# SPARSE MODEL IDENTIFICATION FOR HIGH DIMENSIONAL DATA

by

Nengfeng Zhou

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Statistics)
in The University of Michigan
2009

Doctoral Committee:

Associate Professor Ji Zhu, Chair
Professor George Michailidis
Associate Professor Jionghua Jin
Assistant Professor Elizaveta Levina

To Mom and Dad and to Xintong Luo whose influence made this work possible.

# ACKNOWLEDGEMENTS

I would like to thank all of the wonderful people I have met during my time at the University of Michigan. I am particularly grateful to my advisor, Ji Zhu for his guidance and support during my research and study. His perpetual energy and enthusiasm in research continually motivate all his advisees, including myself. Professor Jionghua Jin, Professor Elizaveta Levina and Professor George Michailidis deserve special thanks as my thesis committee members. I would also like to thank all of the faculty members and staff in the Statistics department. I am grateful to Professor Gareth M. James, Professor Jie Peng, Professor Chiara Sabatti and Professor Pei Wang for their discussion and help during my thesis work. Finally, I would like to thank my fellow students who have made the last four years so enjoyable.

# TABLE OF CONTENTS

# LIST OF FIGURES

Figure

# LIST OF TABLES

**Table**

# CHAPTER I

# Introduction

This thesis focused on analysis of high-dimensional data, and its applications in bioinformatics. The primary topics are group variable selection, partial correlation estimation and transcriptional regulation network construction. Below, I will briefly describe the major components of my thesis.

## 1.1 Group Variable Selection

Variable selection is an essential component of modern statistical data analysis. Starting with a large number of variables, possibly larger than the number of observations, the aim is to determine a smaller subset that includes the most important effects. Traditional methods treat the predictor variables "flatly," considering variables as exchangeable. However, in many science and engineering applications, predictor variables have one or more types of inherent structure. Incorporating such structural information into the modeling procedure poses interesting and challenging questions.

Specifically, the grouping structure in variable selection is considered in this thesis. In many scientific applications, there is a natural grouping of predictor variables. For example, in biological applications, assayed genes or proteins can be grouped by biological roles or biological pathways. Traditional variable selection methods tend to make selection based on the strength of individual variables rather than the strength of groups of variables, often resulting in selecting more groups than necessary. In this thesis, a new group variable selection method is proposed that not only removes unimportant groups effectively, but also keeps the flexibility of selecting variables within a group. We also showed that the new method offers the potential for achieving the theoretical "oracle" property.

## 1.2 Partial Correlation Estimation

Covariance selection is the identification and estimation of non-zero entries in the inverse covariance matrix (concentration matrix). Covariance selection is very useful in elucidating associations among a set of random variables. Under Gaussianity, for example, non-zero entries of the concentration matrix imply conditional dependency (i.e., non-zero partial correlation) between corresponding variable pairs [17]. Traditional methods only work when the sample size ($n$) is larger than the number of variables ($p$) [17, 68]. Recently, a number of methods have been introduced to perform covariance selection for data sets with $p > n$ [36, 37, 43, 51, 56, 71].

In this thesis, a computationally efficient approach —`space`(Sparse PArtial Correlation Estimation)— for selecting non-zero partial correlations is proposed under the high-dimension-low-sample-size setting. This method employs sparse regression techniques for model fitting. It is shown that `space` performs well in both non-zero partial correlation selection and the identification of hub variables, and it also outperforms two existing methods. We then apply `space` to a microarray breast cancer data set and identify a set of hub genes which may provide important insights on genetic regulatory networks.

## 1.3 Transcriptional Regulation Network Construction

In many organisms, the expression levels of each gene are controlled by the activation levels of known “Transcription Factors” (TF). A problem of considerable interest is that of estimating the “Transcription Regulation Networks” (TRN) relating the TFs and genes. While the expression levels of genes can be observed, the activation levels of the corresponding TFs are usually unknown, greatly increasing the difficulty of the problem. Based on previous experimental work, it is often the case that partial information about the TRN is available. For example, certain TFs may be known to regulate a given gene or in other cases a connection may be predicted with a certain probability. In general, the biology of the problem indicates there will be very few connections between TFs and genes. Several methods have been proposed for estimating TRNs. However, they all suffer from problems such as unrealistic assumptions about prior knowledge of the network structure or computational limitations. In this thesis, a new approach is proposed to directly utilize prior information about the network structure in conjunction with observed gene expression data to estimate the TRN. Our approach uses $L_1$ penalties on the network to ensure a sparse structure. This has the advantage of being computationally efficient as well as making many fewer assumptions about the network structure. We used our methodology to construct the TRN for *E. coli* and showed that the estimate is biologically sensible and compares favorably with previous estimates.

# CHAPTER II

# Group Variable Selection via a Hierarchical Lasso and Its Oracle Property

In many engineering and scientific applications, prediction variables are grouped. For example, in biological applications, assayed genes or proteins can be grouped by biological roles or biological pathways. Common statistical analysis methods such as ANOVA factor analysis and functional modeling with basis sets also exhibit natural variable groupings. Existing successful group variable selection methods have the limitation of selecting variables in an "all-in-all-out" fashion, i.e., when one variable in a group is selected, all other variables in the same group are also selected [2, 72, 75]. In many real problems, however, we may want to keep the flexibility of selecting variables within a group, such as in gene-set selection. In this chapter, we develop a new group variable selection method that not only removes unimportant groups effectively, but also keeps the flexibility of selecting variables within a group. We also show that the new method offers the potential for achieving the theoretical "oracle" property [20, 21].

## 2.1 Introduction

Consider the usual regression situation: we have training data, $(\boldsymbol{x}_1, y_1)$, ..., $(\boldsymbol{x}_i, y_i)$, ..., $(\boldsymbol{x}_n, y_n)$, where $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})$ are the predictors and $y_i$ is the response. To model the response $y$ in terms of the predictors $x_1, \ldots, x_p$, one may consider the linear model:

$$y = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p + \varepsilon, \tag{2.1}$$

where $\varepsilon$ is the error term. In many important practical problems, however, prediction variables are "grouped." For example, in ANOVA factor analysis, a factor may have several levels and can be expressed via several dummy variables, and the dummy variables corresponding to the same factor form a natural "group." Similarly, in additive models, each original prediction variable may be expanded into different order polynomials or a set of basis functions, and these polynomials (or basis functions) corresponding to the same original prediction variable form a natural "group." Another example is in biological applications, where assayed genes or proteins can be grouped by biological roles (or biological pathways).

For the rest of this chapter, we assume that the prediction variables can be divided into $K$ groups and the $k$th group contains $p_k$ variables. Specifically, the linear model (2.1) is now written as

$$y_i \;=\; \beta_0 + \sum_{k=1}^{K} \sum_{j=1}^{p_k} \beta_{kj} x_{i,kj} + \varepsilon_i. \tag{2.2}$$

We are interested in finding out which variables, especially which "groups," have an important effect on the response. For example, $(x_{11}, \ldots, x_{1p_1})$, $(x_{21}, \ldots, x_{2p_2})$, $\ldots$, $(x_{K1}, \ldots, x_{Kp_K})$ may represent different biological pathways and $y$ may represent a certain phenotype. We are interested in deciphering which of these biological pathways "work together" to determine the phenotype and how it is done.

There are two important challenges in this problem: prediction accuracy and interpretation. We would like our model to accurately predict future data. Prediction accuracy can often be improved by shrinking the regression coefficients. Shrinkage sacrifices some bias to reduce the variance of the predicted value and hence may improve the overall prediction accuracy. Interpretability is often realized via variable selection. With a large number of prediction variables, we often would like to determine a smaller subset that exhibits the strongest effects.

Variable selection has been studied extensively in the literature [8, 20, 27, 28, 40, 57, 63, 77]. In particular, Lasso [63] has gained much attention in recent years. The Lasso criterion penalizes the $L_1$-norm of the regression coefficients to achieve a sparse model:

$$\max_{\beta_0,\beta_{kj}} -\frac{1}{2}\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{k=1}^{K}\sum_{j=1}^{p_k}\beta_{kj}x_{i,kj}\right)^2 - \lambda\sum_{k=1}^{K}\sum_{j=1}^{p_k}|\beta_{kj}|, \tag{2.3}$$

where $\lambda \geq 0$ is a tuning parameter. Note that by location transformation, we can always assume that the predictors and the response have mean 0, so we can ignore the intercept in equation (2.3).

Due to the singularity at $\beta_{kj} = 0$, the $L_1$-norm penalty can shrink some of the fitted coefficients to be *exactly* zero when making the tuning parameter sufficiently large. However, Lasso and other methods above are for the case when the candidate variables can be treated individually or "flatly." When variables are grouped, ignoring the group structure and directly applying Lasso as in equation (2.3) may be sub-optimal. For example, suppose the $k$th group is unimportant, then Lasso tends to make individual estimated coefficients in the $k$th group zero, rather than the whole group, i.e., Lasso tends to make selections based on the strength of individual variables rather than the strength of the group, often resulting in selecting more groups than necessary.

Group variable selection problem have been addressed in some literature [2, 72, 75]. Antoniadis and Fan [2] proposed to use a blockwise additive penalty in the setting of wavelet approximations. To increase the estimation precision, empirical wavelet coefficients were thresholded or shrunken in blocks (or groups) rather than individually.

