{(2)} = (\boldsymbol{\beta}_{1\cdot 0}^T, \ \ldots\ldots \ , \boldsymbol{\beta}_{p\cdot 0}^T)^T$ denotes the vector of intercepts. In this section, the j^{th} component of $\boldsymbol{\phi}$ is used to denote the j^{th} block of parameters, e.g. $\phi_1 = \boldsymbol{\beta}_{1\cdot 1}$ and so on. Similarly, we rewrite the weight matrix $\mathbf{W}$ in a vector format as $\mathcal{T} = (\tau_j)_{p^2\times 1} = (w_{11}, \ldots, w_{1p}, \ldots\ldots, w_{p1}, \ldots, w_{pp})^T$. We say $\boldsymbol{\phi}$ is acyclic if the graph $\mathcal{G}_{\boldsymbol{\phi}}$ induced by $\boldsymbol{\phi}$ (or the corresponding $\boldsymbol{\beta}$) is acyclic. Let $\boldsymbol{\Omega} = \{\boldsymbol{\phi} : \boldsymbol{\phi}$ is acyclic and $\boldsymbol{\phi}$ satisfies (3.5) and (3.6)$\}$ be the parameter space and $\boldsymbol{\phi}^* \in \boldsymbol{\Omega}$ be the true parameter. Denote the sparsity pattern of $\boldsymbol{\phi}^*$ by $\mathcal{A} = \{j : \phi_j^* = \mathbf{0}, 1 \le j \le p^2\}$ and $\mathcal{B} = \{j : \phi_j^* \neq \mathbf{0}, 1 \le j \le p^2\}$. Let $\mathcal{G}_{\boldsymbol{\phi}^*}$ be the true DAG.

Let $\boldsymbol{\phi}_k$ $(k = 1, \ldots, p)$ be the parameter obtained from $\boldsymbol{\phi}$ by setting $\boldsymbol{\beta}_{k\cdot i} = \mathbf{0}$ for $1 \le i \le p$. Therefore, the difference between $\mathcal{G}_{\boldsymbol{\phi}_k}$ and $\mathcal{G}_{\boldsymbol{\phi}}$ is that all edges pointing to the k^{th} node in $\mathcal{G}_{\boldsymbol{\phi}}$ are deleted in $\mathcal{G}_{\boldsymbol{\phi}_k}$. As demonstrated in Section 1.5, we can model interventional data in the k^{th} block of the data matrix $\mathcal{X}^k$ as *i.i.d.* observations from a distribution factorized according to $\mathcal{G}_{\boldsymbol{\phi}_k}$. For discrete Bayesian networks, we denote the corresponding probability mass function by $P(\mathbf{x}|\boldsymbol{\phi}_k)$, where $\mathbf{x} = (x_1, \ldots, x_p)$ and $x_j \in \{1, \ldots, r_j\}$ for $j = 1, \ldots, p$. Then the penalized log-likelihood function with the adaptive group lasso penalty is defined as

$$R(\boldsymbol{\phi}) = L(\boldsymbol{\phi}) - \lambda_n \sum_{j=1}^{p^2} \tau_j \|\phi_j\|_2 = \sum_{k=1}^{p} L_k(\boldsymbol{\phi}_k) - \lambda_n \sum_{j=1}^{p^2} \tau_j \|\phi_j\|_2, \tag{3.21}$$

where $L_k(\boldsymbol{\phi}_k) = \sum_{h\in\mathcal{I}_k} \log(P(\mathcal{X}_{h\cdot}|\boldsymbol{\phi}_k))$ and $\mathcal{X}_{h\cdot} = (\mathcal{X}_{hj})_{1\le j\le p}$. A penalized likelihood estimator $\hat{\boldsymbol{\phi}}$ is obtained by maximizing $R(\boldsymbol{\phi})$ in the parameter space $\boldsymbol{\Omega}$.

Remark 3.4. The probability mass function $P(\mathbf{x}|\boldsymbol{\phi}_k) = \Pi_{j=1}^p P(x_j|\Pi_j^{\mathcal{G}_{\boldsymbol{\phi}_k}})$ for $k = 1, \ldots, p$. The conditional distribution $P(x_j|\Pi_j^{\mathcal{G}_{\boldsymbol{\phi}_k}})$ is modeled by some parametric function where parameter ϕ_j $(j = 1, \ldots, p^2)$ represents the influence of one variable on the other in this parametric function. In this section, we assume the multi-logit model (3.4) with constraints (3.5) and (3.6) as the functional form of $P(x_j|\Pi_j^{\mathcal{G}_{\boldsymbol{\phi}_k}})$.

However, the theory developed in this section can be applied to other models.

Though interventional data help to distinguish equivalent DAGs, the following notion of natural parameters is used to completely establish identifiability of DAGs for the case where each variable has interventional data. Suppose that X_i is an ancestor of X_j in a DAG $\mathcal{G}$, that is, there exists at least one path from X_i to X_j. Denote the set of ancestors of X_j by $\text{an}(X_j)$.

Definition 3.1 (Natural parameters)**.** We say that $\boldsymbol{\phi}$ is natural if for $1 \leq i, j \leq p$

$$X_i \in \text{an}(X_j) \text{ in } \mathcal{G}_{\boldsymbol{\phi}} \qquad \Longrightarrow \qquad X_j \text{ is dependent of } X_i \text{ in } P(\mathbf{x}|\boldsymbol{\phi}_i). \tag{3.22}$$

We state the following theorems to establish some properties of our penalized likelihood estimator. These theorems are analogous to the ones in Section 2.4 and their proofs are relegated to Section 3.7.

Theorem 3.2. *Suppose that samples in $\mathcal{X}^k$ are i.i.d. with probability mass function $P(\mathbf{x}|\boldsymbol{\phi}_k^*)$ for $k = 1, \ldots, p$. Assume that the true parameter $\boldsymbol{\phi}^*$ is natural. Then*

$$P(\mathbf{x}|\boldsymbol{\phi}_k) = P(\mathbf{x}|\boldsymbol{\phi}_k^*) \quad \textit{a.e. for all } k = 1, \ldots, p \qquad \Longrightarrow \qquad \boldsymbol{\phi} = \boldsymbol{\phi}^*. \tag{3.23}$$

If we further assume that $n_k/n \to \alpha_k > 0$ as $n \to \infty$, then

$$P_{\boldsymbol{\phi}^*}(L(\boldsymbol{\phi}^*) > L(\boldsymbol{\phi})) \to 1 \tag{3.24}$$

for any $\boldsymbol{\phi} \neq \boldsymbol{\phi}^*$.

Let $\hat{\phi}_k^{(m)}$ $(1 \leq k \leq p^2)$ be the estimate of ϕ_k when the corresponding $\boldsymbol{\beta}_{j\cdot i}$ is estimated by solving the unpenalized multi-logit model, that is, by maximizing $\ell_j(\boldsymbol{\beta}_{j\cdot})$ in (3.11). Define $\boldsymbol{\phi}_{\mathcal{A}} = (\phi_j)_{j \in \mathcal{A}}$. It is further assumed in the following two theorems that $\boldsymbol{\phi}^*$ is natural.

