THE UNIVERSITY OF CHICAGO

HIGH-DIMENSIONAL GRAPH ESTIMATION AND DENSITY ESTIMATION

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF STATISTICS

BY

ZHE LIU

CHICAGO, ILLINOIS

JUNE 2016

To my parents

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost I express my sincerest gratitude to my supervisor, Dr. John Lafferty, who has supported me throughout my thesis with his extensive knowledge, expertise and patience. I have been extremely lucky to have a supervisor who is always available to me and cares so much about my work. I appreciate his continued guidance and encouragement throughout my time as his student.

Moreover, I would like to thank the other members of my committee, Dr. Matthew Stephens and Dr. Rina Foygel Barber for their constructive suggestions at all levels of my research projects. I must also acknowledge the rest of the faculty of the Department of Statistics at the University of Chicago for providing me with the outstanding education.

Finally, I wish to thank all of my family and friends for their support throughout my studies, without whose love and encouragement, I would not have finished this thesis.

ABSTRACT

Graphical models have become a common tool in many fields and a useful way of modeling probability distributions. In this thesis, we investigate approaches for graph estimation and density estimation problems in high dimensions.

The estimation and model selection problems in Gaussian graphical models are equivalent to the estimation of precision matrix and identification of its zero-pattern. In Chapter 2, we propose a new estimation method by considering a convex combination of a set of individual estimators with various sparsity patterns. We analyze the risk of this aggregation estimator and show by an oracle that it is comparable to the risk of the best estimator based on a single graph. In Chapter 3, we investigate robust methods for Gaussian graphical models in the presence of possible outliers and corrupted data. We consider the neighborhood selection and graphical lasso algorithms and show that the robust counterparts obtained using the trimmed inner product or the nonparanormal give stronger performance guarantees.

Gaussian graphical models maintain tractable inference, but they are limited in their ability to flexibly model the bivariate and higher order marginals. In Chapter 4, we study tree-based graphical models and develop new density estimation approaches under structural constraints—scale-free network and shared edges among multiple graphs. Our methods arise from a Bayesian formulation as the MAP estimates and solve the optimization problems via a minorize-maximization procedure with Kruskal's algorithm.

The ability of tree-based graphical models to model complex independence graphs is compromised. In Chapter 5, we combine the ideas behind Gaussian graphical models and tree-based graphical models to form a new nonparametric family of graphical models, which relax the normality assumption and increase statistical efficiency by modeling the forest with kernel density estimators and modeling each blossom with the nonparanormal.

Our analysis and experimental results indicate that the newly proposed methods in this thesis can be powerful alternatives to standard approaches for graph estimation and density estimation problems.

CHAPTER 1

INTRODUCTION

Graphical models [26, 52] have now become a common tool in many fields and a useful way of exploring and modeling the distribution. For instance, an undirected graphical model may be used to represent friendships between people in a social network, or represent complex interactions among gene products resulted from biological processes.

Let $X = (X_1, \ldots, X_d)$ be a continuous random vector with density $p^*(x)$. An undirected graph G for p^* has d vertices, collected in a set V, one for each variable. We can represent the edges as a set E of unordered pairs: $(i, j) \in E$ if and only if there is an edge between X_i and X_j.

In a conditional independence graph, an edge between X_i and X_j is absent if X_i and X_j are independent, given the other variables

$$X_i \perp\!\!\!\perp X_j \mid X_{V\setminus\{i,j\}}. \tag{1.0.1}$$

Suppose we observe n random vectors

$$X^{(1)}, \ldots, X^{(n)} \overset{\text{i.i.d.}}{\sim} p^*. \tag{1.0.2}$$

The goal is to estimate the graph structure G and the density p^* from the data.

If p^* is strictly positive, the Hammersley-Clifford theorem implies that the density has the following form

$$p^*(x) \propto \prod_{C\in\mathcal{C}} \varphi_C(x_C) = \exp\left(\sum_{C\in\mathcal{C}} f_C(x_C)\right), \tag{1.0.3}$$

where $\mathcal{C}$ denotes the set of cliques in the graph, and $\varphi_C(x_C) = \exp(f_C(x_C)) > 0$ denotes arbitrary potential functions. This represents a very large and rich set of nonparametric

graphical models. The fundamental difficulty is that it is in general intractable to compute the normalizing constant. A compromise must be made to achieve computationally tractable inference, typically involving assumptions on the functions f_C, on the graph $G = \{\mathcal{C}\}$, or both.

The default model for graphical modeling of continuous data is the multivariate Gaussian. Under the assumption that $X \sim \mathcal{N}_d(\mu, \Sigma)$, it has been shown that there is no edge between X_i and X_j if and only if $\Theta_{ij} = 0$, where $\Theta = \Sigma^{-1}$. Therefore, the estimation and model selection problems in Gaussian graphical models are equivalent to estimation of the precision matrix Θ and identification of its zero-pattern. While this is one of the classical problems in multivariate statistics, with a renewed focus on high-dimensional data in recent years, a number of sparse estimators have been proposed to deal with the problem of precision matrix estimation. Among them, Yuan and Lin [56] and Friedman et al. [19] impose an ℓ_1 penalty on the entries of the precision matrix when maximizing the Gaussian log-likelihood, known as the *graphical lasso*, encouraging some of the entries of the estimated precision matrix to be exactly zero. Specifically, the graphical lasso estimator maximizes the following ℓ_1-regularized log-likelihood:

$$\widehat{\Theta}_{\lambda_n} = \underset{\Theta \succ 0}{\operatorname{argmax}} \left\{ \log\det(\Theta) - \operatorname{tr}(\widehat{\Sigma}_n \Theta) - \lambda_n \sum_{i \neq j} |\Theta_{ij}| \right\}, \tag{1.0.4}$$

where $\widehat{\Sigma}_n$ is the empirical covariance matrix and λ_n is a regularization parameter. Also, Meinshausen and Bühlmann [36] consider and analyze the neighborhood selection method via the lasso. Cai et al. [7] propose a new constrained ℓ_1 minimization approach for sparse inverse covariance matrix estimation. Yuan [55] takes advantage of the connection between multivariate linear regression and entries of the inverse covariance matrix, developing an estimating procedure that can effectively exploit sparsity. Theoretical properties, including consistency in parameter estimation and consistency in recovering the sparsity structure, are discussed in these and other papers [41, 44].

The nonparanormal [31], a form of Gaussian copula, weakens the Gaussian assumption by imposing Gaussianity on the transformed random vector $f(X) = (f_1(X_1), \ldots, f_d(X_d))$, where each f_j is a monotonic and differentiable function. This allows arbitrary single variable marginal probability distributions in the model. Liu et al. [31] derive a method for estimating the nonparanormal, and study the method's theoretical properties.

Both Gaussian graphical models and the nonparanormal maintain tractable inference, but they are limited in their ability to flexibly model the bivariate and higher order marginals. Alternatively, forest-structured graphical models permit arbitrary bivariate marginals and maintain tractability by restricting to acyclic graphs. A nonparametric approach based on forests is developed in [33] for density estimation in high-dimensional settings. In particular, Liu et al. [33] form kernel density estimates of the bivariate and univariate marginals, and apply Kruskal's algorithm to estimate the optimal forest. An oracle inequality on the excess risk of the resulting estimator relative to the risk of the best forest is also developed. However, the ability to model complex independence graphs is compromised.

In this thesis, we study approaches for graph estimation and density estimation problems in high dimensions. The rest of this thesis is organized as follows.

In Chapter 2, we propose and analyze an aggregation method for estimating precision matrices in Gaussian graphical models, by considering a convex combination of a suitable set of individual estimators with various underlying sparsity patterns. We study the risk of this aggregation estimator and show by an oracle that it is comparable to the risk of the best estimator based on a single graph.

In Chapter 3, we explore the estimation methods for Gaussian graphical models that are robust against possible outliers and corrupted data. Two procedures, the trimmed inner product and the nonparanormal, are discussed and analyzed. We consider the neighborhood selection and graphical lasso algorithms in the uncorrupted setting and show that the robust counterparts obtained using the trimmed inner product or the nonparanormal give stronger performance guarantees under corruption.

In Chapter 4, we describe forest-based nonparametric methods for learning scale-free graphical models and joint learning multiple graphs. Our approaches arise from a Bayesian formulation as the MAP estimates and solve the optimization problems via a minorize-maximization procedure with Kruskal's algorithm. Compared with parametric methods that penalize the Gaussian likelihood, the proposed methods result in more accurate estimation of the underlying networks.

In Chapter 5, we introduce a combination of Gaussian graphical models and forest-structured graphical models to form a new nonparametric method for high-dimensional data. Blossom tree models could relax the normality assumption and increase statistical efficiency by modeling the forest with nonparametric kernel density estimators and modeling each blossom with the nonparanormal.

Finally, we conclude in Chapter 6. Our analysis and experimental results indicate that the newly proposed methods in this thesis can be powerful alternatives to standard approaches for graph estimation and density estimation problems. Some future visions are also provided in this chapter.

CHAPTER 2

AGGREGATION METHODS FOR GAUSSIAN GRAPHICAL MODELS

2.1 Introduction

In this chapter, we propose and analyze a new estimation approach for the precision matrix by considering a convex combination of a set of individual estimators with different sparsity patterns. A sparsity pattern is defined as a binary vector with each element indicating whether the corresponding edge of the graph is absent or not. These individual estimators and the corresponding aggregating weights are determined to ensure a competitive rate of convergence for the risk of the aggregation estimator.

A primary motivation for aggregating estimators is that it can improve the estimation risk, as "betting" on multiple models can provide a type of insurance against a single model being poor [28]. Most of the recent work on estimator aggregation deals with regression learning problems. For example, exponential screening for linear models provides a form of frequentist averaging over a large model class, which enjoys strong theoretical properties [42]. An aggregation classifier is proposed in [27] and an optimal rate of convex aggregation for the hinge risk is also obtained. In this work, we extend the idea of aggregation to the estimation of precision matrices in Gaussian graphical models.

Our aggregation method is based on a sample-splitting procedure: the first subsample is set to construct individual estimators and the second subsample is then used to determine the weights and aggregate these estimators. To carry out the analysis of the aggregation step, it is enough to work conditionally on the first subsample so that the problem reduces to aggregation of deterministic estimators [43]. Namely, let $R(\cdot)$ denote a risk function, then given the deterministic estimator Θ_m with sparsity pattern m, one can construct an

aggregation estimator $\widehat{\Theta}_{\text{agg}}$ such that the excess risk

$$r := R(\widehat{\Theta}_{\text{agg}}) - \min_{m \in \mathcal{M}} R(\Theta_m), \tag{2.1.1}$$

is as small as possible, where $\mathcal{M}$ is a candidate set of sparsity patterns. Ideally, we wish to find an aggregation estimator whose risk is as close as possible (in a probabilistic sense) to the minimum risk of individual estimators. The risk function considered in this chapter is the Kullback-Leibler divergence.

The rest of this chapter is organized as follows. In Section 2.2, we describe the aggregation estimator in detail. Theoretical properties are given in Section 2.3. Numerical experiments are presented in Section 2.4. Finally, we conclude in Section 2.5. Detailed proofs are collected in the last section.

2.2 Graphical Aggregation Estimator

2.2.1 Aggregation Methods

The default model for graphical modeling of continuous data is the multivariate Gaussian. Let data $\mathcal{D}_n := \{x_1, \ldots, x_n\}$ be the realizations of n independent samples from a multivariate Gaussian distribution $\mathcal{N}_d(0, \Sigma)$, where Σ is the covariance matrix. The log-likelihood of $\mathcal{D}_n$ is (up to a constant) given by

$$l_n(\Theta) := \frac{n}{2} \log \det(\Theta) - \frac{n}{2} \text{tr}(\widehat{\Sigma}_n \Theta), \tag{2.2.1}$$

where $\Theta = \Sigma^{-1}$ is the precision matrix, that is the inverse covariance matrix, and $\widehat{\Sigma}_n = \frac{1}{n} \sum_{i=1}^n x_i x_i^T$ is the empirical covariance matrix with dimension d.

Let $|\cdot|_1$ denote the ℓ_1 vector norm, $\|\cdot\|$ denote the spectral matrix norm, and $\|\cdot\|_F$ denote the Frobenius matrix norm. For any symmetric positive-definite matrix A we use the notation $A \succ 0$. For any two real numbers a and b we use the notation $a \vee b := \max(a, b)$.

Let $\mathbf{1}(\cdot)$ denote the indicator function.

We call a sparsity pattern any binary vector $m \in \mathcal{M}$, where $\mathcal{M}$ is a candidate set of sparsity patterns and $\mathcal{M} \subset \{0,1\}^{d(d-1)/2}$. The kth coordinate of m can be interpreted as an indicator of presence ($m_k = 1$) or absence ($m_k = 0$) of the edge $(i_k, j_k) \in E$ such that (i_k, j_k) is the kth element of the ordered list $\mathcal{S}_d$, where $\mathcal{S}_d = \{(i,j) : 1 \leq i < j \leq d\}$.

We partition the sample $\mathcal{D}_n$ into two independent subsamples, $\mathcal{D}_{n_1}^{(1)}$ and $\mathcal{D}_{n_2}^{(2)}$, of size n_1 and n_2, respectively, where $n_1 + n_2 = n$. In the aggregation procedure, $\mathcal{D}_{n_1}^{(1)}$ is utilized to construct individual estimators and $\mathcal{D}_{n_2}^{(2)}$ is then used to aggregate these estimators.

This sample-splitting aggregation procedure was originally described in [39], where two independent subsamples are obtained from the initial sample by randomization; individual estimators are constructed from the first subsample while the second is used to perform aggregation.

Given each sparsity pattern $m \in \mathcal{M}$, we first define the individual estimator of the precision matrix Θ.

Definition 2.2.1. *For each $m \in \mathcal{M}$, let E_m be the edge set of the graph with sparsity pattern m. Let $\widehat{\Theta}_m^{(1)}$ be the constrained maximum likelihood estimator, defined by*

$$\widehat{\Theta}_m^{(1)} = \underset{\Theta \in \mathcal{C}_m}{\operatorname{argmax}} \{\log\det(\Theta) - \operatorname{tr}(\widehat{\Sigma}_{n_1}^{(1)}\Theta)\}, \tag{2.2.2}$$

where

$$\mathcal{C}_m = \{\Theta \succ 0 : \Theta_{ij} = 0 \text{ for any } (i,j) \notin E_m \text{ and } i \neq j\}, \tag{2.2.3}$$

and $\widehat{\Sigma}_{n_1}^{(1)}$ denotes the empirical covariance matrix using the first subsample $\mathcal{D}_{n_1}^{(1)}$.

Notice that each individual estimator maximizes the log-likelihood under the constraint that some pre-defined subset of the parameters are zero. If $n_1 \geq q_m$ where q_m is the maximal clique size of a minimal chordal cover of the graph with edge set E_m, estimator $\widehat{\Theta}_m^{(1)}$ exists

and is unique [49].

The following relationships hold regarding $\widehat{\Theta}_m^{(1)}$ and its inverse $(\widehat{\Theta}_m^{(1)})^{-1}$:

$$[\widehat{\Theta}_m^{(1)}]_{ij} = 0,\ \forall (i,j) \notin E_m, \tag{2.2.4}$$

$$[(\widehat{\Theta}_m^{(1)})^{-1}]_{ij} = [\widehat{\Sigma}_{n_1}^{(1)}]_{ij},\ \forall (i,j) \in E_m \cup \{(i,i), i = 1, \ldots, d\}. \tag{2.2.5}$$

Indeed we can drive the above properties via the Lagrange form, where we add Lagrange constants for all missing edges of E_m

$$l_m^{(1)}(\Theta) := \log\det(\Theta) - \mathrm{tr}(\widehat{\Sigma}_{n_1}^{(1)}\Theta) - \sum_{(i,j)\notin E_m} \gamma_{ij}\theta_{ij}, \tag{2.2.6}$$

where γ_{ij} is a Lagrange constant. The gradient equation for maximizing (2.2.6) can be written as

$$\Theta^{-1} - \widehat{\Sigma}_{n_1}^{(1)} - \Gamma = 0, \tag{2.2.7}$$

where Γ is a matrix of Lagrange parameters with nonzero values for all pairs with edges absent.

This is an equality-constrained convex optimization problem, and a number of methods have been proposed for solving it, for example, in Hastie et al. [21].

Now we define the aggregation estimator, which linearly combines a set of individual estimators $\widehat{\Theta}_m^{(1)}$ for $m \in \mathcal{M}$.

Definition 2.2.2. *Let $\widehat{\Theta}_{\mathrm{agg}}^{(1,2)}$ be the aggregation estimator that linearly combines a set of estimators $\widehat{\Theta}_m^{(1)}$, defined by*

$$\widehat{\Theta}_{\mathrm{agg}}^{(1,2)} = \sum_{m\in\mathcal{M}} v_m^{(1,2)}\widehat{\Theta}_m^{(1)}, \tag{2.2.8}$$

where the superscripts denote which subsamples are used for constructions, and the weights

$$v_m^{(1,2)} := \frac{\exp\{(n_2/2)(\text{logdet}(\widehat{\Theta}_m^{(1)}) - \text{tr}(\widehat{\Theta}_m^{(1)}\widehat{S}_{n_2}^{(2)}))\}\pi_m}{\sum_{m'\in\mathcal{M}}\exp\{(n_2/2)(\text{logdet}(\widehat{\Theta}_{m'}^{(1)}) - \text{tr}(\widehat{\Theta}_{m'}^{(1)}\widehat{S}_{n_2}^{(2)}))\}\pi_{m'}}. \tag{2.2.9}$$

Here, $S_{n_2}^{(2)}$ is an estimator of Σ, and π_m is a (prior) probability distribution on the set of sparsity pattern $\mathcal{M}$, defined by

$$\pi_m := \frac{1}{H}\left(\frac{|m|_1}{ed(d-1)}\right)^{|m|_1}, \tag{2.2.10}$$

where H is a normalization factor to ensure that π_m add up to one.

Observe that the aggregation estimator is a linear combination of the set of individual estimators $\widehat{\Theta}_m^{(1)}$ with weights $v_m^{(1,2)}$. It is indeed a convex combination since the weights add up to one.

A natural choice of $S_{n_2}^{(2)}$ would be the empirical covariance matrix $\widehat{\Sigma}_{n_2}^{(2)}$. In this scenario, ignoring the prior distribution π_m, the individual weight is proportional to the likelihood of estimator $\widehat{\Theta}_m^{(1)}$ evaluated on the second subsample. Thus, the higher the predictive ability, the more weighting will be put on that individual estimator. However, as is shown in the next section, this would lead to a deterioration of convergence rate in high-dimensional settings. Instead, we use the hard thresholding estimator proposed in [5]. To be more specific, the thresholding estimation of Σ thresholded at λ is defined by

$$S_{n_2}^{(2)} = T_\lambda(\widehat{\Sigma}_{n_2}^{(2)}) := \{\widehat{\sigma}_{ij}^{(2)} \cdot \mathbf{1}(|\widehat{\sigma}_{ij}^{(2)}| \geq \lambda)\}_{1\leq i,j\leq d}, \tag{2.2.11}$$

where $\widehat{\sigma}_{ij}^{(2)}$ is the (i,j)th element of $\widehat{\Sigma}_{n_2}^{(2)}$.

In practice, we can apply the same procedure for the problem of threshold selection as in [5]: we split the second subsample randomly into two pieces of size $n_2(1 - 1/\log n_2)$ and $n_2/\log n_2$, respectively, and repeat this B times. Let $\widehat{\Sigma}_{1,v}^{(2)}$ and $\widehat{\Sigma}_{2,v}^{(2)}$ be the empirical covariance matrices based on the two pieces, from the vth split. Then the thresholding

parameter λ is determined by

$$\lambda = \underset{\lambda'}{\operatorname{argmin}} \left\{ \frac{1}{B} \sum_{v=1}^{B} \| T_{\lambda'}(\widehat{\Sigma}_{1,v}^{(2)}) - \widehat{\Sigma}_{2,v}^{(2)} \|_F^2 \right\}. \tag{2.2.12}$$

In Definition 2.2.2, we also incorporate a deterministic factor π_m into the weighted averaging to account for model complexity, in a manner that facilitates desirable risk properties [28, 42]. Here, low-complexity models are favored. Alternatively, if the set of sparsity patterns $\mathcal{M} = \{0,1\}^{d(d-1)/2}$, the following deterministic factor specifies a uniform distribution on the cardinality of the subset and a conditionally uniform distribution on the subsets of that size:

$$\pi_m = \left[(d(d-1)/2 + 1) \binom{d(d-1)/2}{|m|_1} \right]^{-1}. \tag{2.2.13}$$

In addition, a simple way is to choose a flat prior, where we set $\pi_m = \pi_{m'}$ for any $m, m' \in \mathcal{M}$.

We show in the following section that, under some technical conditions, the risk of the aggregation estimator is bounded by the risk of the best component estimator plus a low price for aggregating.

2.2.2 Approximation Algorithms

To implement the method, note that exact computation of the aggregation estimator might require the calculation of as many as $2^{d(d-1)/2}$ component estimators. In many cases this number could be extremely large, and we must make a numerical approximation. Observing that the aggregation estimator is actually the expectation of a random variable that has a probability mass proportional to $v_m^{(1,2)}$ on individual estimator $\widehat{\Theta}_m^{(1)}$ for $m \in \mathcal{M}$, the Metropolis-Hastings algorithm can be exploited to provide such an approximation. The detail algorithm is shown in Algorithm 2.1.

Here, the neighbours of any m_t consist of all the sparsity patterns with a Manhattan

Algorithm 2.1 Metropolis-Hastings algorithm for the aggregation estimator

If the set of sparsity patterns $\mathcal{M} = \{0,1\}^{d(d-1)/2}$, then the aggregation estimator can be approximated by running a Metropolis-Hastings algorithm on a $d(d-1)/2$-dimensional hypercube:

(S1) Initialize $m_t = \{0\}^{d(d-1)/2}, t = 0$;

(S2) For each $t \geq 0$, generate m_t' with the uniform distribution on the neighbours of m_t;

(S3) Generate an $[0,1]$-uniformly distributed number r;

(S4) Put $m_{t+1} \leftarrow m_t'$, if $r < \min\{1, v_{m_t'}^{(1,2)}/v_{m_t}^{(1,2)}\}$; otherwise, $m_{t+1} \leftarrow m_t$;

(S5) Compute $\widehat{\Theta}_{m_{t+1}}^{(1)}$. Stop if $t > T_0 + T$; otherwise, update $t \leftarrow t+1$ and go to step (S2).

Then we can approximate $\widehat{\Theta}_{\text{agg}}^{(1,2)}$ by

$$\widehat{\widehat{\Theta}_{\text{agg}}^{(1,2)}} = \frac{1}{T} \sum_{t=T_0+1}^{T_0+T} \widehat{\Theta}_{m_t}^{(1)}, \tag{2.2.14}$$

where T_0 and T are positive integers.

distance of one to m_t. The following proposition shows that the resulting Markov chain ensures the ergodicity.

Proposition 2.2.1. *The Markov chain $\{m_t\}_{0 \leq t \leq T_0+T}$ generated by Algorithm 2.1 satisfies*

$$\lim_{T \to \infty} \frac{1}{T} \sum_{t=T_0+1}^{T_0+T} \widehat{\Theta}_{m_t}^{(1)} = \sum_{m \in \mathcal{M}} v_m^{(1,2)} \widehat{\Theta}_m^{(1)}, \text{ almost surely.} \tag{2.2.15}$$

The proof is straightforward as the Markov chain is clearly $v^{(1,2)}$-irreducible.

Here the Metropolis-Hastings algorithm incorporates a trade-off between sparsity and prediction to decide whether to add or remove an edge. Observe that the aggregation estimator would always estimate a precision matrix in which all the elements are nonzero, since all possible component estimators are linearly mixed. However, the Metropolis-Hastings Algorithm 2.1 leads to a sparse precision matrix estimator as in the regression case [42].

In high-dimensional cases where d is large, the Metropolis-Hastings Algorithm 2.1 might

take many iterations to generate a good random tour over the hypercube with as many as $d(d-1)/2$ dimensions, and thus could become computationally unsuitable. In this case, we need to approximate the model space and select a candidate set of edges before the execution of the Metropolis-Hastings algorithm.

We identify a candidate set of edges by applying some pre-screening method to the first subsample $\mathcal{D}_{n_1}^{(1)}$. Let the ordered list $\mathcal{Q}_d \subset \mathcal{S}_d$ be the set of selected edges. Then our aggregation process is constructed on this subset of edges. In that case, the set of sparsity patterns is given by

$$\mathcal{M} = \prod_{k=1}^{d(d-1)/2} \mathcal{C}_k, \tag{2.2.16}$$

where

$$\mathcal{C}_k := \begin{cases} \{0,1\} & \text{if } (i_k, j_k) \in \mathcal{Q}_d, \text{ where } (i_k, j_k) \text{ is the } k\text{th element of } \mathcal{S}_d \\ \{0\} & \text{otherwise} \end{cases}. \tag{2.2.17}$$

In practice where d is large, the Metropolis-Hastings algorithm introduced in Algorithm 2.1 will be applied to this reduced set of sparsity patterns instead of the set of all possible edges $\{0,1\}^{d(d-1)/2}$. Many pre-screening methods could be considered here, for instance, the graphical lasso [19, 56]. Note that we prefer to choose a small regularization parameter in the graphical lasso method to prevent any true edge being ruled out in this stage.

An alternative algorithm could be based on the ℓ_1 regularization paths from the graphical lasso method. Let $\mathcal{B}$ be a set of regularization parameters. For each $\beta \in \mathcal{B}$, let m_β be the sparsity pattern resulted from the graphical lasso method with tuning parameter β. Then the aggregation method is constructed on the set of sparsity patterns $\mathcal{M} = \{m_\beta : \beta \in \mathcal{B}\}$. However, in practical applications, the Metropolis-Hastings algorithm based on the ℓ_1 regularization paths always concentrations on a single model after convergence, and generally does not perform well.

2.3 Statistical Properties

In this section, we show that under some technical conditions, the risk of the aggregation estimator is bounded by the risk of the best component estimator in the dictionary plus a low aggregating price.

Following the notations in [5], define the uniformity class of covariance matrices invariant under permutations by

$$\mathcal{U}(q,\delta,M)=\Big\{\Sigma\in\mathbb{R}^{d\times d}:\ \Sigma\succ 0,\ \sigma_{ii}\le M,\ \sum_{j=1}^{d}|\sigma_{ij}|^{q}\le\delta,\ \text{for all } i\Big\}, \tag{2.3.1}$$

for $0\le q<1$, where M and δ are constants.

For any estimator $\widehat{\Theta}\succ 0$, we define the risk function

$$R(\widehat{\Theta})=\mathrm{tr}(\widehat{\Theta}\Sigma)-\mathrm{logdet}(\widehat{\Theta}). \tag{2.3.2}$$

Note that Σ is the true covariance matrix. Consider the Kullback-Leibler (KL) divergence

$$\mathrm{KL}(\widehat{\Theta})=R(\widehat{\Theta})-R(\Theta)\ge 0. \tag{2.3.3}$$

For each component estimator corresponding to $m\in\mathcal{M}$, we assume $\widehat{\Theta}_m^{(1)}$ exists and is unique. The following proposition relates the KL risk of the aggregation estimator $\widehat{\Theta}_{\mathrm{agg}}^{(1,2)}$ to the KL risks of individual estimators $\widehat{\Theta}_m^{(1)}$.

Proposition 2.3.1. *The aggregation estimator $\widehat{\Theta}_{\mathrm{agg}}^{(1,2)}$ in Definition 2.2.2 satisfies*

$$\mathrm{KL}(\widehat{\Theta}_{\mathrm{agg}}^{(1,2)})\le\min_{m\in\mathcal{M}}\Big\{\mathrm{KL}(\widehat{\Theta}_m^{(1)})+\frac{2}{n_2}\log\frac{1}{\pi_m}+\mathrm{tr}((\widehat{\Theta}_m^{(1)}-\widehat{\Theta}_{\mathrm{agg}}^{(1,2)})(\widehat{S}_{n_2}^{(2)}-\Sigma))\Big\} \tag{2.3.4}$$

$$\le\min_{m\in\mathcal{M}}\Big\{\mathrm{KL}(\widehat{\Theta}_m^{(1)})+\frac{2}{n_2}\log\frac{1}{\pi_m}+\mathrm{tr}(\widehat{\Theta}_m^{(1)}(\widehat{S}_{n_2}^{(2)}-\Sigma))\Big\}$$
$$+\|\widehat{S}_{n_2}^{(2)}-\Sigma\|\cdot\mathrm{tr}(\widehat{\Theta}_{\mathrm{agg}}^{(1,2)}). \tag{2.3.5}$$

It is shown in [50] that under some conditions

$$\|\widehat{\Sigma}_{n_2}^{(2)} - \Sigma\| = O_P\left(\sqrt{\frac{d}{n_2}}\right), \tag{2.3.6}$$

thus if we use the empirical covariance matrix as $\widehat{S}_{n_2}^{(2)}$, Proposition 2.3.1 implies that this would lead to a deterioration of convergence rate in high dimensions. Then we choose to use the thresholded covariance matrix estimation in [5].

We consider the following assumption for further analysis of the remainder term.

Assumption 2.3.1. *The set of sparsity patterns $\mathcal{M}$ satisfies the following condition:*

$$\max_{m\in\mathcal{M}} \operatorname{tr}(\widehat{\Theta}_m^{(1)}) = O_P(d). \tag{2.3.7}$$

This assumption ensures a fast convergence rate of the aggregation estimator. It is shown in [11] that the inverse of $\widehat{\Theta}_m^{(1)}$ is a solution to the following problem that maximizes the determinant of a symmetric positive definite matrix Z:

$$\begin{aligned} &\max_{Z\succ 0} \quad \operatorname{logdet} Z, \\ &\text{s.t.} \quad Z_{ij} = [\widehat{\Sigma}_{n_1}^{(1)}]_{ij},\ \forall (i,j) \in E_m \cup \{(i,i), i = 1,\ldots,d\}. \end{aligned} \tag{2.3.8}$$

In general, for any two graphs G and G' with sparsity patterns m and m', respectively, if G is a subgraph of G', then the trace of $\widehat{\Theta}_{m'}^{(1)}$ would always be larger than that of $\widehat{\Theta}_m^{(1)}$. Thus the most dense graph in the dictionary would always be able to achieve the maximum trace among all individual estimators. This assumption claims that the diagonal entries of the true precision matrix Θ are well estimated by all individual estimators in the dictionary $\mathcal{M}$.

Let $m^* \in \mathcal{M}$ be the sparsity pattern that attains the minimum KL risk in $\mathcal{M}$:

$$m^* = \operatorname*{argmin}_{m\in\mathcal{M}} \operatorname{KL}(\widehat{\Theta}_m^{(1)}). \tag{2.3.9}$$

The following theorem shows the oracle inequality that the aggregation estimator satisfies.

Theorem 2.3.1. *Suppose Assumption 2.3.1 holds. Uniformly on $\mathcal{U}(q, \delta, M)$, for sufficiently large K, if the thresholding parameter $\lambda = K\sqrt{\frac{\log d}{n_2}}$, and $\frac{\log d}{n_2} = o(1)$, then the aggregation estimator $\widehat{\Theta}_{\text{agg}}^{(1,2)}$ satisfies*

$$\text{KL}(\widehat{\Theta}_{\text{agg}}^{(1,2)}) - \min_{m \in \mathcal{M}} \text{KL}(\widehat{\Theta}_m^{(1)}) = O_P\left(\frac{s^* \log d}{n_2} + d\left(\frac{\log d}{n_2}\right)^{(1-q)/2}\right), \tag{2.3.10}$$

where $s^* = |m^*|_1$ is the number of nonzero off-diagonal elements of $\widehat{\Theta}_{m^*}$.

This theorem yields a rate of convergence of the excess risk. In particular, if the set of sparsity patterns $\mathcal{M}$ also includes the true sparsity pattern m^0. Let s^0 be the number of nonzero off-diagonal elements of Θ. It is shown in [58] that under some technical conditions

$$\text{KL}(\widehat{\Theta}_{m^0}^{(1)}) = O_P\left(\frac{(s^0 + d)\log d}{n_2}\right). \tag{2.3.11}$$

Combine it with Theorem 2.3.1 and assume that $s^0 = O(d)$, we obtain

$$\text{KL}(\widehat{\Theta}_{\text{agg}}^{(1,2)}) = O_P\left(d\left(\frac{\log d}{n_2}\right)^{(1-q)/2}\right), \tag{2.3.12}$$

for $0 \leq q < 1$.

2.4 Experimental Results

Here we provide empirical evidence to illustrate the usefulness of the proposed graphical aggregation method and compare it with other state-of-the-art methods in the parameter estimation and graph recovery using simulated and real datasets.

2.4.1 Analysis of Simulated Data

We generate synthetic datasets with sample size $n = 200$ or 400, and number of nodes $d = 50, 100$ or 200. We use the following three models for simulating graphs and precision matrices. Figure 2.1 displays a typical run of the generated graphs of the precision matrices when $d = 100$.

(a) "AR": The off-diagonal (i, j)th element of the adjacency matrix is set to be 1 if $|i-j| = 1$ and 0 otherwise;

(b) "Hub": The vertices are evenly partitioned into $d/10$ disjoint groups. Each group is associated with a center vertex i in that group and the off-diagonal (i, j)th element of the adjacency matrix is set to 1 if j also belongs to the same group as i and 0 otherwise;

(c) "Random": The off-diagonal (i, j)th element of the adjacency matrix is randomly set to be 1 with certain probability and 0 otherwise.