In [72] and [75], Lasso model (2.3) is extended for group variable selection. Yuan and Lin [72] chose to penalize the $L_2$-norm of the coefficients within each group, i.e., $\sum_{k=1}^{K} \|\boldsymbol{\beta}_k\|_2$, where

$$\|\boldsymbol{\beta}_k\|_2 = \sqrt{\beta_{k1}^2 + \ldots + \beta_{kp_k}^2}. \tag{2.4}$$

Due to the singularity of $\|\boldsymbol{\beta}_k\|_2$ at $\boldsymbol{\beta}_k = \mathbf{0}$, appropriately tuning $\lambda$ can set the whole coefficient vector $\boldsymbol{\beta}_k = \mathbf{0}$, hence the $k$th group is removed from the fitted model. We note that in the setting of wavelet analysis, this method reduces to that proposed by Antoniadis and Fan [2].

Instead of using the $L_2$-norm penalty, Zhao et al. [75] suggested using the $L_\infty$-norm penalty, i.e., $\sum_{k=1}^{K} \|\boldsymbol{\beta}_k\|_\infty$, where

$$\|\boldsymbol{\beta}_k\|_\infty = \max(|\beta_{k1}|, |\beta_{k2}|, \ldots, |\beta_{kp_k}|). \tag{2.5}$$

Similar to the $L_2$-norm, the $L_\infty$-norm of $\boldsymbol{\beta}_k$ is also singular when $\boldsymbol{\beta}_k = \mathbf{0}$; hence when $\lambda$ is appropriately tuned, the $L_\infty$-norm can also effectively remove unimportant groups.

However, there are some possible limitations with these methods: Both the $L_2$-norm penalty and the $L_\infty$-norm penalty select variables in an "all-in-all-out" fashion, i.e., when one variable in a group is selected, *all* other variables in the same group are also selected. The reason is that both $\|\boldsymbol{\beta}_k\|_2$ and $\|\boldsymbol{\beta}_k\|_\infty$ are singular only when the whole vector $\boldsymbol{\beta}_k = \mathbf{0}$. Once a component of $\boldsymbol{\beta}_k$ is non-zero, the two norm functions are no longer singular. This can also be heuristically understood as the following: for the $L_2$-norm (2.4), it is the ridge penalty that is under the square root; since the ridge penalty can not do variable selection (as in ridge regression), once the $L_2$-norm is non-zero (or the corresponding group is selected), all components will be non-zero. For the $L_\infty$-norm (2.5), if the "max$(\cdot)$" is non-zero, there is no increase in the penalty for letting all the individual components move away from zero. Hence if one variable in a group is selected, all other variables are also automatically selected.

In many important real problems, however, we may want to keep the flexibility of selecting variables *within* a group. For example, in the gene-set selection problem, a biological pathway may be related to a certain biological process, but it does not necessarily mean all the genes in the pathway are all related to the biological process. We may want to not only remove unimportant pathways effectively, but also identify important genes within important pathways.

For the $L_\infty$-norm penalty, another possible limitation is that the estimated coefficients within a group tend to have the same magnitude, i.e. $|\beta_{k1}| = |\beta_{k2}| = \ldots = |\beta_{kp_k}|$; and this may cause some serious bias, which jeopardizes the prediction accuracy.

In this chapter, we propose an extension of Lasso for group variable selection, which we call Hierarchical Lasso (HLasso). Our method not only removes unimportant groups effectively, but also keeps the flexibility of selecting variables within a group. Furthermore, asymptotic studies motivate us to improve our model and show that when the tuning parameter is appropriately chosen, the improved model has the *oracle* property [20, 21], i.e., it performs as well as if the correct underlying model were given in advance. Such a theoretical property has not been previously studied for group variable selection at both the group level and the within group level.

The rest of this chapter is organized as follows. In Section 2.2, we introduce our new method: the Hierarchical Lasso. We propose an algorithm to compute the solution for the Hierarchical Lasso in Section 2.3. In Section 2.4, we study the asymptotic behavior of the Hierarchical Lasso and propose an improvement for the Hierarchical Lasso. Numerical results are in Sections 2.5 and 2.6, and we conclude this chapter with Section 2.7.

## 2.2 Hierarchical Lasso

In this section, we extend the Lasso method for group variable selection so that we can effectively remove unimportant variables at both the group level and the within group level.

We reparameterize $\beta_{kj}$ as

$$\beta_{kj} = d_k \alpha_{kj}, \;\; k = 1, \ldots, K; \;\; j = 1, \ldots, p_k, \tag{2.6}$$

where $d_k \geq 0$ (for identifiability reasons). This decomposition reflects the information that $\beta_{kj}, j = 1, \ldots, p_k$, all belong to the $k$th group, by treating each $\beta_{kj}$ hierarchically. At the first level of the hierarchy is $d_k$, controlling $\beta_{kj}, j = 1, \ldots, p_k$, as a group; $\alpha_{kj}$ is at the second level of the hierarchy, reflecting differences within the $k$th group.

For the purpose of variable selection, we consider the following penalized least squares criterion:

$$\begin{aligned} \max_{d_k, \alpha_{kj}} \quad & -\frac{1}{2} \sum_{i=1}^{n} \left( y_i - \sum_{k=1}^{K} d_k \sum_{j=1}^{p_k} \alpha_{kj} x_{i,kj} \right)^2 \\ & -\lambda_1 \cdot \sum_{k=1}^{K} d_k - \lambda_2 \cdot \sum_{k=1}^{K} \sum_{j=1}^{p_k} |\alpha_{kj}| \\ \text{subject to} \quad & d_k \geq 0, \;\; k = 1, \ldots, K, \end{aligned} \tag{2.7}$$

where $\lambda_1 \geq 0$ and $\lambda_2 \geq 0$ are tuning parameters. The estimates at the group level are controlled by $\lambda_1$, and it can effectively remove unimportant groups: if $d_k$ is shrunken to zero, all $\beta_{kj}$ in the $k$th group will be equal to zero. The estimates at the variable-specific level are controlled by $\lambda_2$: if $d_k$ is not equal to zero, some of the $\alpha_{kj}$, hence some of the $\beta_{kj}, j = 1, \ldots, p_k$, still have the possibility of being zero; in this sense, the hierarchical penalty keeps the flexibility of the $L_1$-norm penalty.

One may complain that such a hierarchical penalty may be more complicated to tune in practice. However, it turns out that the two tuning parameters $\lambda_1$ and $\lambda_2$ in equation (2.7) can be simplified into one. Specifically, if we let $\lambda = \lambda_1 \cdot \lambda_2$, we can show that equation (2.7) is equivalent to

$$\begin{aligned} (2.8) \qquad & \max_{d_k, \alpha_{kj}} \quad -\frac{1}{2}\sum_{i=1}^{n}\left(y_i - \sum_{k=1}^{K} d_k \sum_{j=1}^{p_k} \alpha_{kj} x_{i,kj}\right)^2 - \sum_{k=1}^{K} d_k - \lambda \sum_{k=1}^{K}\sum_{j=1}^{p_k} |\alpha_{kj}| \\ & \text{subject to} \quad d_k \geq 0, k = 1, \ldots, K. \end{aligned}$$

**Lemma II.1.** *Let $(\hat{\boldsymbol{d}}^*, \hat{\boldsymbol{\alpha}}^*)$ be a local maximizer of (2.7), then there exists a local maximizer $(\hat{\boldsymbol{d}}^\star, \hat{\boldsymbol{\alpha}}^\star)$ of (2.8) such that $\hat{d}_k^* \hat{\alpha}_{kj}^* = \hat{d}_k^\star \hat{\alpha}_{kj}^\star$. Similarly, if $(\hat{\boldsymbol{d}}^\star, \hat{\boldsymbol{\alpha}}^\star)$ is a local maximizer of (2.8), there exists a local maximizer $(\hat{\boldsymbol{d}}^*, \hat{\boldsymbol{\alpha}}^*)$ of (2.7) such that $\hat{d}_k^* \hat{\alpha}_{kj}^* = \hat{d}_k^\star \hat{\alpha}_{kj}^\star$.*

The proof for this is in Appendix A. This lemma indicates that the final fitted models from (2.7) and (2.8) are the same, although they may provide different $d_k$ and $\alpha_{kj}$. This also implies that in practice, we do not need to tune $\lambda_1$ and $\lambda_2$ separately; we only need to tune one parameter $\lambda = \lambda_1 \cdot \lambda_2$ as in equation (2.8).

## 2.3 Algorithm

To estimate the $d_k$ and $\alpha_{kj}$ in equation (2.8), we can use an iterative approach, i.e., we first fix $d_k$ and estimate $\alpha_{kj}$, then we fix $\alpha_{kj}$ and estimate $d_k$, and we iterate between these two steps until the solution converges. Since at each step, the value of the objective function (2.8) decreases, the solution is guaranteed to converge.

When $d_k$ is fixed, (2.8) becomes a Lasso problem, hence we can use either the LAR/LASSO algorithm [19] or a quadratic programming package to efficiently solve for $\alpha_{kj}$. When $\alpha_{kj}$ is fixed, (2.8) becomes a non-negative garrote problem. Again, we can use either an efficient solution path algorithm or a quadratic programming package to solve for $d_k$. In summary, the algorithm proceeds as follows:

1. (Standardization) Center $\boldsymbol{y}$. Center and normalize $\boldsymbol{x}_{kj}$.