Theorem 3.3. *Assume the adaptive group lasso penalty with weights $\tau_j = \min(\|\hat{\phi}_j^{(m)}\|_2^{-\gamma}, M^\gamma)$ for all j, where $\gamma, M > 0$. As $n \to \infty$, if $\lambda_n/\sqrt{n} \to 0$ and*

$n_k/n \to \alpha_k > 0$ *for* $k = 1, \ldots, p$*, then there exists a local maximizer* $\hat{\boldsymbol{\phi}}$ *of* $R(\boldsymbol{\phi})$ *such that* $\|\hat{\boldsymbol{\phi}} - \boldsymbol{\phi}^*\|_2 = O_p(n^{-1/2})$.

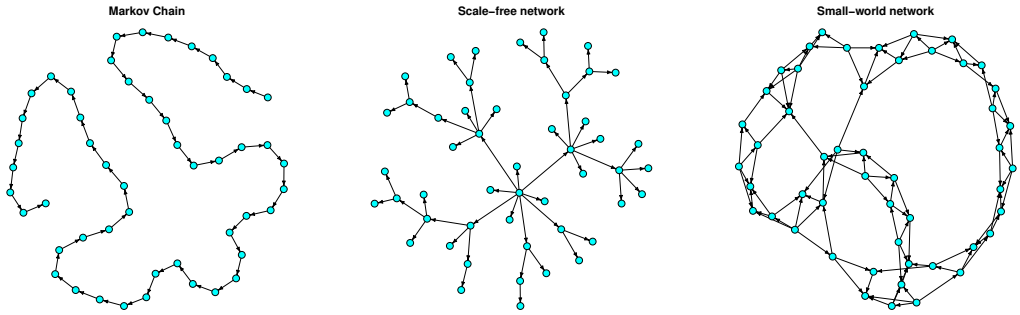
Theorem 3.4. *Assume the adaptive group lasso penalty with weights* $\tau_j = \|\tilde{\phi}_j\|_2^{-\gamma}$ *for some* $\gamma > 0$ *and all* j*, where* $\tilde{\phi}_j$ *is* $\sqrt{n}$*-consistent for* ϕ_j^**. As* $n \to \infty$*, if* $\lambda_n/\sqrt{n} \to 0$*,* $\lambda_n n^{(\gamma-1)/2} \to \infty$ *and* $n_k/n \to \alpha_k > 0$ *for* $k = 1, \ldots, p$*, then there exists a local maximizer* $\hat{\boldsymbol{\phi}}$ *of* $R(\boldsymbol{\phi})$ *such that* $\|\hat{\boldsymbol{\phi}} - \boldsymbol{\phi}^*\|_2 = O_p(n^{-1/2})$*. Furthermore, with probability tending to one, the* $\sqrt{n}$*-consistent local maximizer* $\hat{\boldsymbol{\phi}}$ *must satisfy:* $\hat{\boldsymbol{\phi}}_{\mathcal{A}} = \mathbf{0}$.

3.5 Simulation Study

We evaluated the CD algorithm on simulated data sets. Three types of networks were simulated: a Markov chain, a scale-free network and a small-world network (Figure 3.1). All three networks have $p = 50$ nodes. Given the network structure, 10 data sets were simulated. Each data set contains $n = 500$ samples and consists of p blocks of data $\mathcal{X}^j$, $j = 1, \ldots, p$. The j^{th} block of data $\mathcal{X}^j$ $(1 \leq j \leq p)$ has $n_j = 10$ samples in which only values of X_j were experimentally manipulated, that is, X_j was randomly fixed to one of its levels regardless of the values of X_j's parents. All other variables were generated according to the multi-logit model (3.4). The variables were assumed to be binary with levels in $\{0, 1\}$. If $\Pi_j = \varnothing$, the value of X_j was sampled from $\{0, 1\}$ with equal probability. Otherwise, parameters were chosen such that

$$\begin{aligned} p_{j0}(\mathbf{x}_h) &= \frac{\exp(2\sum_{i\in\Pi_j} I(\mathcal{X}_{hi} = 0))}{\exp(2\sum_{i\in\Pi_j} I(\mathcal{X}_{hi} = 0)) + \exp(2\sum_{i\in\Pi_j} I(\mathcal{X}_{hi} = 1))}, \\ p_{j1}(\mathbf{x}_h) &= \frac{\exp(2\sum_{i\in\Pi_j} I(\mathcal{X}_{hi} = 1))}{\exp(2\sum_{i\in\Pi_j} I(\mathcal{X}_{hi} = 0)) + \exp(2\sum_{i\in\Pi_j} I(\mathcal{X}_{hi} = 1))}. \end{aligned}$$

The scale-free network as well as the small-world network (Figure 3.1B and



(A) (B) (C)

Figure 3.1: Three simulated DAGs: (A) Markov chain, (B) Scale-free network, (C) Small-world network.

Figure 3.1C) was generated using the R package *igraph* (Csárdi and Nepusz 2006). A scale-free network is a network whose degree distribution follows a power law. Many complex networks have been reported to have scale-free topology, such as the Internet (Barabási and Albert 1999; Faloutsos et al. 1999), metabolic networks (Jeong et al. 2000, 2001), semantic networks (Steyvers and Tenenbaum 2005), et cetera. The skeleton of the scale-free network in Figure 3.1B was generated using the Barabási-Albert model (Barabási and Albert 1999). This network has 49 directed edges.

A small-world network is one in which the average path length between two randomly chosen nodes is small relative to the size of the network (Watts and Strogatz 1998). This length grows proportionately to the logarithm of the number of nodes in the network. Another feature of small-world networks is that they have a high clustering coefficient. Scale-free networks are also small-world networks. The small-world network in Figure 3.1C was generated using the Watts-Strogatz model (Watts and Strogatz 1998). The graph initially generated by the Watts-Strogatz model was undirected. In order to convert it to a DAG, edge directions were chosen to be consistent with a randomly generated topological sort. This small-world network has 100 directed edges.

We used the CD algorithm (Algorithm 3.1) to estimate DAGs from each data

set over a grid of penalty parameters, starting from λ_1 defined in (3.19). Model selection was done using criterion (3.20) with $\alpha = 0.1$. Our current implementation of Algorithm 3.1 leads to long running time if the adaptive group lasso weights are set according to Theorem 3.4. Hence, results in this section were obtained with weights set to 1. The accuracy of DAG estimation is measured by true positive rate (TPR) and false discovery rate (FDR), defined as

$$\begin{aligned} \text{TPR} &= \text{E/T}, \\ \text{FDR} &= (\text{R} + \text{FP})/\text{P}, \end{aligned}$$

where P, E, R, FP denote respectively the total number of predicted edges, the number of expected edges, the number of reversed edges and the number of false positive edges (excluding the reversed ones) in a DAG estimate, and T denotes the total number of true edges in the simulated network. Results reported in this section are the average over 10 data sets for each of the three simulated networks.

Table 3.1 summarizes the performance of our CD algorithm on the three simulated networks. It is apparent that both the Markov chain and the scale-free network can be estimated very accurately from data. Though estimation of the denser small-world network did not turn out to be good in terms of TPR, we found that our estimated small-world network has very few false positive edges, which implies that estimation of the skeleton of the small-world network is still satisfactory. The primary reason for a low TPR in this case is that about a third of the predicted edges have wrong directions.