We compare the graphical aggregation (GAggregation) with the graphical lasso (Glasso) estimator [19, 56] and the constrained ℓ_1 minimization method (CLIME) [7] with tuning parameters determined by 10-fold cross-validation (CV). We also consider the method of multiple testing of hypotheses about vanishing partial correlation coefficients [15, 16], where the family-wise error rates for incorrect edge inclusion are controlled by Bonferroni correction at level $\alpha = 0.05$. We refer this method as "PCorTest" in this work. For the GAggregation method, the full dataset is random partitioned into two subsamples with equal size. For other methods, the graphs are estimated using the full dataset in order to make the comparison fair. In addition, we compare with a method that we call the "RefitGlasso", which is a two-step procedure—in the first step, a sparse precision matrix is obtained by the glasso; in the second step, a Gaussian model is refit without regularization, but enforcing the sparsity pattern obtained in the first step.

For any estimator $\widehat{\Theta} \succ 0$, we use the following criteria for the comparisons.

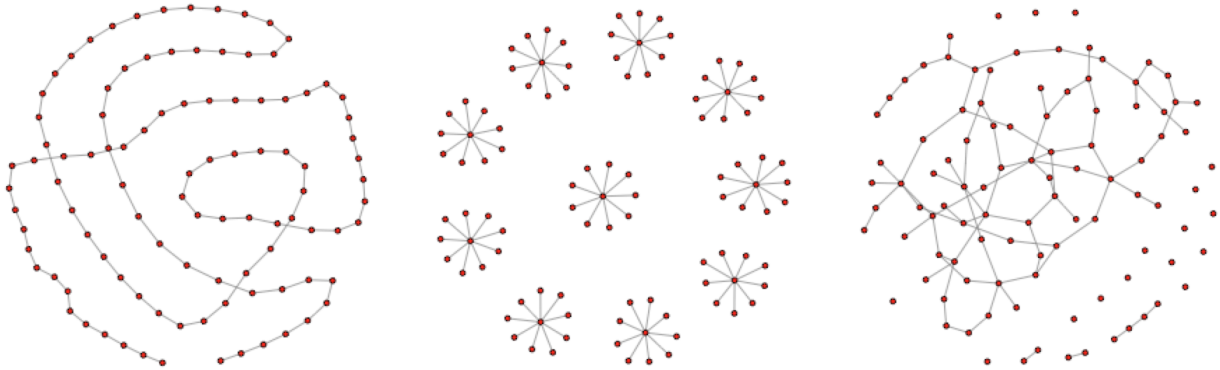


Figure 2.1: An illustration of the three graph patterns with 100 nodes; Left: AR; Middle: Hub; Right: Random.

(1) Squared Frobenius norm errors:

$$\text{Frobenius} = \|\widehat{\Theta} - \Theta\|_F^2; \tag{2.4.1}$$

(2) Kullback-Leibler loss:

$$\text{KL} = -\log\det(\widehat{\Theta}) + \text{trace}(\widehat{\Theta}\Sigma) - (-\log\det(\Theta) + d); \tag{2.4.2}$$

(3) Precision: number of correctly estimated edges divided by the total number of edges in the estimated graph. Let $G = (V, E)$ be a d-dimensional graph and let $\widehat{G} = (\widehat{V}, \widehat{E})$ be an estimated graph. We define

$$\text{Precision} = \frac{|\widehat{E} \cap E|}{|\widehat{E}|}; \tag{2.4.3}$$

(4) Recall: number of correctly estimated edges divided by the total number of edges in the true graph, defined by

$$\text{Recall} = \frac{|\widehat{E} \cap E|}{|E|}; \tag{2.4.4}$$

(5) F_1-score: a weighted average of the precision and recall, defined by

$$F_1\text{-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{2.4.5}$$

where an F_1-score reaches its best value at 1 and worst score at 0.

Table 2.1 shows the results of the quantitative comparison of different methods, where we repeat the experiments 50 times and report the averaged values with their standard errors. We can see that GAggregation, CLIME and RefitGlasso methods perform better than Glasso and PCorTest in term of the squared Frobenius norm errors, while GAggregation, Glasso and RefitGlasso are better than CLIME and PCorTest in term of the Kullback-Leibler loss. For the comparison of graph structure recovery, GAggregation, PCorTest and RefitGlasso outperform other methods. Notice that the graphs selected by Glasso and CLIME are overly too dense—for higher tuning parameter, the graph is sparse but the coefficients are strongly biased to shrinkage, so CV shows poor performance; for lower tuning parameter, most of the excess edges have very small coefficients while the true edges' coefficients do not suffer from too much shrinkage, thus CV would prefer such graph.

Figure 2.2 displays the evolution of the Metropolis-Hastings algorithm, showing evidence that the algorithm converges after 4000 iterations. As is seen from Figure 2.3, the sparsity pattern is well recovered and the estimates are close to the true values.

Figure 2.4 shows a typical realization of the graphical aggregation method, varying the regularization parameter λ in the pre-screening Glasso method, where we can see that the result for graphical aggregation is not quite sensitive to the pre-screening method.

The computational complexity of the graphical lasso algorithm [19] which uses a row-by-row block coordinate method is roughly $O(d^3)$. For the graphical aggregation estimator, the complexity is roughly $O(Ld^3)$, where $L = T + T_0$ is the number of iterations in the Metropolis-Hastings algorithm. Note that L can be dramatically reduced if we keep track of

Table 2.1: Quantitative comparison of different methods on simulated data.

		$(n, d) = (200, 50)$				
Model	Estimator	Frobenius	KL	Precision	Recall	F_1-score
AR	GAggregation	6.06 (0.97)	1.61 (0.29)	0.73 (0.05)	0.97 (0.02)	0.83 (0.03)
	PCorTest	5.91 (1.48)	1.73 (0.49)	0.99 (0.01)	0.89 (0.04)	0.94 (0.02)
	Glasso	6.39 (0.82)	1.51 (0.14)	0.21 (0.03)	1.00 (0.00)	0.35 (0.04)
	RefitGlasso	6.69 (1.08)	1.65 (0.19)	0.66 (0.09)	0.98 (0.02)	0.78 (0.06)
	CLIME	5.46 (0.76)	1.77 (0.25)	0.15 (0.02)	1.00 (0.00)	0.26 (0.03)
Hub	GAggregation	6.29 (1.57)	1.44 (0.31)	0.65 (0.06)	0.97 (0.02)	0.78 (0.04)
	PCorTest	26.56 (3.77)	6.02 (0.67)	0.98 (0.02)	0.53 (0.06)	0.69 (0.05)
	Glasso	16.43 (1.84)	1.53 (0.13)	0.18 (0.02)	1.00 (0.00)	0.31 (0.03)
	RefitGlasso	8.33 (2.36)	1.65 (0.31)	0.57 (0.09)	0.98 (0.02)	0.71 (0.07)
	CLIME	7.34 (0.99)	3.03 (0.43)	0.14 (0.01)	1.00 (0.00)	0.24 (0.02)
Random	GAggregation	5.94 (1.72)	1.72 (0.46)	0.73 (0.05)	0.91 (0.06)	0.81 (0.04)
	PCorTest	8.55 (3.63)	2.45 (0.93)	0.99 (0.02)	0.72 (0.13)	0.83 (0.09)
	Glasso	7.09 (1.18)	1.31 (0.16)	0.20 (0.03)	1.00 (0.01)	0.33 (0.04)
	RefitGlasso	6.07 (1.75)	1.59 (0.37)	0.58 (0.10)	0.92 (0.05)	0.71 (0.07)
	CLIME	4.59 (0.91)	1.68 (0.32)	0.13 (0.02)	1.00 (0.00)	0.23 (0.03)
		$(n, d) = (200, 100)$				
Model	Estimator	Frobenius	KL	Precision	Recall	F_1-score
AR	GAggregation	13.94 (2.58)	3.10 (0.55)	0.84 (0.03)	0.98 (0.02)	0.90 (0.02)
	PCorTest	17.48 (2.70)	4.55 (0.72)	0.91 (0.03)	0.88 (0.03)	0.89 (0.02)
	Glasso	20.89 (1.77)	3.71 (0.21)	0.17 (0.02)	1.00 (0.00)	0.29 (0.02)
	RefitGlasso	15.42 (2.68)	3.18 (0.43)	0.68 (0.07)	0.98 (0.02)	0.80 (0.05)
	CLIME	13.90 (1.47)	4.79 (0.50)	0.13 (0.01)	1.00 (0.00)	0.23 (0.02)
Hub	GAggregation	14.27 (2.77)	3.29 (0.58)	0.77 (0.03)	0.97 (0.02)	0.86 (0.02)
	PCorTest	72.07 (5.98)	14.92 (0.93)	0.81 (0.08)	0.42 (0.04)	0.55 (0.04)
	Glasso	38.56 (1.67)	3.61 (0.17)	0.15 (0.01)	1.00 (0.00)	0.27 (0.02)
	RefitGlasso	15.97 (3.62)	3.42 (0.53)	0.56 (0.06)	0.97 (0.02)	0.71 (0.05)
	CLIME	17.27 (2.14)	8.48 (0.89)	0.12 (0.01)	1.00 (0.00)	0.22 (0.02)
Random	GAggregation	12.89 (3.05)	3.94 (0.82)	0.85 (0.04)	0.88 (0.05)	0.86 (0.04)
	PCorTest	26.05 (7.44)	7.15 (1.70)	0.82 (0.06)	0.58 (0.10)	0.68 (0.08)
	Glasso	16.58 (2.24)	3.04 (0.22)	0.18 (0.02)	1.00 (0.00)	0.30 (0.04)
	RefitGlasso	13.69 (2.18)	3.90 (0.64)	0.55 (0.11)	0.86 (0.07)	0.66 (0.08)
	CLIME	10.39 (1.12)	4.27 (0.50)	0.11 (0.01)	1.00 (0.00)	0.20 (0.02)
		$(n, d) = (400, 100)$				
Model	Estimator	Frobenius	KL	Precision	Recall	F_1-score
AR	GAggregation	5.54 (1.12)	1.14 (0.18)	0.82 (0.03)	1.00 (0.00)	0.90 (0.02)
	PCorTest	2.46 (0.30)	0.51 (0.06)	0.99 (0.01)	1.00 (0.00)	1.00 (0.00)
	Glasso	12.17 (0.95)	1.93 (0.10)	0.17 (0.01)	1.00 (0.00)	0.29 (0.02)
	RefitGlasso	5.57 (0.78)	1.12 (0.10)	0.89 (0.04)	1.00 (0.00)	0.94 (0.02)
	CLIME	7.55 (0.58)	2.08 (0.21)	0.14 (0.01)	1.00 (0.00)	0.25 (0.02)
Hub	GAggregation	5.27 (1.12)	1.16 (0.18)	0.82 (0.03)	0.99 (0.01)	0.90 (0.02)
	PCorTest	11.20 (2.66)	2.96 (0.69)	0.99 (0.01)	0.90 (0.03)	0.94 (0.02)
	Glasso	23.76 (1.76)	1.91 (0.09)	0.15 (0.02)	1.00 (0.00)	0.26 (0.03)
	RefitGlasso	5.46 (0.91)	1.18 (0.18)	0.78 (0.08)	1.00 (0.01)	0.87 (0.05)
	CLIME	8.31 (0.78)	3.20 (0.27)	0.13 (0.00)	1.00 (0.00)	0.23 (0.01)
Random	GAggregation	5.46 (0.85)	1.98 (0.33)	0.84 (0.03)	0.94 (0.04)	0.88 (0.03)
	PCorTest	6.53 (1.76)	2.42 (0.65)	0.99 (0.01)	0.82 (0.06)	0.90 (0.04)
	Glasso	6.48 (0.75)	1.69 (0.15)	0.19 (0.02)	1.00 (0.00)	0.32 (0.03)
	RefitGlasso	5.17 (0.67)	1.79 (0.25)	0.65 (0.08)	0.95 (0.03)	0.76 (0.06)
	CLIME	5.03 (0.54)	1.95 (0.17)	0.14 (0.02)	1.00 (0.00)	0.24 (0.02)
		$(n, d) = (400, 200)$				
Model	Estimator	Frobenius	KL	Precision	Recall	F_1-score
AR	GAggregation	14.27 (2.42)	2.99 (0.53)	0.89 (0.02)	0.99 (0.01)	0.94 (0.01)
	PCorTest	6.02 (1.13)	1.36 (0.28)	0.92 (0.02)	1.00 (0.00)	0.96 (0.01)
	Glasso	29.93 (2.37)	4.62 (0.19)	0.14 (0.02)	1.00 (0.00)	0.25 (0.03)
	RefitGlasso	11.71 (1.25)	2.29 (0.21)	0.89 (0.03)	1.00 (0.00)	0.94 (0.02)
	CLIME	16.33 (1.53)	5.66 (0.61)	0.10 (0.03)	1.00 (0.00)	0.18 (0.05)
Hub	GAggregation	13.49 (2.63)	2.89 (0.54)	0.83 (0.04)	0.99 (0.01)	0.90 (0.03)
	PCorTest	44.47 (4.72)	11.25 (1.11)	0.89 (0.03)	0.79 (0.02)	0.84 (0.02)
	Glasso	54.98 (1.84)	4.45 (0.16)	0.13 (0.00)	1.00 (0.00)	0.23 (0.01)
	RefitGlasso	11.52 (1.55)	2.39 (0.23)	0.75 (0.07)	1.00 (0.00)	0.86 (0.04)
	CLIME	19.65 (1.68)	8.17 (0.50)	0.14 (0.00)	1.00 (0.00)	0.25 (0.01)
Random	GAggregation	10.43 (1.24)	3.88 (0.51)	0.95 (0.02)	0.90 (0.04)	0.92 (0.02)
	PCorTest	13.21 (1.81)	4.95 (0.74)	0.87 (0.04)	0.77 (0.06)	0.81 (0.04)
	Glasso	11.67 (1.18)	3.24 (0.20)	0.16 (0.01)	1.00 (0.00)	0.28 (0.02)
	RefitGlasso	8.91 (1.26)	3.24 (0.48)	0.61 (0.09)	0.93 (0.03)	0.73 (0.07)
	CLIME	9.77 (0.97)	3.56 (0.27)	0.12 (0.01)	1.00 (0.00)	0.21 (0.02)

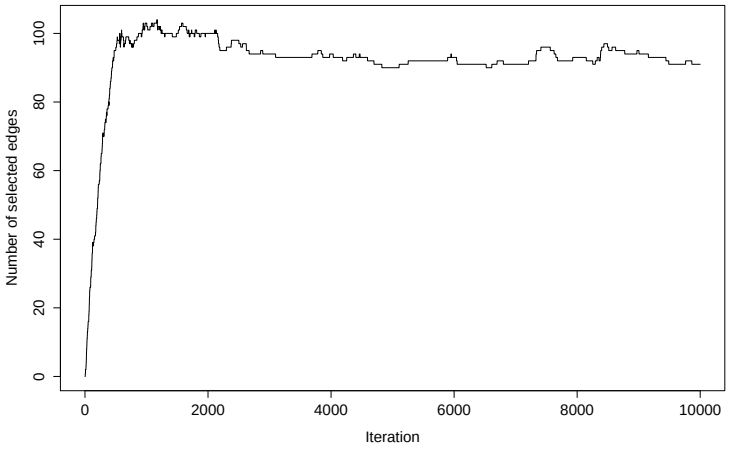


Figure 2.2: Number of selected edges by the Metropolis-Hastings algorithm as a function of iterations, given a typical run of the graphical aggregation method for hub graph on simulated data with $(n, d) = (400, 100)$.

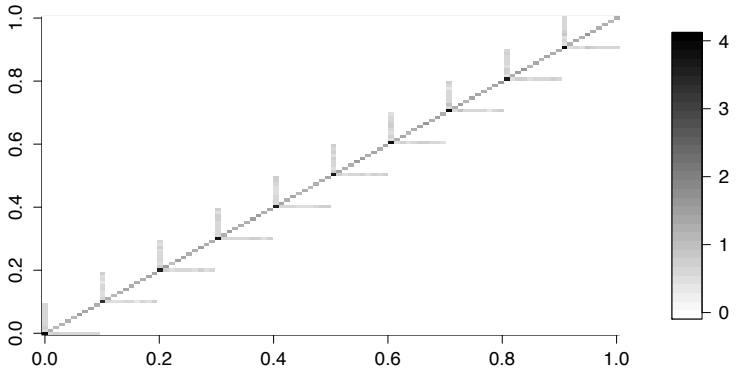


Figure 2.3: A heatmap of the precision matrix estimated by the graphical aggregation method for hub graph on simulated data with $(n, d) = (400, 100)$. The truth $\theta_{ii} = 3.77$, $\theta_{jj} = 1.31$, and $\theta_{ij} = 0.63$, where i is a center node and j belongs to the same group as i.

and store the component estimators to avoid duplicated estimation of precision matrix given the same sparsity pattern.

2.4.2 Analysis of Microarray Data

In this study, we consider a real-world dataset based on Affymetrix GeneChip microarrays for the plant *Arabidopsis thaliana* [53]. The sample size is $n = 118$. A nonparanormal

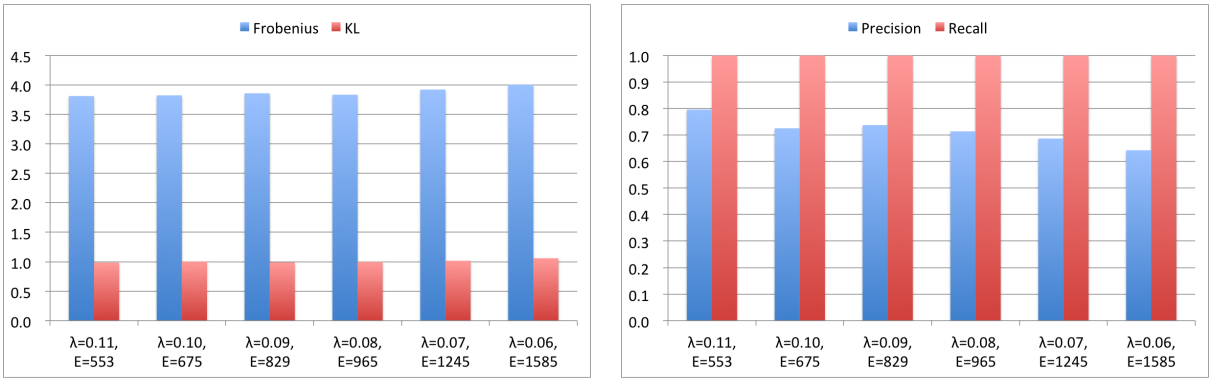


Figure 2.4: A typical realization of the graphical aggregation method for hub graph on simulated data with $(n, d) = (400, 100)$, varying the regularization parameter λ in the pre-screening Glasso method. Number of selected candidate edges by the pre-screening Glasso (marked as E) are also reported.

transformation is estimated and the expression levels for each chip are replaced by their respective normal scores, subject to a Winsorized truncation [31]. A subset of $d = 40$ genes from the isoprenoid pathway are chosen, and we study the associations among them using the graphical aggregation method and graphical lasso with tuning parameter determined by cross-validation.

The graphical aggregation method selects 111 edges and the graphical lasso method selects 378 edges. Among those selected edges, 102 edges are identified by both methods. Figure 2.5 shows grids of rectangles with gray scale corresponding to the absolute values in the estimated precision matrix for each method. Figure 2.6 provides the estimated graphs.

We also analyze a dataset on microarrays for the gene expression levels [37]. Of the 4,238 genes in immortalized B cells of $n = 295$ normal individuals, we select $d = 318$ genes that are associated with phenotypes in genome-wide association studies. We study the estimated graphs obtained from the graphical aggregation method and graphical lasso with tuning parameter determined by cross-validation. The expression levels for each gene are pre-processed by log-transformation and standardization.

The graphical aggregation method selects 1,631 edges and the graphical lasso selects

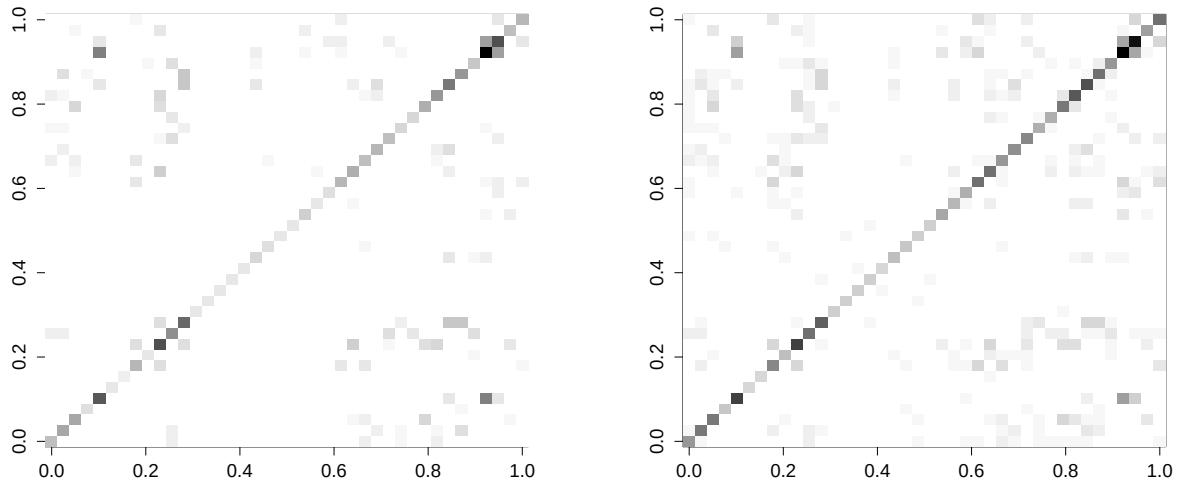


Figure 2.5: Grids of rectangles with gray scale corresponding to the absolute values in the estimated precision matrix for each method; Left: GAggregation; Right: Glasso.

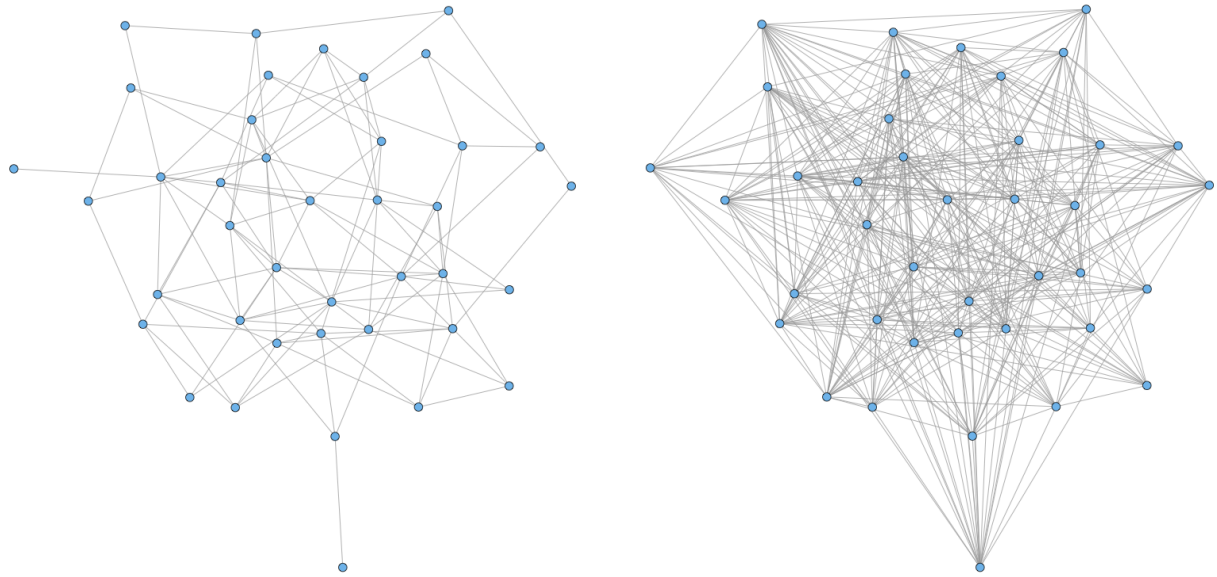


Figure 2.6: Estimated graphs for microarray data example for the plant *Arabidopsis thaliana*; Left: GAggregation; Right: Glasso.

12,514 edges. Among those selected edges, 1,542 edges are identified by both methods. Figure 2.7 provides the estimated graphs. The graph estimated by graphical aggregation is more informative than that estimated by graphical lasso, since the latter graph is too dense.

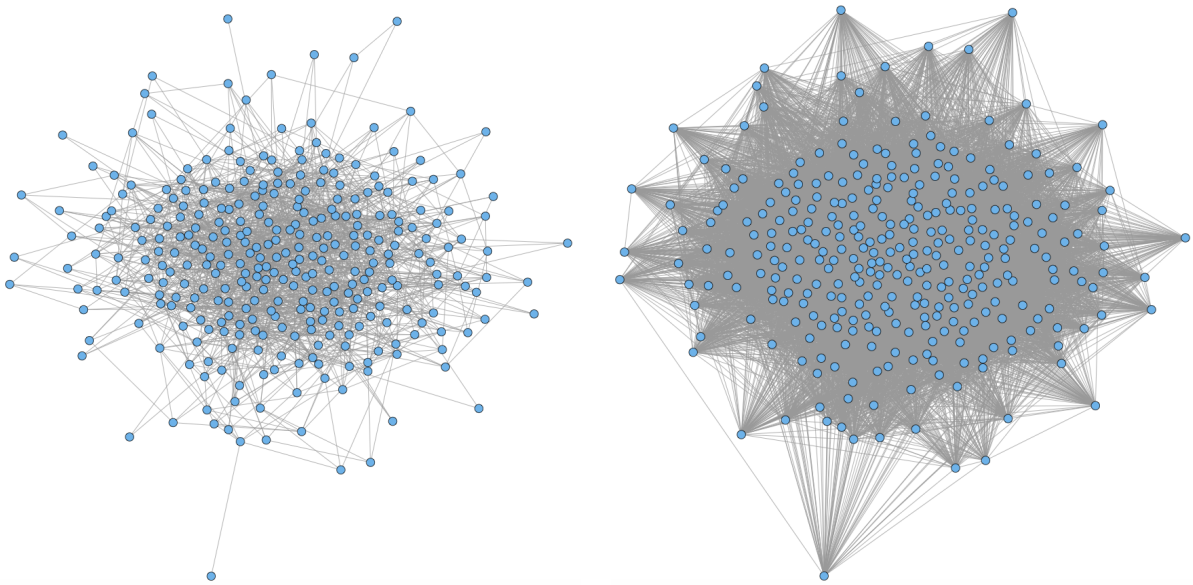


Figure 2.7: Estimated graphs for microarray data example for immortalized human B cells; Left: GAggregation; Right: Glasso.

2.5 Conclusion

In this chapter, we propose a new aggregation method for estimating the precision matrix in Gaussian graphical models, by considering a convex combination of a suitable set of individual estimators with different underlying sparsity patterns. We investigate the risk of this aggregation estimator and show by an oracle that it is comparable to the risk of the best estimator based on a single graph. Experimental results validate the usefulness of our method in practice.

2.6 Proofs

Proof of Proposition 2.3.1.

Proof. For any $m \in \mathcal{M}$ and individual estimator $\widehat{\Theta}_m \succ 0$

$$\mathrm{KL}(\widehat{\Theta}_m^{(1)}) = \mathrm{tr}(\widehat{\Theta}_m^{(1)}\Sigma) - \mathrm{logdet}(\widehat{\Theta}_m^{(1)}) - d + \mathrm{logdet}(\Theta). \tag{2.6.1}$$

Similarly, for the aggregation estimator $\widehat{\Theta}_{\text{agg}}^{(1,2)} \succ 0$

$$\text{KL}(\widehat{\Theta}_{\text{agg}}^{(1,2)}) = \text{tr}(\widehat{\Theta}_{\text{agg}}^{(1,2)}\Sigma) - \text{logdet}(\widehat{\Theta}_{\text{agg}}^{(1,2)}) - d + \text{logdet}(\Theta). \tag{2.6.2}$$

Based on the convexity of $\text{KL}(\cdot)$, we obtain

$$\text{KL}(\widehat{\Theta}_{\text{agg}}^{(1,2)}) \leq \sum_{m\in\mathcal{M}} v_m^{(1,2)} \,\text{KL}(\widehat{\Theta}_m^{(1)}). \tag{2.6.3}$$

Let $\widetilde{m} \in \mathcal{M}$ being any sparsity pattern attaining

$$\min_{m\in\mathcal{M}} \left\{ \text{KL}(\widehat{\Theta}_m^{(1)}) + \frac{2}{n_2}\log\frac{1}{\pi_m} + \text{tr}(\widehat{\Theta}_m^{(1)}(\widehat{S}_{n_2}^{(2)} - \Sigma)) \right\}. \tag{2.6.4}$$

Then according the definition of $v_m^{(1,2)}$, we obtain

$$\begin{aligned} \frac{v_{\widetilde{m}}^{(1,2)}}{v_m^{(1,2)}} =& \frac{\exp\{(n_2/2)(\text{logdet}(\widehat{\Theta}_{\widetilde{m}}^{(1)}) - \text{tr}(\widehat{\Theta}_{\widetilde{m}}^{(1)}\widehat{S}_{n_2}^{(2)}))\}\pi_{\widetilde{m}}}{\exp\{(n_2/2)(\text{logdet}(\widehat{\Theta}_m^{(1)}) - \text{tr}(\widehat{\Theta}_m^{(1)}\widehat{S}_{n_2}^{(2)}))\}\pi_m} \qquad (2.6.5) \\ =& \frac{\pi_{\widetilde{m}}}{\pi_m}\exp\{-(n_2/2)[\text{KL}(\widehat{\Theta}_{\widetilde{m}}^{(1)}) - \text{KL}(\widehat{\Theta}_m^{(1)}) \\ & + \text{tr}((\widehat{\Theta}_{\widetilde{m}}^{(1)} - \widehat{\Theta}_m^{(1)})(\widehat{S}_{n_2}^{(2)} - \Sigma))]\}, \qquad (2.6.6) \end{aligned}$$

where the last equality follows from the fact that

$$\begin{aligned} \text{KL}(\widehat{\Theta}_m^{(1)}) =& \,\text{tr}(\widehat{\Theta}_m^{(1)}(\Sigma - \widehat{S}_{n_2}^{(2)})) - (\text{logdet}(\widehat{\Theta}_m^{(1)}) - \text{tr}(\widehat{\Theta}_m^{(1)}\widehat{S}_{n_2}^{(2)})) \\ & - d + \text{logdet}(\Theta). \end{aligned} \tag{2.6.7}$$

Note that $\sum_{m\in\mathcal{M}} v_m^{(1,2)} = 1$. Then the inequality (2.6.3) can be written as

$$\mathrm{KL}(\widehat{\Theta}_{\mathrm{agg}}^{(1,2)}) \le \mathrm{KL}(\widehat{\Theta}_{\widetilde{m}}^{(1)}) + \sum_{m\in\mathcal{M}} v_m^{(1,2)}(\mathrm{KL}(\widehat{\Theta}_m^{(1)}) - \mathrm{KL}(\widehat{\Theta}_{\widetilde{m}}^{(1)})) \tag{2.6.8}$$

$$\le \mathrm{KL}(\widehat{\Theta}_{\widetilde{m}}^{(1)}) + \frac{2}{n_2}\sum_{m\in\mathcal{M}} v_m^{(1,2)} \log\frac{v_{\widetilde{m}}^{(1,2)}}{v_m^{(1,2)}} + \frac{2}{n_2}\sum_{m\in\mathcal{M}} v_m^{(1,2)} \log\frac{\pi_m}{\pi_{\widetilde{m}}}$$
$$+ \sum_{m\in\mathcal{M}} v_m^{(1,2)} \operatorname{tr}(\widehat{\Theta}_{\widetilde{m}}^{(1)}(\widehat{S}_{n_2}^{(2)} - \Sigma)) - \sum_{m\in\mathcal{M}} v_m^{(1,2)} \operatorname{tr}(\widehat{\Theta}_m^{(1)}(\widehat{S}_{n_2}^{(2)} - \Sigma)) \tag{2.6.9}$$

$$\le \mathrm{KL}(\widehat{\Theta}_{\widetilde{m}}^{(1)}) + \frac{2}{n_2}\sum_{m\in\mathcal{M}} v_m^{(1,2)} \log\frac{\pi_m}{v_m^{(1,2)}} + \frac{2}{n_2}\sum_{m\in\mathcal{M}} v_m^{(1,2)} \log\frac{v_{\widetilde{m}}^{(1,2)}}{\pi_{\widetilde{m}}}$$
$$+ \operatorname{tr}(\widehat{\Theta}_{\widetilde{m}}^{(1)}(\widehat{S}_{n_2}^{(2)} - \Sigma)) - \sum_{m\in\mathcal{M}} v_m^{(1,2)} \operatorname{tr}(\widehat{\Theta}_m^{(1)}(\widehat{S}_{n_2}^{(2)} - \Sigma)). \tag{2.6.10}$$

According to the fact that

$$\log v_{\widetilde{m}}^{(1,2)} \le 0, \text{ and } \sum_{m\in\mathcal{M}} v_m^{(1,2)} \log\frac{\pi_m}{v_m^{(1,2)}} \le 0, \tag{2.6.11}$$

we obtain

$$\mathrm{KL}(\widehat{\Theta}_{\mathrm{agg}}^{(1,2)}) \le \mathrm{KL}(\widehat{\Theta}_{\widetilde{m}}^{(1)}) + \frac{2}{n_2}\log\frac{1}{\pi_{\widetilde{m}}} + \operatorname{tr}(\widehat{\Theta}_{\widetilde{m}}^{(1)}(\widehat{S}_{n_2}^{(2)} - \Sigma))$$
$$- \operatorname{tr}(\widehat{\Theta}_{\mathrm{agg}}^{(1,2)}(\widehat{S}_{n_2}^{(2)} - \Sigma)). \tag{2.6.12}$$

It is shown in [51] that for any $d \times d$ real symmetric matrix A and any $d \times d$ positive semidefinite matrix B

$$\lambda_d(A)\operatorname{tr}(B) \le \operatorname{tr}(AB) \le \lambda_1(A)\operatorname{tr}(B), \tag{2.6.13}$$

where $\lambda_i(A)$ is the ith largest eigenvalue of A.