2. (Initialization) Initialize $d_k^{(0)}$ and $\alpha_{kj}^{(0)}$ with some plausible values. For example, we can set $d_k^{(0)} = 1$ and use the least squares estimates or the simple regression estimates by regressing the response $\boldsymbol{y}$ on each of the $\boldsymbol{x}_{kj}$ for $\alpha_{kj}^{(0)}$. Let $\beta_{kj}^{(0)} = d_k^{(0)} \alpha_{kj}^{(0)}$ and $m = 1$.

3. (Update $\alpha_{kj}$) Let

$$\tilde{x}_{i,kj} = d_k^{(m-1)} x_{i,kj}, \ k = 1, \ldots, K; \ j = 1, \ldots, p_k,$$

then

$$\alpha_{kj}^{(m)} = \arg\max_{\alpha_{kj}} -\frac{1}{2} \sum_{i=1}^{n} \left( y_i - \sum_{k=1}^{K} \sum_{j=1}^{p_k} \alpha_{kj} \tilde{x}_{i,kj} \right)^2 - \lambda \sum_{k=1}^{K} \sum_{j=1}^{p_k} |\alpha_{kj}|.$$

4. (Update $d_k$) Let

$$\tilde{x}_{i,k} = \sum_{j=1}^{p_k} \alpha_{kj}^{(m)} x_{i,kj}, \ k = 1, \ldots, K,$$

then

$$d_k^{(m)} = \arg\max_{d_k \geq 0} -\frac{1}{2} \sum_{i=1}^{n} \left( y_i - \sum_{k=1}^{K} d_k \tilde{x}_{i,k} \right)^2 - \sum_{k=1}^{K} d_k.$$

5. (Update $\beta_{kj}$) Let

$$\beta_{kj}^{(m)} = d_k^{(m)} \alpha_{kj}^{(m)}.$$

6. If $\|\beta_{kj}^{(m)} - \beta_{kj}^{(m-1)}\|$ is small enough, stop the algorithm. Otherwise, let $m \leftarrow m+1$ and go back to Step 3.

**Orthogonal Case**

To gain more insight into the hierarchical penalty, we have also studied the algorithm in the orthogonal design case. This can be useful, for example, in the wavelet setting, where each $\boldsymbol{x}_{kj}$ corresponds to a wavelet basis function, different $k$ may correspond to different "frequency" scales, and different $j$ with the same $k$ correspond to different "time" locations. Specifically, suppose $\boldsymbol{x}_{kj}^{\intercal}\boldsymbol{x}_{kj} = 1$ and $\boldsymbol{x}_{kj}^{\intercal}\boldsymbol{x}_{k'j'} = 0$ if $k \neq k'$ or $j \neq j'$, then Step 3 and Step 4 in the above algorithm have closed form solutions.

Let $\hat{\beta}_{kj}^{\text{ols}} = \boldsymbol{x}_{kj}^{\intercal}\boldsymbol{y}$ be the ordinary least squares solution when $\boldsymbol{x}_{kj}$ are orthonormal to each other.

3. When $d_k$ is fixed,

$$\alpha_{kj}^{(m)} = \mathbb{I}(d_k^{(m-1)} > 0) \cdot \text{sgn}(\hat{\beta}_{kj}^{\text{ols}}) \cdot \left( \frac{|\hat{\beta}_{kj}^{\text{ols}}|}{d_k^{(m-1)}} - \frac{\lambda}{(d_k^{(m-1)})^2} \right)_+ . \tag{2.9}$$

4. When $\alpha_{kj}$ is fixed,

$$d_k^{(m)} = \mathbb{I}(\exists j, \alpha_{kj}^{(m)} \neq 0) \cdot \left( \sum_{j=1}^{p_k} \frac{(\alpha_{kj}^{(m)})^2}{\sum_{j=1}^{p_k} (\alpha_{kj}^{(m)})^2} \frac{\hat{\beta}_{kj}^{\text{ols}}}{\alpha_{kj}^{(m)}} - \frac{1}{\sum_{j=1}^{p_k} (\alpha_{kj}^{(m)})^2} \right)_+ . \tag{2.10}$$

Equations (2.9) and (2.10) show that both $d_k^{(m)}$ and $\alpha_{kj}^{(m)}$ are soft-thresholding estimates. Here we provide some intuitive explanation.

We first look at $\alpha_{kj}^{(m)}$ in equation (2.9). If $d_k^{(m-1)} = 0$, it is natural to estimate all $\alpha_{kj}$ to be zero because of the penalty on $\alpha_{kj}$. If $d_k^{(m-1)} > 0$, then from our reparametrization, we have $\alpha_{kj} = \beta_{kj}/d_k^{(m-1)}$, $j = 1, \ldots, p_k$. Plugging in $\hat{\beta}_{kj}^{\text{ols}}$ for $\beta_{kj}$, we obtain $\tilde{\alpha}_{kj} = \hat{\beta}_{kj}^{\text{ols}}/d_k^{(m-1)}$. Equation (2.9) shrinks $\tilde{\alpha}_{kj}$, and the amount of shrinkage is inversely proportional to $(d_k^{(m-1)})^2$. So when $d_k^{(m-1)}$ is large, which indicates the $k$th group is important, the amount of shrinkage is small, and when

$d_k^{(m-1)}$ is small, which indicates the $k$th group is less important, the amount of shrinkage is large.

Now considering $d_k^{(m)}$ in equation (2.10). If all $\alpha_{kj}^{(m)}$ are zero, it is natural to estimate $d_k^{(m)}$ to also be zero because of the penalty on $d_k$. If not all $\alpha_{kj}^{(m)}$ are 0, say $\alpha_{kj_1}^{(m)}, \ldots, \alpha_{kj_r}^{(m)}$ are not zero, then we have $d_k = \beta_{kj_s}/\alpha_{kj_s}^{(m)}, 1 \leq s \leq r$. Again, plugging in $\hat{\beta}_{kj_s}^{\text{ols}}$ for $\beta_{kj_s}$, we obtain $r$ estimates for $d_k$: $\tilde{d}_k = \hat{\beta}_{kj_s}^{\text{ols}}/\alpha_{kj_s}^{(m)}, 1 \leq s \leq r$. A natural estimate for $d_k$ is then a weighted average of the $\tilde{d}_k$, and equation (2.10) provides such a (shrunken) average, with weights proportional to $(\alpha_{kj}^{(m)})^2$.

## 2.4 Asymptotic Theory

In this section, we explore the asymptotic behavior of the Hierarchical Lasso method.

The Hierarchical Lasso criterion (2.8) uses $d_k$ and $\alpha_{kj}$. We first show that it can also be written in an equivalent form using the original regression coefficients $\beta_{kj}$.

**Theorem II.2.** *If $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{\alpha}})$ is a local maximizer of (2.8), then $\hat{\boldsymbol{\beta}}$, where $\hat{\beta}_{kj} = \hat{d}_k\hat{\alpha}_{kj}$, is a local maximizer of*

$$\begin{aligned} \max_{\beta_{kj}} \quad & -\frac{1}{2}\sum_{i=1}^{n}\left(y_i - \sum_{k=1}^{K}\sum_{j=1}^{p_k} x_{i,kj}\beta_{kj}\right)^2 \\ & -2\sqrt{\lambda}\cdot\sum_{k=1}^{K}\sqrt{|\beta_{k1}| + |\beta_{k2}| + \ldots + |\beta_{kp_k}|}. \end{aligned} \tag{2.11}$$

*On the other hand, if $\hat{\boldsymbol{\beta}}$ is a local maximizer of (2.11), then we define $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{\alpha}})$, where $\hat{d}_k = 0, \hat{\boldsymbol{\alpha}}_k = 0$ if $\|\hat{\boldsymbol{\beta}}_k\|_1 = 0$, and $\hat{d}_k = \sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k\|_1}, \hat{\boldsymbol{\alpha}}_k = \frac{\hat{\boldsymbol{\beta}}_k}{\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k\|_1}}$ if $\|\hat{\boldsymbol{\beta}}_k\|_1 \neq 0$. Then the so-defined $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{\alpha}})$ is a local maximizer of (2.8).*

The proof for this is in Appendix A. Note that the penalty term in (2.11) is similar to the $L_2$-norm penalty (2.4), except that under each square root, we now penalize the $L_1$-norm of $\boldsymbol{\beta}_k$, rather than the sum of squares. However, unlike the $L_2$-norm, which is singular only at the point $\boldsymbol{\beta}_k = \mathbf{0}$, (i.e., the whole vector is equal to $\mathbf{0}$), the square root of the $L_1$-norm is singular at $\beta_{kj} = 0$ no matter the values of other $\beta_{kj}$'s. This explains, from a different perspective, why the Hierarchical Lasso can remove not only groups, but also variables within a group even when the group is selected. Equation (2.11) also implies that the Hierarchical Lasso belongs to the "CAP" family [75].