As we did in Section 2.5, we also made a comparison between the CD algorithm and the PC-based method. Results of this comparison are reported in Table 3.2. Recall that the reported results represent an upper bound that can be obtained from a two-step PC-based method as we explained in Section 2.5. Based on Table 3.2, we conclude that the performance of our CD algorithm is comparable

Table 3.1: The average number of predicted (P), expected (E), reversed (R), missed (M) and false positive (FP) edges and the average TPRs and FDRs for DAGs learned by the CD algorithm from discrete data sets

Network	CD algorithm						
	P	E	R	M	FP	TPR	FDR
Markov chain	53.8	44.0	5.0	0.0	4.8	0.898 (0.035)	0.181 (0.052)
Scale-free	52.0	44.3	4.7	0.0	3.0	0.904 (0.048)	0.146 (0.078)
Small-world	89.7	54.1	30.6	15.3	5.0	0.541 (0.058)	0.397 (0.035)

Note: The numbers in parentheses are the standard deviations across 10 data sets.

Table 3.2: The average performance of the two-step PC-based method and the average performance of the CD algorithm on discrete data sets when models are selected to match the number of edges of those learned by the PC-based method

Network	PC-based method				CD algorithm		
	P	Bi	TPR	FDR	P	TPR	FDR
Markov chain	49.2	46.9	0.957 (0.136)	0.046 (0.139)	49.2	0.898 (0.035)	0.106 (0.034)
Scale-free	41.2	39.8	0.824 (0.032)	0.019 (0.029)	46.7	0.869 (0.087)	0.089 (0.042)
Small-world	64.9	15.1	0.405 (0.035)	0.375 (0.058)	65.1	0.415 (0.037)	0.362 (0.057)

Note: The column "Bi" lists the average number of bidirected/undirected edges for CPDAGs learned by the PC algorithm. The numbers in parentheses are the standard deviations across 10 data sets.

to the PC-based method. Note that for Markov chain and the scale-free network, directionality of most predicted edges cannot be determined by the PC algorithm alone. This is because the two networks are very sparse and the size of their corresponding equivalence class is large.

3.6 Conclusions

We have extended the penalized likelihood framework for estimating Gaussian Bayesian networks to discrete Bayesian networks in this chapter. We adopt the multi-logit model to model causal interactions in a discrete Bayesian network so that the development of the penalized likelihood method is relatively straightforward. The adaptive lasso penalty is no longer appropriate in the discrete setting since it penalizes separately individual dummy variables corresponding to the same discrete variable. Instead, the adaptive group lasso penalty is utilized to encourage selection of grouped variables together. A blockwise coordinate descent algorithm has been proposed where each coordinate descent step is solved by iteratively applying a quadratic approximation. Computation needed for estimating discrete Bayesian networks is much more demanding than that for Gaussian case and improvements over the current implementation are left for future work. We have also established asymptotic theory parallel to the one developed in the last chapter for the proposed learning procedure. Our method has been evaluated on three types of simulated networks: a Markov chain, a scale-free network and a small-world network. The latter two are thought to resemble many real-world networks. We have demonstrated that the performance of the discrete CD algorithm is comparable to the two-step PC-based procedure.

3.7 Appendix

3.7.1 Proof of Theorem 3.2

We prove the first claim (3.23) by contradiction. Suppose $\boldsymbol{\phi} \neq \boldsymbol{\phi}^*$ and $P(\mathbf{x}|\boldsymbol{\phi}_k) = P(\mathbf{x}|\boldsymbol{\phi}_k^*)$ a.e. for $k = 1, \ldots, p$. Let $\mathcal{S}(\mathcal{G})$ denote the set of topological sorts of a DAG $\mathcal{G}$. There are two cases for $\mathcal{G}_{\boldsymbol{\phi}}$ and $\mathcal{G}_{\boldsymbol{\phi}^*}$ if $\boldsymbol{\phi}$ is different from $\boldsymbol{\phi}^*$:

Case 1: $\mathcal{S}(\mathcal{G}_{\boldsymbol{\phi}}) \cap \mathcal{S}(\mathcal{G}_{\boldsymbol{\phi}^*}) \neq \varnothing$. Let $\sqsubset \in \mathcal{S}(\mathcal{G}_{\boldsymbol{\phi}}) \cap \mathcal{S}(\mathcal{G}_{\boldsymbol{\phi}^*})$, i.e., an ordering compatible with both $\mathcal{G}_{\boldsymbol{\phi}}$ and $\mathcal{G}_{\boldsymbol{\phi}^*}$. Assume without loss of generality that in this ordering $i \prec j$ if $i < j$. Apparently, $\sqsubset$ is also compatible with $\mathcal{G}_{\boldsymbol{\phi}_k}$ and $\mathcal{G}_{\boldsymbol{\phi}_k^*}$ for $k = 1, \ldots, p$. Then we can write $P(\mathbf{x}|\boldsymbol{\phi}_k) = \prod_{i=1}^p P(x_i|x_1, \ldots, x_{i-1}, \boldsymbol{\phi}_k) = \prod_{i=1}^p P(x_i|\Pi_i^{\mathcal{G}_{\boldsymbol{\phi}_k}}, \boldsymbol{\phi}_k)$ and $P(\mathbf{x}|\boldsymbol{\phi}_k^*) = \prod_{i=1}^p P(x_i|x_1, \ldots, x_{i-1}, \boldsymbol{\phi}_k^*) = \prod_{i=1}^p P(x_i|\Pi_i^{\mathcal{G}_{\boldsymbol{\phi}_k^*}}, \boldsymbol{\phi}_k^*)$. Since $P(\mathbf{x}|\boldsymbol{\phi}_k) = P(\mathbf{x}|\boldsymbol{\phi}_k^*)$, it follows that $\Pi_i^{\mathcal{G}_{\boldsymbol{\phi}_k}} = \Pi_i^{\mathcal{G}_{\boldsymbol{\phi}_k^*}}$ for all i and thus $\mathcal{G}_{\boldsymbol{\phi}_k} = \mathcal{G}_{\boldsymbol{\phi}_k^*}$ for all k. However, since $\boldsymbol{\phi} \neq \boldsymbol{\phi}^*$, there exists some k such that $\boldsymbol{\phi}_k \neq \boldsymbol{\phi}_k^*$. Therefore, there exists a k such that, the common probability mass function $P(\mathbf{x}|\boldsymbol{\phi}_k) = P(\mathbf{x}|\boldsymbol{\phi}_k^*)$, factorized according to a common structure $\mathcal{G}_{\boldsymbol{\phi}_k} = \mathcal{G}_{\boldsymbol{\phi}_k^*}$, can be parameterized by two different parameters $\boldsymbol{\phi}_k$ and $\boldsymbol{\phi}_k^*$. This is apparently impossible.

Case 2: $\mathcal{S}(\mathcal{G}_{\boldsymbol{\phi}}) \cap \mathcal{S}(\mathcal{G}_{\boldsymbol{\phi}^*}) = \varnothing$, that is, none of the orderings of $\mathcal{G}_{\boldsymbol{\phi}^*}$ is compatible with $\mathcal{G}_{\boldsymbol{\phi}}$. In this case, there must exist a pair of indices (i, j) such that in $\mathcal{G}_{\boldsymbol{\phi}^*}$ $X_i \in \text{an}(X_j)$, but in $\mathcal{G}_{\boldsymbol{\phi}}$ X_j is a non-descendant of X_i. Then X_j is independent of X_i in $P(\mathbf{x}|\boldsymbol{\phi}_i)$, since in $\mathcal{G}_{\boldsymbol{\phi}_i}$ X_i has no parents and X_j is a non-descendant of X_i. However, in $\mathcal{G}_{\boldsymbol{\phi}_i^*}$ we still have $X_i \in \text{an}(X_j)$. Since $\boldsymbol{\phi}^*$ is natural, X_i and X_j are dependent in $P(\mathbf{x}|\boldsymbol{\phi}_i^*)$. Therefore, there exists $1 \leq i \leq p$ such that $P(\mathbf{x}|\boldsymbol{\phi}_i) \neq P(\mathbf{x}|\boldsymbol{\phi}_i^*)$, which contradicts our assumption.