Following this property, we obtain

$$\left|\mathrm{tr}(\widehat{\Theta}_{\mathrm{agg}}^{(1,2)}(\widehat{S}_{n_2}^{(2)}-\Sigma))\right| \leq \|\widehat{S}_{n_2}^{(2)}-\Sigma\| \cdot \mathrm{tr}(\widehat{\Theta}_{\mathrm{agg}}^{(1,2)}). \tag{2.6.14}$$

Then we write the inequality (2.6.12) as

$$\begin{aligned}\mathrm{KL}(\widehat{\Theta}_{\mathrm{agg}}^{(1,2)}) \leq & \mathrm{KL}(\widehat{\Theta}_{\widetilde{m}}^{(1)}) + \frac{2}{n_2}\log\frac{1}{\pi_{\widetilde{m}}} + \mathrm{tr}(\widehat{\Theta}_{\widetilde{m}}^{(1)}(\widehat{S}_{n_2}^{(2)}-\Sigma)) \\ & + \|\widehat{S}_{n_2}^{(2)}-\Sigma\| \cdot \mathrm{tr}(\widehat{\Theta}_{\mathrm{agg}}^{(1,2)}).\end{aligned} \tag{2.6.15}$$

According to the definition of $\widetilde{m}$ as in (2.6.4), the proposition then follows. □

Proof of Theorem 2.3.1.

Proof. Since $m^* \in \mathcal{M}$ is any sparsity pattern attaining $\min_{m\in\mathcal{M}} \mathrm{KL}(\widehat{\Theta}_m^{(1)})$, then

$$\begin{aligned}\mathrm{KL}(\widehat{\Theta}_{\mathrm{agg}}^{(1,2)}) \leq & \mathrm{KL}(\widehat{\Theta}_{m^*}^{(1)}) + \frac{2}{n_2}\log\frac{1}{\pi_{m^*}} + \mathrm{tr}(\widehat{\Theta}_{m^*}^{(1)}(\widehat{S}_{n_2}^{(2)}-\Sigma)) \\ & + \|\widehat{S}_{n_2}^{(2)}-\Sigma\| \cdot \mathrm{tr}(\widehat{\Theta}_{\mathrm{agg}}^{(1,2)})\end{aligned} \tag{2.6.16}$$

$$\begin{aligned}= & \min_{m\in\mathcal{M}} \mathrm{KL}(\widehat{\Theta}_m^{(1)}) + \frac{2}{n_2}\log\frac{1}{\pi_{m^*}} + \mathrm{tr}(\widehat{\Theta}_{m^*}^{(1)}(\widehat{S}_{n_2}^{(2)}-\Sigma)) \\ & + \|\widehat{S}_{n_2}^{(2)}-\Sigma\| \cdot \mathrm{tr}(\widehat{\Theta}_{\mathrm{agg}}^{(1,2)}).\end{aligned} \tag{2.6.17}$$

The normalization factor H of the prior probability π_m in Definition 2.2.2 satisfies the

following inequality

$$H = \sum_{m \in \mathcal{M}} \left(\frac{|m|_1}{ed(d-1)} \right)^{|m|_1} \tag{2.6.18}$$

$$\leq \sum_{k=0}^{d(d-1)/2} \binom{d(d-1)/2}{k} \left(\frac{k}{ed(d-1)} \right)^k \tag{2.6.19}$$

$$\leq \sum_{k=0}^{d(d-1)/2} \left(\frac{ed(d-1)/2}{k} \right)^k \left(\frac{k}{ed(d-1)} \right)^k \tag{2.6.20}$$

$$\leq 2. \tag{2.6.21}$$

Then for any $m \in \mathcal{M}$ we have

$$\log \left(\frac{1}{\pi_m} \right) \leq |m|_1 \log \left(\frac{ed(d-1)}{|m|_1 \vee 1} \right) + \log 2. \tag{2.6.22}$$

Thus we obtain

$$\frac{2}{n_2} \log \frac{1}{\pi_{m^*}} = O_P \left(\frac{s^* \log d}{n_2} \right), \tag{2.6.23}$$

where $s^* = |m^*|_1$ is the number of nonzero off-diagonal elements of $\widehat{\Theta}_{m^*}$.

Note that

$$\operatorname{tr}(\widehat{\Theta}^{(1)}_{m^*}(\widehat{S}^{(2)}_{n_2} - \Sigma)) \leq \|\widehat{S}^{(2)}_{n_2} - \Sigma\| \cdot \operatorname{tr}(\widehat{\Theta}^{(1)}_{m^*}) \tag{2.6.24}$$

$$\leq \|\widehat{S}^{(2)}_{n_2} - \Sigma\| \cdot \max_{m \in \mathcal{M}} \operatorname{tr}(\widehat{\Theta}^{(1)}_{m}), \tag{2.6.25}$$

and

$$\|\widehat{S}^{(2)}_{n_2} - \Sigma\| \cdot \operatorname{tr}(\widehat{\Theta}^{(1,2)}_{\mathrm{agg}}) = \|\widehat{S}^{(2)}_{n_2} - \Sigma\| \cdot \sum_{m \in \mathcal{M}} v^{(1,2)}_m \operatorname{tr}(\widehat{\Theta}^{(1)}_{m}) \tag{2.6.26}$$

$$\leq \|\widehat{S}^{(2)}_{n_2} - \Sigma\| \cdot \max_{m \in \mathcal{M}} \operatorname{tr}(\widehat{\Theta}^{(1)}_{m}). \tag{2.6.27}$$

Given the assumptions of the theorem, it is shown in [5] that

$$\|\widehat{S}_{n_2}^{(2)} - \Sigma\| = O_P\left(\left(\frac{\log d}{n_2}\right)^{(1-q)/2}\right). \tag{2.6.28}$$

Assumption 2.3.1 provides a stochastic bound for $\max_{m\in\mathcal{M}} \mathrm{tr}(\widehat{\Theta}_m^{(1)})$. Combining it with the results of (2.6.23) and (2.6.28), the theorem follows. □

CHAPTER 3

ROBUST GAUSSIAN GRAPHICAL MODELS UNDER CORRUPTION

3.1 Introduction

Robustness is of great interest and importance both in practice and in theory. It is well-known that the performance of standard algorithms would break down even in the face of just a few corrupted points. In high-dimensional precision matrix estimation for Gaussian graphical models, outliers and corrupted data entries are frequently encountered and would lead to wrong inference of graph structures. For example, gene expression data typically have heavier tails and more outlying components than the Gaussian distribution. It is thus fundamental to develop robust sparse precision estimation methods to account for the contamination of data.

In recent years, some approaches have been proposed to deal with outliers and corrupted data for graphical models. Among them, Sun and Li [46] consider a robustified-likelihood function which is weighted according to how the observation is deviated. Then an efficient algorithm based on the coordinate gradient descent is developed to obtain the maximizer of the penalized robustified-likelihood. Finegold and Drton [17] use multivariate t-distributions for more robust inference of graphs. Yang and Lozano [54] introduce the trimmed graphical lasso, which seeks to minimize a weighted version of the negative log-likelihood regularized by the ℓ_1 penalty on the precision matrix for Gaussian graphical models and under some simple constraints on the weights.

In most of the previous work on robust methods for Gaussian graphical models, however, the identified outlying observation is entirely discarded from the training, which might be unnecessary and impractical. For example, in high-dimensional gene expression data, it is likely that most observations would have a few corrupted measurements on some genes. Instead of identifying each sample as an outlier or not, it is desirable to develop robust

methods that could deal with corrupted data entries.

In this chapter, we propose estimation procedures for Gaussian graphical models that are robust against possible outliers and corrupted data components. Two robust procedures in the regression cases, the trimmed inner product [9] and the nonparanormal skeptic [30], are discussed and analyzed. Then we consider the Meinshausen and Bühlmann neighborhood selection [36] and graphical lasso [19] algorithms in the uncorrupted setting and show that the robust counterparts obtained using the trimmed inner product or the nonparanormal skeptic give stronger performance guarantees under corruption.

The rest of this chapter is organized as follows. In the next section, we explore robust methods for sparse regression problems in high dimensions. In Section 3.3, we describe robust versions of the neighborhood selection and graphical lasso methods under corruption. We conclude in Section 3.4. Detailed proofs are collected in the last section.

3.2 Robust Sparse Regression

3.2.1 Problem Setup and Related Work

In this section, we consider the sparse regression problem with both corrupted covariates matrix and corrupted response. Suppose the response vector $y \in \mathbb{R}^N$ is generated using the following model:

$$y^* = X^* \beta^*, \tag{3.2.1}$$

$$X = X^* + B, \tag{3.2.2}$$

$$y = y^* + b + \epsilon, \tag{3.2.3}$$

where X^* is sampled from a Gaussian distribution $\mathcal{N}_p(0, \Sigma_x)$, B is a sparse matrix which represents the corruption on X^*, b is a sparse vector which represents the corruption on y^*, and additive noise ϵ is from a Gaussian distribution $\mathcal{N}(0, \sigma_e^2)$. In this setting, note that our

observations are corrupted covariates matrix X and corrupted response y. We also assume the unknown parameter $\beta^* \in \mathbb{R}^p$ is k-sparse ($k < p$), i.e., has only k nonzeros. Throughout this section we use x_i and X_i to denote the ith row and ith column of X, respectively.

Recently, the authors in [25, 29] consider a modified lasso that accounts for corruption in the response via an additional variable:

$$\min_{\beta,z} \ \|X\beta - y - z\|_2^2 + \lambda_1\|\beta\|_1 + \lambda_2\|z\|_1. \tag{3.2.4}$$

This optimization problem is convex and could lead to excellent recovery guarantees under a constant fraction of outliers. However, this approach faces serious difficulties when the covariates are also corrupted, and would fail in this setting. On the other hand, this approach can be modified to account for corruption in the covariates [59]:

$$\min_{\beta,E} \ \|(X - E)\beta - y\|_2^2 + \lambda_1\|\beta\|_1 + \lambda_2\|E\|_F. \tag{3.2.5}$$

Nevertheless, this results in a non-convex problem, where we are not aware of any tractable algorithm with provable performance guarantees. Bhatia et al. [4] propose a robust regression method based on hard thresholding. Their algorithm follows a most natural iterative strategy of, alternately, estimating an active set of points which have the least residual error on the current regressor, and then updating the regressor to provide a better fit on this active set. Once the algorithm converges, we could obtain an estimate of the true regressor β^*, and a set of points that are corruption free. However, this method would also fail when the covariates matrix is corrupted.

To handle corruption in both covariates matrix and response, Chen et al. [9] does not try to accurately identify outliers; instead, they replace standard calculation—the inner product—with robust counterparts less susceptible to corruption. We describe their method in detail in the next subsection.

As described in [9], the following two corruption models are considered in this section.

Definition 3.2.1. *(Regression: Row Corruption Model) Out of the N samples of (y, X), at most n_1 of them are arbitrarily corrupted.*

Definition 3.2.2. *(Regression: Distributed Corruption Model) Out of the N pairs of each column of X and y, at most n_1 of them are arbitrarily corrupted.*

We can see that in the row corruption model, each sample we observe is either due to the true generative model, or is corrupted in some arbitrary way, with the only restriction that at most n_1 such samples are corrupted. In the distributed corruption model, each column of X and y, have at most n_1 pairs that are arbitrarily corrupted. Note that in the second model, the corrupted entries need not lie in the same n_1 rows. Evidently this includes the first model as a special case.

Following the two corruption models, two types of robust procedures, trimmed inner product-based methods and nonparanormal skeptic-based methods, are studied in the next two subsections. For notational simplicity, let $n = N - n_1$ throughout the remaining section.

3.2.2 Trimmed Inner Product-Based Methods

Instead of performing any outlier detection procedure, Chen et al. [9] explore a simple idea: replace the essential calculation—the inner product—with a robust counterpart that cannot be greatly affected by a controlled number of arbitrarily corrupted points: the *trimmed inner product*. The trimmed inner product is presented in Algorithm 3.1.

Algorithm 3.1 Trimmed inner product $\langle a, b \rangle_{n_1}$

Input: $a \in \mathbb{R}^N, b \in \mathbb{R}^N, n_1$

Compute $q_i = a_i b_i, i = 1, \ldots, N$.

Sort $\{|q_i|\}$ and select the smallest $(N - n_1)$ ones.

Let Ω be the set of selected indices.

Output: $h = \sum_{i \in \Omega} q_i$.

Here we consider the robustified versions of the lasso and iterative hard thresholding (IHT). Note that the standard lasso and IHT estimates are functions of the inner products X^TX and X^Ty. In the robustified versions, these inner products are replaced by the trimmed inner products.

For the trimmed inner product-based methods [9], no assumption on the corrupted pairs is required—they might be unbounded and dependent on the authentic samples.

Algorithm 3.2 presents the trimmed inner product-based lasso algorithm. Notice that the optimization (3.2.7) is non-convex because $\widehat{\Gamma}$ might have negative eigenvalues. But we can still use the projected gradient descent method. The algorithm requires two parameters, R and n_1. Note that knowledge of the exact number of outliers is not needed—n_1 can be any upper bound of the number of outliers. In practice, cross-validation could be useful here to determine these parameters.

Algorithm 3.2 Trimmed inner product-based lasso

Input: X, y, R, n_1

Compute for all $i, j = 1, \ldots, p$,

$$\widehat{\Gamma}_{ij} = \langle X_i, X_j \rangle_{n_1}, \quad \widehat{\gamma}_i = \langle X_i, y \rangle_{n_1}. \tag{3.2.6}$$

Use projected gradient descent to solve and output:

$$\widehat{\beta} = \underset{\beta}{\operatorname{argmin}} \frac{1}{2}\beta^T\widehat{\Gamma}\beta - \widehat{\gamma}^T\beta \tag{3.2.7}$$

$$\text{s.t. } \|\beta\|_1 \leq R. \tag{3.2.8}$$

Algorithm 3.3 shows the trimmed inner product-based IHT algorithm. The projection operator $P_s(z)$ projects vector $z \in \mathbb{R}^p$ onto the set of s-sparsity vector by selecting the s largest elements (in magnitude) of z. The algorithm requires three parameters, step size η, s and n_1.

Algorithm 3.3 Trimmed inner product-based iterative hard thresholding

Input: X, y, s, η, n_1

Compute $\widehat{\Gamma}$ and $\widehat{\gamma}$ using (3.2.6).

Initialize $\beta^{(0)} = 0$, $t = 0$.

Repeat until convergence and output:

$$\beta^{(t+1)} = P_s(\beta^{(t)} - \eta \cdot (\widehat{\Gamma}\beta^{(t)} - \widehat{\gamma})). \tag{3.2.9}$$

For the trimmed inner product-based lasso method, Chen et. al [9] provide bound on the number of corrupted points it can tolerate, while still guaranteeing support recovery and small ℓ_2 error. In particular, if R is chosen as $\|\beta^*\|_1$ and suppose

$$n \gtrsim \frac{\sigma_{\max}^4}{\lambda_{\min}^2(\Sigma_x)} k \log p, \quad n_1 \lesssim \frac{\lambda_{\min}^2(\Sigma_x)}{\sigma_{\max}^2} \frac{n}{k \log p}, \tag{3.2.10}$$

where $\sigma_{\max}^2 = \max_i(\Sigma_x)_{ii}$, $\lambda_{\min}(\Sigma_x)$ denotes the smallest eigenvalue of Σ_x, and $\lesssim$ (resp. $\gtrsim$) means that there exists a constant, independent of k, p, n, such that the statement $<$ (resp. $>$) hold. Then the following ℓ_2 error bound (with high probability) is obtained:

$$\|\widehat{\beta} - \beta^*\|_2 \lesssim \frac{1}{\lambda_{\min}(\Sigma_x)} \left(\sigma_e \sigma_{\max} \sqrt{\frac{k \log p}{n}} + \frac{k n_1 \log p}{n} \sigma_{\max}^2 \|\beta^*\|_2 + \frac{\sqrt{k} n_1 \log p}{n} \sigma_{\max} \sqrt{\sigma_e^2 + \sigma_{\max}^2 \|\beta^*\|_2^2} \right). \tag{3.2.11}$$

3.2.3 Nonparanormal Skeptic-Based Methods

For the graphical modeling of random vector $X = (X_1, \ldots, X_d)$, the nonparanormal [31], a form of Gaussian copula, weakens the Gaussian assumption by imposing Gaussianity on the transformed random vector $f(X) = (f_1(X_1), \ldots, f_d(X_d))$, where each f_j is a monotonic and differentiable function. In particular, if $\Sigma^0 \in \mathbb{R}^{d \times d}$ is a positive-definite correlation matrix,

we say X has a nonparanormal distribution $X \sim \mathcal{NPN}_d(f, \Sigma^0)$ if $f(X) \sim \mathcal{N}_d(0, \Sigma^0)$. It is shown in [31] that the conditional independence graph is encoded by the sparsity pattern of $(\Sigma^0)^{-1}$. This allows arbitrary single variable marginal probability distributions in the model.

Assuming $X \sim \mathcal{NPN}_d(f, \Sigma^0)$, it is shown in [24] that $\Sigma^0_{ij} = 2\sin\left(\frac{\pi}{6}\rho_{ij}\right)$, where ρ_{ij} is the population version of Spearman's rho correlation between X_i and X_j. Motivated by this property, Liu et. al [30] propose a semiparametric method called the *nonparanormal skeptic* for estimating high-dimensional graphical models. Specifically, Liu et. al [30] estimate the correlation matrix by $\widetilde{S}^{\rho}_{ij} = 2\sin\left(\frac{\pi}{6}\widetilde{\rho}_{ij}\right)$, where $\widetilde{\rho}_{ij}$ is the sample version of Spearman's rho correlation between X_i and X_j. Then $\widetilde{S}^{\rho}_{ij}$ can be directly plugged into different parametric Gaussian graph estimators to obtain the final precision matrix and graph estimates.

In this subsection, we employ the idea of nonparanormal skeptic to introduce another robustified version of the inner product. Let $\widetilde{\rho}_{xy}$ be the sample version of Spearman's rho correlation vector of X and y, and $\widetilde{\rho}_x$ be the sample Spearman's rho correlation matrix of X. Algorithm 3.4 shows the nonparanormal skeptic-based lasso method. We first compute the nonparanormal skeptic estimator $\widetilde{S}^{\rho}$ of the correlation matrix underlying (y, X), and then we form a robustified version of the inner product of X_i and X_j by $[\widetilde{S}^{\rho}_x]_{ij}$ multiplied by the estimated standard deviations of X_i and X_j. Likewise for a robustified version of the inner product of X_i and y. This algorithm replies on the following assumption.

Assumption 3.2.1. *Each sample of (y, X) follows a nonparanormal distribution:*

$$(y_i, x_i) \overset{\text{i.i.d.}}{\sim} \mathcal{NPN}_{p+1}(f, \Sigma), \tag{3.2.12}$$

where

$$\Sigma = \begin{bmatrix} \beta^{*T}\Sigma_x\beta^* + \sigma_e^2 & \Sigma_x\beta^* \\ \beta^{*T}\Sigma_x & \Sigma_x \end{bmatrix}. \tag{3.2.13}$$

The following theorem provides bound on the number of corrupted points that the nonparanormal skeptic-based lasso can tolerate, while still guaranteeing small ℓ_2 error.

Assumption 3.2.2. *The eigenvalues of* Σ *are bounded from below and from above:* $\lambda_{\min}(\Sigma) \geq c > 0$ *and* $\lambda_{\max}(\Sigma) \leq C$.

Algorithm 3.4 Nonparanormal skeptic-based lasso

Input: X, y, R, n_1

Compute the nonparanormal skeptic estimator $\widetilde{S}^{\rho}$:

$$\widetilde{S}^{\rho} = \begin{bmatrix} 1 & \widetilde{S}_{xy}^{\rho^T} \\ \widetilde{S}_{xy}^{\rho} & \widetilde{S}_{x}^{\rho} \end{bmatrix}, \tag{3.2.14}$$

where

$$[\widetilde{S}_{xy}^{\rho}]_i = 2\sin\left(\frac{\pi}{6}[\widetilde{\rho}_{xy}]_i\right), \quad [\widetilde{S}_{x}^{\rho}]_{ij} = 2\sin\left(\frac{\pi}{6}[\widetilde{\rho}_{x}]_{ij}\right). \tag{3.2.15}$$

Compute $\widehat{\sigma}_y^2$ and $[\widehat{\sigma}_x^2]_i$ for $i = 1, \ldots, p$ using (3.2.6):

$$\widehat{\sigma}_y^2 = \frac{1}{n}\langle y, y\rangle_{n_1}, \quad [\widehat{\sigma}_x^2]_i = \frac{1}{n}\langle X_i, X_i\rangle_{n_1}. \tag{3.2.16}$$

Compute for all $i, j = 1, \ldots, p$,

$$\widetilde{\Gamma}_{ij} = [\widehat{\sigma}_x]_i \cdot [\widetilde{S}_{x}^{\rho}]_{ij} \cdot [\widehat{\sigma}_x]_j, \quad \widetilde{\gamma}_i = [\widehat{\sigma}_x]_i \cdot [\widetilde{S}_{xy}^{\rho}]_i \cdot \widehat{\sigma}_y. \tag{3.2.17}$$

Use projected gradient descent to solve and output:

$$\widetilde{\beta} = \underset{\beta}{\operatorname{argmin}} \frac{1}{2}\beta^T\widetilde{\Gamma}\beta - \widetilde{\gamma}^T\beta \tag{3.2.18}$$

$$\text{s.t. } \|\beta\|_1 \leq R. \tag{3.2.19}$$

Theorem 3.2.1. *Suppose the following is satisfied:*

$$n \gtrsim k \log p, \quad n_1 \lesssim \frac{n}{k \log p}, \tag{3.2.20}$$

Under Assumptions 3.2.1 and 3.2.2, if we choose the following parameter:

$$R = \|\beta^*\|_1, \tag{3.2.21}$$

then with probability tending to 1, the output of nonparanormal skeptic-based lasso algorithm satisfies the following ℓ_2 error bound:

$$\|\widetilde{\beta} - \beta^*\|_2 \lesssim \frac{1}{\lambda_{\min}(\Sigma_x)} \left((\sigma_e \sigma_{\max} + \sigma_{\max}^2 \|\beta^*\|_1) \sqrt{\frac{k \log p}{n}} + \frac{k n_1 \log p}{n} \sigma_{\max}^2 \|\beta^*\|_2 \right.$$
$$\left. + \frac{\sqrt{k} n_1 \log p}{n} \sigma_{\max} \sqrt{\sigma_e^2 + \sigma_{\max}^2 \|\beta^*\|_2^2} \right). \tag{3.2.22}$$

3.2.4 Analysis of Simulated Data

In this section, we conduct simulation studies to compare the performance of standard lasso, trimmed inner product-based lasso and IHT, nonparanormal skeptic-based lasso and IHT for sparse recovery.

We generate data with sample size $n = 400$ and number of variables $p = 1,000$ under the Gaussian design model with $\Sigma_x = I_p$, $k = 10$ and $\sigma_e = 1$. The non-zero elements of β^* is set to be random ± 1. The corrupted components of B and b are generated using a Gaussian distribution $\mathcal{N}(8, 0.5^2)$.

For standard and robust lasso methods, we choose the *oracle* tuning parameter $R = \|\beta^*\|_1$. For robust IHT methods, we set $s = 1.2k$, where s is the number of variables to keep in each step of IHT. For trimmed inner product-based methods, we set $n_1 = 2\alpha n$, where α is the fraction of outliers on each column of X and y.

Figure 3.1 shows the support recovery and ℓ_2 recovery errors for different methods with

independent columns of X, when the fraction of outliers α varies.

We next consider X with correlated columns. The data is generated using $[\Sigma_x]_{ii} = 1$ for all i, and $[\Sigma_x]_{ij} = 0.4$ for all $i \neq j$; other parameters are the same as before. Figure 3.2 shows the support recovery and ℓ_2 recovery errors for different methods with correlated columns of X, when the fraction of outliers α varies.

From the simulation results, we can see that all robust methods outperform standard lasso, especially when the fraction of outliers is not large. Also, nonparanormal skeptic-based methods always work better than trimmed inner product-based methods. In the scenario of correlated columns of X, we notice that nonparanormal skeptic-based lasso achieves the best performance among all methods.

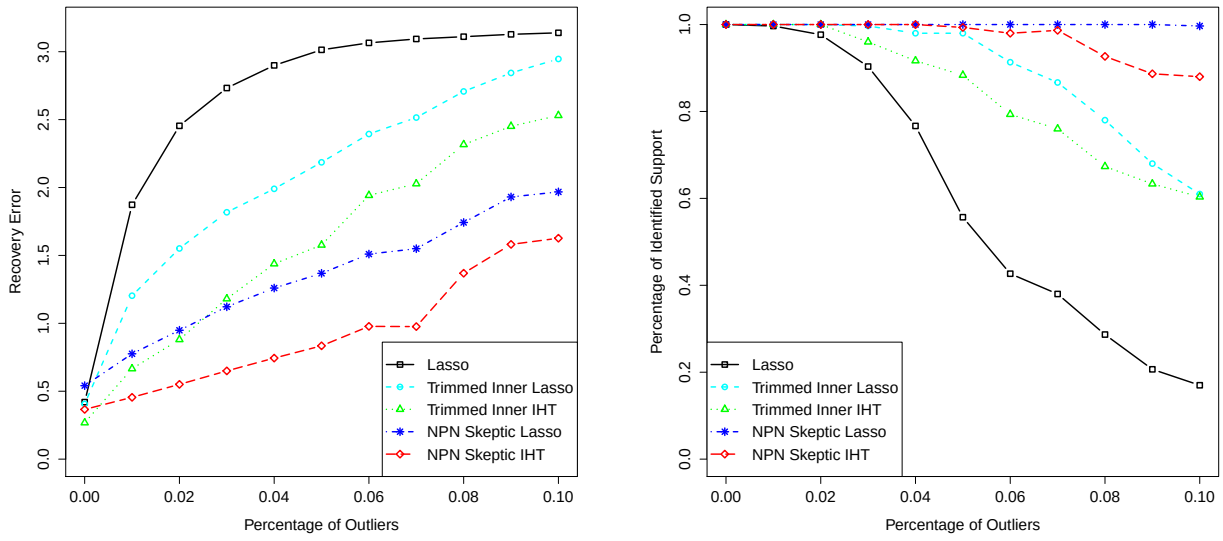


Figure 3.1: Simulation results on robust regression. Support recovery and ℓ_2 recovery errors for different methods with independent columns of X.

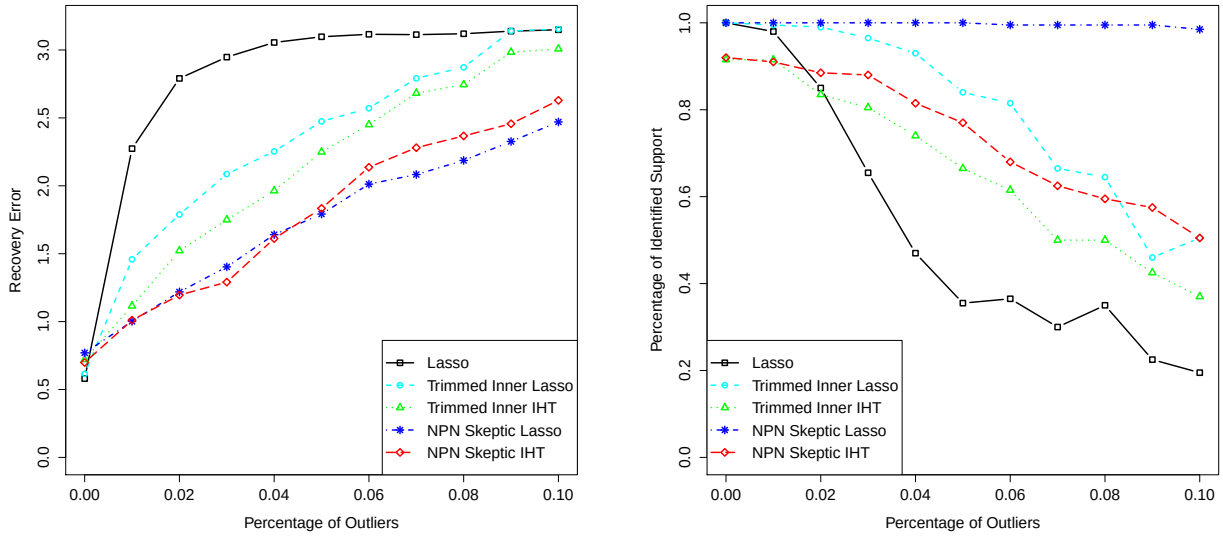


Figure 3.2: Simulation results on robust regression. Support recovery and ℓ_2 recovery errors for different methods with correlated columns of X.

3.3 Robust Gaussian Graphical Models

3.3.1 Problem Setup and Related Work

Suppose we are given a data matrix $Y \in \mathbb{R}^{N \times d}$, where N is the number of samples with d variables corresponding to a random vector $X = (X_1, \ldots, X_d)$. For example, Y_{ij} would represent the expression level of the jth gene on the ith microarray experiment.

Consider modeling the data matrix Y as a *matrix-variate Gaussian* distribution with observation corruptions:

$$Y = Z + B, \tag{3.3.1}$$

$$Z \sim \mathcal{MN}(\mathbf{0},\ I_N,\ \Sigma) \quad (\text{i.e. } \mathrm{Vec}(Z) \sim \mathcal{N}(\mathbf{0},\ \Sigma \otimes I_N)), \tag{3.3.2}$$

where $\Sigma \in \mathbb{R}^{d \times d}$ is the covariance matrix for X, I_N is an identity matrix, and $B \in \mathbb{R}^{N \times d}$ is a sparse corruption matrix. Here, $\mathrm{Vec}(A)$ represents the vectorization of matrix A, and

$A \otimes B$ represents the Kronecker product of matrices A and B.

Note that the above model allows two kinds of perturbations to observations Y: dense but bounded noise Z, as well as sparse but potentially unbounded corruptions B. Such corruption matrix can be chosen in an adversarial manner, i.e., both the support and entries of B are selected adversarially after Z has been generated.

Let $\Theta = \Sigma^{-1}$ be the precision matrix which encodes the sparsity patterns of graph $G = (V, E)$ corresponding to the random vector X. In high-dimensional settings where $p > N$, our goal is to estimate a sparse precision matrix and make inference of the underlying graph that is robust against possible outliers and corrupted data.

For the sparse precision matrix estimation problem in Gaussian graphical models, one natural way to handle possible outliers and corrupted data is to exclude the entire samples from the modeling. However, in many cases we would be left with too few observations so that the power of statistical analysis becomes questionable. Also, discarding samples with corrupted entries is a waste of available information.

In this section, let Y_i be the ith column of observations Y, and $Y_{\setminus i}$ be all columns of Y except the ith column. Likewise for B_i and $B_{\setminus i}$. Also, let y_i be the ith row of Y. Similar to the regression case, the following two corruption models are considered.

Definition 3.3.1. *(Graph: Row Corruption Model) Out of the N samples of Y, at most n_1 of them are arbitrarily corrupted.*

Definition 3.3.2. *(Graph: Distributed Corruption Model) Out of the N pairs of each Y_i and Y_j, at most n_1 of them are arbitrarily corrupted.*

In the next two subsections, we propose and analyze robust neighborhood selection and robust graphical lasso methods which extend the trimmed inner product and nonparanormal skeptic to the graphical modeling case.

For nonparanormal skeptic-based methods, we require the following assumption.

Assumption 3.3.1. *Each sample of Y follows a nonparanormal distribution:*

$$y_i \overset{\text{i.i.d.}}{\sim} \mathcal{NPN}_d(f, \Sigma_d), \tag{3.3.3}$$

Again, we let $n = N - n_1$ throughout the remaining section.

3.3.2 Robust Neighborhood Selection

According to the model (3.3.1)–(3.3.2), it is straightforward to show that

$$Y_i = (Y_{\setminus i} - B_{\setminus i})\beta_i^* + \epsilon_i + B_i, \tag{3.3.4}$$

where the true regressor $\beta_i^* = (\Sigma_{\setminus i,\setminus i})^{-1}\Sigma_{\setminus i,i}$, noise vector $\epsilon_i \sim \mathcal{N}(0, \sigma_i^2 I_N)$ with $\sigma_i^2 = \Sigma_{ii} - \Sigma_{i,\setminus i}(\Sigma_{\setminus i,\setminus i})^{-1}\Sigma_{\setminus i,i}$. By the block matrix inversion formula, we have

$$\Theta_{ii} = \sigma_i^{-2}, \tag{3.3.5}$$

$$\Theta_{\setminus i,i} = -\sigma_i^{-2}\beta_i^*. \tag{3.3.6}$$

Notice that the sparsity of each β_i^* encodes the sparsity of $\Theta_{\setminus i,i}$. Thus we can recover the sparsity of Θ in a column by column manner by regressing Y_i on $Y_{\setminus i}$ as in (3.3.4) for $i = 1, \ldots, d$. From the regression equation (3.3.4), we can see that both the covariates matrix and response are corrupted, in which case we can apply the robust methods introduced in the last section to account for the contamination of data.