We study the asymptotic properties allowing the total number of variables $P_n$, as well as the number of groups $K_n$ and the number of variables within each group $p_{nk}$, to go to $\infty$, where $P_n = \sum_{k=1}^{K_n} p_{nk}$. Note that we add a subscript "$n$" to $K$ and $p_k$ to denote that these quantities can change with $n$. Accordingly, $\boldsymbol{\beta}$, $y_i$ and $x_{i,kj}$ are also changed to $\boldsymbol{\beta}_n$, $y_{ni}$ and $x_{ni,kj}$. We write $2\sqrt{\lambda}$ in (2.11) as $n\lambda_n$, and the criterion (2.11) is re-written as

$$\begin{aligned} \max_{\beta_{n,kj}} \quad & -\frac{1}{2}\sum_{i=1}^{n}\left(y_{ni} - \sum_{k=1}^{K_n}\sum_{j=1}^{p_{nk}} x_{ni,kj}\beta_{n,kj}\right)^2 \\ & -n\lambda_n \cdot \sum_{k=1}^{K_n}\sqrt{|\beta_{n,k1}| + |\beta_{n,k2}| + \ldots + |\beta_{n,kp_{nk}}|}. \end{aligned} \tag{2.12}$$

Our asymptotic analysis in this section is based on criterion (2.12).

Let $\boldsymbol{\beta}_n^0 = (\boldsymbol{\beta}_{\mathcal{A}_n}^0, \boldsymbol{\beta}_{\mathcal{B}_n}^0, \boldsymbol{\beta}_{\mathcal{C}_n}^0)^{\mathsf{T}}$ be the underlying true parameters, where

$$\begin{aligned} \mathcal{A}_n &= \{(k,j) : \beta_{n,kj}^0 \neq 0\}, \\ \mathcal{B}_n &= \{(k,j) : \beta_{n,kj}^0 = 0, \boldsymbol{\beta}_{nk}^0 \neq 0\}, \\ \mathcal{C}_n &= \{(k,j) : \boldsymbol{\beta}_{nk}^0 = 0\}, \\ \mathcal{D}_n &= \mathcal{B}_n \cup \mathcal{C}_n. \end{aligned} \tag{2.13}$$

Note that $\mathcal{A}_n$ contains the indices of coefficients which are truly non-zero, $\mathcal{C}_n$ contains the indices where the whole "groups" are truly zero, and $\mathcal{B}_n$ contains the indices of zero coefficients, but they belong to some non-zero groups. So $\mathcal{A}_n$, $\mathcal{B}_n$ and $\mathcal{C}_n$ are disjoint and they partition all the indices. We have the following theorem.

**Theorem II.3.** *If* $\sqrt{n}\lambda_n = O(1)$, *then there exists a root-*$(n/P_n)$ *consistent local maximizer* $\hat{\boldsymbol{\beta}}_n = (\hat{\boldsymbol{\beta}}_{\mathcal{A}_n}, \hat{\boldsymbol{\beta}}_{\mathcal{B}_n}, \hat{\boldsymbol{\beta}}_{\mathcal{C}_n})^{\mathsf{T}}$ *of (2.12), and if also* $P_n n^{-3/4}/\lambda_n \to 0$ *as* $n \to \infty$, *then* $\Pr(\hat{\boldsymbol{\beta}}_{\mathcal{C}_n} = 0) \to 1$.

The proof for this is in Appendix A. Theorem II.3 implies that the Hierarchical Lasso method can effectively remove unimportant *groups*. For the above root-$(n/P_n)$ consistent estimate, however, if $\mathcal{B}_n \neq \emptyset$ (empty set), then $\Pr(\hat{\boldsymbol{\beta}}_{\mathcal{B}_n} = 0) \to 1$ is not always true. This means that although the Hierarchical Lasso method can effectively remove *all* unimportant *groups* and *some* of the unimportant *variables* within important groups, it cannot effectively remove *all* unimportant *variables* within important groups.

Next, we improve the Hierarchical Lasso method to tackle this limitation.

### 2.4.1 Further Improvement and Generalization

To improve the Hierarchical Lasso method, we apply the adaptive idea [8, 57, 67, 73, 74, 76], i.e., to penalize different coefficients differently. Specifically, we consider

$$\begin{aligned} \max_{\beta_{n,kj}} \quad & -\frac{1}{2}\sum_{i=1}^{n}\left(y_{ni} - \sum_{k=1}^{K_n}\sum_{j=1}^{p_k} x_{ni,kj}\beta_{n,kj}\right)^2 \\ & -n\lambda_n \cdot \sum_{k=1}^{K_n}\sqrt{w_{n,k1}|\beta_{n,k1}| + w_{n,k2}|\beta_{n,k2}| + \ldots + w_{n,kp_k}|\beta_{n,kp_{nk}}|}, \end{aligned} \tag{2.14}$$

where $w_{n,kj}$ are pre-specified weights. The intuition here is that if the effect of a variable is strong, we would like the corresponding weight to be small, hence the corresponding coefficient is lightly penalized. If the effect of a variable is not strong, we would like the corresponding weight to be large, hence the corresponding coefficient is heavily penalized. In practice, we may consider using the ordinary least squares estimates or the ridge regression estimates to help us compute the weights, for example,

$$w_{n,kj} = \frac{1}{|\hat{\beta}^{\text{ols}}_{n,kj}|^{\gamma}} \text{ or } w_{n,kj} = \frac{1}{|\hat{\beta}^{\text{ridge}}_{n,kj}|^{\gamma}}, \tag{2.15}$$

where $\gamma$ is a positive constant.

### 2.4.2 Oracle Property

**Problem Setup**

Since the theoretical results we develop for (2.14) are not restricted to the squared error loss, for the rest of the section, we consider the generalized linear model. For generalized linear models, statistical inferences are based on underlying likelihood functions. We assume that the data $\boldsymbol{V}_{ni} = (\boldsymbol{X}_{ni}, Y_{ni}),\ i = 1, \ldots, n$ are independent and identically distributed for every $n$. Conditioning on $\boldsymbol{X}_{ni} = \boldsymbol{x}_{ni}$, $Y_{ni}$ has a density $f_n(g_n(\boldsymbol{x}_{ni}^{\intercal}\boldsymbol{\beta}_n), Y_{ni})$, where $g_n(\cdot)$ is a known link function. We maximize the penalized log-likelihood

$$\begin{aligned} \max_{\beta_{n,kj}} Q_n(\boldsymbol{\beta}_n) &= L_n(\boldsymbol{\beta}_n) - J_n(\boldsymbol{\beta}_n) \\ &= \sum_{i=1}^{n} \ell_n(g_n(\boldsymbol{x}_{ni}^{\intercal}\boldsymbol{\beta}_n), y_{ni}) - n \sum_{k=1}^{K_n} p_{\lambda_n, \boldsymbol{w}_n}(\boldsymbol{\beta}_{nk}), \end{aligned} \tag{2.16}$$

where $\ell_n(\cdot, \cdot) = \log f_n(\cdot, \cdot)$ denotes the conditional log-likelihood of $Y$, and

$$p_{\lambda_n, \boldsymbol{w}_n}(\boldsymbol{\beta}_{nk}) = \lambda_n \sqrt{w_{n,k1}|\beta_{n,k1}| + \ldots + w_{n,kp_k}|\beta_{n,kp_{nk}}|}.$$

Note that under the normal distribution, $\ell_n(g_n(\boldsymbol{x}_{ni}^{\intercal}\boldsymbol{\beta}_n), y_{ni}) = -\frac{(y_{ni} - \boldsymbol{x}_{ni}^{\intercal}\boldsymbol{\beta}_n)^2}{2C_1} + C_2$, hence (2.16) reduces to (2.14).

The asymptotic properties of (2.16) are described in the following theorems, and the proofs are in Appendix A. We note that the proofs follow the spirit of previous work [20, 21], but due to the grouping structure and the adaptive weights, they are non-trivial extensions of these work.

To control the adaptive weights, we define:

$$\begin{aligned} a_n &= \max\{w_{n,kj} : \beta^0_{n,kj} \neq 0\}, \\ b_n &= \min\{w_{n,kj} : \beta^0_{n,kj} = 0\}. \end{aligned}$$

We assume that

$$0 < c_1 < \min\{\beta^0_{n,kj} : \beta^0_{n,kj} \neq 0\} < \max\{\beta^0_{n,kj} : \beta^0_{n,kj} \neq 0\} < c_2 < \infty.$$

We then have the following theorems.

**Theorem II.4.** *For every $n$, the observations $\{\boldsymbol{V}_{ni}, i = 1, 2, \ldots, n\}$ are independent and identically distributed, each with a density $f_n(\boldsymbol{V}_{n1}, \boldsymbol{\beta}_n)$ that satisfies conditions (A1)-(A3) in Appendix A. If $\frac{P_n^4}{n} \to 0$ and $P_n^2 \lambda_n \sqrt{a_n} = o_p(1)$, then there exists a local maximizer $\hat{\boldsymbol{\beta}}_n$ of $Q_n(\boldsymbol{\beta}_n)$ such that $\|\hat{\boldsymbol{\beta}}_n - \boldsymbol{\beta}^0_n\| = O_p(\sqrt{P_n}(n^{-1/2} + \lambda_n \sqrt{a_n}))$.*

Hence by choosing $\lambda_n \sqrt{a_n} = O_p(n^{-1/2})$, there exists a root-$(n/P_n)$ consistent penalized likelihood estimate.