So in both *case 1* and *case 2* we have a contradiction. Thus, the first claim holds.

For the second claim (3.24), first note that by the law of large numbers,

$$\frac{1}{n}(L(\boldsymbol{\phi}) - L(\boldsymbol{\phi}^*)) = \sum_{k=1}^{p} \frac{n_k}{n} \frac{1}{n_k} \sum_{h \in \mathcal{I}_k} \log \frac{P(\mathcal{X}_{h\cdot}|\boldsymbol{\phi}_k)}{P(\mathcal{X}_{h\cdot}|\boldsymbol{\phi}_k^*)} \longrightarrow_p \sum_{k=1}^{p} \alpha_k \mathbf{E}_{\boldsymbol{\phi}_k^*} \left[\log \frac{P(\mathbf{Y}|\boldsymbol{\phi}_k)}{P(\mathbf{Y}|\boldsymbol{\phi}_k^*)} \right], \tag{3.25}$$

where $\mathbf{Y}$ is a random vector with probability mass function $P(\mathbf{x}|\boldsymbol{\phi}_k^*)$. Then the desired result follows immediately using Jensen's inequality and (3.23).

3.7.2 Proof of Theorem 3.3

Let

$$\mathbf{I}(\boldsymbol{\phi}_k) = \mathbf{E}_{\boldsymbol{\phi}_k} \left\{ \left[\frac{\partial}{\partial \boldsymbol{\phi}_k} \log P(\mathbf{x}|\boldsymbol{\phi}_k) \right] \left[\frac{\partial}{\partial \boldsymbol{\phi}_k} \log P(\mathbf{x}|\boldsymbol{\phi}_k) \right]^T \right\}$$

be the Fisher information matrix.

Consider $\boldsymbol{\phi} \in \text{nb}(\boldsymbol{\phi}^*)$, where $\text{nb}(\boldsymbol{\phi}^*)$ is an arbitrarily small neighborhood of $\boldsymbol{\phi}^*$. The components of $\boldsymbol{\phi}$ must satisfy $\phi_i \neq \mathbf{0}$ if $\phi_i^* \neq \mathbf{0}$ $(i = 1, \ldots, p^2)$, since otherwise $\|\boldsymbol{\phi} - \boldsymbol{\phi}^*\|_2 \geq \min_{j:\phi_j^* \neq \mathbf{0}} \|\phi_j^*\|_2$. In particular, this implies that if $\boldsymbol{\phi} \in \text{nb}(\boldsymbol{\phi}^*)$, $i \to j$ in $\mathcal{G}_{\boldsymbol{\phi}}$ for all $i \to j$ in $\mathcal{G}_{\boldsymbol{\phi}^*}$ and thus $\mathcal{G}_{\boldsymbol{\phi}}$ and $\mathcal{G}_{\boldsymbol{\phi}^*}$ have compatible orderings. If we restrict to the lower dimensional space $\boldsymbol{\Omega}_k = \{\boldsymbol{\phi}_k : \boldsymbol{\phi} \in \boldsymbol{\Omega}\}$, the same arguments apply to an arbitrarily small neighborhood of $\boldsymbol{\phi}_k^*$ in this space, that is, $\mathcal{G}_{\boldsymbol{\phi}_k}$ and $\mathcal{G}_{\boldsymbol{\phi}_k^*}$ have compatible orderings. Then it follows from the arguments used in *Case 1* in the proof of Theorem 3.2 that $P(\mathbf{x}|\boldsymbol{\phi}_k) \neq P(\mathbf{x}|\boldsymbol{\phi}_k^*)$ for $\boldsymbol{\phi}_k \in \text{nb}(\boldsymbol{\phi}_k^*) \backslash \{\boldsymbol{\phi}_k^*\}$. It follows that $\mathbf{I}(\boldsymbol{\phi}_k^*)$ is positive definite for all k.

Let $\mathbf{u} \in \left\{\mathbf{u} : \boldsymbol{\phi}^* + n^{-1/2}\mathbf{u} \in \boldsymbol{\Omega}\right\}$ and u_j be its j^{th} component. Here u_j is defined in the same way as ϕ_j. Further, let $\mathbf{u}_k$ be the vector defined similarly as $\boldsymbol{\phi}_k$, $k = 1, \ldots, p$. Note that $\sum_{k=1}^{p} \|\mathbf{u}_k\|_2^2 \geq \|\mathbf{u}\|_2^2$. Let $\delta_{\min}^k > 0$ be the minimal eigenvalue of $\mathbf{I}(\boldsymbol{\phi}_k^*)$ and $\rho = \min_k(\alpha_k \delta_{\min}^k/2)$. Then

$$\sum_{k=1}^{p} \frac{\alpha_k}{2} \mathbf{u}_k^T \mathbf{I}(\boldsymbol{\phi}_k^*) \mathbf{u}_k \geq \sum_{k=1}^{p} \frac{\alpha_k}{2} \delta_{\min}^k \|\mathbf{u}_k\|_2^2 \geq \rho \sum_{k=1}^{p} \|\mathbf{u}_k\|_2^2 \geq \rho \|\mathbf{u}\|_2^2. \tag{3.26}$$

Now we study the behavior of $R(\boldsymbol{\phi})$ in a small neighborhood of the true value $\boldsymbol{\phi}^*$ by expanding $L(\boldsymbol{\phi})$ around $\boldsymbol{\phi}^*$. We have, as $n \to \infty$,

$$
\begin{aligned}
& R(\boldsymbol{\phi}^* + n^{-1/2}\mathbf{u}) - R(\boldsymbol{\phi}^*) \\
\leq\ & L(\boldsymbol{\phi}^* + n^{-1/2}\mathbf{u}) - L(\boldsymbol{\phi}^*) - \lambda_n \sum_{j \in \mathcal{B}} \tau_j (\|\phi_j^* + n^{-1/2} u_j\|_2 - \|\phi_j^*\|_2) \\
\leq\ & \sum_{k=1}^{p} \left[L_k(\boldsymbol{\phi}_k^* + n^{-1/2}\mathbf{u}_k) - L_k(\boldsymbol{\phi}_k^*) \right] + \lambda_n n^{-1/2} \sum_{j \in \mathcal{B}} \tau_j \|u_j\|_2 \\
=\ & \sum_{k=1}^{p} \left[n^{-1/2} L_k'(\boldsymbol{\phi}_k^*)^T \mathbf{u}_k - \frac{1}{2} n_k n^{-1} \mathbf{u}_k^T \mathbf{I}(\boldsymbol{\phi}_k^*) \mathbf{u}_k \left\{1 + o_p(1)\right\} \right] \\
& + \lambda_n n^{-1/2} \sum_{j \in \mathcal{B}} \tau_j \|u_j\|_2 \\
=\ & \sum_{k=1}^{p} \left[\sqrt{\alpha_k} \frac{L_k'(\boldsymbol{\phi}_k^*)^T}{\sqrt{n_k}} \mathbf{u}_k \left\{1 + o_p(1)\right\} - \frac{\alpha_k}{2} \mathbf{u}_k^T \mathbf{I}(\boldsymbol{\phi}_k^*) \mathbf{u}_k \left\{1 + o_p(1)\right\} \right] \\
& + \lambda_n n^{-1/2} \sum_{j \in \mathcal{B}} \tau_j \|u_j\|_2 \\
\leq\ & \sum_{k=1}^{p} \left[\sqrt{\alpha_k} \frac{L_k'(\boldsymbol{\phi}_k^*)^T}{\sqrt{n_k}} \mathbf{u}_k \left\{1 + o_p(1)\right\} \right] - \rho \|\mathbf{u}\|_2^2 \left\{1 + o_p(1)\right\} \\
& + \lambda_n n^{-1/2} \sum_{j \in \mathcal{B}} \tau_j \|\mathbf{u}\|_2. \qquad (3.27)
\end{aligned}
$$