Algorithm 3.5 shows the robust neighborhood selection method for Gaussian graphical models. Note that in each regression of Y_i on $Y_{\setminus i}$, we employ the algorithm of trimmed inner product-based lasso or nonparanormal skeptic-based lasso. Once we have the estimates $\widehat{\beta}_i$'s, let $[\widehat{\beta}_i]_j$ represent the estimated regression coefficient for variable X_j when regressing Y_i on $Y_{\setminus i}$, and likewise for $[\widehat{\beta}_j]_i$, then we set $(i,j) \in \widehat{E}^{\text{NS}}$ if and only if $[\widehat{\beta}_i]_j \neq 0$ AND $[\widehat{\beta}_j]_i \neq 0$. Alternatively, we can use the criterion that $(i,j) \in \widehat{E}^{\text{NS}}$ if and only if $[\widehat{\beta}_i]_j \neq 0$ OR $[\widehat{\beta}_j]_i \neq 0$.

Algorithm 3.5 Robust neighborhood selection

Input: Y, R, n_1

For $i = 1, \ldots, d$:

In each regression of Y_i on $Y_{\setminus i}$, apply Algorithms 3.2, 3.3 or 3.4 to obtain $\widehat{\beta}_i$.

Output $\widehat{E}^{\mathrm{NS}} = \{(i,j) : [\widehat{\beta}_i]_j \neq 0 \text{ AND } [\widehat{\beta}_j]_i \neq 0\}$.

Let m be the maximum degree in the graph, corresponding to the maximum number of nonzero off-diagonal elements in any row of Θ. Define the support set of Θ as:

$$H = \{(i,j) \in \{1, \ldots, d\} \times \{1, \ldots, d\} : \ \Theta_{ij} \neq 0\}. \tag{3.3.7}$$

We make the following assumption about the true covariance matrix Σ.

Assumption 3.3.2. *The eigenvalues of* Σ *are bounded from below and from above:* $\lambda_{\min}(\Sigma) \geq c > 0$ *and* $\lambda_{\max}(\Sigma) \leq C$.

We obtain the following theorem on graph recovery consistency.

Theorem 3.3.1. *Under Assumption 3.3.2, suppose the following is satisfied*

$$n \gtrsim m \log d, \quad n_1 \lesssim \sqrt{\frac{n}{m \log d}}. \tag{3.3.8}$$

Further, suppose for a sufficiently large constant K

$$\min_{(i,j) \in H} |\Theta_{ij}| > K \sqrt{\Theta_{ii} \vee \Theta_{jj}} \sqrt{\frac{m \log d}{n}}. \tag{3.3.9}$$

If the regularization parameter R_i *for the regression of* Y_i *on* $Y_{\setminus i}$ *is chosen as*

$$R_i = \left\| \frac{\Theta_{\setminus i,i}}{\Theta_{ii}} \right\|_1, \tag{3.3.10}$$

then the estimated edge set $\widehat{E}^{\mathrm{NS}}$ of Algorithm 3.5 using the trimmed inner product satisfies

$$\mathbb{P}(E(\Theta) \subset \widehat{E}^{\mathrm{NS}}) \to 1. \tag{3.3.11}$$

It is possible that each $\widehat{\beta}_i$ in Algorithm 3.5 have some very small nonzero values. To achieve exact recovery, we can hard threshold $\widehat{\beta}_i$ to get a sparser estimate:

$$[\widehat{\beta}_i^M]_j = \begin{cases} [\widehat{\beta}_i]_j & \text{if } |[\widehat{\beta}_i]_j| > M\sqrt{\frac{m \log d}{n}} \\ 0 & \text{otherwise} \end{cases}. \tag{3.3.12}$$

It can be shown that there exists a constant M such that the above hard thresholding estimator can achieve exact recovery.

3.3.3 Robust Graphical Lasso

Let $\widehat{S}$ be the estimated covariance matrix for random variables $X_1, \ldots, X_d$, where

$$\widehat{S}_{ij} = \frac{1}{n}\langle X_i, X_j \rangle_{n_1}. \tag{3.3.13}$$

Consider the following ℓ_1-regularized estimator

$$\widehat{\Theta}_{\lambda_n} = \underset{\Theta}{\operatorname{argmin}} \left\{ \operatorname{tr}(\widehat{S}\Theta) - \log\det(\Theta) + \lambda_n \sum_{i \neq j} |\Theta_{ij}| \right\}, \tag{3.3.14}$$

where λ_n is a regularization parameter.

Alternatively, we can use the nonparanormal skeptic to estimate a robustified version of covariance matrix, which is essentially equivalent to the method proposed in [30].

One thing to note is that $\widehat{S}$ may not be positive semidefinite. Certain algorithms to solve (3.3.14) (like the blockwise-coordinate descent) may fail. However, other algorithms such as projected gradient descent do not have such positive semi-definiteness assumptions.

Algorithm 3.6 Trimmed inner product-based graphical lasso

Input: Y, λ_n, n_1

For all $i, j = 1, \ldots, d$, compute $\widehat{S}_{ij}$ using the trimmed inner product (3.3.13).

Use projected gradient descent solve (3.3.14).

Output $\widehat{\Theta}_{\lambda_n}$ and $\widehat{E}^{\text{Glasso}} = \{(i,j) : [\widehat{\Theta}_{\lambda_n}]_{ij} \neq 0\}$.

Lemma 3.3.1. *Suppose Assumption 3.3.2 holds. With probability tending to 1, we have*

$$\max_{ij} |\widehat{S}_{ij} - \Sigma_{ij}| = O_P\left(\sqrt{\frac{\log d}{n}} + \frac{n_1 \log d}{n}\right). \tag{3.3.15}$$

The following theorem yields estimation consistency result of trimmed inner product-based graphical lasso in the Frobenius norm.

Theorem 3.3.2. *Suppose Assumption 3.3.2 holds. If the regularization parameter λ_n is chosen as*

$$\lambda_n \asymp \sqrt{\frac{\log d}{n}} + \frac{n_1 \log d}{n}, \tag{3.3.16}$$

then the estimator $\widehat{\Theta}_{\lambda_n}$ of Algorithm 3.6 satisfies

$$\|\widehat{\Theta}_{\lambda_n} - \Theta\|_F = O_P\left(\sqrt{\frac{(s+d)\log d}{n}} + \frac{n_1\sqrt{s+d}\log d}{n}\right), \tag{3.3.17}$$

where s is the number of nonzero off-diagonal elements in Θ.

To prove the support recovery consistency result, we need further assumptions. Following [41], let $\Omega = \Sigma \otimes \Sigma$ and H be the support set of Θ. Let H^c be the complement of H in the set $\{1, \ldots, d\} \times \{1, \ldots, d\}$. For any two subsets U and V of $\{1, \ldots, d\} \times \{1, \ldots, d\}$, Ω_{UV} denotes the sub-matrix with rows and columns of Ω indexed by U and V, respectively.

Assumption 3.3.3. *There exists some $\tau \in (0, 1]$, such that $\|\Omega_{H^cH}(\Omega_{HH})^{-1}\|_\infty \leq 1 - \tau$.*

This assumption requires that the pairs of variables which are non-edges cannot have an overtly strong effect on the pairs of variables which form edges of the underlying graph.

Assumption 3.3.4. *The quantities* $\|\Sigma\|_\infty$ *and* $\|(\Omega_{HH})^{-1}\|_\infty$ *are bounded from above.*

This condition requires that the entries along any row of Σ and $(\Omega_{HH})^{-1}$ have bounded ℓ_1 norms.

Let m be the maximum degree in the underlying graph. The following theorem provides the sparsistency result for trimmed inner product-based graphical lasso.

Theorem 3.3.3. *Under Assumptions 3.3.2–3.3.4, suppose the following is satisfied:*

$$n \gtrsim m^2 \log d, \quad n_1 \lesssim \sqrt{\frac{n}{\log d}}. \tag{3.3.18}$$

Further, suppose for a sufficiently large constant K

$$\min_{(i,j)\in H} |\Theta_{ij}| > K\sqrt{\frac{\log d}{n}}. \tag{3.3.19}$$

If the regularization parameter λ_n *is chosen as*

$$\lambda_n \asymp \sqrt{\frac{\log d}{n}}, \tag{3.3.20}$$

then the estimated edge set $\widehat{E}^{\text{Glasso}}$ *of Algorithm 3.6 satisfies*

$$\mathbb{P}(\widehat{E}^{\text{Glasso}} = E(\Theta)) \to 1. \tag{3.3.21}$$

3.3.4 Analysis of Simulated Data

Here we conduct simulation studies to evaluate the performance of different approaches in graph estimation under corruption. We generate data with sample size $n = 200$ and number of nodes $d = 400$. Consider the random graph structure as follows: given the adjacency

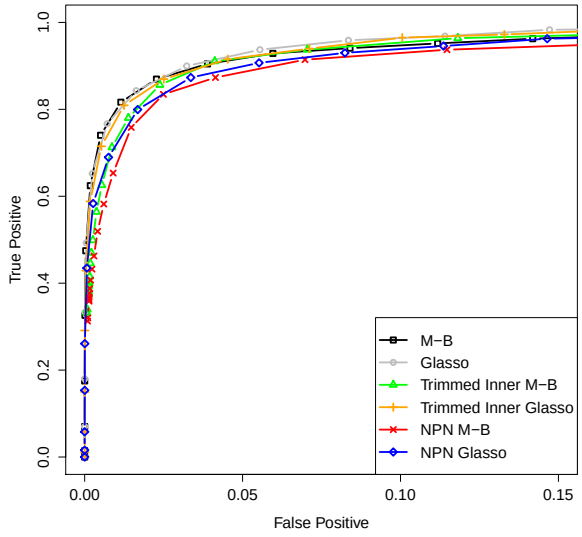
matrix Q, each pair of off-diagonal elements are randomly set $Q_{ij} = Q_{ji} = 1$ for $i \neq j$ with probability 0.01, and 0 otherwise, which results in about $0.01 \cdot d(d-1)/2$ edges in the graph.

In the simulation, $(\alpha \cdot nd)$-sized observed components were set to be corrupted and each corrupted component was randomly set to be $-U + \delta$ or $U + \delta$ multiplied by the marginal standard deviance of the corresponding variable, where $\delta \sim \mathcal{N}(0, 0.5^2)$, α controls the fraction of corrupted components and U controls the magnitude of corruption.

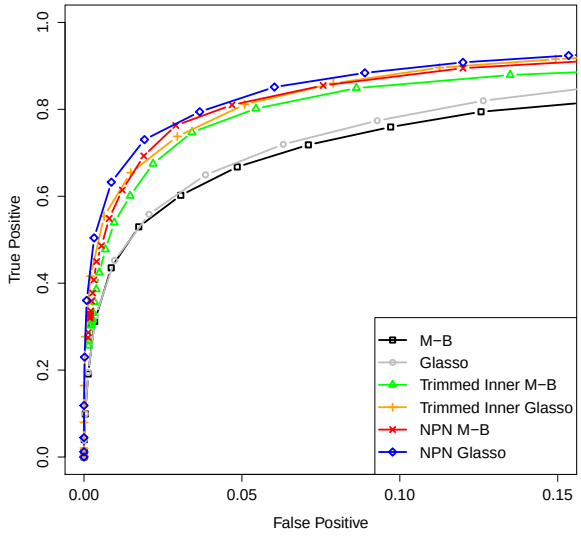
Consider the following methods in the comparison:

- M-B, Glasso: vanilla versions of Meinshausen and Bühlmann neighborhood selection and graphical lasso methods;
- Trimmed Inner M-B, Trimmed Inner Glasso: robustified versions of M-B and Glasso, where each inner product of X_i and X_j is obtained using the trimmed inner product; we also set $n_1 = 2\alpha n$;
- NPN M-B, NPN Glasso: robustified versions of M-B and Glasso, where each inner product of X_i and X_j is obtained using the nonparanormal skeptic.

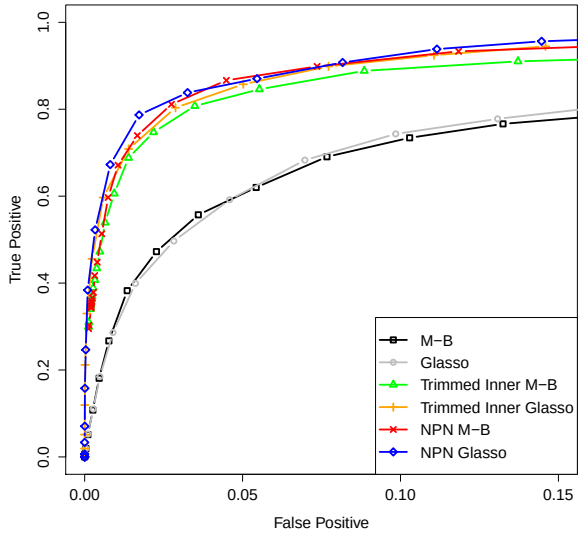
Figure 3.3 shows the ROC curves of different methods, when α and U vary. From the results, we can see that in the scenario of no corruption (i.e. $\alpha = 0$), the performance of robust methods is as good as that of standard M-B and Glasso methods; in the presence of outliers, robust methods are superior to standard M-B and Glasso in term of support recovery. In particular, trimmed inner product-based methods perform as well as that of nonparanormal skeptic-based methods, although the latter ones are still slightly better.



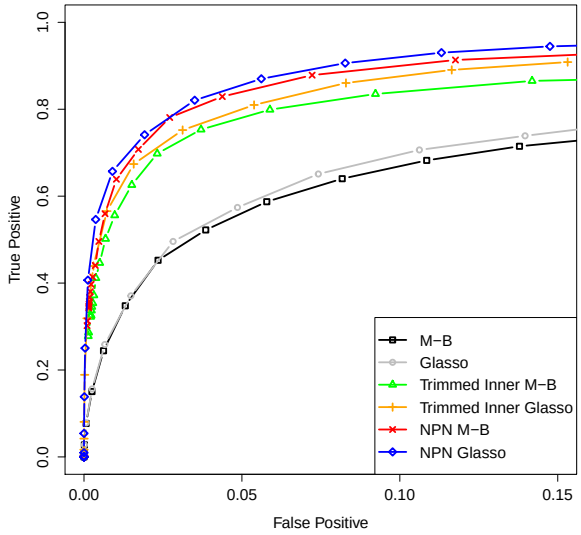
(a) $\alpha = 0$



(b) $\alpha = 0.01$, $U = 6$



(c) $\alpha = 0.01$, $U = 8$



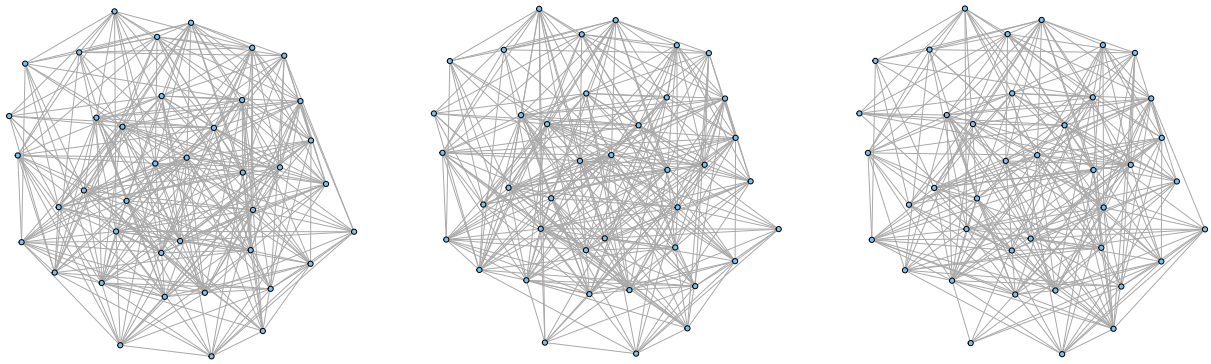
(d) $\alpha = 0.02$, $U = 6$

Figure 3.3: ROC curves of different methods on simulated data for random graphs, when α and U vary.

3.3.5 Analysis of Stock Price and Microarray Data

We first test the robust methods on stock price data from Yahoo! Finance. We collected the daily closing prices for $d = 432$ stocks in the S&P 500 index from January 5, 2009 to December 31, 2009. Let $S_k^{(t)}$ be the closing price for stock k on day t, and $X_k^{(t)} = \log(S_k^{(t)}/S_k^{(t-1)})$ be the log return. We treat $X^{(t)}$'s as independent random vectors, though they form a time series. We study the associations among stocks using standard and robust graphical lasso methods with tuning parameters determined by BIC. The standard graphical lasso selects 7,446 edges, trimmed inner product-based graphical lasso selects 7,069 edges, and nonparanormal skeptic-based graphical lasso selects 6,856 edges.

As a second example, we study a dataset on Affymetrix GeneChip microarrays for the plant *Arabidopsis thaliana* [53]. The gene expression levels of $n = 118$ samples are measured on $d = 40$ genes from the isoprenoid pathway. To investigate the associations among genes, the standard graphical lasso selects 326 edges, trimmed inner product-based graphical lasso selects 310 edges, and nonparanormal skeptic-based graphical lasso selects 275 edges. Among the selected edges, 218 edges are identified by all the three methods. Figure 3.4 provides the estimated graphs of these methods. The standard graphical lasso tends to estimate more links than the robust methods; the lack of robustness might lead to inaccurate graph estimations and falsely identified edges.



(a) Glasso (b) Trimmed Inner Glasso (c) NPN Glasso

Figure 3.4: Estimated graphs for microarray data example for the plant *Arabidopsis thaliana*; Left: Glasso; Middle: Trimmed Inner Glasso; Right: NPN Glasso.

3.4 Conclusion

In this chapter, we study the estimation methods for Gaussian graphical models that are robust against corrupted data. Two procedures to deal with possible outliers, the trimmed inner product and the nonparanormal skeptic, are discussed and analyzed. We consider the neighborhood selection and graphical lasso algorithms in the uncorrupted setting and show that the robust counterparts obtained using the trimmed inner product or the nonparanormal skeptic give stronger performance guarantees under corruption.

3.5 Proofs

Proof of Theorem 3.2.1.

Proof. This proof follows closely to the proof of Theorem 4 of [9]. For simplicity, we only prove the theorem for the row corruption model. Let $y = [y^{\mathcal{A}}; y^{\mathcal{O}}]$ and $X = [X^{\mathcal{A}}; X^{\mathcal{O}}]$, where $(y^{\mathcal{A}}, X^{\mathcal{A}})$ are the authentic rows and $(y^{\mathcal{O}}, X^{\mathcal{O}})$ are the corrupted rows. Note that $y^{\mathcal{A}} = X^{\mathcal{A}}\beta^* + \epsilon$.

Let $\Delta := \widetilde{\beta} - \beta^*$ and $S = \text{support}(\beta^*)$. For simplicity of notation, we write $\widehat{\sigma}_i$, $\widetilde{S}^{\rho}_{ij}$ and $\widetilde{S}^{\rho}_{iy}$ instead of $[\widehat{\sigma}_x]_i$, $[\widetilde{S}^{\rho}_x]_{ij}$ and $[\widetilde{S}^{\rho}_{xy}]_i$, respectively.

Since $\widetilde{\beta}$ satisfies the constraint, we have

$$\|\beta^*\|_1 \geq \|\beta^* + \Delta\|_1 = \|\beta^* + \Delta_S\|_1 + \|\Delta_{S^c}\|_1 \geq \|\beta^*\|_1 - \|\Delta_S\|_1 + \|\Delta_{S^c}\|_1 \tag{3.5.1}$$

$$\implies \|\Delta_{S^c}\|_1 \leq \|\Delta_S\|_1. \tag{3.5.2}$$

Then we have

$$\|\Delta\|_1 = \|\Delta_S\|_1 + \|\Delta_{S^c}\|_1 \leq 2\|\Delta_S\|_1 \leq 2\sqrt{k}\|\Delta_S\|_2 \tag{3.5.3}$$

$$\implies \|\Delta\|_1 \leq 2\sqrt{k}\|\Delta\|_2. \tag{3.5.4}$$

By optimality of $\widetilde{\beta}$, we have

$$\frac{1}{2}\widetilde{\beta}^T\widetilde{\Gamma}\widetilde{\beta} - \widetilde{\gamma}^T\widetilde{\beta} \leq \frac{1}{2}\beta^{*T}\widetilde{\Gamma}\beta^* - \widetilde{\gamma}^T\beta^*, \tag{3.5.5}$$

which implies

$$\frac{1}{2}\Delta^T\widetilde{\Gamma}\Delta \leq \langle\widetilde{\gamma} - \widetilde{\Gamma}\beta^*, \Delta\rangle. \tag{3.5.6}$$

Let $F = \widetilde{\Gamma} - \frac{1}{n}\langle X^{\mathcal{A}}, X^{\mathcal{A}}\rangle$ and $\|F\|_{\max} = \max_{ij}|F_{ij}|$. Since $X^{\mathcal{A}}$ satisfies the restricted eigenvalue condition under the assumption of the theorem, for any u such that $\|u\|_1 \leq 2\sqrt{k}\|u\|_2$, we have

$$\frac{1}{n}u^T\langle X_i^{\mathcal{A}}, X_j^{\mathcal{A}}\rangle u \geq \frac{1}{4}\lambda_{\min}(\Sigma_x)\|u\|_2^2. \tag{3.5.7}$$

Then we obtain

$$\Delta^T\widetilde{\Gamma}\Delta = \frac{1}{n}\Delta^T\langle X_i^{\mathcal{A}}, X_j^{\mathcal{A}}\rangle\Delta + \Delta^T F\Delta \tag{3.5.8}$$

$$\geq \frac{1}{4}\lambda_{\min}(\Sigma_x)\|\Delta\|_2^2 - \|F\|_{\max}\cdot\sum_{ij}|\Delta_i||\Delta_j| \tag{3.5.9}$$

$$= \frac{1}{4}\lambda_{\min}(\Sigma_x)\|\Delta\|_2^2 - \|F\|_{\max}\cdot\|\Delta\|_1^2 \tag{3.5.10}$$

$$\geq \frac{1}{4}\lambda_{\min}(\Sigma_x)\|\Delta\|_2^2 - 4k\|F\|_{\max}\cdot\|\Delta\|_2^2. \tag{3.5.11}$$

In order to bound $\|F\|_{\max}$, we can write

$$F_{ij} = (\widetilde{\sigma}_i\widetilde{S}_{ij}^{\rho}\widetilde{\sigma}_j - [\Sigma_x]_{ij}) + ([\Sigma_x]_{ij} - \frac{1}{n}\langle X_i^{\mathcal{A}}, X_j^{\mathcal{A}}\rangle) \tag{3.5.12}$$

$$= (\widetilde{\sigma}_i\widetilde{S}_{ij}^{\rho}\widetilde{\sigma}_j - \sigma_i[\Sigma_x^0]_{ij}\sigma_j) + ([\Sigma_x]_{ij} - \frac{1}{n}\langle X_i^{\mathcal{A}}, X_j^{\mathcal{A}}\rangle) \tag{3.5.13}$$

$$= \sigma_i\sigma_j(\widetilde{S}_{ij}^{\rho} - [\Sigma_x^0]_{ij}) + \widetilde{S}_{ij}^{\rho}(\widetilde{\sigma}_i\widetilde{\sigma}_j - \sigma_i\sigma_j) + ([\Sigma_x]_{ij} - \frac{1}{n}\langle X_i^{\mathcal{A}}, X_j^{\mathcal{A}}\rangle). \tag{3.5.14}$$

For the first term, it is shown in [31] that with high probability, we have

$$\max_{ij}|\widetilde{S}_{ij}^{\rho} - [\Sigma_x^0]_{ij}| \lesssim \sqrt{\frac{\log p}{n}}. \tag{3.5.15}$$

For the second term, it is shown in [9] that

$$|\widetilde{\sigma}_i^2 - \frac{1}{n}\langle X_i^{\mathcal{A}}, X_i^{\mathcal{A}}\rangle| \leq \frac{2n_1}{n}\left(\max_{k\in\mathcal{A}}|X_{ki}|\right)^2 \lesssim 2n_1\frac{\log p}{n}\sigma_i^2. \tag{3.5.16}$$

For the third term, under the assumption that the largest eigenvalue of Σ_x is bounded, it is shown in [6] that with high probability, we have

$$\max_{ij}|[\Sigma_x]_{ij} - \frac{1}{n}\langle X_i^{\mathcal{A}}, X_j^{\mathcal{A}}\rangle| \lesssim \sqrt{\frac{\log p}{n}}. \tag{3.5.17}$$

Under the assumption that

$$n \gtrsim k \log p, \quad n_1 \lesssim \frac{n}{k \log p}, \tag{3.5.18}$$

we have

$$\|F\|_{\max} \lesssim \frac{\lambda_{\min}(\Sigma_x)}{32k}, \tag{3.5.19}$$

and thus

$$\Delta^T \widetilde{\Gamma} \Delta \geq \frac{\lambda_{\min}(\Sigma_x)}{4} \|\Delta\|_2^2. \tag{3.5.20}$$

According to Holder's inequality, we have

$$\langle \widetilde{\gamma} - \widetilde{\Gamma}\beta^*, \Delta \rangle \leq \|\widetilde{\gamma} - \widetilde{\Gamma}\beta^*\|_\infty \cdot \|\Delta\|_1 \tag{3.5.21}$$

$$\leq \|\widetilde{\gamma} - \widetilde{\Gamma}\beta^*\|_\infty \cdot 2\sqrt{k}\|\Delta\|_2. \tag{3.5.22}$$

Note that

$$\|\widetilde{\gamma} - \widetilde{\Gamma}\beta^*\|_\infty \leq \|F\beta^*\|_\infty + \|\widetilde{\gamma} - \frac{1}{n}\langle X^{\mathcal{A}}, X^{\mathcal{A}}\rangle \beta^*\|_\infty. \tag{3.5.23}$$

For the first term, we have

$$\|F\beta^*\|_\infty \lesssim \sigma_{\max}^2 \sqrt{k} n_1 \frac{\log p}{n} \|\beta^*\|_2 + \sigma_{\max}^2 \sqrt{\frac{\log p}{n}} \|\beta^*\|_1. \tag{3.5.24}$$

For the second term,

$$\widetilde{\gamma}_i - \frac{1}{n}\langle X_i^{\mathcal{A}}, X^{\mathcal{A}}\beta^* \rangle = \frac{1}{n}\langle X_i^{\mathcal{A}}, \epsilon \rangle + \widetilde{\gamma}_i - \frac{1}{n}\langle X_i^{\mathcal{A}}, y^{\mathcal{A}} \rangle \tag{3.5.25}$$

$$= \frac{1}{n}\langle X_i^{\mathcal{A}}, \epsilon \rangle + (\widetilde{\sigma}_i \widetilde{S}_{iy}^{\rho} \widetilde{\sigma}_y - \sigma_i \Sigma_{iy}^0 \sigma_y) + (\Sigma_{iy} - \frac{1}{n}\langle X_i^{\mathcal{A}}, y^{\mathcal{A}} \rangle). \tag{3.5.26}$$

We have

$$\frac{1}{n}\langle X_i^{\mathcal{A}}, \epsilon\rangle \lesssim \sigma_e\sigma_{\max}\sqrt{\frac{\log p}{n}}, \quad |\Sigma_{iy} - \frac{1}{n}\langle X_i^{\mathcal{A}}, y^{\mathcal{A}}\rangle| \lesssim \sqrt{\frac{\log p}{n}}. \tag{3.5.27}$$

Note that

$$|\widetilde{\sigma}_y^2 - \frac{1}{n}\langle y^{\mathcal{A}}, y^{\mathcal{A}}\rangle| \lesssim n_1\frac{\log p}{n}(\sigma_e^2 + \sigma_{\max}^2\|\beta^*\|_2^2). \tag{3.5.28}$$

Thus we have

$$\begin{aligned}\|\widetilde{\gamma} - \frac{1}{n}\langle X^{\mathcal{A}}, X^{\mathcal{A}}\rangle\beta^*\|_\infty \lesssim \sigma_e\sigma_{\max}\sqrt{\frac{\log p}{n}} + \sigma_{\max}\sqrt{\sigma_e^2 + \sigma_{\max}^2\|\beta^*\|_2^2}\sqrt{\frac{\log p}{n}} \\ + n_1\frac{\log p}{n}\sigma_{\max}\sqrt{\sigma_e^2 + \sigma_{\max}^2\|\beta^*\|_2^2}.\end{aligned} \tag{3.5.29}$$

Combining pieces, we obtain

$$\begin{aligned}\langle\widetilde{\gamma} - \widetilde{\Gamma}\beta^*, \Delta\rangle \lesssim \sqrt{k}\|\Delta\|_2\Bigg(\sigma_{\max}^2\sqrt{k}n_1\frac{\log p}{n}\|\beta^*\|_2 + \sigma_{\max}^2\sqrt{\frac{\log p}{n}}\|\beta^*\|_1 + \sigma_e\sigma_{\max}\sqrt{\frac{\log p}{n}} \\ +\sigma_{\max}\sqrt{\sigma_e^2 + \sigma_{\max}^2\|\beta^*\|_2^2}\sqrt{\frac{\log p}{n}} + n_1\frac{\log p}{n}\sigma_{\max}\sqrt{\sigma_e^2 + \sigma_{\max}^2\|\beta^*\|_2^2}\Bigg).\end{aligned} \tag{3.5.30}$$

According to (3.5.6), we have

$$\begin{aligned}\|\Delta\|_2 \lesssim \frac{1}{\lambda_{\min}(\Sigma_x)}\Bigg((\sigma_e\sigma_{\max} + \sigma_{\max}^2\|\beta^*\|_1)\sqrt{\frac{k\log p}{n}} + \frac{kn_1\log p}{n}\sigma_{\max}^2\|\beta^*\|_2 \\ +\frac{\sqrt{k}n_1\log p}{n}\sigma_{\max}\sqrt{\sigma_e^2 + \sigma_{\max}^2\|\beta^*\|_2^2}\Bigg),\end{aligned} \tag{3.5.31}$$

which concludes the proof of the theorem. □

Proof of Theorem 3.3.1.

Proof. Note that $\beta_i^* = -\Theta_{\backslash i,i}/\Theta_{ii}$ and $\sigma_i^2 = 1/\Theta_{ii}$. Following the results from Theorem 4 in [9], in the regression of Y_i on $Y_{\backslash i}$, we obtain for any $j \neq i$ such that $(i,j) \in H$:

$$|[\beta_i^*]_j| - |[\widehat{\beta}_i]_j| \leq |[\widehat{\beta}_i]_j - [\beta_i^*]_j| \leq \|\widehat{\beta}_i - \beta_i^*\|_2. \tag{3.5.32}$$

Under the assumption of the theorem, we have

$$\|\widehat{\beta}_i - \beta_i^*\|_2 = O_P\left(\sigma_i \sqrt{\frac{m \log p}{n}}\right), \tag{3.5.33}$$

and

$$|[\widehat{\beta}_i]_j| \geq |[\beta_i^*]_j| - K\sigma_i \sqrt{\frac{m \log p}{n}} = \frac{1}{\Theta_{ii}}\left(|\Theta_{ij}| - K\sqrt{\Theta_{ii}}\sqrt{\frac{m \log p}{n}}\right) > 0 \tag{3.5.34}$$

$$\implies [\widehat{\beta}_i]_j > 0. \tag{3.5.35}$$

Similarly, we can also obtain $[\widehat{\beta}_j]_i > 0$. Thus we have $(i,j) \in \widehat{E}^{\mathrm{NS}}$ which concludes the proof of the theorem. □

Proof of Lemma 3.3.1.

Proof. Note that

$$\max_{ij} |\widehat{S}_{ij} - \Sigma_{ij}| \leq \max_{ij} \left|\widehat{S}_{ij} - \frac{1}{n}\langle X_i^{\mathcal{A}}, X_j^{\mathcal{A}}\rangle\right| + \max_{ij} \left|\frac{1}{n}\langle X_i^{\mathcal{A}}, X_j^{\mathcal{A}}\rangle - \Sigma_{ij}\right|. \tag{3.5.36}$$

For the first term, it is shown in [9] that with high probability

$$\max_{ij} \left|\widehat{S}_{ij} - \frac{1}{n}\langle X_i^{\mathcal{A}}, X_j^{\mathcal{A}}\rangle\right| = O_P\left(\frac{n_1 \log d}{n}\right). \tag{3.5.37}$$

For the second term, it is shown in [6] that under the assumption of the lemma, with high

probability

$$\max_{ij} \left| \frac{1}{n} \langle X_i^{\mathcal{A}}, X_j^{\mathcal{A}} \rangle - \Sigma_{ij} \right| = O_P \left(\sqrt{\frac{\log d}{n}} \right). \tag{3.5.38}$$

The lemma then follows. □

Proof of Theorem 3.3.2.

Proof. The proof follows the same arguments as the proof of Theorem 1 in [44], replacing their Lemma 1 with our Lemma 3.3.1. □

Proof of Theorem 3.3.3.

Proof. The proof uses Lemma 3.3.1 and follows the same arguments as the proof of Theorem 1 in [41]. □

CHAPTER 4

TREE-BASED GRAPHICAL MODELS WITH STRUCTURAL CONSTRAINTS

4.1 Introduction

Both Gaussian graphical models and the nonparanormal maintain tractable inference, but they are limited in their ability to flexibly model the bivariate and higher order marginals. Tree-based graphical models permit arbitrary bivariate marginals and maintain tractability by restricting to acyclic graphs. An nonparametric approach based on forests is developed in [33] for density estimation in high-dimensional settings.