**Theorem II.5.** *For every $n$, the observations $\{\boldsymbol{V}_{ni}, i = 1, 2, \ldots, n\}$ are independent and identically distributed, each with a density $f_n(\boldsymbol{V}_{n1}, \boldsymbol{\beta}_n)$ that satisfies conditions (A1)-(A3) in Appendix A. If $\frac{P_n^4}{n} \to 0$, $\lambda_n \sqrt{a_n} = O_p(n^{-1/2})$ and $\frac{P_n^2}{\lambda_n^2 b_n} = o_p(n)$, then there exists a root-$(n/P_n)$ consistent local maximizers $\hat{\boldsymbol{\beta}}_n$ such that:*

*(a) Sparsity: $\Pr(\hat{\boldsymbol{\beta}}_{n,\mathcal{D}_n} = 0) \to 1$, where $\mathcal{D}_n = \mathcal{B}_n \cup \mathcal{C}_n$.*

*(b) Asymptotic normality: If $\lambda_n \sqrt{a_n} = o_p((nP_n)^{-1/2})$ and $\frac{P_n^5}{n} \to 0$ as $n \to \infty$, then we also have:*

$$\sqrt{n}\boldsymbol{A}_n \boldsymbol{I}_n^{1/2}(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{\beta}^0_{n,\mathcal{A}_n}) \to \mathcal{N}(\boldsymbol{0}, \boldsymbol{G}),$$

*where $\boldsymbol{A}_n$ is a $q \times |\mathcal{A}_n|$ matrix such that $\boldsymbol{A}_n \boldsymbol{A}_n^\intercal \to \boldsymbol{G}$ and $\boldsymbol{G}$ is a $q \times q$ nonnegative symmetric matrix. $\boldsymbol{I}_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})$ is the Fisher information matrix which knows $\boldsymbol{\beta}^0_{\mathcal{D}_n} = 0$.*

The above requirements, $\lambda_n\sqrt{a_n} = o_p((nP_n)^{-1/2})$ and $\frac{P_n^2}{\lambda_n^2 b_n} = o_p(n)$ as $n \to \infty$, can be satisfied by selecting $\lambda_n$ and $w_{n,kj}$ appropriately. For example, we may let $\lambda_n = \frac{(nP_n)^{-1/2}}{\log n}$ and $w_{n,kj} = \frac{1}{|\hat{\beta}^0_{n,kj}|^2}$, where $\hat{\beta}^0_{n,kj}$ is the un-penalized likelihood estimate of $\beta^0_{n,kj}$, which is root-$(n/P_n)$ consistent. Then we have $a_n = O_p(1)$ and $\frac{1}{b_n} = O_p(\frac{P_n}{n})$. Hence $\lambda_n\sqrt{a_n} = o_p((nP_n)^{-1/2})$ and $\frac{P_n^2}{\lambda_n^2 b_n} = o_p(n)$ are satisfied when $\frac{P_n^5}{n} \to 0$.

**Likelihood ratio test**

Similarly as in the paper of Fan and Peng [21], we can do a likelihood ratio test. Consider the problem of testing linear hypotheses:

$$H_0 : \boldsymbol{A}_n\boldsymbol{\beta}^0_{n,\mathcal{A}_n} = 0 \text{ vs. } H_1 : \boldsymbol{A}_n\boldsymbol{\beta}^0_{n,\mathcal{A}_n} \neq 0,$$

where $\boldsymbol{A}_n$ is a $q \times |\mathcal{A}_n|$ matrix and $\boldsymbol{A}_n\boldsymbol{A}_n^{\intercal} \to \boldsymbol{I}_q$ for a fixed $q$. This problem includes testing simultaneously the significance of a few covariate variables.

We can introduce a natural likelihood ratio test for the problem

$$T_n = 2\left\{\sup_{\Omega_n} Q_n(\boldsymbol{\beta}_n|\boldsymbol{V}) - \sup_{\Omega_n, \boldsymbol{A}_n\boldsymbol{\beta}_{n,\mathcal{A}_n}=0} Q_n(\boldsymbol{\beta}_n|\boldsymbol{V})\right\},$$

where $\Omega_n$ is the parameter space for $\boldsymbol{\beta}_n$.

We can derive the following theorem about the asymptotic null distribution of the test statistic.

**Theorem II.6.** *When conditions in (b) of Theorem II.5 are satisfied, under $H_0$ we have*

$$T_n \to \chi^2_q,$$

*as $n \to \infty$.*

## 2.5 Simulation Study

In this section, we use simulations to demonstrate the Hierarchical Lasso (HLasso) method, and compare the results with those of some existing methods. Specifically, we first compare Hierarchical Lasso with some other group variable selection methods, i.e., the $L_2$-norm Group Lasso (2.4) and the $L_\infty$-norm Group Lasso (2.5). Then we compare the Adaptive Hierarchical Lasso with some other "oracle" (but non-group variable selection) methods, i.e., the SCAD and the Adaptive Lasso.

We extended the simulations in [72], and considered a model which had both categorical and continuous prediction variables. We first generated seventeen independent standard normal variables, $Z_1, \ldots, Z_{16}$ and W. The covariates were then defined as $X_j = (Z_j + W)/\sqrt{2}$. Then the last eight covariates $X_9, \ldots, X_{16}$ were discretized to 0, 1, 2, and 3 by $\Phi^{-1}(1/4)$, $\Phi^{-1}(1/2)$ and $\Phi^{-1}(3/4)$. Each of $X_1, \ldots, X_8$ was expanded through a fourth-order polynomial, and only the main effects of $X_9, \ldots, X_{16}$ were considered. This gave us a total of eight continuous groups with four variables in each group and eight categorical groups with three variables in each group. We considered two cases.

**Case 1.** "All-in-all-out"

$$\begin{aligned} Y &= \left[X_3 + 0.5X_3^2 + 0.1X_3^3 + 0.1X_3^4\right] + \left[X_6 - 0.5X_6^2 + 0.15X_6^3 + 0.1X_6^4\right] \\ &\quad + \left[\mathbb{I}(X_9 = 0) + \mathbb{I}(X_9 = 1) + \mathbb{I}(X_9 = 2)\right] + \varepsilon. \end{aligned}$$

**Case 2.** "Not all-in-all-out"

$$Y = \left(X_3 + X_3^2\right) + \left(2X_6 - 1.5X_6^2\right) + \left[\mathbb{I}(X_9 = 0) + 2\,\mathbb{I}(X_9 = 1)\right] + \varepsilon.$$

For all the simulations above, the error term $\varepsilon$ follows a normal distribution $\mathrm{N}(0, \sigma^2)$, where $\sigma^2$ was set such that each signal-to-noise ratio, $\mathrm{Var}(\boldsymbol{X}^\intercal\boldsymbol{\beta})/\mathrm{Var}(\epsilon)$, was equal to 3. We generated $n = 400$ training observations from each of the above models, along with 200 validation observations and 10,000 test observations. The validation set was used to select the tuning parameters $\lambda$'s that minimized the validation error. Using these selected $\lambda$'s, we calculated the mean squared error (MSE) with the test set. We repeated this 200 times and computed the average MSEs and their corresponding standard errors. We also recorded how frequently the important variables were selected and how frequently the unimportant variables were removed. The results are summarized in Table 2.1.

As we can see, all shrinkage methods perform much better than OLS; this illustrates that some regularization is crucial for prediction accuracy. In terms of prediction accuracy, we can also see that when variables in a group follow the "all-in-all-out" pattern, the $L_2$-norm (Group Lasso) method performs slightly better than the Hierarchical Lasso method (Case 1 of Table 2.1). When variables in a group do not follow the "all-in-all-out" pattern, however, the Hierarchical Lasso method performs slightly better than the $L_2$-norm method (Case 2 of Table 2.1). For variable selection, in terms of identifying important variables, the four shrinkage methods, the Lasso, the $L_\infty$-norm, the $L_2$-norm, and the Hierarchical Lasso all perform similarly in both Case 1 and Case 2 ("Non-zero Var." of Table 2.1). However, the $L_2$-norm method and the Hierarchical Lasso method are more effective at removing unimportant variables than Lasso and the $L_\infty$-norm method in both Case 1 and Case 2 ("Zero Var." of Table 2.1).

In the above analysis, we used either criterion (2.8) or criterion (2.11) for the Hierarchical Lasso, i.e., we did not use the adaptive weights $w_{kj}$ to penalize different coefficients differently. To assess the improved version of the Hierarchical Lasso, i.e. criterion (2.14), we also considered using adaptive weights. Specifically, we applied the OLS weights in (2.15) to (2.14) with $\gamma = 1$. We compared the results with those of SCAD and the Adaptive Lasso, which also enjoy the oracle property. However, we note that SCAD and the Adaptive Lasso do not take advantage of the grouping structure information. As a benchmark, we also computed the Oracle OLS results,

Table 2.1: Comparison of several group variable selection methods, including the $L_2$-norm Group Lasso, the $L_\infty$-norm Group Lasso and the Hierarchical Lasso. The OLS and the regular Lasso are used as benchmarks. The upper half shows results for Case 1, and the lower half shows results for Case 2. "MSE" is the mean squared error of the test set. "Zero Var." is the percentage of correctly removed unimportant variables. "Non-zero Var." is the percentage of correctly identified important variables. All the numbers before parentheses are means over 200 repetitions, and the numbers within the parentheses are the corresponding standard errors.