The last inequality is due to (3.26). From the central limit theorem, $n_k^{-1/2} L_k'(\boldsymbol{\phi}_k^*) = O_p(1)$ for all k. By assumption, $\tau_j = O_p(1)$ for $j = 1, \ldots, p^2$ and $\lambda_n / \sqrt{n} = o_p(1)$. Therefore, for a sufficiently large C, the second order term in the last line of (3.27) dominates the first and third terms uniformly in $\left\{\mathbf{u} : \|\mathbf{u}\|_2 = C, \boldsymbol{\phi}^* + n^{-1/2}\mathbf{u} \in \boldsymbol{\Omega}\right\}$. Hence, for any given $\varepsilon > 0$, there exists a sufficiently large C such that

$$
P\left(\sup_{\|\mathbf{u}\|_2 = C} R(\boldsymbol{\phi}^* + n^{-1/2}\mathbf{u}) < R(\boldsymbol{\phi}^*) \right) \geq 1 - \varepsilon, \qquad (3.28)
$$

which implies that with probability at least $1 - \varepsilon$, there exists a local maximizer

$\hat{\boldsymbol{\phi}}$ of $R(\boldsymbol{\phi})$ in the ball $\left\{\boldsymbol{\phi}^* + n^{-1/2}\mathbf{u} \in \boldsymbol{\Omega} : \|\mathbf{u}\|_2 \leq C\right\}$. Thus, there exists a local maximizer $\hat{\boldsymbol{\phi}}$ of $R(\boldsymbol{\phi})$ such that $\|\hat{\boldsymbol{\phi}} - \boldsymbol{\phi}^*\|_2 = O_p(n^{-1/2})$.

3.7.3 Proof of Theorem 3.4

We omit the proof of the first part of Theorem 3.4, since it is similar to that of Theorem 3.3. To prove the second part, let us first, by permuting the indices, rewrite the parameter $\boldsymbol{\phi}$ as $\boldsymbol{\phi} = (\boldsymbol{\phi}_a^T, \boldsymbol{\phi}_b^T)^T = (\boldsymbol{\phi}_{\mathcal{A}}^T, \boldsymbol{\phi}_{\mathcal{B}}^T, \boldsymbol{\phi}_{(2)}^T)^T$, where $\boldsymbol{\phi}_a = \boldsymbol{\phi}_{\mathcal{A}}$ and $\boldsymbol{\phi}_b = (\boldsymbol{\phi}_{\mathcal{B}}^T, \boldsymbol{\phi}_{(2)}^T)^T$. Let $r = |\mathcal{A}|$ be the number of zero components of $\boldsymbol{\phi}^*$.

Now we only need to show that with probability tending to 1, for any $\boldsymbol{\phi}_b$ satisfying $\|\boldsymbol{\phi}_b - \boldsymbol{\phi}_b^*\| = O_p(n^{-1/2})$ and any constant $C > 0$,

$$(\mathbf{0}^T, \boldsymbol{\phi}_b^T)^T = \underset{\|\boldsymbol{\phi}_a\|_2 \leq C/\sqrt{n}}{\arg\max} R\big((\boldsymbol{\phi}_a^T, \boldsymbol{\phi}_b^T)^T\big). \tag{3.29}$$

To establish (3.29), we again study the behavior of $R(\boldsymbol{\phi})$ around the point $(\mathbf{0}^T, \boldsymbol{\phi}_b^T)^T$ by expanding $L(\boldsymbol{\phi})$ around $(\mathbf{0}^T, \boldsymbol{\phi}_b^T)^T$. Let $\boldsymbol{\phi}^o = (\mathbf{0}^T, \boldsymbol{\phi}_b^T)^T$, and $\boldsymbol{\phi} = \boldsymbol{\phi}^o + n^{-1/2}\mathbf{u} \in \boldsymbol{\Omega}$, where $\mathbf{u} = (\mathbf{u}_a^T, \mathbf{u}_b^T)^T$, $\|\mathbf{u}\|_2 \leq C$ and $\mathbf{u}_b = \mathbf{0}$. Then we have the following result similar to that in Theorem 3.3:

$$\begin{aligned}
& R(\boldsymbol{\phi}^o + n^{-1/2}\mathbf{u}) - R(\boldsymbol{\phi}^o) \\
= & \sum_{k=1}^{p} \left[\sqrt{\alpha_k} \frac{L_k'(\boldsymbol{\phi}_k^o)^T}{\sqrt{n_k}} \mathbf{u}_k \left\{1 + o_p(1)\right\} - \frac{\alpha_k}{2} \mathbf{u}_k^T \mathbf{I}(\boldsymbol{\phi}_k^o) \mathbf{u}_k \left\{1 + o_p(1)\right\} \right] \\
& - \frac{\lambda_n}{\sqrt{n}} \sum_{j=1}^{r} \tau_j \|u_j\|_2 \\
= & \sum_{k=1}^{p} \left[\sqrt{\alpha_k} \frac{L_k'(\boldsymbol{\phi}_k^o)^T}{\sqrt{n_k}} \mathbf{u}_k \left\{1 + o_p(1)\right\} - \frac{\alpha_k}{2} \mathbf{u}_k^T \mathbf{I}(\boldsymbol{\phi}_k^o) \mathbf{u}_k \left\{1 + o_p(1)\right\} \right] \\
& - \frac{\lambda_n}{\sqrt{n}} n^{\gamma/2} \sum_{j=1}^{r} \|\sqrt{n}\tilde{\phi}_j\|_2^{-\gamma} \|u_j\|_2.
\end{aligned} \tag{3.30}$$

Note that both the first and second terms in the last line of (3.30) are on

the order of $O_p(1)$ for any fixed constant C. Since $\tilde{\phi}_j$ is $\sqrt{n}$-consistent, we have $\|\sqrt{n}\tilde{\phi}_j\|_2 = O_p(1)$, for $j = 1, \ldots, r$. Then the third term in the last line of (3.30) is on the order of $\lambda_n n^{(\gamma-1)/2} \to \infty$. Therefore, (3.29) holds, and the proof is complete.

CHAPTER 4

A Real Data Example

In this chapter, we illustrate the usage of our CD algorithm in analyzing a flow cytometry data set. This data set has been extensively studied. It is ideal for our purpose since a continuous version as well as a discrete version of the data set is available. After giving a general description of the data set, we report results of our analysis using methodologies developed in the last two chapters.