Let $X = (X_1, \ldots, X_d)$ be a d-dimensional continuous random vector with density function $p^*(x) > 0$. We say an undirected graph is a forest if it is acyclic. Let $F = (V_F, E_F)$ be a forest with vertices $V_F = \{1, \ldots, d\}$ and edge set $E_F \in V_F \times V_F$. We say that X, or equivalently, its density p^*, is Markov to F if X_i and X_j are conditionally independent given the other random variables whenever edge (i, j) is missing in E_F. Any density $p > 0$ that is Markov to F has the following factorization

$$p(x) = \prod_{(i,j)\in E_F} \frac{p_{ij}(x_i, x_j)}{p_i(x_i)p_j(x_j)} \prod_{k\in V_F} p_k(x_k), \tag{4.1.1}$$

where each $p_{ij}(x_i, x_j)$ is a bivariate density, and each $p_k(x_k)$ is a univariate density. With this factorization, we can write the expected log likelihood as

$$\mathbb{E} \log p(X) = \int p(x) \left(\sum_{(i,j)\in E_F} \log \frac{p_{ij}(x_i, x_j)}{p_i(x_i)p_j(x_j)} + \sum_{k\in V_F} \log p_k(x_k) \right) dx \tag{4.1.2}$$

$$= \sum_{(i,j)\in E_F} I(X_i; X_j) - \sum_{k\in V_F} H(X_k), \tag{4.1.3}$$

where

$$I(X_i; X_j) = \int p_{ij}(x_i, x_j) \log \frac{p_{ij}(x_i, x_j)}{p_i(x_i) p_j(x_j)} dx_i dx_j, \tag{4.1.4}$$

$$H(X_k) = -\int p_k(x_k) \log p_k(x_k) dx_k, \tag{4.1.5}$$

are the mutual information between X_i and X_j, and the entropy of X_k, respectively. We maximize the right hand side of (4.1.3) to find the optimal F^*

$$F^* = \arg\max_{F \in \mathcal{F}_d} \sum_{(i,j) \in E_F} I(X_i; X_j), \tag{4.1.6}$$

where $\mathcal{F}_d$ is the collection of spanning trees on vertices $\{1, \ldots, d\}$. We let $\mathcal{F}_d$ contain only spanning trees because there is always a spanning tree that solves the problem (4.1.3). This problem can be recast as the problem of finding the maximum spanning tree for a weighted graph, where the weight w_{ij} of the edge between node i and j is $I(X_i; X_j)$. Kruskal's algorithm [23] is a greedy algorithm that is guaranteed to find an optimal solution, while Chow and Liu [10] propose the procedure in the setting of discrete distribution estimation. The method is described in Algorithm 4.1.

Algorithm 4.1 Kruskal's (Chow-Liu) algorithm

Input Weight matrix $W = (w_{ij})_{d \times d}$

Initialize $E^{(0)} \leftarrow \varnothing$

for $k = 1, \ldots, d-1$ **do**

$(i^{(k)}, j^{(k)}) \leftarrow \arg\max_{(i,j)} w_{ij}$ such that $E^{(k-1)} \cup \{(i^{(k)}, j^{(k)})\}$ doesn't contain a cycle

$E^{(k)} \leftarrow E^{(k-1)} \cup \{(i^{(k)}, j^{(k)})\}$

end for

Output edge set $E^{(d-1)}$.

However, this procedure is not practical since the true density p^* is unknown. Suppose instead that we have $X^{(1)}, \ldots, X^{(n)}$, which are n i.i.d. copies of the random vector X. We

replace the population mutual information by their corresponding estimates, defined by

$$\widehat{I}(X_i; X_j) = \int \widehat{p}_{ij}(x_i, x_j) \log \frac{\widehat{p}_{ij}(x_i, x_j)}{\widehat{p}_i(x_i)\widehat{p}_j(x_j)} dx_i dx_j, \tag{4.1.7}$$

where $\widehat{p}_{ij}(x_i, x_j)$ and $\widehat{p}_k(x_k)$ are the kernel estimators of the bivariate and univariate marginal densities:

$$\widehat{p}_{ij}(x_i, x_j) = \frac{1}{n} \sum_{t=1}^{n} \frac{1}{h_2^2} K\left(\frac{X_i^{(t)} - x_i}{h_2}\right) K\left(\frac{X_j^{(t)} - x_j}{h_2}\right), \tag{4.1.8}$$

$$\widehat{p}_k(x_k) = \frac{1}{n} \sum_{t=1}^{n} \frac{1}{h_1} K\left(\frac{X_k^{(t)} - x_k}{h_1}\right), \tag{4.1.9}$$

with a kernel function K and bandwidths h_2 and h_1. The resulting estimator of the graph becomes

$$\widehat{F}^* = \underset{F \in \mathcal{F}_d}{\arg\max} \sum_{(i,j) \in E_F} \widehat{I}(X_i; X_j). \tag{4.1.10}$$

A held-out set is usually used to prune the spanning tree $\widehat{F}^*$ by stopping early in Algorithm 4.1 when the likelihood on the held-out set is maximized. Thus we obtain a forest estimate of the graph.

The topological structure of a graph is an important part of graphical modeling and network analysis, such as graph sparsity, scale-free behavior, the presence of hubs, and shared graph structures among multiple graphs [18]. Incorporating such topological knowledge in graph estimation and density estimation is a fundamental goal. Sparsity has been successfully studied via ℓ_1 penalization to obtain consistent estimators of the precision matrix, but little work has been done with other graph-topological properties, especially in the case of tree-based graphical models.

In this chapter, we propose two extensions to the basic tree-based graph estimation and density estimation methods. The first extension is based on a scale-free graph constraint,

while the second extension focuses on joint learning of multiple graphs which have similar structures, although not necessarily identical. By incorporating prior structural knowledge into the estimation of the underlying graphical models, our approaches arise from a Bayesian formulation as a maximum a posteriori (MAP) estimate and solve the optimization problems via a minorize-maximization procedure with Kruskal's algorithm. Compared with parametric methods that penalize the Gaussian likelihood, the proposed methods result in more accurate estimation of the underlying networks.

The rest of this chapter is organized as follows. In Section 4.2, we propose nonparametric tree-based methods for estimating scale-free graphs in high dimensions. In Section 4.3, we describe methods for joint learning of multiple forests. We conclude in Section 4.4. Detailed proofs are collected in the last section.

4.2 Scale-Free Graphical Models

4.2.1 Background and Related Work

A wide variety of the networks, such as protein, gene, and social networks, are reported to be scale-free. That is, the degree distribution of the vertices follows a power law: $\mathbb{P}(\text{degree} = k) \propto k^{-\alpha}$ for some $\alpha > 1$. In such scale-free networks, some vertices have many more connections than others, and these highest-degree vertices are usually called hubs and serve significant roles in their networks. For instance, in a gene network, a hub usually represents a gene playing functions in many biological processes [57]. The methods described above, however, take no prior information of such graphical structure into account. They usually do not fit well on a scale-free graphical model, and fail to identify the important hubs, as the methods implicitly assume *a priori* that each edge is equally likely.

To utilize the prior belief that an underlying network may be scale-free, or more generally, contains some dominating hubs, several approaches have been proposed, including [13, 34, 47, 48]. Nevertheless, to the best of our knowledge, all the existing methods assume normality

of the data distribution. In particular, they try to maximize the Gaussian likelihood as a function of the precision matrix, with some specifically designed penalty functions to encourage the scale-free feature in the estimated graph. Such distributional assumptions can be quite unrealistic and unnecessary in many applications. Even though the marginal distribution of each variable can be transformed to approximately Gaussian, which allows arbitrary univariate distributions, the joint dependence is still restricted under the Gaussian assumption.

In this section, we relax such distributional assumptions and try to estimate scale-free graphs using a nonparametric method. We build on the forest density estimation (FDE) method introduced in [33]. When the graph is known to be scale-free, it is likely that the graph structure is close to a forest. In fact, if a graph is generated by the Barabási and Albert model [1], a standard generative model for scale-free networks, it is indeed a tree. Thus, a forest structure can be viewed as a reasonable assumption or approximation. The FDE approach can be viewed as maximizing a log likelihood as a function of the forest, we can incorporate the prior information to favor the scale-free graphs. Motivated from such a Bayesian perspective, we formulate a problem of finding the maximum spanning tree of a weighted graph with a penalty term on the logarithm of the node degrees. We will devise an algorithm based on a minorize-maximization procedure and Kruskal's algorithm [23] to find a local optimal solution. An R package of our method has been developed and is publicly available on the Web (`https://github.com/zhejosephliu/scalefreeForest`).

Before proceeding to present our nonparametric method, we pause to review some of the existing approaches. As mentioned, the existing methods for estimating a scale-free network or a network with hubs assume that the data are from a multivariate Gaussian distribution. Under this assumption, learning the graph is equivalent to estimating the zero pattern of the precision matrix Θ. The default method for estimating a sparse Θ is the graphical lasso [19]. To encourage the power law distribution of the vertex degrees, Liu and Ihler [34] propose to

replace the ℓ_1 penalty by a power law regularization term and solve

$$\widehat{\Theta}_{\text{SFGlasso}} = \underset{\Theta \succ 0}{\arg\max} \left\{ \log\det(\Theta) - \text{tr}(S\Theta) - \lambda \sum_i \log\left(\|\Theta_{-i,i}\|_1 + \varepsilon_i \right) \right\}, \tag{4.2.1}$$

where S is the empirical covariance matrix, $\|\Omega_{-i,i}\|_1 = \sum_{j=1, j\neq i}^{d} |\Omega_{ij}|$, and ε_i's are some small quantities. The term $\|\Omega_{-i,i}\|_1$ acts as a surrogate of the estimated degree of the ith node. Along the same line, Defazio and Caetano [13] impose a convex penalty by using submodular functions and their Lovász extension. Essentially, both methods try to penalize the log degree of each node, but end up using a continuous/convex surrogate to avoid the combinatorial problems involving the degrees. Another attempt is made by Tan et al. [47] to learn networks with hubs. To account for the existence of hubs, they design the following penalty term $P(\Theta)$:

$$\min_{V,Z:\ \Theta = V + V^T + Z} \left\{ \lambda_1 \|Z - \text{diag}(Z)\|_1 + \lambda_2 \|V - \text{diag}(V)\|_1 + \lambda_3 \sum_{j=1}^{d} \|(V - \text{diag}(V))_j\|_2 \right\}, \tag{4.2.2}$$

where the sparse elements of Z represent edges between non-hub nodes, and the non-zero columns of V correspond to hub nodes. The estimate is then obtained to be

$$\widehat{\Theta}_{\text{HGlasso}} = \underset{\Theta \succ 0}{\arg\max} \left\{ \log\det(\Theta) - \text{tr}(S\Theta) - P(\Theta) \right\}. \tag{4.2.3}$$

We motivate our method of estimating scale-free graphical models in the next subsection and introduce our estimation algorithm.

4.2.2 Scale-Free Forest Density Estimation

To avoid the Gaussian assumption in estimating a scale-free graphical model, we start from the FDE approach, and incorporate the prior information to improve the estimator obtained

in (4.1.10), especially when sample size is small and the estimates $\widehat{I}(X_i; X_j)$ have relatively large statistical error. To encourage the resulting spanning tree to be scale-free, we add a penalty on the logarithm of the node degrees and solve the following problem

$$\widehat{F}^*_\lambda = \arg\max_{F \in \mathcal{F}_d} \left\{ \sum_{(i,j) \in E_F} \widehat{I}(X_i; X_j) - \lambda \sum_{k \in V_F} \log(\delta(F,k)) \right\}, \tag{4.2.4}$$

where $\delta(F,k)$ is the degree of the kth vertex of F, and λ is a tuning parameter. In fact, this optimization problem can be motivated from a Bayesian perspective.

Bayesian motivation

Consider a prior distribution on the spanning trees on d vertices which satisfies that

$$\mathbb{P}(F) \propto \prod_{k \in V_F} \delta(F,k)^{-\alpha}, \tag{4.2.5}$$

for $F \in \mathcal{F}_d$ and some $\alpha > 1$. This prior distribution favors the spanning trees whose degrees have a power law distribution, and thus reflects our prior beliefs. Given the data $X^{(1:n)} = \{X^{(1)}, \ldots, X^{(n)}\}$ and assuming that the true density p^* is known and Markov to the spanning tree F, we can write the likelihood as

$$\mathbb{P}(X^{(1:n)}|F) = \prod_{t=1}^{n} \left(\prod_{(i,j) \in E_F} \frac{p^*_{ij}(X_i^{(t)}, X_j^{(t)})}{p^*_i(X_i^{(t)}) p^*_j(X_j^{(t)})} \prod_{k \in V_F} p^*_k(X_k^{(t)}) \right). \tag{4.2.6}$$

Now the posterior probability of F is

$$\mathbb{P}(F|X^{(1:n)}) \propto \mathbb{P}(X^{(1:n)}|F)\mathbb{P}(F) \tag{4.2.7}$$

$$\propto \prod_{t=1}^{n} \left(\prod_{(i,j) \in E_F} \frac{p^*_{ij}(X_i^{(t)}, X_j^{(t)})}{p^*_i(X_i^{(t)}) p^*_j(X_j^{(t)})} \prod_{k \in V_F} p^*_k(X_k^{(t)}) \right) \prod_{k \in V_F} \delta(F,k)^{-\alpha}. \tag{4.2.8}$$

The maximum a posteriori (MAP) estimate is given by

$$\widehat{F}_{\mathrm{MAP}} = \underset{F \in \mathcal{F}_d}{\arg\max} \left\{ \sum_{(i,j) \in E_F} \sum_{t=1}^{n} \frac{1}{n} \log \frac{p^*_{ij}(X_i^{(t)}, X_j^{(t)})}{p^*_i(X_i^{(t)}) p^*_j(X_j^{(t)})} - \frac{\alpha}{n} \sum_{k \in V_F} \log \delta(F, k) \right\}. \tag{4.2.9}$$

We can re-parameterize by setting $\lambda = \alpha/n$. Since the true density p^* is unknown, we propose the following two approaches to approximate the MAP estimate. First we split the data set into two disjoint groups according to the index sets $\mathcal{D}_1$ and $\mathcal{D}_2$, such that $|\mathcal{D}_i| = n_i$ and $\mathcal{D}_1 \cup \mathcal{D}_2 = \{1, \ldots, n\}$.

- **Option 1:** Use (4.1.7) to estimate the mutual information based on data $\{X^{(t)} : t \in \mathcal{D}_1\}$ and obtain $\widehat{F}^*_\lambda$ in (4.2.4) as an approximation of $\widehat{F}_{\mathrm{MAP}}$. In fact, $\widehat{F}^*_\lambda$ is obtained by replacing the true marginal densities and the empirical distributions in (4.2.9) by their corresponding density estimates. Subset $\{X^{(t)} : t \in \mathcal{D}_2\}$ is used to prune the resulting spanning tree to obtain a final estimate that maximizes the held-out likelihood.

- **Option 2:** First form the bivariate and univariate marginal density estimates $\widehat{p}_{ij}$'s and $\widehat{p}_k$'s using $\{X^{(t)} : t \in \mathcal{D}_1\}$ and then solve for

$$\check{F}^*_\lambda = \underset{F \in \mathcal{F}_d}{\arg\max} \left\{ \sum_{(i,j) \in E_F} \sum_{t \in \mathcal{D}_2} \frac{1}{n_2} \log \frac{\widehat{p}_{ij}(X_i^{(t)}, X_j^{(t)})}{\widehat{p}_i(X_i^{(t)}) \widehat{p}_j(X_j^{(t)})} - \lambda \sum_{k \in V_F} \log \delta(F, k) \right\}. \tag{4.2.10}$$

  Restrict $\check{F}^*_\lambda$ to the edges with positive weights to obtain a final estimate.

Optimization

To solve either of the optimization problems (4.2.4) and (4.2.10), we first rewrite the objective function as

$$f(Z) = \sum_{i<j} w_{ij} z_{ij} - \lambda \sum_{i=1}^{d} \log \left(\sum_{j=1}^{d} z_{ij} \right), \tag{4.2.11}$$

where w_{ij} is the edge weight given in (4.2.4) or (4.2.10), and $Z = (z_{ij})_{d\times d}$ is the adjacency matrix of F. Note that we have the additional constraint that the graph corresponding to Z is a spanning tree. To maximize $f(Z)$ under the constraint, we use a minorize-maximization approach [22]. Let $\widetilde{Z} = (\widetilde{z}_{ij})_{d\times d}$ be our current solution. We first lower bound $f(z)$ by linearizing $\log(\cdot)$ at $\widetilde{Z}$:

$$f(Z) \geq \sum_{i<j} w_{ij} z_{ij} - \lambda \sum_{i=1}^{d} \left(\log\left(\sum_{j=1}^{d} \widetilde{z}_{ij} \right) + \frac{\sum_{j=1}^{d} z_{ij} - \sum_{j=1}^{d} \widetilde{z}_{ij}}{\sum_{j=1}^{d} \widetilde{z}_{ij}} \right) \tag{4.2.12}$$

$$= \underbrace{\sum_{i<j} \left(w_{ij} - \frac{\lambda}{\sum_{k=1}^{d} \widetilde{z}_{ik}} - \frac{\lambda}{\sum_{k=1}^{d} \widetilde{z}_{jk}} \right) z_{ij}}_{f(Z;\widetilde{Z})} + C, \tag{4.2.13}$$

where C is a constant which doesn't depend on Z. We maximize this lower bound $f(Z;\check{Z})$ as a function of Z under the constraints. We then iterate through this pair of minorization and maximization procedures. Since the objective function is always increasing, the algorithm is guaranteed to converge to a local maximum.

Weights updating rule

Within each iteration of the minorize-maximization approach, we need to maximize the lower bound $f(Z;\check{Z})$. By (4.2.13), maximizing $f(Z;\widetilde{Z})$ is equivalent to finding the maximum spanning tree with edge weights given by

$$\widetilde{w}_{ij} = w_{ij} - \frac{\lambda}{\sum_{k=1}^{d} \widetilde{z}_{ik}} - \frac{\lambda}{\sum_{k=1}^{d} \widetilde{z}_{jk}}. \tag{4.2.14}$$

We see that the weights are updated at each iteration based on the current estimate of the graph. Each edge weight is penalized by two quantities that are inversely proportional to the degrees of the two endpoints of the edge. An edge weight is thus penalized less if its endpoints are already highly connected and vice versa. This is sort of a "rich gets richer"

procedure, and in this way it encourages some vertices to have high connectivity and hence the overall degree distribution to have a heavy tail.

The method can be thus thought as an iteration of Kruskal's algorithm with edges reweighted at each round. It is summarized in Algorithm 4.2.

Algorithm 4.2 Minorize-maximization algorithm for scale-free forest density estimation

Input Weight matrix $W = (w_{ij})_{d\times d}$ and tuning parameter λ

Initialize Z to be the adjacency matrix of the graph obtained by applying Algorithm 4.1 on W

do

$\widetilde{Z} \leftarrow Z$

$\widetilde{w}_{ij} \leftarrow w_{ij} - \frac{\lambda}{\sum_{k=1}^{d} \widetilde{z}_{ik}} - \frac{\lambda}{\sum_{k=1}^{d} \widetilde{z}_{jk}}$ and $\widetilde{W} \leftarrow (\widetilde{w}_{ij})_{d\times d}$

$Z \leftarrow \arg\max_{Z'} f(Z'; \widetilde{Z})$ by applying Algorithm 4.1 on $\widetilde{W}$

while $Z \neq \widetilde{Z}$

Output Z and the corresponding graph F.

4.2.3 Analysis of Simulated Data

Here we evaluate the performance of the proposed method and other existing methods on synthetic data. Consider the following two types of graph structures with $d = 100$ vertices.

- **Scale-free graph:** We use a preferential attachment process to generate a scale-free graph [1]. Specifically, we start with a chain of 4 nodes (i.e., with edges 1–2, 2–3, and 3–4). New nodes are added one at every time step, and each new node is connected to one of the existing nodes with probability $p_i \propto \delta_i^{\alpha}$, where δ_i is the current degree of the ith node, and α is a parameter, which we set to be 1.5 in our experiments. A typical realization of such networks is shown in Figure 4.1(a) in the top left corner.

- **Stars:** The graph has 5 stars of size 20. Each star is a tree with one internal node and 19 leaves. An illustration is shown in Figure 4.1(b) in the top left corner.

Given a particular graph, we generate $n_1 = 200$ samples according to two types of probability distributions that are Markov to the graph: Gaussian copulas and t copulas [14]. The Gaussian copula (resp., the t copula) can be thought of as representing the dependence structure implicit in a multivariate Gaussian (multivariate t) distribution, while each variable follows a uniform distribution on $[0, 1]$ marginally. Since both graph structures we consider are trees or forests, we generate the data sequentially, first sampling for an arbitrary node in a tree, and then drawing samples for the neighboring nodes according to the conditional distribution given by the copula until going through all nodes in the tree. In our simulations, the degree of freedom of the t copula is set to be 1, and the correlation coefficients are chosen to be 0.4 and 0.25 for the Gaussian and the t copula.

In the presentation that follows, the usual forest density estimation [33] will be referred to as FDE, and our method as SF-FDE. For the two methods, we use a held-out set of size $n_2 = 100$ to prune the estimated spanning trees. The tuning parameter of SF-FDE is chosen to maximize the likelihood on the held-out data after pruning. We also evaluate the performance of three approaches for learning Gaussian graphical models: the graphical lasso [19] (Glasso), the scale-free network approach [34] (SFGlasso), and the graphical lasso for graphs with hubs [47] (HGlasso). To apply these methods, we first transform the data marginally to be approximately Gaussian. There are tuning parameters for all three methods. We choose them by searching through a fine grid, and selecting those that maximize the likelihood on the held-out set. We refer to this as *held-out tuning*. The results obtained by the held-out tuning reflect the performance of the methods in a fully data-driven way. In addition, we also consider what we call *oracle tuning*, where the tuning parameters are chosen to maximize the accuracy criterion (in particular, the F_1 score, to be discussed below) of the estimated graph. This tuning method requires the knowledge of the true graph, and hence it's not obvious that there would exist a data-driven way to achieve this. We include the oracle tuning as a way to show the optimal performance possibly achieved by the methods.

We carry out four sets of experiments, with data generated from the two types of graphs and the two types of distributions. For each set of experiments, we apply the five methods and repeat the simulations 10 times. We record the F_1 scores of the estimated graphs for each method. An F_1 score is the harmonic mean of a method's precision and recall and hence a measure of its accuracy. It's a number between 0 and 1; a higher score means better accuracy and 1 means perfect labelling. The average F_1 scores are shown in Table 4.1. From the table, we see that SF-FDE always performs better than FDE on these particular graphs. SF-FDE and FDE perform better than the other three methods using held-out tuning as the penalized likelihood methods tend to select more edges when tuning parameters are chosen to maximize the held-out likelihood. When the true copula is Gaussian, Glasso, SFGlasso and HGlasso all have very high scores if oracle tuning is used; they fail to deliver good performance when the true copula is no longer Gaussian. On the other hand, SF-FDE is not affected too much by the true distribution as long as the graph is scale-free or has dominating hubs. We also include in Figure 4.1 two sets of typical realizations and estimations. Note that the data are generated according to the t copula, and the tuning parameters for the three parametric methods are chosen to maximize the F_1 scores, which is unrealistic in a data-driven way. Even with the oracle tuning, we observe that SF-FDE significantly outperforms the three parametric methods.

Graph × Dist.	FDE	SF-FDE	Glasso		SFGlasso		HGlasso	
			held-out	oracle	held-out	oracle	held-out	oracle
Scale-free × $\mathcal{N}$	0.49	0.69	0.24	0.91	0.42	0.92	0.16	0.88
Stars × $\mathcal{N}$	0.49	0.81	0.25	0.93	0.46	0.98	0.08	0.99
Scale-free × t	0.89	0.98	0.30	0.43	0.47	0.53	0.07	0.55
Stars × t	0.93	0.98	0.32	0.56	0.50	0.67	0.09	0.79

Table 4.1: F_1 scores for graph estimation with four types of probability models ($\mathcal{N}$ represents Gaussian copula, and t for t copula). The scores are averaged over 10 replicates. For Glasso, SFGlasso and HGlasso, scores for both held-out tuning and oracle tuning are listed.

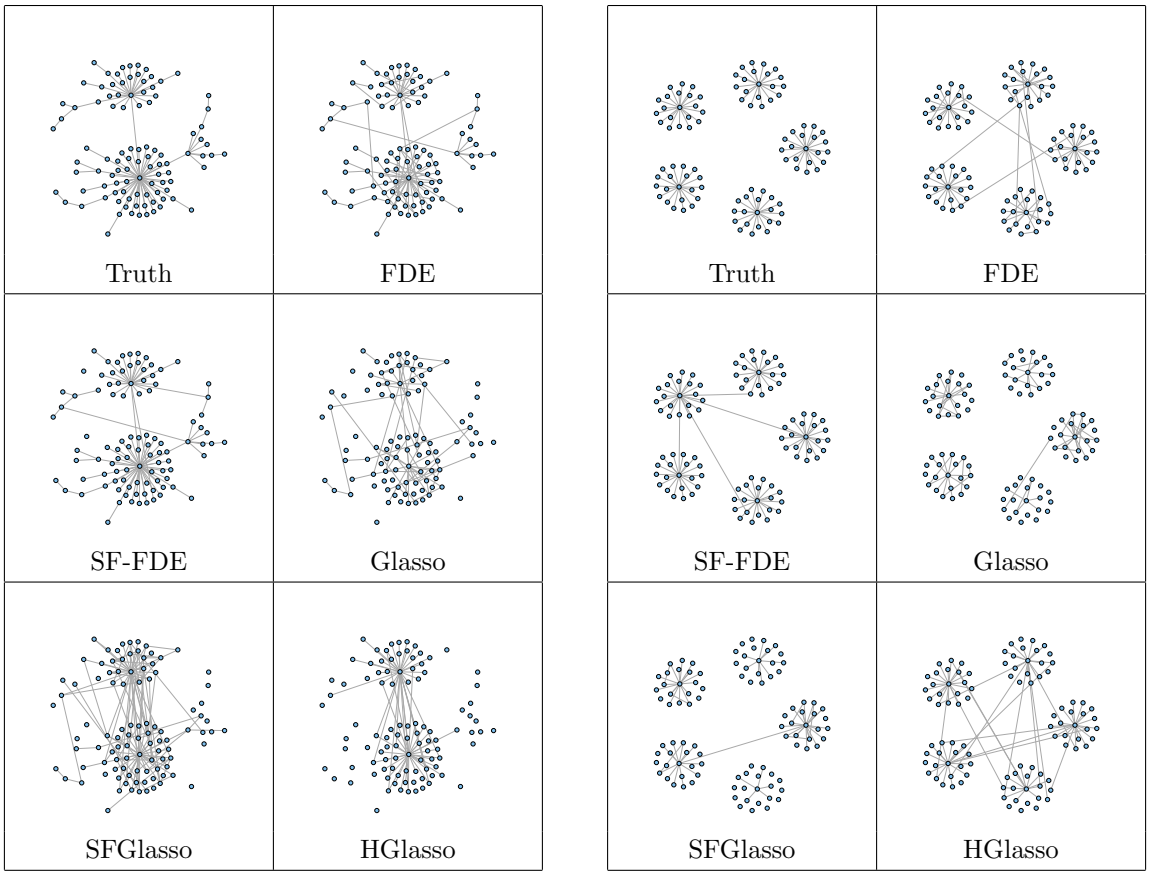


Figure 4.1: Typical realizations and estimated graphs with $d = 100$ nodes for (a) scale-free graph and (b) stars. $n_1 = 200$ samples are generated according to a t copula. Another $n_2 = 100$ samples are used to prune the spanning trees. The tuning parameters of Glasso, SFGlasso and HGlasso are chosen to maximize the F_1 scores.

4.2.4 Analysis of Stock Price and Microarray Data

We first test our method on stock price data from Yahoo! Finance. We collected the daily closing prices for $d = 417$ stocks that were consistently in the S&P 500 index from January 1, 2011 to December 31, 2014, excluding some stocks with anomalous price behavior. Let $S_k^{(t)}$ be the closing price for stock k on day t, and $X_k^{(t)} = \log(S_k^{(t)}/S_k^{(t-1)})$ be the log return. We marginally transform the log returns of each stock to a normal distribution. We treat $X^{(t)}$'s as independent random vectors, though they form a time series. We use the data from the first 9 months of 2014 as the training data and the data from the last 3 months of 2014 as the held-out part. We apply FDE and SF-FDE on this data set. We notice that SF-FDE

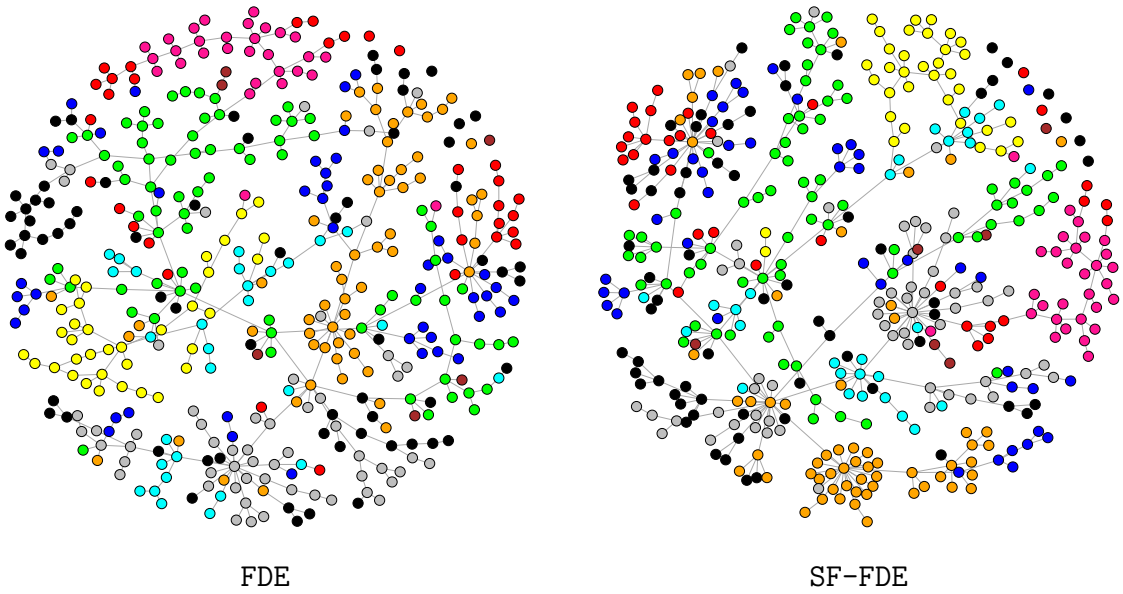


Figure 4.2: Estimated graphs via FDE and SF-FDE for the stock price data. The stocks are colored according to their Global Industry Classification Standard categories.

yields a larger likelihood on the held-out data, implying that a scale-free approximation is helpful in estimating/predicting the relationships. The estimated graphs are shown in Figure 4.2, and the stocks are colored according to their Global Industry Classification Standard categories. We see that the stocks with a same color are a bit more clustered in the graph estimated via SF-FDE.

As a second example, we consider a microarray data set from [38]. The data set contains Affymetrics chip measured expression levels of 4,238 genes for 295 normal subjects in the Centre d'Etude du Polymorphisme Humain and the International HapMap collections. For the purpose of presentation, we select a subset of 300 genes and apply FDE and SF-FDE. Once again, the held-out likelihood is higher for SF-FDE, indicating an improvement in the predictive power by a scale-free estimation. The estimated graphs are shown in Figure 4.3. In the FDE estimated graph, only one node has degree of 10 (colored in red), while in the SF-FDE graph, we see 8 highly connected nodes of degree no less than 10. Hopefully this can help identify important genes and inspire follow-up studies.

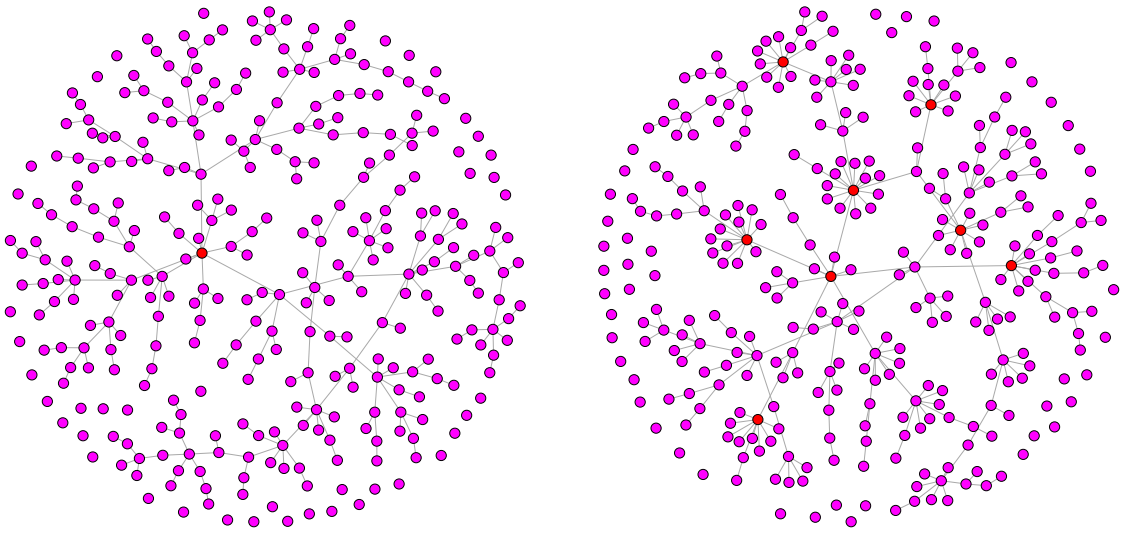


Figure 4.3: Estimated graphs via FDE and SF-FDE for the microarray data. Genes that have at least 10 connections are marked in red.