| Case 1: "All-in-all-out" | | | | | |
|---|---|---|---|---|---|
| | OLS | Lasso | $L_\infty$ | $L_2$ | HLasso |
| MSE | 0.92 (0.018) | 0.47 (0.011) | 0.31 (0.008) | 0.18 (0.009) | 0.24 (0.008) |
| Zero Var. | - | 57% (1.6%) | 29% (1.4%) | 96% (0.8%) | 94% (0.7%) |
| Non-Zero Var. | - | 92% (0.6%) | 100% (0%) | 100% (0%) | 98% (0.3%) |
| Case 2: "Not all-in-all-out" | | | | | |
| | OLS | Lasso | $L_\infty$ | $L_2$ | HLasso |
| MSE | 0.91 (0.018) | 0.26 (0.008) | 0.46 (0.012) | 0.21 (0.01) | 0.15 (0.006) |
| Zero Var. | - | 70% (1%) | 17% (1.2%) | 87% (0.8%) | 91% (0.5%) |
| Non-zero Var. | - | 99% (0.3%) | 100% (0%) | 100% (0.2%) | 100% (0.1%) |

i.e., OLS using only the important variables. The results are summarized in Table 2.2. We can see that in the "all-in-all-out" case, the Adaptive Hierarchical Lasso removes unimportant variables more effectively than SCAD and Adaptive Lasso, and consequently, the Adaptive Hierarchical Lasso outperforms SCAD and Adaptive Lasso by a significant margin in terms of prediction accuracy. In the "not all-in-all-out" case, the advantage of knowing the grouping structure information is reduced, however, the Adaptive Hierarchical Lasso still performs slightly better than SCAD and Adaptive Lasso, especially in terms of removing unimportant variables.

To assess how the sample size affects different "oracle" methods, we also considered $n$=200, 400, 800, 1600 and 3200. The results are summarized in Figure 2.1, where the upper half corresponds to the "all-in-all-out" case, and the lower half corresponds to the "not all-in-all-out" case. Not surprisingly, as the sample size increases, the performances of different methods all improve: in terms of prediction accuracy, the MSE's all decrease (at about the same rate) and approach that of the Oracle OLS; in terms of variable selection, the probabilities of identifying the correct model all increase and approach one. However, overall, the Adaptive Hierarchical Lasso always performs the best among the three "oracle" methods, and the gap is especially prominent in terms of removing unimportant variables when the sample size is moderate.

Table 2.2: Comparison of several "oracle" methods, including the Adaptive Hierarchical Lasso, SCAD and the Adaptive Lasso. SCAD and Adaptive Lasso do not take advantage of the grouping structure information. The Oracle OLS uses only important variables. Descriptions for the rows are the same as those in the caption of Table 2.1.

| Case 1: "All-in-all-out" | | | | |
|---|---|---|---|---|
| | Oracle OLS | Ada Lasso | SCAD | Ada HLasso |
| MSE | 0.16 (0.006) | 0.37 (0.011) | 0.35 (0.011) | 0.24 (0.009) |
| Zero Var. | - | 77% (0.7%) | 79% (1.1%) | 98% (0.3%) |
| Non-Zero Var. | - | 94% (0.5%) | 91% (0.6%) | 96% (0.5%) |
| Case 2: "Not all-in-all-out" | | | | |
| | Oracle OLS | Ada Lasso | SCAD | Ada HLasso |
| MSE | 0.07 (0.003) | 0.13 (0.005) | 0.11 (0.004) | 0.10 (0.005) |
| Zero Var. | - | 91% (0.3%) | 91% (0.4%) | 98% (0.1%) |
| Non-zero Var. | - | 98% (0.4%) | 99% (0.3%) | 99% (0.3%) |

## 2.6 Real Data Analysis

In this section, we use a gene expression dataset from the NCI-60 collection of cancer cell lines to further illustrate the Hierarchical Lasso method. We sought to use this dataset to identify targets of the transcription factor p53, which regulates gene expression in response to various signals of cellular stress. The mutational status of the p53 gene has been reported for 50 of the NCI-60 cell lines, with 17 being classified as normal and 33 as carrying mutations in the gene [49].

Instead of single-gene analysis, gene-set information has recently been used to analyze gene expression data. For example, Subramanian et al. [59] developed the Gene Set Enrichment Analysis (GSEA), which is found to be more stable and more powerful than single-gene analysis. Efron and Tibshirani [18] improved the GSEA method by using a new statistics for summarizing gene-sets. Both methods are based

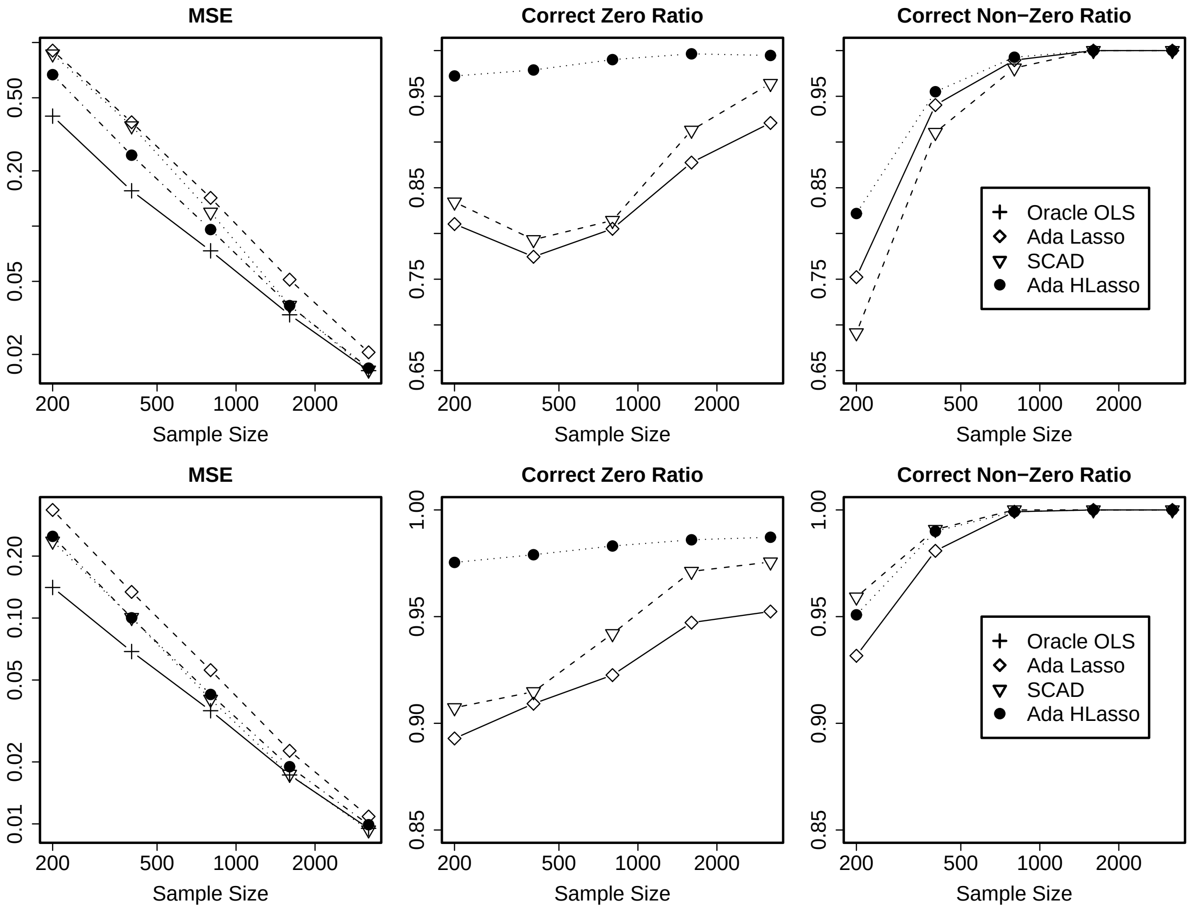


Figure 2.1: Comparison of several oracle methods, including the SCAD, the Adaptive Lasso and the Adaptive Hierarchical Lasso. SCAD and Adaptive Lasso do not take advantage of the grouping structure information. The Oracle OLS uses only important variables. The first row corresponds to the "all-in-all-out" case, and the second row corresponds to the "not all-in-all-out" case. "Correct zero ratio" records the percentage of correctly removed unimportant variables. "Correct non-zero ratio" records the percentage of correctly identified important variables.

on hypothesis testing. In this analysis, we consider using the Hierarchical Lasso method for gene-set selection. The gene-sets used here are the cytogenetic gene-sets and the functionals gene-sets from the GSEA package [59]. We only considered the 391 overlapping gene-sets with a size greater than 15 genes.

Since the response here is binary (normal vs mutation), following the discussion in Section 2.4, we use the logistic Hierarchical Lasso regression, instead of the least square Hierarchical Lasso. Note that a gene may belong to multiple gene-sets, we also extend the Hierarchical Lasso to the case of overlapping groups. Suppose there are $K$ groups and $J$ variables. Let $\mathcal{G}_k$ denote the set of indices of the variables in the $k$th group. One way to model the overlapping situation is to extend criterion (2.8) as the following:

$$\begin{aligned} \max_{d_k, \alpha_j} \quad & \sum_{i=1}^{n} \ell\left( \sum_{k=1}^{K} d_k \sum_{j: j \in \mathcal{G}_k} \alpha_j x_{i,j},\ y_i \right) \\ & - \sum_{k=1}^{K} d_k - \lambda \cdot \sum_{j=1}^{J} |\alpha_j| \\ \text{subject to} \quad & d_k \geq 0,\ k = 1, \ldots, K, \end{aligned} \tag{2.17}$$

where $\alpha_j$ can be considered as the "intrinsic" effect of a variable (no matter which group it belongs to), and different group effects are represented via different $d_k$. In the formulation, $\ell(\eta_i, y_i) = y_i \eta_i - \log(1 + e^{\eta_i})$ is the logistic log-likelihood function with $y_i$ being a 0/1 response. Also notice that if each variable belongs to only one group, the model reduces to the non-overlapping criterion (2.8).