4.1 Description of the Data Set

The data set was generated from a flow cytometry experiment conducted by Sachs et al. (2005). Flow cytometry is a technique frequently employed for cell sorting and biomarker detection. It allows simultaneous measurement of the expression levels as well as modification states of multiple proteins and other cellular molecules in thousands of individual cells (De Rosa et al. 2001; Perez and Nolan 2002). Therefore, it can be used to produce large multivariate data sets containing interventional data.

Sachs et al. (2005) studied a well-known signaling network in human primary CD4+ T-cells of the immune system. This chosen network was perturbed by 9 stimulatory and inhibitory interventions. Each interventional condition was applied to an individual component of the network and a total of 8 components were perturbed in their experiment. Simultaneous measurements were taken on $p = 11$ proteins and phospholipids of this network under each condition. Since 3

interventions were targeted to proteins that were not measured, samples collected under these conditions were observational. Data were collected from $n = 7466$ cells and contained continuous measurements. Hence, the original data set is continuous with dimension 7466×11 and consists of a mixture of observational and interventional samples. Among the 11 measured components, 5 proteins and phospholipids were perturbed.

Figure 4.1A shows the known causal interactions among the 11 measured components of this signaling network. These causal relationships are well established, and no consensus has been reached on interactions beyond those present in the network. Thus, this network structure is often used as the benchmark to assess the accuracy of an estimated network structure, and we therefore call it the consensus model.

The discrete flow cytometry data set, analyzed originally by Sachs et al. (2005), was obtained from the source data after some preprocessing by Sachs et al. (2005). The preprocessing includes elimination of outliers and discretization using some information-preserving technique (Hartemink 2001). The discretization transforms the variables into three levels, high, medium and low, which are coded as 2, 1 and 0, respectively. As a result, the magnitudes of the original measurements are partially preserved in the discrete data set. The discrete data set has dimension 5400×11 and 600 samples were sampled for each of the 9 interventions.

In order to distinguish the two CD algorithms developed respectively in the last two chapters, we will denote Algorithm 2.1 by the gCD algorithm and Algorithm 3.1 by the dCD algorithm in this chapter.

4.2 Analysis of the Continuous Data Set

A number of researchers studied the flow cytometry data set, among whom Friedman et al. (2008) and Shojaie and Michailidis (2010) analyzed the continuous

version. Friedman et al. (2008) applied their graphical lasso procedure to this data set and estimated a number of graphical models using different values of the L_1 penalty. Their models are all undirected and they observed moderate agreement between one of their estimates and the consensus model. Shojaie and Michailidis (2010) also analyzed the same data set and estimated directed acyclic graphs using penalized likelihood method by assuming the ordering of the variables is known *a priori*. Their estimated DAG using the adaptive lasso penalty is shown in Figure 4.1B. This graph has 27 directed edges in total, among which 14 are expected and 13 are false positives. We obtained a sequence of estimated DAGs after applying the gCD algorithm to the continuous flow cytometry data. One of them is shown in Figure 4.1C. Our model also has a total of 27 directed edges, of which 8 are expected, 6 are reversed, and 13 are false positives. It seems that the performance of the gCD algorithm, if ignoring the directionality, is very comparable to the method assuming a known ordering.

4.3 Analysis of the Discrete Data Set

To test the robustness of our method, we applied the gCD algorithm to the discrete flow cytometry data set. A DAG with 26 edges is shown in Figure 4.1D. To our surprise, this graph is qualitatively better than the one estimated using the continuous data set. In this graph, there are 11 expected edges and 15 false predictions (R+FP) (4^{th} row of Table 4.1). We also applied the gCD algorithm to 100 bootstrap samples generated from the discrete data set to assess the sensitivity of our method to data perturbations. For each bootstrap sample, we selected a model with 26 edges and found that on average it shared 23.3 edges with the model shown in Figure 4.1D, which confirms that our method is quite robust to data perturbations. Moreover, though the gCD algorithm was designed for Gaussian data, we were still able to obtain a reasonable network structure from

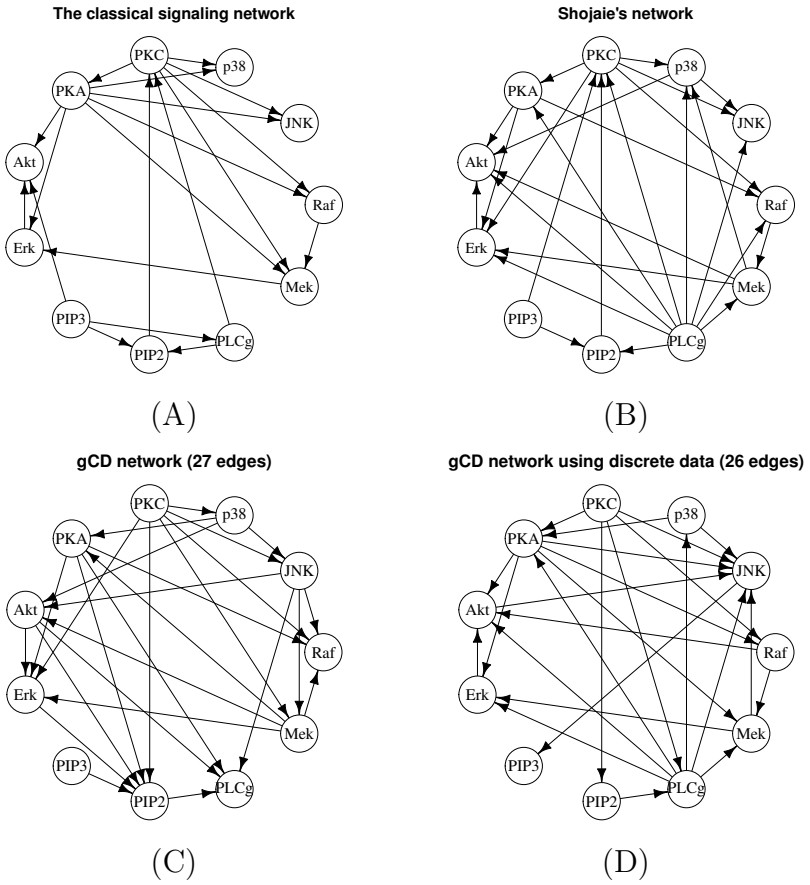


Figure 4.1: (A) The classical signaling network of human immune system cells, (B) Shojaie's network estimated from the continuous flow cytometry data set. The gCD networks estimated from (C) the continuous flow cytometry data set and (D) the discrete flow cytometry data set.