4.3 Multiple Graphical Models

4.3.1 Background and Related Work

In many applications, we may have multiple sets of data across distinct classes or units. For example, gene expression measurements are collected on a set of normal tissue samples and a set of cancer tissue samples. When the units of study are related, we expect observations from different units would have similar conditional independence structures, although not necessarily identical. It is thus natural to ask whether such similarity can be modeled by borrowing strength across different classes.

Methods for inferring Gaussian graphical models for multiple units have been proposed in recent years. Specifically, a joint graphical lasso approach is developed in [12], which is based upon maximizing a penalized log-likelihood by employing generalized fused lasso or group lasso penalties. Peterson et. al [40] propose a Bayesian approach for inference on multiple Gaussian graphical models. A hierarchical model for multiple Gaussian graphs is considered in [60], which leads to an optimization problem with a nonconvex log-shift penalty function.

In this section, our goal is to relax the Gaussian assumption imposed by most current methods in learning multiple graphs, and develop a method for estimating multiple forest-based graphical models. We formulate a hierarchical model for multiple related forests and derive an optimization problem which combines a log-likelihood term with a non-convex penalty. According to the forest factorization, the log-likelihood term only involves bivariate and univariate marginal distributions which can be estimated by nonparametric kernel-based methods. We efficiently find a local optimum of the resulting optimization problem using a minorize-maximization procedure, in which the objective function can be optimized by a sequence of iteratively reweighted Kruskal's algorithm. For density estimation, we do not assume the true density corresponds to a forest; instead, we estimate the optimal multiple forests that share certain locations of edges. Viewing the forest sizes as tuning parameters, we use a data-splitting scheme to select the best pruned forest-based structures based on held-out risk.

In the following subsection, we present our method, which includes a hierarchical model for multiple forests, a minorize-maximization procedure for the optimization problem, and the corresponding density estimator.

4.3.2 Learning Multiple Forest Graphical Models

Suppose we are given K classes or units of study, with $K > 1$. For each $k \in \{1, \ldots, K\}$, let $p^{(k)}$ be a density function corresponding to a random vector $X^{(k)} = (X_1^{(k)}, \ldots, X_d^{(k)})$. The undirected graph $G^{(k)}$ has d nodes associated with $X^{(k)}$, and missing edges (i, j) whenever $X_i^{(k)}$ and $X_j^{(k)}$ are conditionally independent given other variables. Let $\mathcal{X}_i^{(k)}$ denote the range of $X_i^{(k)}$.

The random vectors $X^{(k)}$ are assumed to be mutually independent across the K units. The joint density of $X = (X^{(1)}, \ldots, X^{(k)})$ can then be written as $p(x) = \prod_{k=1}^{K} p^{(k)}(x^{(k)})$, where $x = (x^{(1)}, \ldots, x^{(K)})$. Let $\mathcal{F}^{(k)}$ be the family of forests for unit k. Given some forest $F^{(k)} \in \mathcal{F}^{(k)}$, for any distribution $q^{(k)}$ that factorizes according to forest $F^{(k)}$, it is shown

in [2] that the negative log-likelihood risk $R^{(k)}(q^{(k)}) := -\mathbb{E}_{p^{(k)}}(\log q^{(k)})$ is maximized when $q^{(k)}$ and $p^{(k)}$ have the same bivariate and univariate marginal distributions.

For each unit k, observe data $Z^{(k)} = \{z_1^{(k)}, \ldots, z_{n_k}^{(k)}\}$ consisting of n_k samples measured on the random vector $X^{(k)}$, and these observations are assumed to be identically distributed with $p^{(k)}$. We also assume all $n = \sum_{k=1}^{K} n_k$ observations are independent.

A hierarchical model for multiple forest graphs

Here we formulate a hierarchical model for forests $F^{(1)}, \ldots, F^{(K)}$. For each k, the structure of $F^{(k)}$ can be represented as a symmetric binary matrix $g^{(k)}$ where the off-diagonal entry $g_{ij}^{(k)}$ indicates the inclusion of edge (i, j) in $F^{(k)}$. The inclusion of edge (i, j) in forests $F^{(1)}, \ldots, F^{(K)}$ is represented by the binary vector $g_{ij} = (g_{ij}^{(1)}, \ldots, g_{ij}^{(K)})$. In this way, the structure of the forests $F^{(1)}, \ldots, F^{(K)}$ is encoded by the parameter $g = (g^{(1)}, \ldots, g^{(K)})$. With a slight abuse of notation, we will use $g^{(k)}$ also for the induced forest that corresponds to the symmetric binary matrix itself.

We are now ready to state the hierarchical model for multiple forests:

$$\tau_{ij} \sim \text{Beta}(\alpha, \beta) \text{ for all } i < j, \tag{4.3.1}$$

$$g_{ij}^{(k)} | \tau_{ij} \sim \text{Bernoulli}(\tau_{ij}) \text{ for all } k, \text{ for all } i < j, \tag{4.3.2}$$

$$Z^{(k)} | g^{(k)} \sim \prod_{u=1}^{n_k} p_{g^{(k)}}^{(k)}(z_u^{(k)}) \text{ for all } k, \tag{4.3.3}$$

where the likelihood from observation $z_u^{(k)}$ given the forest $g^{(k)}$ can be written as

$$p_{g^{(k)}}^{(k)}(z_u^{(k)}) = \prod_{(i,j) \in E(g^{(k)})} \frac{p^{(k)}(z_{ui}^{(k)}, z_{uj}^{(k)})}{p^{(k)}(z_{ui}^{(k)}) p^{(k)}(z_{uj}^{(k)})} \prod_{i \in V(g^{(k)})} p^{(k)}(z_{ui}^{(k)}). \tag{4.3.4}$$

Note that $g^{(k)}$ might not form a forest for each k; thus we need to introduce a forest indicator function in the derivation of marginal prior distribution of g. This hierarchical

model characterizes our prior belief on shared structure across the K forests, since the Bernoulli parameter τ_{ij} for edge (i, j) is common across the units. Intuitively, τ_{ij} can be understood as the connectivity between nodes i and j. The hyperparameters α and β control the sparsity pattern of graphs.

The marginal prior distribution of g can be written as

$$
\begin{aligned}
p(g) &\propto \prod_{i<j} p(g_{ij}) \cdot \mathbb{1}\{g^{(k)} \in \mathcal{F}^{(k)} \text{ for all } k\} && (4.3.5)\\
&\propto \prod_{i<j} \int_{\tau_{ij}} p(g_{ij}|\tau_{ij}) p(\tau_{ij}) d\tau_{ij} \cdot \mathbb{1}\{g^{(k)} \in \mathcal{F}^{(k)} \text{ for all } k\} && (4.3.6)\\
&= \prod_{i<j} \int_{\tau_{ij}} \left[\prod_{k=1}^{K} p(g_{ij}^{(k)}|\tau_{ij}) \right] p(\tau_{ij}) d\tau_{ij} \cdot \mathbb{1}\{g^{(k)} \in \mathcal{F}^{(k)} \text{ for all } k\} && (4.3.7)\\
&\propto \prod_{i<j} \int_{\tau_{ij}} \tau_{ij}^{\alpha+\|g_{ij}\|_1-1} (1-\tau_{ij})^{\beta+K-\|g_{ij}\|_1-1} d\tau_{ij} \cdot \mathbb{1}\{g^{(k)} \in \mathcal{F}^{(k)} \text{ for all } k\} && (4.3.8)\\
&= \prod_{i<j} B(\alpha + \|g_{ij}\|_1, \beta + K - \|g_{ij}\|_1) \cdot \mathbb{1}\{g^{(k)} \in \mathcal{F}^{(k)} \text{ for all } k\}, && (4.3.9)
\end{aligned}
$$

where $B(\cdot, \cdot)$ denotes the Beta function.

If $p^{(1)}, \ldots, p^{(K)}$ is known, combining the marginal prior on g with the likelihood from the K units of data, the MAP estimate of g is given by

$$
\widehat{g} = \underset{g^{(k)} \in \mathcal{F}^{(k)} \text{ for all } k}{\operatorname{argmax}} \quad \sum_{k=1}^{K} \sum_{i<j} g_{ij}^{(k)} \cdot w_{ij}^{(k)} + \sum_{i<j} \log B(\alpha + \|g_{ij}\|_1, \beta + K - \|g_{ij}\|_1), \tag{4.3.10}
$$

where the weight on edge (i, j) for unit k is given by

$$
w_{ij}^{(k)} = \sum_{u=1}^{n_k} \log \frac{p^{(k)}(z_{ui}^{(k)}, z_{uj}^{(k)})}{p^{(k)}(z_{ui}^{(k)}) p^{(k)}(z_{uj}^{(k)})}. \tag{4.3.11}
$$

The above MAP procedure is not practical since the true density is unknown. We estimate

each $w_{ij}^{(k)}$ directly from data using estimates of bivariate and univariate marginals. For simplicity, we assume the data lie in a d-dimensional unit cube $[0,1]^d$. We thus estimate $w_{ij}^{(k)}$ by discretization:

$$\widehat{w}_{ij}^{(k)} = n_k \cdot \frac{1}{m^2} \sum_{u=1}^{m} \sum_{v=1}^{m} \widehat{p}^{(k)}(y_{ui}^{(k)}, y_{vj}^{(k)}) \log \frac{\widehat{p}^{(k)}(y_{ui}^{(k)}, y_{vj}^{(k)})}{\widehat{p}^{(k)}(y_{ui}^{(k)}) \widehat{p}^{(k)}(y_{vj}^{(k)})}, \tag{4.3.12}$$

where we calculate the kernel density estimates on a grid of points by choosing m evaluation points $y_{1i}^{(k)} < \cdots < y_{mi}^{(k)}$ on each dimension for the ith variable on any unit k.

Note that $\widehat{w}_{ij}^{(k)}/n_k$ is equivalent to the plug-in estimate of $I_{ij}^{(k)}(\widehat{p}^{(k)})$ between $X_i^{(k)}$ and $X_j^{(k)}$. Furthermore, $\mathbb{E}(w_{ij}^{(k)}/n_k) = I_{ij}^{(k)}(p^{(k)})$.

Define the *oracle* joint forests density $p_{g^*}^* := \operatorname{argmax}_q \mathbb{E}_p(\log q)$, where the density $q = \prod_{k=1}^K q^{(k)}$ and each $q^{(k)}$ factorizes according to some forest. Define the *oracle* joint forests as the corresponding joint forests $g^* = (g^{(1)*}, \ldots, g^{(K)*})$ that supports $p_{g^*}^*$.

Proposition 4.3.1. *The oracle joint forests g^* satisfies*

$$g^* = \underset{g^{(k)} \in \mathcal{F}^{(k)} \text{ for all } k}{\operatorname{argmax}} \sum_{k=1}^{K} \sum_{i<j} g_{ij}^{(k)} I_{ij}^{(k)}(p^{(k)}). \tag{4.3.13}$$

With $I_{ij}^{(k)}(p^{(k)})$ replaced by its plug-in estimate, maximizing $\sum_{k=1}^K \sum_{i<j} g_{ij}^{(k)} \widehat{w}_{ij}^{(k)}$ would lead to an estimate of oracle joint forests, under the case where $n_k = n_{k'}$ for any k, k'. Therefore, we do not assume the true density p corresponds to some joint forests; rather, we estimate the oracle joint forests with some constraints on the shared certain locations of edges.

The above procedure leads to the following optimization problem

$$\widehat{g} = \underset{g^{(k)} \in \mathcal{F}^{(k)} \text{ for all } k}{\operatorname{argmax}} f(g), \tag{4.3.14}$$

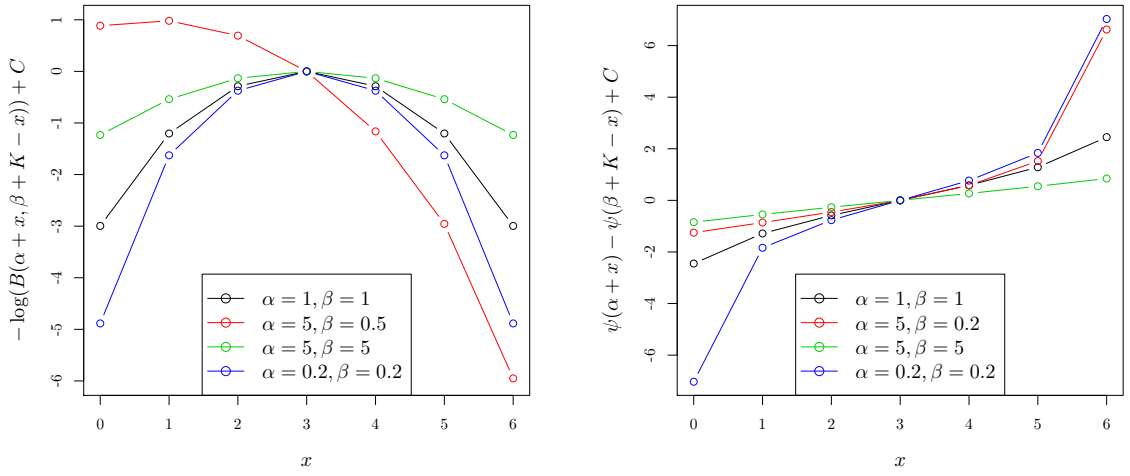
where the objective function is given by

$$f(g) := \sum_{k=1}^{K} \sum_{i<j} g_{ij}^{(k)} \widehat{w}_{ij}^{(k)} + \lambda \sum_{i<j} \log B(\alpha + \|g_{ij}\|_1, \beta + K - \|g_{ij}\|_1). \tag{4.3.15}$$

Here we also include an additional complexity parameter $\lambda \geq 0$ for the convenience of adjusting the magnitudes of the penalty function. The introduction of λ makes the process of tuning hyperparameters simpler.

The following proposition shows that the penalty term in the objective function is non-convex.

Proposition 4.3.2. *For any* $\alpha > 0, \beta > 0$, $-\log B(\alpha + x, \beta + K - x)$ *is concave for* $0 \leq x \leq K$.



(a) $-\log B(\alpha + x, \beta + K - x)$ (b) $\psi(\alpha + x) - \psi(\beta + K - x)$

Figure 4.4: Behavior of $-\log B(\alpha + x, \beta + K - x)$ and $\psi(\alpha + x) - \psi(\beta + K - x)$ as α and β vary, with $K = 6$. Curves are shifted to have a common joint at $x = 3$.

As the values of α and β vary, Figure 4.4(a) shows the behavior of $-\log(B(\alpha + x, \beta + K - x))$ when $K = 6$. We can see that α and β control the concavity of the penalty function.

Intuitively, in balanced cases where $\alpha = \beta$, if $\|g_{ij}\|_1$ is close to 0 or K, meaning edge (i, j) is either absent or present in a majority of the K units, a "bonus" will be awarded to the objective function. Since the total number of edges in a forest is at most $d-1$, such a bonus would encourage the appearance of common edges across units. Furthermore, the smaller the values of α and β, the larger the (relative) bonus will be awarded; in the unbalanced case where the ratio of α/β is large, the bonus only occurs when $\|g_{ij}\|_1$ approaches K.

Optimization via minorize-maximization

To optimize $f(g)$ in (4.3.14), we employ a minorize-maximization procedure. Let $g(t)$ be the estimate of g on iteration t. According to Proposition 4.3.2, for any (i, j), we obtain an upper bound of the concave function $-\log B(\alpha + \|g_{ij}\|_1, \beta + K - \|g_{ij}\|_1)$ by a linear approximation (first-order Taylor expansion) centered at $g_{ij}(t)$:

$$-\log B(\alpha + \|g_{ij}\|_1, \beta + K - \|g_{ij}\|_1) \leq -\log B(\alpha + \|g_{ij}(t)\|_1, \beta + K - \|g_{ij}(t)\|_1) - (\|g_{ij}\|_1 - \|g_{ij}(t)\|_1) \cdot (\psi(\alpha + \|g_{ij}(t)\|_1) - \psi(\beta + K - \|g_{ij}(t)\|_1)), \tag{4.3.16}$$

where $\psi(x) = \frac{d}{dx} \log \Gamma(x)$ is the digamma function.

We compute the next iteration $g(t+1)$ via

$$g(t+1) = \underset{g^{(k)} \in \mathcal{F}^{(k)} \text{ for all } k}{\operatorname{argmax}} Q(g; g(t)), \tag{4.3.17}$$

where

$$Q(g; g(t)) := \sum_{k=1}^{K} \sum_{i<j} g_{ij}^{(k)} \widehat{w}_{ij}^{(k)} + \lambda \sum_{i<j} \|g_{ij}\|_1 \cdot (\psi(\alpha + \|g_{ij}(t)\|_1) - \psi(\beta + K - \|g_{ij}(t)\|_1)). \tag{4.3.18}$$

Here, we have omitted constant terms that do not affect the optimization. Note that in

iteration t of the minorize-maximization procedure, $Q(g; g(t))$ can be further written as

$$Q(g; g(t)) = \sum_{k=1}^{K} \Big(\sum_{i<j} g_{ij}^{(k)} \widetilde{w}_{ij}^{(k)} \Big), \tag{4.3.19}$$

where

$$\widetilde{w}_{ij}^{(k)} := \widehat{w}_{ij}^{(k)} + \lambda(\psi(\alpha + \|\widetilde{g}_{ij}\|_1) - \psi(\beta + K - \|\widetilde{g}_{ij}\|_1)). \tag{4.3.20}$$

Then the optimal K forests that maximize $Q(g; g(t))$ can be determined by employing Kruskal's algorithm K times, once for each unit. The objective function (4.3.14) can be optimized by a sequence of iteratively reweighted Kruskal's algorithms. In iteration t, for each unit k, we update the weights according to (4.3.20) and use Kruskal's algorithm to grow a maximum weight spanning tree of a graph with edge weight $\widetilde{w}_{ij}^{(k)}$. The resulting K forests are thus the optimal forests $g(t+1)$ that maximizes $Q(g; g(t))$. This process is repeated until the algorithm converges.

As is demonstrated by Figure 4.4(b), the weight update (4.3.20) in each iteration of the minorize-maximization has an appealing intuition: it increases the weights for each edge (i, j) as $\|\widetilde{g}_{ij}\|_1$ approaches K and thus encourages the appearance of common locations of edges across the K units. Since the total number of edges in any K forests is bounded by $K(d-1)$, this "common-get-more-common" phenomenon on edge preference also tends to eliminate any edge (i, j) if $\|\widetilde{g}_{ij}\|_1$ is small, unless the weight for that edge is large enough to overcome such penalty, that is, there exists strong evidence in the data to support the appearance of that edge.

Joint forests estimator

We are now ready to summarize the *joint forests* method for learning multiple related forests. We are concerned with graph estimation and density estimation for continuous random

variables. The method for graphical models on discrete variables is closely analogous and thus omitted here.

For any unit k, any full spanning tree obtained in the minorize-maximization procedure above might have high variance when dimension d is large, which could lead to overfitting in density estimation. Following a similar procedure as in [33], we prune each tree for unit k and choose the optimal number of edges by maximizing the log-likelihood of a set of held-out data collected also from the same unit. The hyperparameters α, β and tuning parameter λ will also be selected using the held-out data.

First, for each unit k, randomly partition the data $Z^{(k)}$ into two parts $Z^{(k)}_{train}$ and $Z^{(k)}_{held}$. Let $\mathcal{D}_{train} = \{Z^{(k)}_{train}\}_{k=1}^{K}$ and $\mathcal{D}_{held} = \{Z^{(k)}_{held}\}_{k=1}^{K}$. Then apply the steps below.

1. Using $\mathcal{D}_{train}$, estimate each bivariate and univariate marginal density using kernel density estimation. Then calculate the estimates $\widehat{w}^{(k)}_{ij}$ for all k and all (i, j);

2. Initialization of the minorize-maximization procedure: we apply Kruskal's algorithm to each unit data of $Z^{(1)}_{train}, \ldots, Z^{(K)}_{train}$ to learn K spanning trees, used as the initials of the minorize-maximization procedure;

3. For $t = 1, 2, \ldots,$ repeat Kruskal's algorithm on the reweighted graphs to solve the optimization problem (4.3.17). Stop when the structures of K forests become intact;

4. Using $\mathcal{D}_{held}$, choose the optimal K forests that maximizes the held-out log-likelihood.

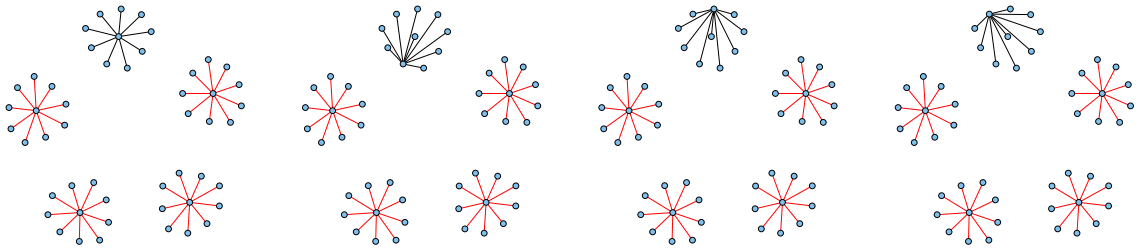
4.3.3 Analysis of Simulated Data

We present numerical results on simulated data to evaluate the performance of the *joint forests* approach. The *forest* density estimation [33] is considered as a competitive forest-based method, which amounts to separately model each unit of data. We also test the joint estimation method (*joint glasso*) [20], and the group graphical lasso method (*group glasso*) [12] for comparison. In dimension $d = 50$, we generate $K = 4$ units of data based on the following two types of graphs.

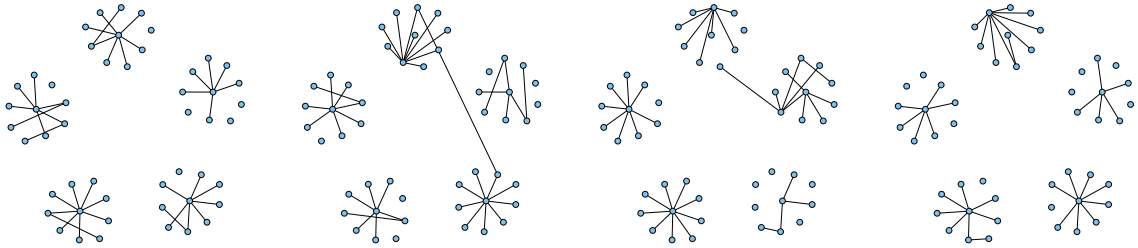
- **Hub graphs:** The d nodes are evenly partitioned into 5 disjoint groups. Within each group, one node is selected as the hub center and we add edges between the hub center and the other $d/5 - 1$ nodes in that group. To generate graph structures for K units, we let 4 of the 5 hubs share the same structures across the K units; the only left 1 hub has a different hub center in each unit.

- **Random graphs:** We consider the following generative model for growing a tree graph. We start with 2 connected nodes and add new node one at a time. The new node is connected to one of the existing nodes with probability proportional to the current degrees of the existing nodes. We apply this generative model to grow a common tree of size $4d/5$ to be shared across the K units. Each unit then continues this growing process independently until obtaining a tree of d vertices.

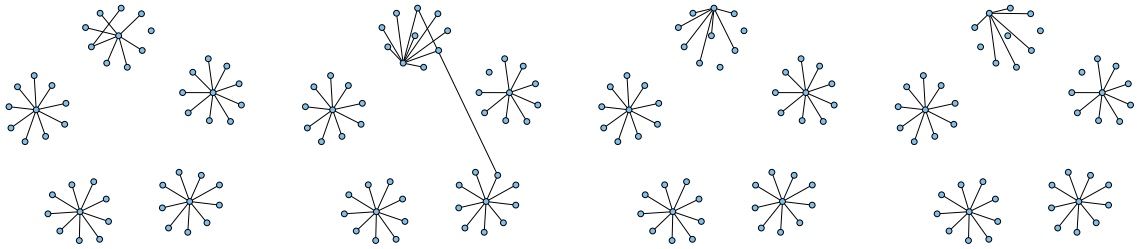
Once the forest structures are obtained, we generate data sequentially using a bivariate t-copula: first sample data for an arbitrary node, and then draw samples for the neighboring nodes according to the conditional distribution given by a bivariate t-copula until all the nodes are traversed. For each unit k, we sample $n_k = 150$ data points for training, with the same amount of data held out for the selection of complexity parameters. Note that for the joint glasso and group glasso methods, the data has been pre-processed by marginal transformations to Gaussian.



(a) *true* graphs with common edges colored as red

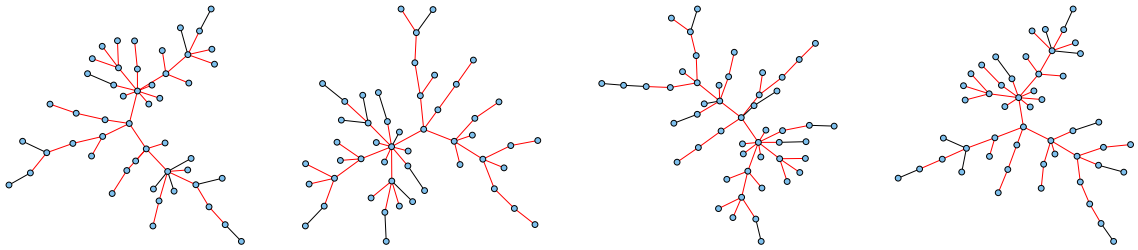


(b) estimated graphs by *forest*

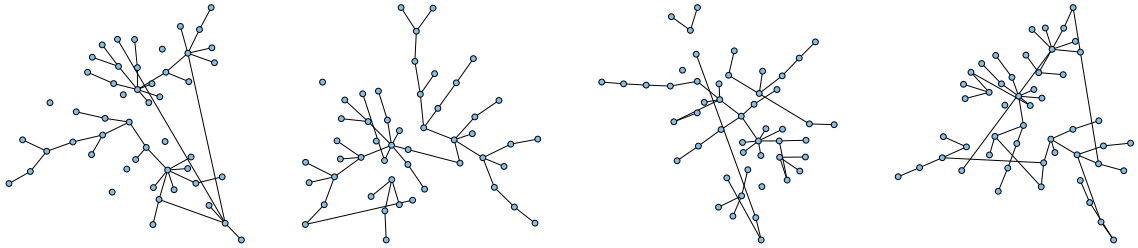


(c) estimated graphs by *joint forests*

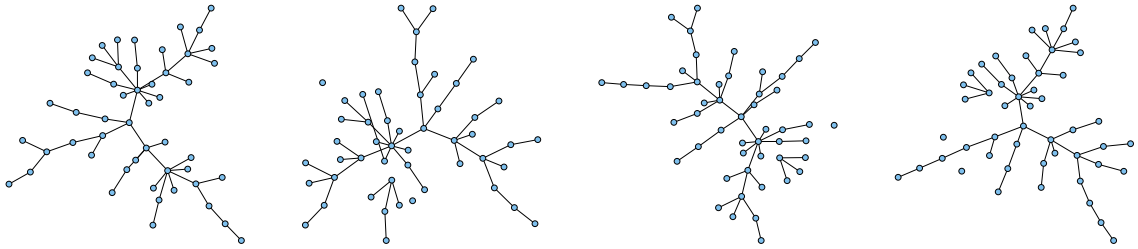
Figure 4.5: Graph recovery results on simulated data for hub graphs ($d = 50$) for the joint forests, forest, joint glasso, and group glasso methods.



(a) *true* graphs with common edges colored as red

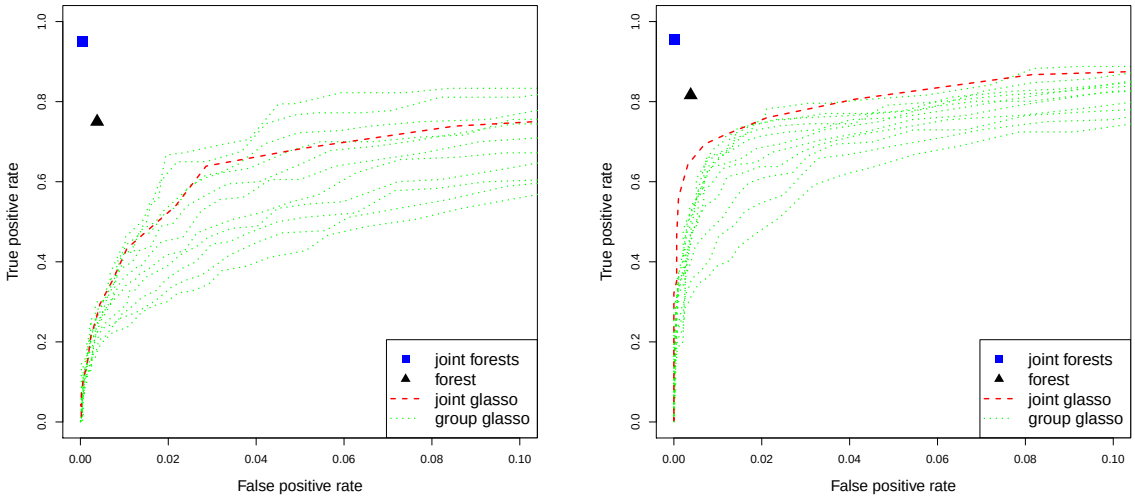


(b) estimated graphs by *forest*



(c) estimated graphs by *joint forests*

Figure 4.6: Graph recovery results on simulated data for random graphs ($d = 50$) for the joint forests, forest, joint glasso, and group glasso methods.



(a) hub graphs

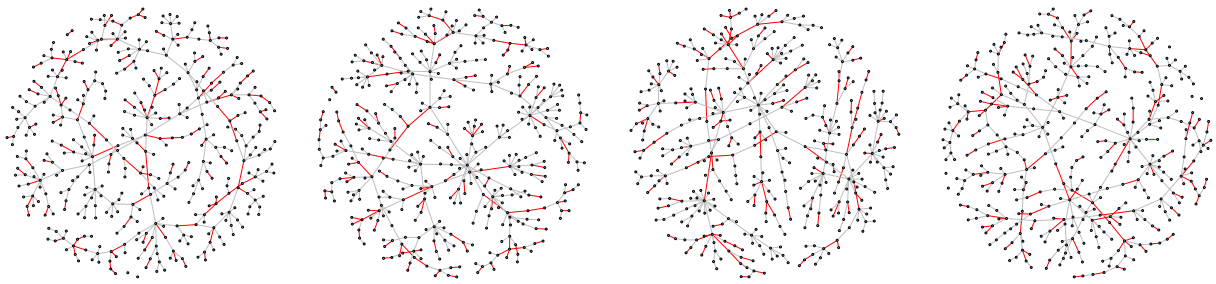
(b) random graphs

Figure 4.7: ROC curves on simulated data ($d = 50$) for the joint forests, forest, joint glasso, and group glasso methods.

A typical run showing the estimated graphs and the receiver operating characteristic (ROC) curves is provided in Figures 4.5, 4.6 and 4.7. For the forest and joint forest methods, we mark the points corresponding to their performance on the ROC space. We see that the graph recovery performance of the joint forests method is significantly better than those of other three methods. The joint forest method is superior to performing separate forest estimation due to the fact that the latter does not share information between units and is thus unable to exploit the graph similarity by borrowing strength across different units; for the generated non-Gaussian data, it is also expected that the joint forests method outperforms the joint glasso and group glasso methods. Moreover, the held-out likelihood of the joint forests method is significantly higher than that of forest estimation method.

4.3.4 Analysis of Stock Price and Cell Signalling Data

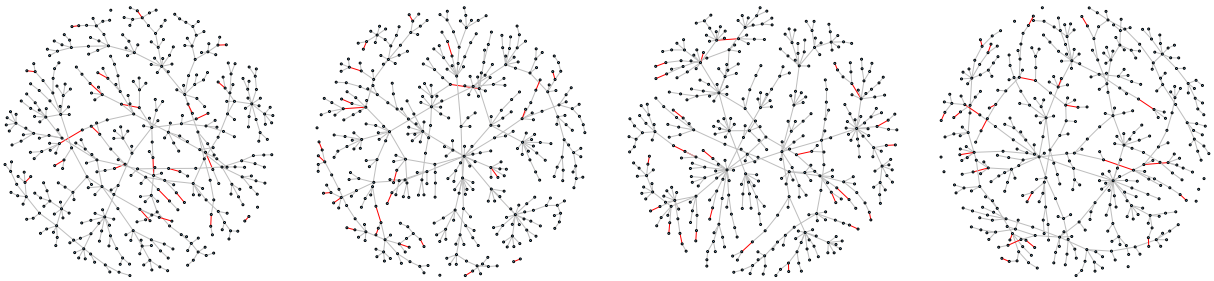
In the first example, we consider daily stock closing prices from Yahoo! Finance for $d = 432$ stocks in the S&P 500 index from January 5, 2009 to December 31, 2012. We partition the data into 4 time periods, one for a year. A nonparanormal transformation is estimated and the log returns of each stock are replaced by their respective normal scores, subject to a Winsorized truncation [31]. We model the 4-unit datasets using forest density estimation [33] and joint forests methods. The estimated graphs by joint forests method are provided in Figure 4.8. The aggregated held-out log-likelihood over the 4 units are 185.5 for forest method and 193.4 for joint forests method. The numbers of common edges across the 4 graphs are 24 for forest method and 111 for joint forests method.