We randomly split the 50 samples into the training and test sets 100 times; for each split, 33 samples (22 carrying mutations in the gene and 11 being normal) were used for training and the remaining 17 samples (11 carrying mutations in the gene and 6 being normal) were for testing. For each split, we applied three methods, the logistic Lasso, the logistic $L_2$-norm Group Lasso and the logistic Hierarchical Lasso.

For each of the splits, we pre-selected 2000 genes from 10,100 genes according to the t-statistics of the training sample. Tuning parameters were chosen using five-fold cross-validation.

We first compare the prediction accuracy of the three methods. Over the 100 random splits, the logistic Hierarchical Lasso has an average misclassification rate of 14% with the standard error 1.8%, which is smaller than 23%(1.7%) of the logistic Lasso and 32%(1.2%) of the logistic Group Lasso. To assess the stability of the prediction, we recorded the frequency in which each sample, as a test observation, was correctly classified. For example, if a sample appeared in 40 test sets among the 100 random splits, and out of the 40 predictions, the sample was correctly classified 36 times, we recorded 36/40 for this sample. The results are shown in Figure 2.2. As we can see, for most samples, the logistic Hierarchical Lasso classified them correctly for most of the random splits, and the predictions seemed to be slightly more stable than the logistic Lasso and the logistic $L_2$-norm Group Lasso.

Next, we compare gene-set selection of these three methods. The most notable difference is that both logistic Lasso and the logistic Hierarchical Lasso selected gene CDKN1A most frequently out of the 100 random split, while the logistic $L_2$-norm Group Lasso rarely selected it. CDKN1A is also named as wild-type p53 activated fragment-1 (p21), and it is known that the expression of gene CDKN1A is tightly controlled by the tumor suppressor protein p53, through which this protein mediates the p53-dependent cell cycle G1 phase arrest in response to a variety of stress stimuli (http://www.ncbi.nlm.nih.gov/).

We also compared the gene-sets selected by the logistic Hierarchical Lasso with those selected by the GSEA of Subramanian et al. [59] and the GSA of Efron and Tibshirani [18]. The two most frequently selected gene-sets by the Hierarchical Lasso are *atm pathway* and *radiation sensitivity*. The most frequently selected genes in *atm pathway* by the logistic Hierarchical Lasso are CDKN1A, MDM2 and RELA, and the most frequently selected genes in *radiation sensitivity* are CDKN1A, MDM2 and BCL2. It is known that MDM2, the second commonly selected gene, is a target gene of the transcription factor tumor protein p53, and the encoded protein in MDM2 is a nuclear phosphoprotein that binds and inhibits transactivation by tumor protein p53, as part of an autoregulatory negative feedback loop (http://www.ncbi.nlm.nih.gov/). Note that the gene-set *radiation sensitivity* was also selected by GSEA and GSA. Though the gene-set *atm pathway* was not selected by GSEA and GSA, it shares 7, 8, 6, and 3 genes with gene-sets *radiation sensitivity*, *p53 signalling*, *p53 hypoxia pathway* and *p53 Up* respectively, which were all selected by GSEA and GSA. We also note that GSEA and GSA are based on the *marginal* strength of each gene-set, while the logistic Hierarchical Lasso fits an "additive" model and uses the *joint* strengths of gene-sets.

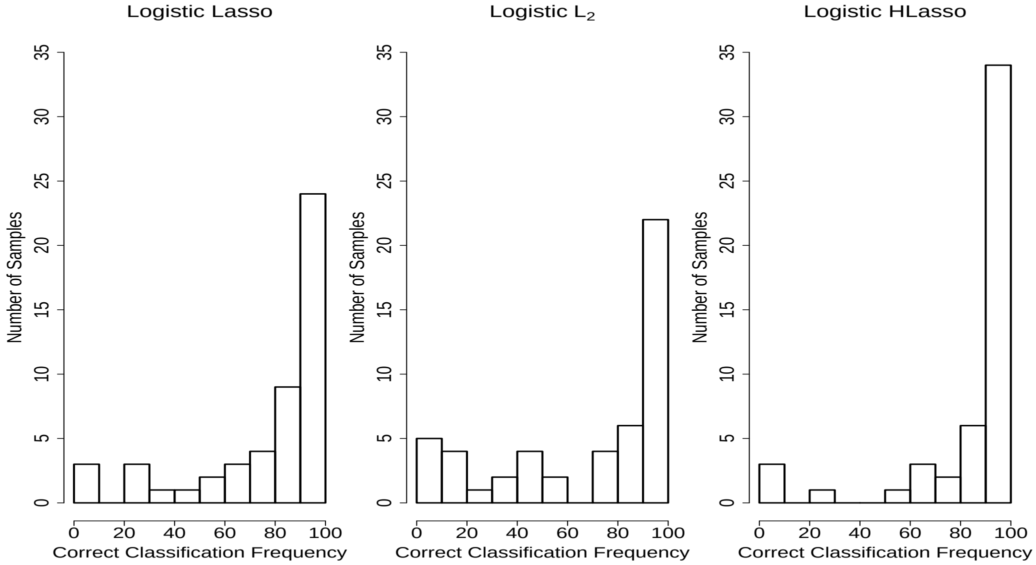


Figure 2.2: The number of samples vs the frequency that a sample was correctly classified on 100 random splits of the p53 data.

## 2.7 Discussion

In this chapter, we have proposed a Hierarchical Lasso method for group variable selection. Different variable selection methods have their own advantages in different scenarios. The Hierarchical Lasso method not only effectively removes unimportant groups, but also keeps the flexibility of selecting variables within a group. We show that the improved Hierarchical Lasso method enjoys an oracle property, i.e., it performs as well as if the true sub-model were given in advance. Numerical results indicate that our method works well, especially when variables in a group are associated with the response in a "not all-in-all-out" fashion.

The grouping idea is also applicable to other regression and classification settings, for example, the multi-response regression and multi-class classification problems. In these problems, a grouping structure may not exist among the prediction variables, but instead, natural grouping structures exist among *parameters*. We use the multi-response regression problem to illustrate the point [9, 65]. Suppose we observe $(\boldsymbol{x}_1, \boldsymbol{y}_1)$, ..., $(\boldsymbol{x}_n, \boldsymbol{y}_n)$, where each $\boldsymbol{y}_i = (y_{i1}, \ldots, y_{iK})$ is a vector containing $K$ responses, and we are interested in selecting a subset of the prediction variables that predict well for all of the multiple responses. Standard techniques estimate $K$ prediction functions, one for each of the $K$ responses, $f_k(\boldsymbol{x}) = \beta_{k1}x_1 + \cdots + \beta_{kp}x_p, k = 1, \ldots, K$. The prediction variables $(x_1, \ldots, x_p)$ may not have a grouping structure, however, we may consider the coefficients corresponding to the same prediction variable form a natural group, i.e., $(\beta_{1j}, \beta_{2j}, \ldots, \beta_{Kj})$. Using our Hierarchical Lasso idea, we reparameterize $\beta_{kj} = d_j\alpha_{kj}$, $d_j \geq 0$, and we consider

$$\max_{d_j \geq 0, \alpha_{kj}} \quad -\frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{n}\left(y_{ik} - \sum_{j=1}^{p} d_j\alpha_{kj}x_{ij}\right)^2 \\ -\lambda_1 \cdot \sum_{j=1}^{p} d_j - \lambda_2 \cdot \sum_{j=1}^{p}\sum_{k=1}^{K}|\alpha_{kj}|.$$

Note that if $d_j$ is shrunk to zero, all $\beta_{kj}, k = 1, \ldots, K$ will be equal to zero, hence the $j$th prediction variable will be removed from all $K$ predictions. If $d_j$ is not equal to zero, then some of the $\alpha_{kj}$ and hence some of the $\beta_{kj}$, $k = 1, \ldots, K$, still have the possibility of being zero. Therefore, the $j$th variable may be predictive for some responses but non-predictive for others.

## 2.8 Appendix A

### Proof of Lemma II.1

Let $Q^*(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{\alpha})$ be the criterion that we would like to maximize in equation (2.7) and let $Q^\star(\lambda, \boldsymbol{d}, \boldsymbol{\alpha})$ be the corresponding criterion in equation (2.8).

Let $(\hat{\boldsymbol{d}}^*, \hat{\boldsymbol{\alpha}}^*)$ be a local maximizer of $Q^*(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{\alpha})$. We would like to prove $(\hat{\boldsymbol{d}}^\star = \lambda_1\hat{\boldsymbol{d}}^*, \hat{\boldsymbol{\alpha}}^\star = \hat{\boldsymbol{\alpha}}^*/\lambda_1)$ is a local maximizer of $Q^\star(\lambda, \boldsymbol{d}, \boldsymbol{\alpha})$.