Table 4.1: Comparison among the gCD algorithm, the dCD algorithm, the order-graph sampler and the multi-domain sampler applied to the discrete flow cytometry data set

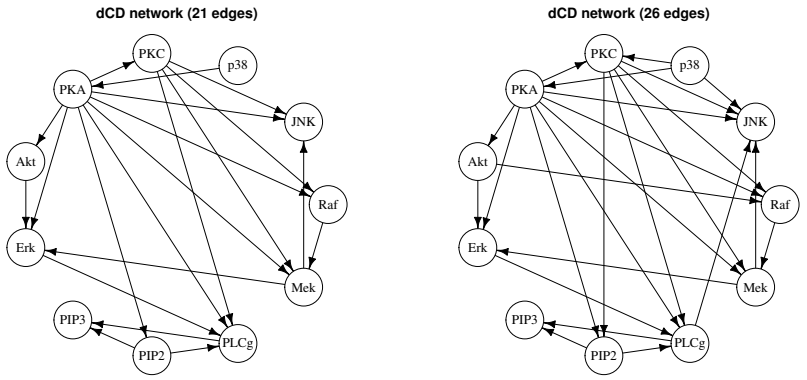
method	P	E	R	M	FP
gCD algorithm (20 edges)	20	9	2	9	9
dCD algorithm (21 edges)	21	10	7	3	4
Order-Graph sampler	20	8	4	8	8
gCD algorithm (26 edges)	26	11	4	5	11
dCD algorithm (26 edges)	26	10	9	1	7
Multi-domain sampler	25.9	15.55	2.05	2.4	8.3

Note: The order-graph sampler result comes from the mean graph (Figure 11) in Ellis and Wong (2008), while the multi-domain sampler result is the average over 20 independent runs (see Table 3 of Zhou (2011)).

the discretized data set which does not satisfy the Gaussian assumption.

Compared to the estimate obtained by Ellis and Wong (2008) using their order-graph sampler, the result of our gCD algorithm is slightly better in terms of the number of expected edges (E) and false predictions (R+FP) (1^{st} row of Table 4.1). The multi-domain sampler, recently developed by Zhou (2011) for Bayesian inference, yields better result than the gCD algorithm. However, the gCD algorithm is much faster than these Monte Carlo sampling approaches. For large networks with hundreds of nodes, the gCD algorithm can still be used to obtain reasonably good estimates of DAGs, while Monte Carlo methods may not be applicable due to their long running time.

We also estimated DAGs from the discrete flow cytometry data set using the dCD algorithm. Two networks with 21 and 26 edges respectively are shown in Figure 4.2. As expected, both DAGs are qualitatively closer to the consensus model than those estimated using the gCD algorithm (Figure 4.1), which confirms that a principled generalization of our methodology to discrete data types does improve the quality of estimation. The detailed performance measures are



(A) (B)

Figure 4.2: Two DAGs estimated from the discrete flow cytometry data set using the dCD algorithm.

reported in Table 4.1. Though the DAG estimated by the multi-domain sampler still has higher TPR than the DAG estimated by the dCD algorithm, our result is slightly better in terms of estimating the skeleton of the consensus model.

4.4 Conclusions

We performed an analysis of a flow cytometry data set generated from a signaling network in human immune system cells. We analyzed both the continuous and the discrete version of the data set and found that the quality of the estimated DAGs obtained using the dCD algorithm was considerably better than those estimated using the gCD algorithm. Results of comparisons with other methods are also reported. Though we used the consensus model as the benchmark to assess the accuracy of DAG estimation, the true network is not available. Hence, DAGs estimated using our method with different penalties provide new causal interactions that could be tested in future experiments.

CHAPTER 5

Summary and Discussion

The problem of learning the structure of Bayesian networks has been studied for a long time. We developed in this dissertation a unified penalized likelihood framework to estimate sparse causal Bayesian networks using experimental data. We considered both the Gaussian Bayesian networks and discrete Bayesian networks, which are two common types of Bayesian networks studied most in the past. Our work was largely motivated by recent work on regularization methods which became a dominating theme in statistics and machine learning literature.

We formulated estimation of Bayesian networks as an optimization problem and proposed to solve this problem using coordinate descent types of algorithms. Coordinate descent methods have been proved successful in various settings (Fu 1998; Friedman et al. 2007; Wu and Lange 2008) and their implementations are relatively straightforward. The acyclic constraint imposed on the structure of Bayesian networks can be solved in a natural way by coordinate descent algorithms. Our CD algorithms estimate a number of Bayesian network models with varying degrees of complexities over a grid of penalties.

For Gaussian Bayesian networks, the penalty we used is the adaptive lasso penalty proposed by Zou (2006). The advantage of the adaptive lasso penalty over the regular lasso penalty is that it satisfies the oracle properties (Fan and Li 2001). Using the adaptive lasso penalty, we have established that model selection consistency can be achieved if the model parameter is natural and the proper penalty parameter is chosen. Our CD algorithm developed for Gaussian Bayesian

networks is fast. In fact, it is often faster than the PC algorithm in terms of the time spent in estimating a single model. Sample sizes do not have significant impact on the running time of the CD algorithm, because the key quantities needed for CD updates are computed beforehand, an idea mentioned in Friedman et al. (2007). However, the algorithm may require fewer iterations to stop if the data set is large. An important issue that is not addressed in this dissertation is the convergence problem of the CD algorithm. This is nontrivial since the corresponding optimization is highly nonconvex. Both the objective function and the constraint set are nonconvex. We plan to investigate this problem in the future.

For discrete Bayesian networks, we used the so-called adaptive group lasso penalty, an adaptive version of the group lasso penalty (Bakin 1999; Yuan and Lin 2006). The group lasso type of penalties are more appropriate for discrete variables than the lasso type of penalties since they encourage sparsity at the factor level. For easy application of the penalized likelihood framework, we adopted the multi-logit model instead of the multinomial model for modeling causal interactions in a discrete Bayesian network. Despite the difference of the form of models and penalties, asymptotic theories developed for Gaussian Bayesian networks can be readily extended to the discrete case. The CD algorithm for discrete Bayesian networks solves each CD step using a quadratic approximation iteratively. Unlike the Gaussian case, the speed of the current implementation of the discrete CD algorithm is not quite satisfactory. One reason is that for each CD step, updating the gradient in (3.13) involves a summation over the majority of samples under the simulation settings we used. Another reason is that we aim to control the amount of memory used when implementing the algorithm and therefore some quantities are recomputed. For the algorithm to be truly effective in estimating very large networks, much of our future work will be centered on improving the computational efficiency of the discrete CD algorithm.

Since the output from our CD algorithm consists of a sequence of Bayesian networks, models should be selected using an appropriate criterion. Traditional model selection criteria such as cross-validation and BIC turn out to be unsatisfactory. Hence, we proposed an empirical model selection rule that is intuitive and performs well in practice. However, we think that a more rigorous investigation of the model selection problem for network estimation is worthwhile in future.

Finally, we are interested in extending our work to a more general setting where Bayesian networks consist of both continuous and discrete variables. It seems that coping with mixed data types will be straightforward given the development of our penalized likelihood framework for both Gaussian and discrete Bayesian networks, and it is left for future work as well.

Bibliography

Francis R. Bach. Consistency of the group lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1225, 2008.

Sergey Bakin. *Adaptive Regression and Model Selection in Data Mining Problems*. Ph.D. thesis, Australian National University, 1999.

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

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

Remco R. Bouckaert. Probabilistic network construction using the minimum description length principle. In *Symbolic and Quantitative Approaches to Reasoning and Uncertainty: European Conference ECSQARU '93*, volume 747 of *Lecture Notes in Computer Science*, pages 41–48. Springer, 1993.

Remco R. Bouckaert. Probabilistic network construction using the minimum description length principle. Technical Report RUU-CS-94-27, Department of Computer Science, Utrecht University, 1994.

Wray Buntine. Theory refinement on Bayesian networks. In *Proceedings of the Seventh Annual Conference on Uncertainty in Artificial Intelligence*, pages 52–60. Morgan Kaufmann, 1991.

David Maxwell Chickering. Learning Bayesian networks is NP-complete. In *Learning from Data: Artificial Intelligence and Statistics V.*, pages 121–130. Springer-Verlag, 1996.