(a) 2009 (b) 2010 (c) 2011 (d) 2012

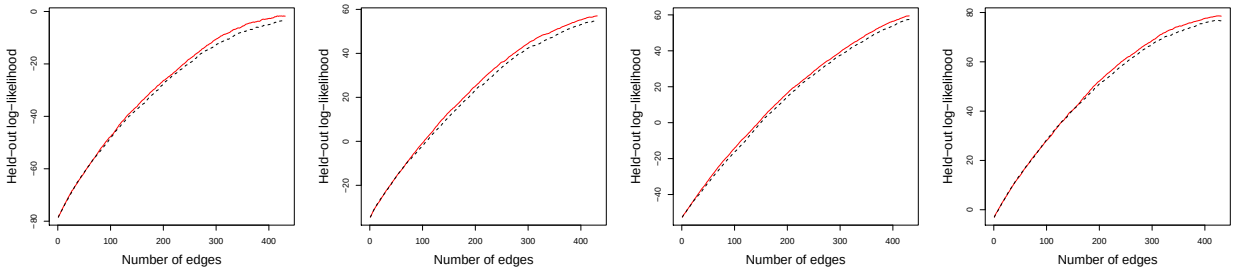
Figure 4.8: Results on stock datasets on 4 units, one for a year of 2009–2012. Estimated graphs by joint forests density estimation method. Common edges across the 4 forests are colored as red.

Figure 4.9 shows the estimated graphs on stock data using forest method. Figure 4.10 shows the held-out log-likelihood for forest and joint forests methods on stock data. We can see that the held-out likelihood of the joint forests method is higher than that of forest estimation method.



(a) 2009 (b) 2010 (c) 2011 (d) 2012

Figure 4.9: Results on stock datasets on 4 units, one for a year of 2009–2012. Estimated graphs by forest density estimation method. Common edges across the 4 forests are colored as red.



(a) 2009 (b) 2010 (c) 2011 (d) 2012

Figure 4.10: Results on stock datasets on 4 units, one for a year of 2009–2012. Held-out log-likelihood of forest method (dashed line) and joint forests method (solid red line).

In the second example, we analyze a flow cytometry dataset on $d = 11$ proteins under $K = 9$ different experimental conditions [45]. The sample size for each condition is between 700 and 1,000. We use the joint forests method to infer a forest for each of the 9 conditions allowing for possible shared edges. Figure 4.11 shows the number of appearance of each edge across the 9 conditions (edges absent in all units are not displayed). We can see that 7 edges are shared by most of conditions.

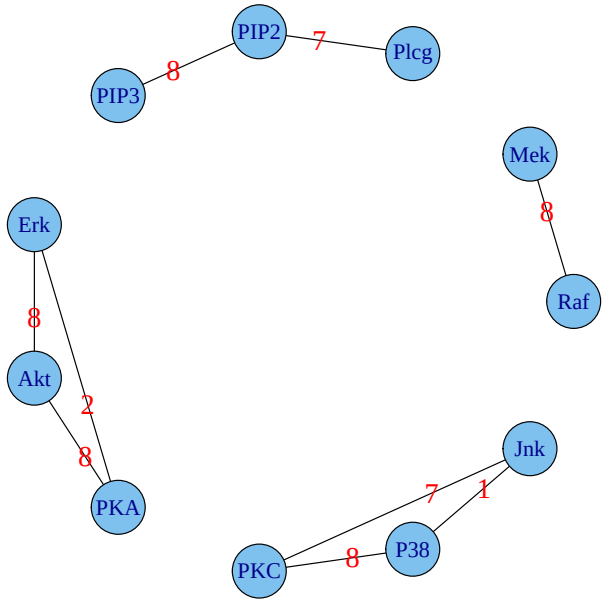


Figure 4.11: Results on cell signalling data on 9 units. For each edge, number of appearance across units is marked.

4.4 Conclusion

In this chapter we introduce nonparametric tree-based methods for estimating graphical models with structural constraints. It allows us to relax the Gaussian assumption and utilize a family of estimators based on forests. Our approaches arise from a Bayesian formulation as the MAP estimate and solve the optimization problems via a minorize-maximization procedure with Kruskal's algorithm. Compared with parametric methods that penalize the Gaussian likelihood, the proposed method leads to more accurate and robust estimation of the underlying networks.

4.5 Proofs

Proof of Proposition 4.3.1.

Proof. Given a set of K forests $F^{(1:K)} = \{F^{(1)}, \ldots, F^{(K)}\}$ with each $F^{(k)} \in \mathcal{F}^{(k)}$, we want to model the true density p using some density function $q = \prod_{k=1}^{K} q^{(k)}$ with each $q^{(k)}$

factorizes according to $F^{(k)}$.

Consider the negative log-likelihood risk

$$
\begin{aligned}
R(q) :&= -\mathbb{E}_p \log q && (4.5.1)\\
&= -\sum_{k=1}^{K} \mathbb{E}_{p^{(k)}} \log q^{(k)} && (4.5.2)\\
&= -\sum_{k=1}^{K} \mathbb{E}_{p^{(k)}} \log q^{(k)} + \sum_{k=1}^{K} \mathbb{E}_{p^{(k)}_{F^{(k)}}} \log p^{(k)}_{F^{(k)}} - \sum_{k=1}^{K} \mathbb{E}_{p^{(k)}_{F^{(k)}}} \log p^{(k)}_{F^{(k)}} && (4.5.3)\\
&= -\sum_{k=1}^{K} \mathbb{E}_{p^{(k)}_{F^{(k)}}} \log q^{(k)} + \sum_{k=1}^{K} \mathbb{E}_{p^{(k)}_{F^{(k)}}} \log p^{(k)}_{F^{(k)}} - \sum_{k=1}^{K} \mathbb{E}_{p^{(k)}} \log p^{(k)}_{F^{(k)}} && (4.5.4)\\
&= \mathrm{KL}(p^{(k)}_{F^{(k)}} \| q^{(k)}) - \sum_{k=1}^{K} \mathbb{E}_{p^{(k)}} \log p^{(k)}_{F^{(k)}}, && (4.5.5)
\end{aligned}
$$

where for any unit k,

$$
p^{(k)}_{F^{(k)}}(x^{(k)}) := \prod_{(i,j)\in E(F^{(k)})} \frac{p^{(k)}(x^{(k)}_i, x^{(k)}_j)}{p^{(k)}(x^{(k)}_i) p^{(k)}(x^{(k)}_j)} \prod_{i \in V(F^{(k)})} p^{(k)}(x^{(k)}_i), \tag{4.5.6}
$$

and $\mathrm{KL}(\cdot\|\cdot)$ represents the Kullback-Leibler divergence.

Given any joint forests $F^{(1:K)}$, we can see that the density function

$$
p_{F^{(1:K)}}(x) := \prod_{k=1}^{K} p^{(k)}_{F^{(k)}}(x^{(k)}) \tag{4.5.7}
$$

achieves the minimum risk.

Let $\mathcal{X}^{(k)} = \mathcal{X}^{(k)}_1 \times \cdots \times \mathcal{X}^{(k)}_d$, then the second term of (4.5.5) can be written as

$$
\begin{aligned}
\sum_{k=1}^{K} \mathbb{E}_{p^{(k)}} \log p^{(k)}_{F^{(k)}} &= \sum_{k=1}^{K} \int_{\mathcal{X}^{(k)}} \log p^{(k)}_{F^{(k)}}(x^{(k)}) \cdot p^{(k)}(x^{(k)}) dx^{(k)} && (4.5.8)\\
&= \sum_{k=1}^{K} \sum_{(i,j)\in E(F^{(k)})} I^{(k)}_{ij}(p^{(k)}) - \sum_{k=1}^{K} \sum_{i \in V(F^{(k)})} H(X^{(k)}_i), && (4.5.9)
\end{aligned}
$$

where the mutual information between pair of variables $X_i^{(k)}$ and $X_j^{(k)}$

$$I_{ij}^{(k)}(p^{(k)}) := \int_{\mathcal{X}_i^{(k)} \times \mathcal{X}_j^{(k)}} p^{(k)}(x_i^{(k)}, x_j^{(k)}) \log \frac{p^{(k)}(x_i^{(k)}, x_j^{(k)})}{p^{(k)}(x_i^{(k)}) p^{(k)}(x_j^{(k)})} dx_i^{(k)} dx_j^{(k)}, \tag{4.5.10}$$

and the entropy of variable $X_i^{(k)}$

$$H(X_i^{(k)}) := -\int_{\mathcal{X}_i^{(k)}} p^{(k)}(x_i^{(k)}) \log p^{(k)}(x_i^{(k)}) dx_i^{(k)}. \tag{4.5.11}$$

Observe that the second term of (4.5.9) is a constant as a function of $F^{(1:K)}$.

Then the oracle joint forests satisfies

$$g^* = \underset{F^{(k)} \in \mathcal{F}^{(k)} \text{ for all } k}{\operatorname{argmax}} \sum_{k=1}^{K} \sum_{(i,j) \in E(F^{(k)})} I_{ij}^{(k)}(p^{(k)}). \tag{4.5.12}$$

Also, the corresponding oracle joint forests density is given by

$$p_{g^*}^*(x) = \prod_{k=1}^{K} p_{g^{(k)*}}^{(k)}(x^{(k)}). \tag{4.5.13}$$

□

Proof of Proposition 4.3.2.

Proof. Note that $\log B(\alpha+x, \beta+K-x) = \log \Gamma(\alpha+x) + \log \Gamma(\beta+K-x) - \log \Gamma(\alpha+\beta+K)$. Using the fact that the logarithm of Gamma function is convex on positive real numbers, for

any $0 \leq x_1, x_2 \leq K$ and $t \in [0,1]$, we have

$$\begin{aligned}
&\log\Gamma(\alpha + tx_1 + (1-t)x_2) + \log\Gamma(\beta + K - tx_1 - (1-t)x_2) \\
&= \log\Gamma(t(\alpha + x_1) + (1-t)(\alpha + x_2)) + \log\Gamma(t(\beta + K - x_1) + (1-t)(\beta + K - x_2)) \\
&\leq t\log\Gamma(\alpha + x_1) + (1-t)\log\Gamma(\alpha + x_2) + t\log\Gamma(\beta + K - x_1) + (1-t)\log\Gamma(\beta + K - x_2) \\
&= t(\log\Gamma(\alpha + x_1) + \log\Gamma(\beta + K - x_1)) + (1-t)(\log\Gamma(\alpha + x_2) + \log\Gamma(\beta + K - x_2))).
\end{aligned} \tag{4.5.14}$$

Thus the proposition follows. □

CHAPTER 5

BLOSSOM TREE GRAPHICAL MODELS

5.1 Introduction

Both the Gaussian graphical model and the nonparanormal maintain tractable inference without placing limitations on the independence graph. But they are limited in their ability to flexibly model the bivariate and higher order marginals. At another extreme, forest-structured graphical models permit arbitrary bivariate marginals, but maintain tractability by restricting to acyclic graphs. Thus the ability to model complex independence graphs is compromised.

In this chapter we bring together the Gaussian, nonparanormal, and forest graphical models, using what we call blossom tree graphical models [35]. Informally, a blossom tree consists of a forest of trees, and a collection of subgraphs, the blossoms, possibly containing many cycles. The vertex sets of the blossoms are disjoint, and each blossom contains at most one node of a tree. We estimate nonparanormal graphical models over the blossoms, and nonparametric bivariate densities over the branches (edges) of the trees. Using the properties of the nonparanormal, these components can be combined, or factored, to give a valid joint density for $X = (X_1, \ldots, X_d)$. The details of our construction are given in Section 5.2. We develop an estimation procedure for blossom tree graphical models, including an algorithm for selecting tree branches, partition the remaining vertices into potential blossoms, and then estimating the graphical structures of the blossoms. Since an objective is to relax the Gaussian assumption, our criterion for selecting tree branches is deviation from Gaussianity. Toward this end, we use the negentropy, showing that it has strong statistical properties in high dimensions. In order to partition the nodes into blossoms, we employ a nonparametric partial correlation statistic. We use a data-splitting scheme to select the optimal blossom tree structure based on held-out risk. An R package of our method has been developed and is available on the Web (`https://github.com/zhejosephliu/blossomTree`).

In the following section, we present the details of our methods, including definitions of blossom tree graphs, the associated family of graphical models, and our estimation methods. Statistical properties of the estimators are given in Section 5.3. In Section 5.4, we present experiments with simulated and real data. Finally, we conclude in Section 5.5. Detailed proofs are collected in the last section.

5.2 Blossom Tree Graphs and Estimation Methods

5.2.1 Blossom Tree Graphs

To unify the Gaussian, nonparanormal, and forest graphical models we make the following definition.

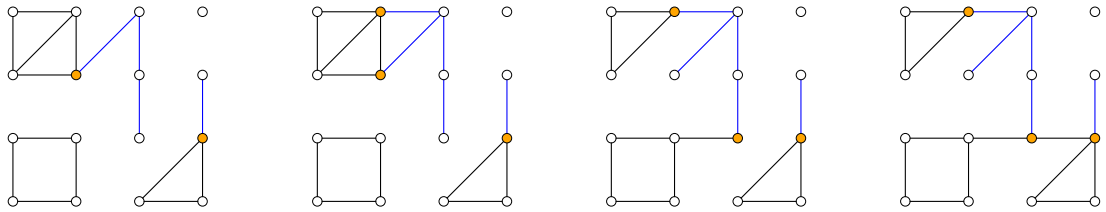
Definition 5.2.1. *A **blossom tree** on a node set $V = \{1, 2, \ldots, d\}$ is a graph $G = (V, E)$, together with a decomposition of the edge set E as $E = F \cup \{\cup_{B \in \mathcal{B}} B\}$ satisfying the following properties:*

1. *F is acyclic;*
2. *$V(B) \cap V(B') = \varnothing$, for $B, B' \in \mathcal{B}$ with $B \neq B'$, where $V(B)$ denotes the vertex set of B;*
3. *$|V(B) \cap V(F)| \leq 1$ for each $B \in \mathcal{B}$;*
4. *$V(F) \cup \bigcup_{\mathcal{B}} V(B) = V$.*

*The subgraphs $B \in \mathcal{B}$ are called **blossoms**. The unique node $\rho(B) \in V(B) \cap V(F)$, which may be empty, is called the **pedicel** of the blossom. The set of pedicels is $\mathcal{P}(F) \subset V(F)$.*

Property 1 says that the set of edges F forms a union of trees—a forest. Property 2 says that distinct blossoms share no vertices or edges in common. Property 3 says that each blossom is connected to at most one tree node. Property 4 says that every node in the graph is either in a tree or a blossom. Note that the blossoms are not required to be connected, but must have at most one vertex in common with the forest—this is the pedicel node.

Figure 5.1 displays four graphs, among which graphs (a) and (c) correspond to blossom trees, while graphs (b) and (d) violate the restriction that each blossom has only a single pedicel, or attachment to a tree.



(a) blossom tree (b) violation (c) blossom tree (d) violation

Figure 5.1: Four graphs, two blossom trees. The tree edges are colored blue, the blossom edges are colored black, and pedicels are orange.

Suppose $p(x) = p(x_1, \ldots, x_d)$ is the density of a distribution that has an independence graph given by a blossom tree $F \cup \{\cup_{\mathcal{B}} B\}$. Then from the blossom tree properties we have that

$$p(X) = p(X_{V(F)}) \prod_{B \in \mathcal{B}} p(X_{V(B)} \,|\, X_{V(F)}) \tag{5.2.1}$$

$$= p(X_{V(F)}) \prod_{B \in \mathcal{B}} p(X_{V(B)} \,|\, X_{\rho(B)}) \tag{5.2.2}$$

$$= p(X_{V(F)}) \prod_{B \in \mathcal{B}} \frac{p(X_{V(B)})}{p(X_{\rho(B)})} \tag{5.2.3}$$

$$= \prod_{(s,t) \in F} \frac{p(X_s, X_t)}{p(X_s) p(X_t)} \prod_{s \in V(F)} p(X_s) \prod_{B \in \mathcal{B}} \frac{p(X_{V(B)})}{p(X_{\rho(B)})} \tag{5.2.4}$$

$$= \prod_{(s,t) \in F} \frac{p(X_s, X_t)}{p(X_s) p(X_t)} \prod_{s \in V(F) \setminus \mathcal{P}(F)} p(X_s) \prod_{B \in \mathcal{B}} p(X_{V(B)}). \tag{5.2.5}$$

The first equality follows from disjointness of the blossoms. The second equality follows from the existence of a single pedicel node attaching the blossom to a tree. The fourth equality follows from the standard factorization of forests, and the last equality follows from the fact that each non-empty pedicel for a blossom is unique. We call the set of distributions that

factor in this way the family of *blossom tree graphical models*.

A key property of the nonparanormal [31] is that the single node marginal probabilities $p(X_s)$ are arbitrary. This allows us to form graphical models where each blossom distribution satisfies $X_{V(B)} \sim \mathcal{NPN}(\mu_B, \Sigma_B, f_B)$, while enforcing that the single node marginal of the pedicel $\rho(B)$ agrees with the marginals of this node defined by the forest. This allows us to define and estimate distributions that are consistent with the factorization (5.2.5).

5.2.2 Estimation Methods

Let $X^{(1)}, \ldots, X^{(n)}$ be n i.i.d. $\mathbb{R}^d$-valued data vectors sampled from density $p^*(x)$ where $X^{(l)} = (X_1^{(l)}, \ldots, X_d^{(l)})$. Our goal is to derive a method for high-dimensional undirected graph estimation and density estimation, using a family of semiparametric estimators based on the blossom tree structure. Let F_B denote the blossom tree structure $F \cup \{\cup_{\mathcal{B}} B\}$. Our estimation procedure is the following.

First, randomly partition $X^{(1)}, \ldots, X^{(n)}$ into two data sets $\mathcal{D}_1$ and $\mathcal{D}_2$ of sample size n_1 and n_2. Then apply the following steps.

1. Using $\mathcal{D}_1$, estimate the bivariate densities $p^*(x_i, x_j)$ using kernel density estimators. Also, estimate the covariance for each pair of variables. Apply Kruskal's algorithm on the estimated pairwise negentropy matrix to construct a family of forests $\{\widehat{F}^{(k)}\}$ with $k = 0, \ldots, d-1$ edges;
2. Using $\mathcal{D}_1$, for each forest $\widehat{F}^{(k)}$ obtained in Step 1, build the blossom tree-structured graph $\widehat{F}^{(k)}_{\widehat{B}}$. The forest structure $\widehat{F}^{(k)}$ is then modeled by kernel density estimators, while each blossom structure $\widehat{B}_i^{(k)}$ is modeled by the graphical lasso or nonparanormal. A family of graphs is obtained by computing regularization paths for the blossoms;
3. Using $\mathcal{D}_2$, choose $\widehat{F}^{(\widehat{k})}_{\widehat{B}}$ from this family of blossom tree models that maximizes the held-out log-likelihood.

The details of each step are presented below.

Step 1: Construct a family of forests

In information theory and statistics, negentropy is used as a measure of distance to normality. The negentropy is zero for Gaussian densities and is always nonnegative. The negentropy between variables X_i and X_j is defined as

$$J(X_i; X_j) = H(\phi(x_i, x_j)) - H(p^*(x_i, x_j)), \tag{5.2.6}$$

where $H(\cdot)$ denotes the differential entropy of a density, and $\phi(x_i, x_j)$ is an Gaussian density with the same mean and covariance matrix as $p^*(x_i, x_j)$.

Kruskal's algorithm [23] is a greedy algorithm to find a maximum weight spanning tree of a weighted graph. At each step it includes an edge connecting the pair of nodes with the maximum weight among all unvisited pairs, if doing so does not form a cycle. The algorithm also results in the best k-edge weighted forest after $k < d$ edges have been included.

In our setting, we define the weight $w(i, j)$ of nodes i and j as the negentropy between X_i and X_j, and use Kruskal's algorithm to build the maximum weight spanning forest $\widehat{F}^{(k)}$ with k edges where $k < d$. In such a way, the pairs of nodes that are less likely to be a bivariate Gaussian are included in the forest and then are modeled nonparametrically.

Since the true density p^* is unknown, we replace the population negentropy $J(X_i; X_j)$ by the estimate

$$\widehat{J}_{n_1}(X_i; X_j) = H(\widehat{\phi}_{n_1}(x_i, x_j)) - \widehat{H}(\widehat{p}_{n_1}(x_i, x_j)), \tag{5.2.7}$$

where $\widehat{\phi}_{n_1}(x_i, x_j)$ is an estimate of the Gaussian density $\phi(x_i, x_j)$ for X_i and X_j using $\mathcal{D}_1$, $\widehat{p}_{n_1}(x_i, x_j)$ is a bivariate kernel density estimate for X_i and X_j, and $\widehat{H}(\cdot)$ denotes the empirical differential entropy. In particular, let Σ^{ij} be the covariance of X_i and X_j. Denote $\widehat{\Sigma}^{ij}_{n_1}$ as the empirical covariance of X_i and X_j based on $\mathcal{D}_1$, then the plug-in estimate

$$H(\widehat{\phi}_{n_1}(x_i, x_j)) = 1 + \log(2\pi) + \frac{1}{2}\text{logdet}(\widehat{\Sigma}^{ij}_{n_1}). \tag{5.2.8}$$

Let $K(\cdot)$ be a univariate kernel function. Given an evaluation point (x_i, x_j), the bivariate kernel density estimate for (X_i, X_j) based on observations $\{X_i^{(l)}, X_j^{(l)}\}_{l\in\mathcal{D}_1}$ is given by

$$\widehat{p}_{n_1}(x_i, x_j) = \frac{1}{n_1}\sum_{l\in\mathcal{D}_1}\frac{1}{h_{2i}h_{2j}}K\left(\frac{X_i^{(l)} - x_i}{h_{2i}}\right)K\left(\frac{X_j^{(l)} - x_j}{h_{2j}}\right), \tag{5.2.9}$$

where h_{2i} and h_{2j} are bandwidth parameters for (X_i, X_j). Given an evaluation point x_i, the univariate kernel density estimate for X_i based on observations $\{X_i^{(l)}\}_{l\in\mathcal{D}_1}$ is given by

$$\widehat{p}_{n_1}(x_i) = \frac{1}{n_1}\sum_{l\in\mathcal{D}_1}\frac{1}{h_{1i}}K\left(\frac{X_i^{(l)} - x_i}{h_{1i}}\right), \tag{5.2.10}$$

where h_{1i} is the bandwidth parameter for X_i. To compute the empirical differential entropy $\widehat{H}(\widehat{p}_{n_1}(x_i, x_j))$, we numerically evaluate a two-dimensional integral by choosing m evaluation points $x_k^{(c_1)} < x_k^{(c_2)} < \cdots < x_k^{(c_m)}$ for variable X_k, where $k = i$ or j. Then

$$\widehat{H}(\widehat{p}_{n_1}(x_i, x_j)) = -\frac{\widehat{R}_{n_1}(X_i)\widehat{R}_{n_1}(X_j)}{(m-1)^2}\sum_{u=1}^{m}\sum_{v=1}^{m}\widehat{p}_{n_1}(x_i^{(c_u)}, x_j^{(c_v)})\log\widehat{p}_{n_1}(x_i^{(c_u)}, x_j^{(c_v)}), \tag{5.2.11}$$

where $\widehat{R}_{n_1}(X_k)$ indicates the range of observations $\{X_k^{(l)}\}_{l\in\mathcal{D}_1}$, where $k = i$ or j.

Once the estimated negentropy matrix $\left[\widehat{J}_{n_1}(X_i; X_j)\right]_{d\times d}$ is obtained, we apply Kruskal's algorithm to construct a family of forests $\{\widehat{F}^{(k)}\}_{k=0\ldots d-1}$.

Step 2: Build and model the blossom tree graphs

Suppose that we have a forest-structured graph F with $|V(F)| < d$ vertices. Then for each remaining non-forest node, we need to determine which blossom it belongs to. We exploit the following basic fact.

Proposition 5.2.1. *Suppose that $X \sim p$ follows a blossom tree graphical model with forest F. Let $i \notin V(F)$ and $s \in V(F)$. Then node i is not in a blossom attached to tree node s if*

and only if

$$X_i \perp\!\!\!\perp X_s \,|\, X_t \quad \textit{for some node } t \in V(F) \textit{ such that } (s,t) \in E(F). \tag{5.2.12}$$

We use this property, together with a measure of partial correlation, in order to partition the non-forest nodes into blossoms. Partial correlation measures the degree of association between two random variables, with the effect of a set of controlling variables removed. Traditionally, the partial correlation between variables X_i and X_s given a controlling variable X_t is the correlation between the residuals $\epsilon_{i\setminus t}$ and $\epsilon_{s\setminus t}$ resulting from the linear regression of X_i with X_t and of X_s with X_t, respectively. However, if the underlying joint Gaussian or nonparanormal assumption is not satisfied, linear regression cannot remove all of the effects of the controlling variable.

We thus use a nonparametric version of partial correlation. Following [3], suppose $X_i = g(X_t) + \epsilon_{i\setminus t}$ and $X_s = h(X_t) + \epsilon_{s\setminus t}$, for certain functions g and h such that $\mathbb{E}(\epsilon_{i\setminus t} \,|\, X_t) = 0$ and $\mathbb{E}(\epsilon_{s\setminus t} \,|\, X_t) = 0$. Define the *nonparametric partial correlation* as

$$\rho_{is\cdot t} = \mathbb{E}(\epsilon_{i\setminus t}\epsilon_{s\setminus t}) \Big/ \sqrt{\mathbb{E}(\epsilon_{i\setminus t}^2)\,\mathbb{E}(\epsilon_{s\setminus t}^2)}. \tag{5.2.13}$$

It is shown in [3] that if $X_i \perp\!\!\!\perp X_s \,|\, X_t$, then $\rho_{is\cdot t} = 0$. We thus conclude the following.

Proposition 5.2.2. *If $\rho_{is\cdot t} \neq 0$ for all t such that $(s,t) \in E(F)$, node i is in a blossom attached to node s.*

Let $\widehat{g}$ and $\widehat{h}$ be local polynomial estimators of g and h, and $\widehat{\epsilon}_{i\setminus t}^{(l)} = X_i^{(l)} - \widehat{g}(X_t^{(l)})$, $\widehat{\epsilon}_{s\setminus t}^{(l)} = X_s^{(l)} - \widehat{h}(X_t^{(l)})$ for any $l \in \mathcal{D}_1$, then an estimate of $\rho_{is\cdot t}$ is given by

$$\widehat{\rho}_{is\cdot t} = \sum_{l\in\mathcal{D}_1} (\widehat{\epsilon}_{i\setminus t}^{(l)}\,\widehat{\epsilon}_{s\setminus t}^{(l)}) \Big/ \sqrt{\sum_{l\in\mathcal{D}_1} (\widehat{\epsilon}_{i\setminus t}^{(l)})^2 \sum_{l\in\mathcal{D}_1} (\widehat{\epsilon}_{s\setminus t}^{(l)})^2}. \tag{5.2.14}$$

Based on Proposition 5.2.2, for each forest $\widehat{F}^{(k)}$ obtained in Step 1, we then assign each

non-forest node i to the blossom with the pedicel given by

$$\widehat{s}_i = \underset{s \in V(\widehat{F}^{(k)})}{\operatorname{argmax}} \min_{\{t:\,(s,t)\in E(\widehat{F}^{(k)})\}} |\widehat{\rho}_{is\cdot t}|. \tag{5.2.15}$$

After iterating over all non-forest nodes, we obtain a blossom tree-structured graph $\widehat{F}^{(k)}_{\widehat{B}}$. Then the forest structure is nonparametrically modeled by the bivariate and univariate kernel density estimations, while each blossom structure is modeled with the graphical lasso or nonparanormal. In particular, when $k = 0$ that there is no forest node, our method is reduced to modeling the entire graph by the graphical lasso or nonparanormal.

Alternative testing procedures based on nonparametric partial correlations could be adopted for partitioning nodes into blossoms. However, these methods might have high computational cost, and low power for small sample sizes.

Note that while each non-forest node is associated with a pedicel in this step, after graph estimation for the blossoms, the node may well become disconnected from the forest.

Step 3: Optimize the blossom tree graphs

The full blossom tree graph $\widehat{F}^{(d-1)}_{\widehat{B}}$ obtained in Steps 1 and 2 might result in overfitting in the density estimate. Thus we need to choose an optimal graph with the number of forest edges $k \leq d-1$. Besides, the tuning parameters involved in the fitting of each blossom by the graphical lasso or nonparanormal also induce a bias-variance tradeoff.

To optimize the blossom tree structures over $\{\widehat{F}^{(k)}_{\widehat{B}}\}_{k=0\ldots d-1}$, we choose the complexity parameter $\widehat{k}$ as the one that maximizes the log-likelihood on $\mathcal{D}_2$:

$$\widehat{k} = \underset{k\in\{0,\ldots,d-1\}}{\operatorname{argmax}} \frac{1}{n_2} \sum_{l\in\mathcal{D}_2} \log \left(\prod_{(i,j)\in E(\widehat{F}^{(k)})} \frac{\widehat{p}_{n_1}(X_i^{(l)}, X_j^{(l)})}{\widehat{p}_{n_1}(X_i^{(l)})\widehat{p}_{n_1}(X_j^{(l)})} \cdot \prod_{s\in V(\widehat{F}^{(k)})\setminus\mathcal{P}(\widehat{F}^{(k)})} \widehat{p}_{n_1}(X_s^{(l)}) \prod_{i=1}^{k} \widehat{\phi}_{n_1}^{\lambda_i^{(k)}}\left(X^{(l)}_{V(\widehat{B}_i^{(k)})}\right) \right), \tag{5.2.16}$$

where $\widehat{\phi}_{n_1}^{\lambda_i^{(k)}}$ is the density estimate for blossoms by the graphical lasso or nonparanormal, with the tuning parameter $\lambda_i^{(k)}$ selected to maximize the held-out log-likelihood. That is, the complexity of each blossom is also optimized on $\mathcal{D}_2$.

Thus the final blossom tree density estimator is given by

$$\widehat{p}_{\widehat{F}^{(\widehat{k})}_{\widehat{B}}}(x) = \prod_{(i,j)\in E(\widehat{F}^{(\widehat{k})})} \frac{\widehat{p}_{n_1}(x_i, x_j)}{\widehat{p}_{n_1}(x_i)\widehat{p}_{n_1}(x_j)} \prod_{s\in V(\widehat{F}^{(\widehat{k})})\setminus\mathcal{P}(\widehat{F}^{(\widehat{k})})} \widehat{p}_{n_1}(x_s) \prod_{i=1}^{\widehat{k}} \widehat{\phi}_{n_1}^{\lambda_i^{(\widehat{k})}}(x_{\widehat{B}_i^{(\widehat{k})}}). \quad (5.2.17)$$

In practice, Step 3 can be carried out simultaneously with Steps 1 and 2. Whenever a new edge is added to the current forest in Kruskal's algorithm, the blossoms are re-constructed and re-modeled. Then the held-out log-likelihood of the obtained density estimator can be immediately computed. In addition, since there are no overlapping nodes between different blossoms, the sparsity tuning parameters are selected separately for each blossom, which reduces the computational cost considerably.

5.3 Statistical Properties

Here we discuss statistical properties of the estimates of negentropy and nonparametric partial correlation. For any pair of variables X_i and X_j, let $(X_i^{(1)}, X_j^{(1)}), \ldots, (X_i^{(n)}, X_j^{(n)})$ be n i.i.d. $\mathbb{R}^2$-valued data vectors sampled from $p^*(x_i, x_j)$. Suppose $\mathcal{X} = [0,1]^2$ is the domain of (X_i, X_j).

Assumption 5.3.1. *For any pair of variables X_i and X_j, assume the density $p^*(x_i, x_j)$ belongs to a second-order Hölder class and is bounded away from zero and infinity.*

Assumption 5.3.2. *For any sequence converging to a boundary point, suppose the density $p^*(x_i, x_j)$ of any pair of variables X_i and X_j has vanishing first-order partial derivatives.*

Assumption 5.3.3. *Assume the kernel function $K(\cdot)$ is nonnegative and has a bounded support $[-1, 1]$ with $\int_{-1}^{1} K(x)dx = 1$ and $\int_{-1}^{1} xK(x)dx = 0$.*

Under the assumptions, the authors in [32] prove an exponential concentration inequality based on a clipped "mirror image" kernel density estimator $\widetilde{p}_{n,h}(x_i, x_j)$, where h is the bandwidth parameter. Combined with bounds on the log determinant of sample covariance matrices presented in [8], we obtain the following.

Theorem 5.3.1. *Under Assumptions 5.3.1–5.3.3, for any $\epsilon > 0$, if we choose the bandwidth according to $h \asymp n^{-1/4}$, then there exists a constant N such that if $n > N$,*

$$\mathbb{P}(|\widetilde{J}_n(X_i; X_j) - J(X_i; X_j)| > \epsilon) \leq 3 \exp\left(-\frac{n\epsilon^2}{144\kappa^2}\right), \tag{5.3.1}$$

for any $i, j \in \{1, \ldots, d\}$ with $i \neq j$, where

$$\widetilde{J}_n(X_i; X_j) = 1 + \log(2\pi) + \frac{1}{2}\text{logdet}(\widehat{\Sigma}_n^{ij}) - \frac{3}{2(n-1)} - H(\widetilde{p}_{n,h}(x_i, x_j)), \tag{5.3.2}$$

and $\kappa = \max\{|\log(\min_{\mathcal{X}} p^*(x_i, x_j))|, |\log(\max_{\mathcal{X}} p^*(x_i, x_j))|\} + 1$.