David Maxwell Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554, 2002.

David Maxwell Chickering and David Heckerman. Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables. *Machine Learning*, 29:181–212, 1997.

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

Gregory F. Cooper and Edward Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, 3rd edition, 2009.

Yves Croissant. Estimation of multinomial logit models in R: The mlogit packages. Available at: http://cran.r-project.org/web/packages/mlogit/vignettes/mlogit.pdf, 2011.

Gábor Csárdi and Tamás Nepusz. The igraph software package for complex network research. *Proceedings of the International Conference on Complex Systems*, 2006.

Stephen C. De Rosa, Leonard A. Herzenberg, Leonore A. Herzenberg, and Mario Roederer. 11-color, 13-parameter flow cytometry: Identification of human naive T cells by phenotype, function, and T-cell receptor diversity. *Nature Medicine*, 7:245–248, 2001.

David L. Donoho and Iain M. Johnstone. Adapting to unknown smoothness via wavelet shrinkage. *Journal of the American Statistical Association*, 90:1200–1224, 1995.

Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *The Annals of Statistics*, 32:407–499, 2004.

Byron Ellis. *Inference on Bayesian Network Structures.* Ph.D. dissertation, Harvard University, 2006.

Byron Ellis and Wing Hung Wong. Learning causal Bayesian network structures from experimental data. *Journal of the American Statistical Association*, 103: 778–789, 2008.

Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the Internet topology. *Computer Communications Review*, 29: 251–262, 1999.

Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96: 1348–1360, 2001.

Jianqing Fan, Yang Feng, and Yichao Wu. Network exploration via the adaptive lasso and SCAD penalties. *The Annals of Applied Statistics*, 3:521–541, 2009.

Jerome Friedman, Trevor Hastie, Holger Höfling, and Robert Tibshirani. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 1:302–332, 2007.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9:432–441, 2008.

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33:1–22, 2010.

Nir Friedman and Daphne Koller. Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50:95–125, 2003.

Wenjiang Fu. Penalized regressions: The bridge versus the lasso. *Journal of Computational and Graphical Statistics*, 7:397–416, 1998.

Steven B. Gillispie and Michael D. Perlman. Enumerating Markov equivalence classes of acyclic digraph models. In *Proceedings of the Seventeenth Annual Conference on Uncertainty in Artificial Intelligence*, pages 171–177. Morgan Kaufmann, 2001.

Alexander John Hartemink. *Principled Computational Methods for the Validation and Discovery of Genetic Regulatory Networks*. Ph.D. thesis, Massachusetts Institute of Technology, 2001.

Alain Hauser and Peter Bühlmann. Characterization and greedy learning of interventional Markov equivalence classes of directed acyclic graphs. Preprint, available at `http://arxiv.org/abs/1104.2808`, 2011.

David Heckerman, Dan Geiger, and David M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.

Edward Herskovits and Gregory Cooper. Kutató: An entropy-driven system for construction of probabilistic expert systems from databases. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*, pages 54–62, 1990.

Hawoong Jeong, Bálint Tombor, Réka Albert, Zoltán N. Oltvai, and Albert-László Barabási. The large-scale organization of metabolic networks. *Nature*, 407:651–654, 2000.

Hawoong Jeong, Sean P. Mason, Albert-László Barabási, and Zoltán N. Oltvai. Lethality and centrality in protein networks. *Nature*, 411:41–42, 2001.

Markus Kalisch and Peter Bühlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *Journal of Machine Learning Research*, 8:613–636, 2007.

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*, 47:1–26, 2012.

Yuwon Kim, Jinseog Kim, and Yongdai Kim. Blockwise sparse regression. *Statistica Sinica*, 16:375–390, 2006.

Clifford Lam and Jianqing Fan. Sparsistency and rates of convergence in large covariance matrix estimation. *The Annals of Statistics*, 37:4254–4278, 2009.

Wai Lam and Fahiem Bacchus. Learning Bayesian belief networks: An approach based on the MDL principle. *Computational Intelligence*, 10:269–293, 1994.

Steffen L. Lauritzen. *Graphical Models*. Oxford University Press, 1996.

David Madigan and Jeremy York. Bayesian graphical models for discrete data. *International Statistical Review*, 63:215–232, 1995.

Christopher Meek. Causal inference and causal explanation with background knowledge. In *Proceedings of Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 403–410. Morgan Kaufmann, 1995a.

Christopher Meek. Strong completeness and faithfulness in Bayesian networks. In *Proceedings of the Eleventh Annual Conference on Uncertainty in Artificial Intelligence*, pages 411–418, 1995b.

Lukas Meier, Sara van de Geer, and Peter Bühlmann. The group Lasso for logistic regression. *Journal of the Royal Statistical Society. Series B.*, 70:53–71, 2008.

Nicolai Meinshausen and Peter Bühlmann. High-dimensional graphs and variable selection with the Lasso. *The Annals of Statistics*, 34:1436–1462, 2006.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

Omar D. Perez and Garry P. Nolan. Simultaneous measurement of multiple active kinase states using polychromatic flow cytometry. *Nature Biotechnology*, 20: 155–162, 2002.

Robert W. Robinson. Counting labeled acyclic digraphs. In F. Harary, editor, *New Directions in the Theory of Graphs*, pages 239–273. Academic Press, New York, 1973.

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

Ali Shojaie and George Michailidis. Penalized likelihood methods for estimation of sparse high-dimensional directed acyclic graphs. *Biometrika*, 97:519–538, 2010.

Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. Springer-Verlag, 1993.

Mark Steyvers and Joshua B. Tenenbaum. The large-scale structure of semantic networks: Statistical analyses and a model of semantic growth. *Cognitive Science*, 29:41–78, 2005.

Joe Suzuki. A construction of Bayesian networks from databases based on an MDL principle. In *Proceedings of the Ninth Annual Conference on Uncertainty in Artificial Intelligence*, pages 266–273, 1993.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B.*, 58:267–288, 1996.

Paul Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109: 475–494, 2001.

Paul Tseng and Sangwoon Yun. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming B*, 117:387–423, 2009.

Lieven Vandenberghe, Stephen Boyd, and Shao-Po Wu. Determinant maximization with linear matrix inequality constraints. *SIAM Journal on Matrix Analysis and Applications*, 19:499–533, 1998.

Thomas Verma and Judea Pearl. Causal networks: Semantics and expressiveness. In *Proceedings of the Fourth Annual Conference on Uncertainty in Artificial Intelligence*, pages 352–359. Elsevier Science, 1988.

Thomas Verma and Judea Pearl. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*, pages 220–227, 1990.

Hansheng Wang and Chenlei Leng. A note on adaptive group lasso. *Computational Statistics and Data Analysis*, 52:5277–5286, 2008.

Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.

Fengrong Wei and Jian Huang. Consistent group selection in high-dimensional linear regression. *Bernoulli*, 16:1369–1384, 2010.

TongTong Wu and Kenneth Lange. Coordinate descent procedures for lasso penalized regression. *The Annals of Applied Statistics*, 2:224–244, 2008.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society. Series B*, 68:49–67, 2006.

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

Qing Zhou. Multi-domain sampling with applications to structural inference of Bayesian networks. *Journal of the American Statistical Association*, 106:1317–1330, 2011.

Ji Zhu and Trevor Hastie. Classification of gene microarrays by penalized logistic regression. *Biostatistics*, 5:427–443, 2004.

Hui Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101:1418–1429, 2006.