The advantage of having this exponential inequality is that we can guarantee accurate estimation of the negentropy for many pairs of variables simultaneously. It is easy to see that the above theorem implies a $\sqrt{n}$-rate of convergence in mean absolute error $\mathbb{E}(|\widetilde{J}_n(X_i; X_j) - J(X_i; X_j)|) = O_P(n^{-1/2})$.

To justify the estimation of nonparametric partial correlation, it is shown in [3] that, for some $p_1, p_2 > 0$,

$$\sqrt{n}(\widehat{\rho}_{is\cdot t} - \rho_{is\cdot t}) = \sqrt{n}(r_{is\cdot t} - \rho_{is\cdot t}) + O_P(n^{-\min\{p_1, p_2\}}), \tag{5.3.3}$$

where $r_{is\cdot t}$ is the empirical correlation coefficient based on the errors $\epsilon_{i\backslash t}^{(l)} = X_i^{(l)} - g(X_t^{(l)})$, $\epsilon_{s\backslash t}^{(l)} = X_s^{(l)} - h(X_t^{(l)})$ for any $l \in \mathcal{D}_1$,

$$r_{is\cdot t} = \sum_{l \in \mathcal{D}_1} (\epsilon_{i\backslash t}^{(l)} \epsilon_{s\backslash t}^{(l)}) \Big/ \sqrt{\sum_{l \in \mathcal{D}_1} (\epsilon_{i\backslash t}^{(l)})^2 \sum_{l \in \mathcal{D}_1} (\epsilon_{s\backslash t}^{(l)})^2}, \tag{5.3.4}$$

and $n^{p_1}(\widehat{g}(x) - g(x)) = O_P(1)$, $n^{p_2}(\widehat{h}(x) - h(x)) = O_P(1)$.

This result justifies the estimation procedure of nonparametric partial correlation.

5.4 Experimental Results

5.4.1 Analysis of Simulated Data

Here we present numerical results based on simulations. We compare the blossom tree density estimator with the graphical lasso [19] and forest density estimator [33]. Since the graphical lasso always results in a biased estimation due to the ℓ_1 penalization, we use a two-step procedure instead: in the first step, a sparse precision matrix is obtained by the graphical lasso; in the second step, a Gaussian model is refitted, enforcing the same sparsity pattern achieved in the first step [33]. To evaluate the performance of these estimators, we compute and compare the log-likelihood of each method on held-out data.

We simulate data in dimension $d = 80$ which are consistent with an undirected graph. We generate multivariate non-Gaussian data set using a sequence of mixtures of two Gaussian distributions with contrary correlation and equal weights. Then a subset of variables are chosen to generate the blossoms that are distributed as multivariate Gaussians. We sample $n_1 = n_2 = 400$ data points from this synthetic distribution.

A typical run showing the held-out log-likelihood and estimated graphs is provided in Figures 5.2 and 5.3. The term "trunk" is used to represent the edge added to the forest in a blossom tree graph. We can see that the blossom tree density estimator is superior to other methods in terms of generalization performance. In particular, the graphical lasso is unable to uncover the edges that are generated nonparametrically. This is expected, since different blossoms have zero correlations among each other and are thus regarded as independent by the algorithm of graphical lasso. For the modeling of the variables that are contained in a blossom and are thus multivariate Gaussian distributed, there is an efficiency loss in the forest density estimator, compared to the graphical lasso. This illustrates the advantage of

blossom tree density estimator. As is seen from the number of selected edges by each method shown in Figure 5.2, the blossom tree density estimator selects a graph with a similar sparsity pattern as the true graph.

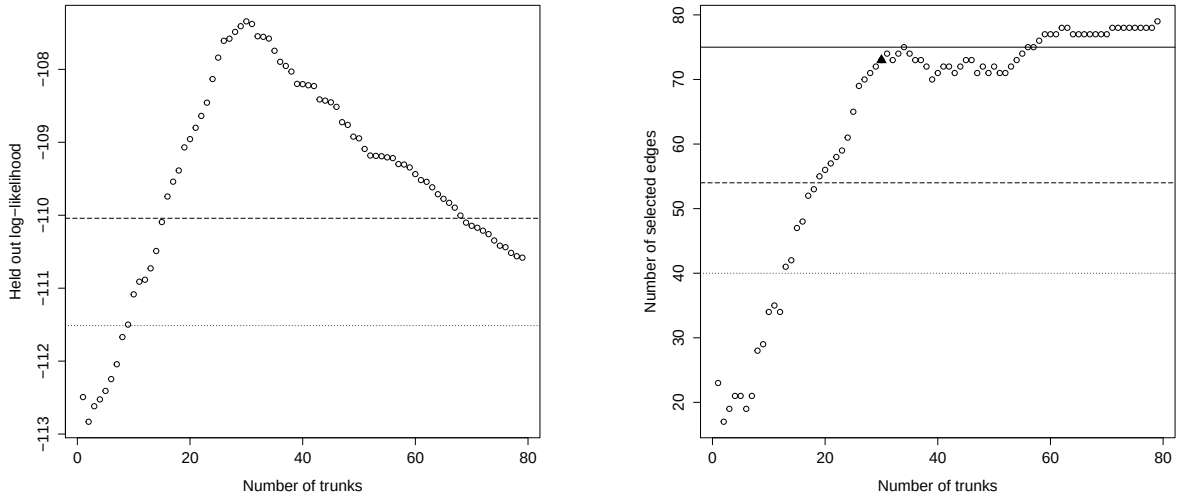
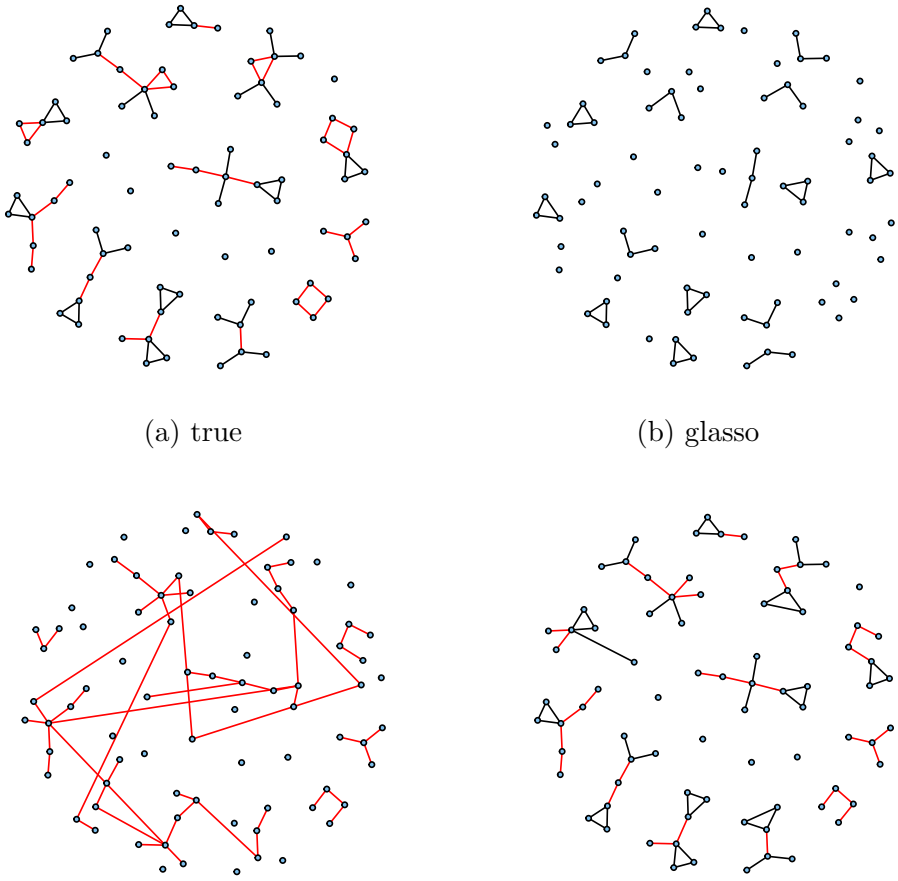


Figure 5.2: Results on simulations. Left: Held-out log-likelihood of the graphical lasso (dotted line), forest density estimator (dashed line), and blossom tree density estimator (circles); Right: Number of selected edges by these methods. The solid line indicates the number of edges in the true graph, and the solid triangle indicates the best blossom tree graph. The first circle for blossom tree refers to the 1-trunk case.

It is also of our interest to evaluate the performance of these methods when the number of nodes within each blossom $|V(B_i)|$ varies. We generate non-Gaussian data for 5 variables with a Toeplitz dependence structure using Student's t-copula, and simulate multivariate Gaussian data for the blossoms connected to each of these 5 nodes. The number of nodes within each blossom are set to be equal across different blossoms and varies from 0 to 6. Table 5.1 compares the held-out log-likelihood for each method. We can see that when $|V(B_i)|$ is small, the forest density estimator has the best generalization performance. As $|V(B_i)|$ increases, the blossom tree density estimator dominates the other two methods. When $|V(B_i)|$ becomes large, the graphical lasso estimator performs better than the forest density estimator, although the blossom tree density estimator is still the best.



(a) true (b) glasso

(c) forest (d) blossom tree

Figure 5.3: Results on simulations. Graph (a) corresponds to the true graph. Graphs (b), (c) and (d) correspond to the estimated graphs by the graphical lasso, forest density estimator, and blossom tree density estimator, respectively. The tree edges are colored red, and the blossom edges are colored black.

Table 5.1: Results on simulations. Held-out log-likelihood of the graphical lasso, forest density estimator, and blossom tree density estimator as a function of the number of nodes within each blossom $|V(B_i)|$, which is set to be equal across different blossoms and varies from 0 to 6.

$\lvert V(B_i)\rvert =$	0	1	2	3	4	5	6
glasso	-6.61	-13.50	-20.66	-27.84	-34.69	-41.25	-48.19
forest	-6.11	-13.04	-20.19	-27.35	-34.55	-41.23	-48.42
blossom tree	-6.32	-13.08	-20.05	-27.21	-34.20	-40.79	-47.84

More simulations are conducted to evaluate the probability that a non-forest node could be assigned to the right blossom using our procedure.

Suppose variables X_s, X_t and X_i satisfy the following distributions:

$$(X_s, X_t) \sim \begin{cases} \mathcal{N}\left(\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1 & 0.8\\0.8 & 1\end{bmatrix}\right) & \text{with probability } 0.5 \\ \mathcal{N}\left(\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1 & -0.8\\-0.8 & 1\end{bmatrix}\right) & \text{with probability } 0.5 \end{cases}, \tag{5.4.1}$$

$$(X_i, X_s) \sim \mathcal{N}\left(\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1 & r\\r & 1\end{bmatrix}\right). \tag{5.4.2}$$

Assume a forest structure F that only contains one edge (s, t) between node s and node t. We would like to evaluate the success probability that another node i is assigned to the blossom attached to node s rather than node t using the procedure described in Section 5.2.

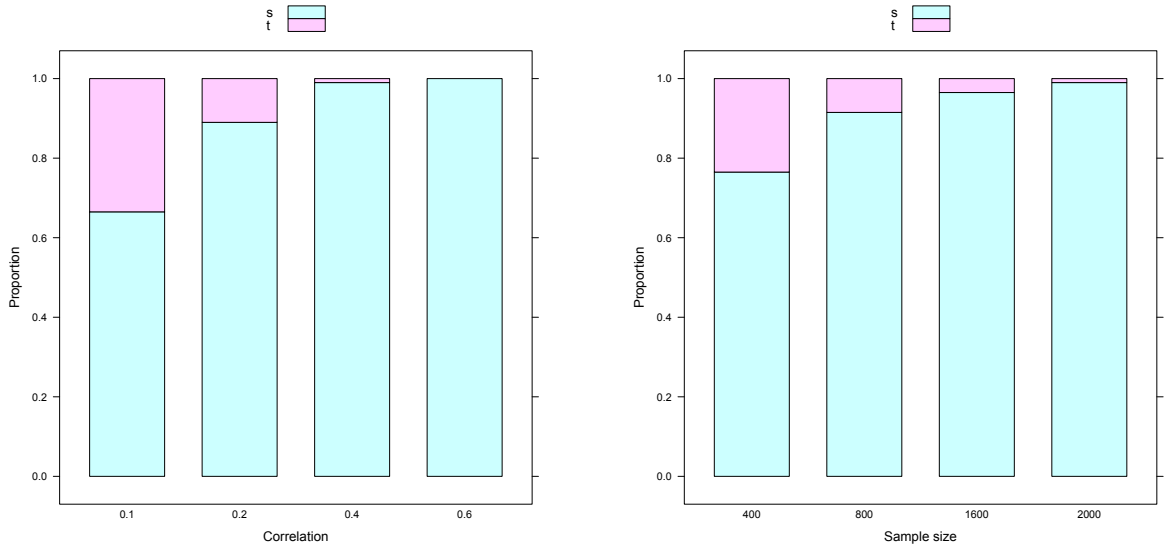


Figure 5.4: Results on simulations. Proportions of times among 200 replications that node i was classified to the blossom attached to forest node s or t, as a function of the correlation coefficient r with sample size $n = 200$ (Left), or as a function of sample size with correlation $r = 0.1$ (Right).

Figure 5.4 reports the proportions of times among 200 replications that node i was classified to the blossom attached to forest node s or t, as a function of the correlation coefficient r or sample size. It is as expected that the probability that node i is successfully assigned to the blossom connected to node s becomes larger as the correlation or sample size increases.

5.4.2 Analysis of Cell Signalling Data

We analyze a flow cytometry dataset on $d = 11$ proteins from [45]. A subset of $n = 853$ cells were chosen. A nonparanormal transformation was estimated and the observations, for each variable, were replaced by their respective normal scores, subject to a Winsorized truncation [31]. We study the associations among the proteins using the graphical lasso, forest density estimator, and blossom tree forest density estimator.

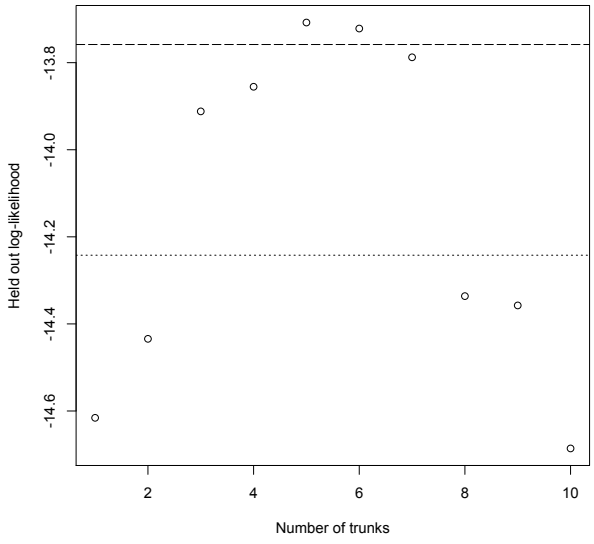
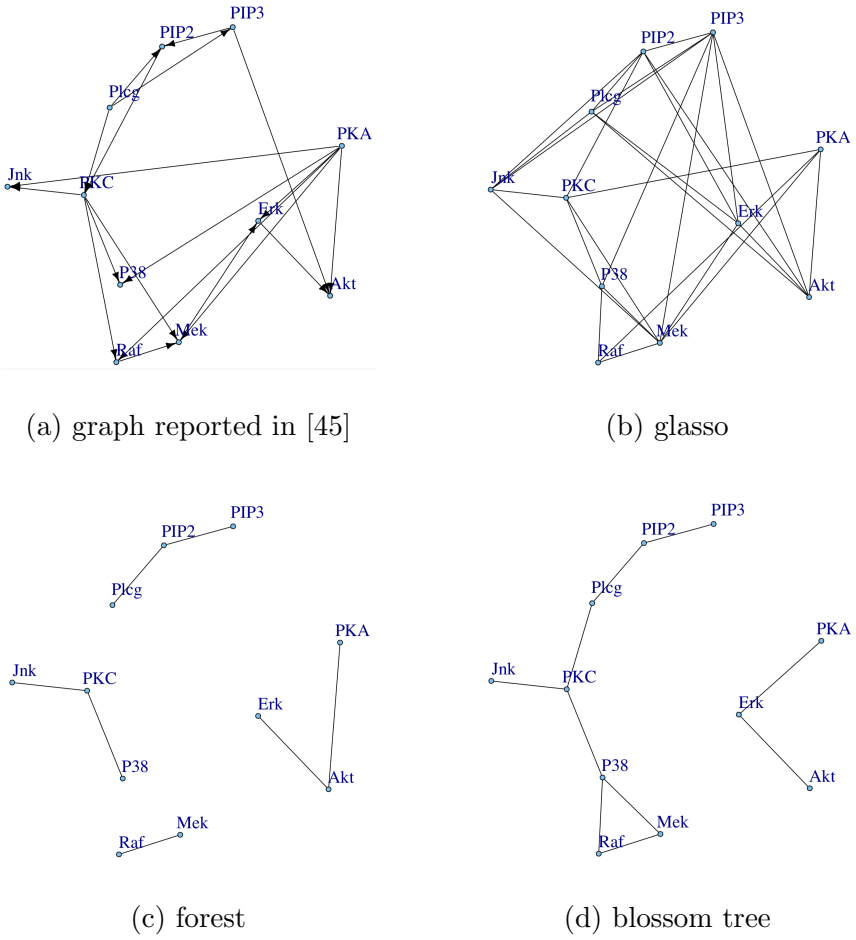


Figure 5.5: Results on signalling data. Held-out log-likelihood of the graphical lasso (dotted line), forest density estimator (dashed line), and blossom tree density estimator (circles).

Figure 5.5 shows the results of the held-out log-likelihood of the graphical lasso, forest density estimator, and blossom tree density estimator. The maximum held-out log-likelihood

for glasso, forest, and blossom tree are -14.3, -13.8, and -13.7, respectively. We can see that blossom tree is slighter better than forest in terms of the generalization performance, both of which outperform glasso. Results of estimated graphs are provided in Figure 5.6. When the maximum of held-out log-likelihood curve is achieved, glasso selects 28 edges, forest selects 7 edges, and blossom tree selects 10 edges. The two graphs uncovered by forest and blossom tree agree on most edges, although the latter contains cycles.



(a) graph reported in [45]

(b) glasso

(c) forest

(d) blossom tree

Figure 5.6: Results on cell signalling data. Graph (a) refers to the fitted graph reported in [45]. Graphs (b), (c) and (d) correspond to the estimated graphs by the graphical lasso, forest density estimator, and blossom tree density estimator, respectively.

5.5 Conclusion

We have proposed a combination of tree-based graphical models and Gaussian graphical models to form a new nonparametric approach for high-dimensional data. Blossom tree models relax the normality assumption and increase statistical efficiency by modeling the forest with nonparametric kernel density estimators and modeling each blossom with the graphical lasso or nonparanormal. Our experimental results indicate that this method can be a practical alternative to standard approaches to graph and density estimation.

5.6 Proofs

Proof of Proposition 5.2.1.

Proof. If node i is not in a blossom attached to node s, then either node i is not attached to any forest node (its pedicel is $\varnothing$), or there exists some forest node t such that node i belongs to the blossom attached to t. In either case, we have $X_i \perp\!\!\!\perp X_s \,|\, X_t$. For the other direction, assume on the contrary that node i is in a blossom attached to node s. Then nodes i and s are not independent conditional on node t. □

Proof of Proposition 5.2.2.

Proof. Suppose node i is not in a blossom attached to forest node s. Then according to Proposition 5.2.1, there exists some node t such that $(s,t) \in E(F)$, satisfying $X_i \perp\!\!\!\perp X_s \,|\, X_t$. This implies $\rho_{is \cdot t} = 0$, contradicting the assumption. □

Proof of Theorem 5.3.1.

Proof. The authors in [32] show that under Assumptions 5.3.1–5.3.3, for any $\epsilon > 0$, if we

choose the bandwidth according to $h \asymp n^{-1/4}$, then if n is large enough,

$$\mathbb{P}(|H(\widetilde{p}_n(x_i, x_j)) - H(p^*(x_i, x_j))| > \epsilon) \leq 2 \exp\left(-\frac{n\epsilon^2}{36\kappa^2}\right). \tag{5.6.1}$$

A limiting law of the log determinant of the sample covariance matrix is presented in [8], where as $n \to \infty$

$$\frac{\text{logdet}(\widehat{\Sigma}_n^{ij}) - 3/(n-1) - \text{logdet}(\Sigma_n^{ij})}{2/\sqrt{n-1}} \xrightarrow{L} \mathcal{N}(0, 1). \tag{5.6.2}$$

Then if n is large enough, we obtain

$$\mathbb{P}(|\text{logdet}(\widehat{\Sigma}_n^{ij}) - 3/(n-1) - \text{logdet}(\Sigma_n^{ij})| > \epsilon) \leq \sqrt{\frac{8}{(n-1)\pi\epsilon^2}} \exp\left(-\frac{(n-1)\epsilon^2}{8}\right). \tag{5.6.3}$$

Following from the union bound, there exists a constant N such that for all $n > N$,

$$\mathbb{P}(|\widetilde{J}_n(X_i; X_j) - J(X_i; X_j)| > \epsilon) \leq \sqrt{\frac{8}{(n-1)\pi\epsilon^2}} \exp\left(-\frac{(n-1)\epsilon^2}{8}\right) + 2\exp\left(-\frac{n\epsilon^2}{144\kappa^2}\right) \tag{5.6.4}$$

$$\leq 3\exp\left(-\frac{n\epsilon^2}{144\kappa^2}\right). \tag{5.6.5}$$

This completes the proof. □

CHAPTER 6

CONCLUSION AND DISCUSSION

In this thesis, we develop new methods and theories for graphical model structure learning and density estimation in high dimensions, and evaluate the performance of estimators using synthetic and real-world datasets.

In Chapter 2, we propose and analyze a new approach for precision matrix estimation via aggregation. We study the risk of this aggregation estimator and show that it is comparable to the risk of the best estimator based on a single graph. In Chapter 3, we investigate robust neighborhood selection and graphical lasso methods for Gaussian graphical models in the presence of corrupted data. In Chapter 4, we explore tree-based density estimation methods given topological constraints on graph structures—scale-free networks and shared edges among multiple graphs. In Chapter 5, we combine tree-based graphical models and Gaussian graphical models to form a new nonparametric approach for high dimensional data. Our analysis and experimental results indicate that the newly proposed methods in this thesis can be powerful alternatives to standard approaches for graph estimation and density estimation.

Here we summarize the contributions of this thesis in terms of methods, theory, and applications as follows:

- *Methods.* We develop new methods for graphical model structure learning and density estimation in high dimensions;
- *Theory.* We investigate statistical properties of nonparametric inference, estimation consistency, and model selection/graph recovery consistency;
- *Applications.* We evaluate the performance of the proposed estimators for learning graphical models using scientific datasets.

The results of this thesis lead to many future directions. We summarize some as follows:

- *Methods*. The Gaussian assumptions for graphical models might be unrealistic and unnecessary in some applications. Thus it is interesting to continue a line of work to go beyond the Gaussian assumptions and develop new nonparametric families of graphical models. Also, with the rapid advances in computing technologies during the last decade, taking advantage of parallel computing could make computationally expensive methods feasible. Therefore it would be fruitful to investigate the interaction of parallel computing with high dimensional graphical model learning.

- *Theory*. It is important to study the convergence guarantees of global optimization with non-convex constraints. In Chapter 4, we employ the minorization-maximization algorithm to solve the optimization problems raised in learning forest-based graphical models with structural constraints. However, the algorithm is lack of any convergence guarantees of global maximum. Moreover, theoretical results are needed to quantify the risk of model misspecification.

- *Applications*. The proposed methods in this thesis could have applications in more scientific areas and we expect to validate the usefulness of graphical models in more real-world datasets. In many modern fields the datasets usually have very high dimensions and are generated by some unknown complex processes, which pose new challenges in modeling and analysis. It is of importance to demonstrate how graphical modeling will play a significant role and inspire follow-up studies.

REFERENCES

[1] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47, 2002.

[2] Francis R. Bach and Michael I. Jordan. Beyond independent components: trees and clusters. *The Journal of Machine Learning Research*, 4:1205–1233, 2003.

[3] Wicher Bergsma. A note on the distribution of the partial correlation coefficient with nonparametrically estimated marginal regressions. *arXiv:1101.4616*, 2011.

[4] Kush Bhatia, Prateek Jain, and Purushottam Kar. Robust regression via hard thresholding. *arXiv:1506.02428*, 2015.

[5] Peter J. Bickel and Elizaveta Levina. Covariance regularization by thresholding. *The Annals of Statistics*, 36(6):2577–2604, 2008.

[6] Peter J. Bickel and Elizaveta Levina. Regularized estimation of large covariance matrices. *The Annals of Statistics*, 36(1):199–227, 2008.

[7] Tony Cai, Weidong Liu, and Xi Luo. A constrained ℓ_1 minimization approach to sparse precision matrix estimation. *Journal of the American Statistical Association*, 106(494):594–607, 2011.

[8] Tony T. Cai, Tengyuan Liang, and Harrison H. Zhou. Law of log determinant of sample covariance matrix and optimal estimation of differential entropy for high-dimensional Gaussian distributions. *arXiv:1309.0482*, 2013.

[9] Yudong Chen, Constantine Caramanis, and Shie Mannor. Robust sparse regression under adversarial corruption. In *Proceedings of the 30th International Conference on Machine Learning*, 2013.

[10] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.

[11] Joachim Dahl, Vwani Roychowdhury, and Lieven Vandenberghe. Maximum likelihood estimation of Gaussian graphical models: numerical implementation and topology selection. *Preprint*, 2005.

[12] Patrick Danaher, Pei Wang, and Daniela M. Witten. The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(2):373–397, 2014.

[13] Aaron Defazio and Tiberio S. Caetano. A convex formulation for learning scale-free networks via submodular relaxation. In *Advances in Neural Information Processing Systems, 25*, pages 1250–1258, 2012.

[14] Stefano Demarta and Alexander J. McNeil. The t copula and related copulas. *International Statistical Review*, 73(1):111–129, 2005.

[15] Mathias Drton and Michael D. Perlman. Model selection for Gaussian concentration graphs. *Biometrika*, 91(3):591–602, 2004.

[16] Mathias Drton and Michael D. Perlman. Multiple testing and error control in Gaussian graphical model selection. *Statistical Science*, pages 430–449, 2007.

[17] Michael Finegold and Mathias Drton. Robust graphical modeling of gene networks using classical and alternative t-distributions. *The Annals of Applied Statistics*, 5(2A):1057–1080, 2011.

[18] Marcelo Fiori, Pablo Musé, and Guillermo Sapiro. Topology constraints in graphical models. In *Advances in Neural Information Processing Systems, 25*, 2012.

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

[20] Jian Guo, Elizaveta Levina, George Michailidis, and Ji Zhu. Joint estimation of multiple graphical models. *Biometrika*, 98(1):1–15, 2011.

[21] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning*. Springer, 2009.

[22] David R. Hunter and Kenneth Lange. A tutorial on MM algorithms. *The American Statistician*, 58(1):30–37, 2004.

[23] Joseph B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. In *Proceedings of the American Mathematical Society*, volume 7, pages 48–50, 1956.

[24] William H. Kruskal. Ordinal measures of association. *Journal of the American Statistical Association*, 53(284):814–861, 1958.

[25] Jason N. Laska, Mark A. Davenport, and Richard G. Baraniuk. Exact signal recovery from sparsely corrupted measurements through the pursuit of justice. In *Asilomar Conference on Signals, Systems and Computers*, 2009.

[26] Steffen L. Lauritzen. *Graphical models*. Oxford University Press, 1996.

[27] Guillaume Lecué. Optimal rates of aggregation in classification under low noise assumption. *Bernoulli*, 13(4):1000–1022, 2007.

[28] Gilbert Leung and Andrew R. Barron. Information theory and mixing least-squares regressions. *IEEE Transactions on Information Theory*, 52(8):3396–3410, 2006.

[29] Xiaodong Li. Compressed sensing and matrix completion with constant proportion of corruptions. *arXiv:1104.1041*, 2011.

[30] Han Liu, Fang Han, Ming Yuan, John Lafferty, and Larry Wasserman. The nonparanormal skeptic. *arXiv:1206.6488*, 2012.

[31] Han Liu, John Lafferty, and Larry Wasserman. The nonparanormal: semiparametric estimation of high dimensional undirected graphs. *The Journal of Machine Learning Research*, 10:2295–2328, 2009.

[32] Han Liu, Larry Wasserman, and John D. Lafferty. Exponential concentration for mutual information estimation with application to forests. In *Advances in Neural Information Processing Systems, 25*, 2012.

[33] Han Liu, Min Xu, Haijie Gu, Anupam Gupta, John Lafferty, and Larry Wasserman. Forest density estimation. *The Journal of Machine Learning Research*, 12:907–951, 2011.

[34] Qiang Liu and Alexander T. Ihler. Learning scale free networks by reweighted ℓ_1 regularization. In *International Conference on Artificial Intelligence and Statistics*, pages 40–48, 2011.

[35] Zhe Liu and John Lafferty. Blossom tree graphical models. In *Advances in Neural Information Processing Systems, 27*, pages 1458–1465, 2014.

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

[37] Renuka R. Nayak, Michael Kearns, Richard S. Spielman, and Vivian G. Cheung. Coexpression network based on natural variation in human gene expression reveals gene interactions and functions. *Genome Research*, 19(11):1953–1962, 2009.

[38] Renuka R. Nayak, Michael Kearns, Richard S. Spielman, and Vivian G. Cheung. Coexpression network based on natural variation in human gene expression reveals gene interactions and functions. *Genome Research*, 19(11):1953–1962, 2009.

[39] Arkadi Nemirovski. *Topics in non-parametric statistics. Lectures on probability theory and statistics*. Springer, 2000.

[40] Christine Peterson, Francesco C. Stingo, and Marina Vannucci. Bayesian inference of multiple Gaussian graphical models. *Journal of the American Statistical Association*, 110(509):159–174, 2015.

[41] Pradeep Ravikumar, Garvesh Raskutti, Martin J. Wainwright, and Bin Yu. Model selection in Gaussian graphical models: high-dimensional consistency of ℓ_1-regularized MLE. In *Advances in Neural Information Processing Systems*, pages 1329–1336, 2008.

[42] Philippe Rigollet and Alexandre Tsybakov. Exponential screening and optimal rates of sparse estimation. *The Annals of Statistics*, 39(2):731–771, 2011.

[43] Philippe Rigollet and Alexandre B. Tsybakov. Sparse estimation by exponential weighting. *Statistical Science*, 27(4):558–575, 2012.

[44] Adam J. Rothman, Peter J. Bickel, Elizaveta Levina, and Ji Zhu. Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics*, 2:494–515, 2008.

[45] 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(5721):523–529, 2003.

[46] Hokeun Sun and Hongzhe Li. Robust Gaussian graphical modeling via ℓ_1 penalization. *Biometrics*, 68(4):1197–1206, 2012.

[47] Kean Ming Tan, Palma London, Karthik Mohan, Su-In Lee, Maryam Fazel, and Daniela Witten. Learning graphical models with hubs. *The Journal of Machine Learning Research*, 15(1):3297–3331, 2014.

[48] Qingming Tang, Siqi Sun, and Jinbo Xu. Learning scale-free networks by dynamic node specific degree prior. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2247–2255, 2015.

[49] Caroline Uhler. Geometry of maximum likelihood estimation in Gaussian graphical models. *The Annals of Statistics*, 40(1):238–261, 2012.

[50] Roman Vershynin. How close is the sample covariance matrix to the actual covariance matrix? *Journal of Theoretical Probability*, 25(3):655–686, 2012.

[51] Sheng-De Wang, Te-Son Kuo, and Chen-Fa Hsu. Trace bounds on the solution of the algebraic matrix Riccati and Lyapunov equation. *IEEE Transactions on Automatic Control*, 31(7):654–656, 1986.

[52] Joe Whittaker. *Graphical models in applied multivariate statistics*. Wiley Publishing, 2009.

[53] Anja Wille, Philip Zimmermann, Eva Vranová, Andreas Fürholz, Oliver Laule, Stefan Bleuler, Lars Hennig, Amela Prelic, Peter von Rohr, and Lothar Thiele. Sparse graphical Gaussian modeling of the isoprenoid gene network in Arabidopsis thaliana. *Genome Biology*, 5(11):R92, 2004.

[54] Eunho Yang and Aurélie C. Lozano. Robust Gaussian graphical modeling with the trimmed graphical lasso. In *Advances in Neural Information Processing Systems, 28*, 2015.

[55] Ming Yuan. High dimensional inverse covariance matrix estimation via linear programming. *The Journal of Machine Learning Research*, 11:2261–2286, 2010.

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

[57] Bin Zhang and Steve Horvath. A general framework for weighted gene co-expression network analysis. *Statistical Applications in Genetics and Molecular Biology*, 4(1), 2005.

[58] Shuheng Zhou, Philipp Rütimann, Min Xu, and Peter Bühlmann. High-dimensional covariance estimation based on Gaussian graphical models. *The Journal of Machine Learning Research*, 12:2975–3026, 2011.

[59] Hao Zhu, Geert Leus, and Georgios B. Giannakis. Sparsity-cognizant total least-squares for perturbed compressive sampling. *IEEE Transactions on Signal Processing*, 59(5):2002–2016, 2011.

[60] Yuancheng Zhu and Rina Foygel Barber. The log-shift penalty for adaptive estimation of multiple Gaussian graphical models. *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, pages 1153–1161, 2015.