We immediately have

$$Q^*(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{\alpha}) = Q^\star(\lambda, \lambda_1\boldsymbol{d}, \boldsymbol{\alpha}/\lambda_1).$$

Since $(\hat{\boldsymbol{d}}^*, \hat{\boldsymbol{\alpha}}^*)$ is a local maximizer of $Q^*(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{\alpha})$, there exists $\delta > 0$ such that if $\boldsymbol{d}'$, $\boldsymbol{\alpha}'$ satisfy $\|\boldsymbol{d}' - \hat{\boldsymbol{d}}^*\| + \|\boldsymbol{\alpha}' - \hat{\boldsymbol{\alpha}}^*\| < \delta$ then $Q^*(\lambda_1, \lambda_2, \boldsymbol{d}', \boldsymbol{\alpha}') \leq Q^*(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}^*, \hat{\boldsymbol{\alpha}}^*)$.

Choosing $\delta'$ such that $\frac{\delta'}{\min\left(\lambda_1, \frac{1}{\lambda_1}\right)} \leq \delta$, for any $(\boldsymbol{d}'', \boldsymbol{\alpha}'')$ satisfying $\|\boldsymbol{d}'' - \hat{\boldsymbol{d}}^\star\| + \|\boldsymbol{\alpha}'' -$

$\hat{\boldsymbol{\alpha}}^\star\| < \delta'$ we have

$$\begin{aligned}
\left\|\frac{\boldsymbol{d}''}{\lambda_1} - \hat{\boldsymbol{d}}^*\right\| + \|\lambda_1\boldsymbol{\alpha}'' - \hat{\boldsymbol{\alpha}}^*\| &\le \frac{\lambda_1\left\|\frac{\boldsymbol{d}''}{\lambda_1} - \hat{\boldsymbol{d}}^*\right\| + \frac{1}{\lambda_1}\|\lambda_1\boldsymbol{\alpha}'' - \hat{\boldsymbol{\alpha}}^*\|}{\min\left(\lambda_1, \frac{1}{\lambda_1}\right)} \\
&= \frac{\|\boldsymbol{d}'' - \hat{\boldsymbol{d}}^\star\| + \|\boldsymbol{\alpha}'' - \hat{\boldsymbol{\alpha}}^\star\|}{\min\left(\lambda_1, \frac{1}{\lambda_1}\right)} \\
&< \frac{\delta'}{\min\left(\lambda_1, \frac{1}{\lambda_1}\right)} \\
&< \delta.
\end{aligned}$$

Hence

$$\begin{aligned}
Q^\star(\lambda, \hat{\boldsymbol{d}}'', \hat{\boldsymbol{\alpha}}'') &= Q^*(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}''/\lambda_1, \lambda_1\hat{\boldsymbol{\alpha}}'') \\
&\le Q^*(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}^*, \hat{\boldsymbol{\alpha}}^*) \\
&= Q^\star(\lambda, \hat{\boldsymbol{d}}^\star, \hat{\boldsymbol{\alpha}}^\star).
\end{aligned}$$

Therefore, $(\hat{\boldsymbol{d}}^\star = \lambda_1\hat{\boldsymbol{d}}^*, \hat{\boldsymbol{\alpha}}^\star = \hat{\boldsymbol{\alpha}}^*/\lambda_1)$ is a local maximizer of $Q^\star(\lambda, \boldsymbol{d}, \boldsymbol{\alpha})$.

Similarly we can prove that for any local maximizer $(\hat{\boldsymbol{d}}^\star, \hat{\boldsymbol{\alpha}}^\star)$ of $Q^\star(\lambda, \boldsymbol{d}, \boldsymbol{\alpha})$, there is a corresponding local maximizer $(\hat{\boldsymbol{d}}^*, \hat{\boldsymbol{\alpha}}^*)$ of $Q^*(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{\alpha})$ such that $\hat{d}_k^*\hat{\alpha}_{kj}^* = \hat{d}_k^\star\hat{\alpha}_{kj}^\star$.

**Lemma II.7.** *Suppose $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{\alpha}})$ is a local maximizer of 2.8. Let $\hat{\boldsymbol{\beta}}$ be the Hierarchical Lasso estimate related to $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{\alpha}})$, i.e., $\hat{\beta}_{kj} = \hat{d}_k\hat{\alpha}_{kj}$. If $\hat{d}_k = 0$, then $\hat{\boldsymbol{\alpha}}_k = 0$; if $\hat{d}_k \neq 0$, then $\|\hat{\boldsymbol{\beta}}_k\|_1 \neq 0$ and $\hat{d}_k = \sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k\|_1}, \hat{\boldsymbol{\alpha}}_k = \frac{\hat{\boldsymbol{\beta}}_k}{\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k\|_1}}$.*

**Proof of Lemma II.7**

If $\hat{d}_k = 0$, then $\hat{\boldsymbol{\alpha}}_k = 0$ is quite obvious. Similarly, if $\hat{\boldsymbol{\alpha}}_k = 0$, then $\hat{d}_k = 0$. Therefore, if $\hat{d}_k \neq 0$, then $\hat{\boldsymbol{\alpha}}_k \neq 0$ and $\|\hat{\boldsymbol{\beta}}_k\|_1 \neq 0$.

We prove $\hat{d}_k = \sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k\|_1}, \hat{\boldsymbol{\alpha}}_k = \frac{\hat{\boldsymbol{\beta}}_k}{\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k\|_1}}$ for $\hat{d}_k \neq 0$ by contradiction. Suppose $\exists k'$ such that $\hat{d}_{k'} \neq 0$ and $\hat{d}_{k'} \neq \sqrt{\lambda\|\hat{\boldsymbol{\beta}}_{k'}\|_1}$. Let $\frac{\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_{k'}\|_1}}{\hat{d}_{k'}} = c$. Then $\hat{\boldsymbol{\alpha}}_k = c\frac{\hat{\boldsymbol{\beta}}_k}{\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k\|_1}}$. Suppose $c > 1$.

Let $\tilde{d}_k = \hat{d}_k$ and $\tilde{\boldsymbol{\alpha}}_k = \hat{\boldsymbol{\alpha}}_k$ for $k \neq k'$; and let $\tilde{d}_{k'} = \delta' \hat{d}_{k'}$ and $\tilde{\boldsymbol{\alpha}}_{k'} = \hat{\boldsymbol{\alpha}}_{k'} \frac{1}{\delta'}$, where $\delta'$ satisfies $c > \delta' > 1$ and is very close to 1 such that $\|\tilde{d}_{k'} - \hat{d}_{k'}\|_1 + \|\tilde{\boldsymbol{\alpha}}_{k'} - \hat{\boldsymbol{\alpha}}_{k'}\|_1 < \delta$ for some $\delta > 0$.

Then we have

$$\begin{aligned}
Q^\star(\lambda, \tilde{\boldsymbol{d}}, \tilde{\boldsymbol{\alpha}}) - Q^\star(\lambda, \hat{\boldsymbol{d}}, \hat{\boldsymbol{\alpha}}) &= -\delta' |\hat{d}_{k'}| - \frac{1}{\delta'} \lambda \|\hat{\boldsymbol{\alpha}}_{k'}\|_1 + |\hat{d}_{k'}| + \lambda \|\hat{\boldsymbol{\alpha}}_{k'}\|_1 \\
&= \left( -\frac{\delta'}{c} - \frac{c}{\delta'} + \frac{1}{c} + c \right) \sqrt{\lambda \|\hat{\boldsymbol{\beta}}_{k'}\|_1} \\
&= \frac{1}{c} (\delta' - 1) \left( \frac{c^2}{\delta'} - 1 \right) \sqrt{\lambda \|\hat{\boldsymbol{\beta}}_{k'}\|_1} \\
&> 0.
\end{aligned}$$

Therefore, for any $\delta > 0$, we can find $\tilde{\boldsymbol{d}}, \tilde{\boldsymbol{\alpha}}$ such that $\|\tilde{\boldsymbol{d}} - \hat{\boldsymbol{d}}\|_1 + \|\tilde{\boldsymbol{\alpha}} - \hat{\boldsymbol{\alpha}}\|_1 < \delta$ and $Q^\star(\lambda, \tilde{\boldsymbol{d}}, \tilde{\boldsymbol{\alpha}}) > Q^\star(\lambda, \hat{\boldsymbol{d}}, \hat{\boldsymbol{\alpha}})$. These contradict with $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{\alpha}})$ being a local maximizer.

Similarly for the case when $c < 1$. Hence, we have the result that if $\hat{d}_k \neq 0$, then $\hat{d}_k = \sqrt{\lambda \|\hat{\boldsymbol{\beta}}_k\|_1}, \hat{\boldsymbol{\alpha}}_k = \frac{\hat{\boldsymbol{\beta}}_k}{\sqrt{\lambda \|\hat{\boldsymbol{\beta}}_k\|_1}}$.

**Proof of Theorem II.2**

Let $Q(\lambda, \boldsymbol{\beta})$ be the corresponding criterion in equation (2.11).

Suppose $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{\alpha}})$ is a local maximizer of $Q^\star(\lambda, \boldsymbol{d}, \boldsymbol{\alpha})$, we first show that $\hat{\boldsymbol{\beta}}$, where $\hat{\beta}_{kj} = \hat{d}_k \hat{\alpha}_{kj}$, is a local maximizer of $Q(\lambda, \boldsymbol{\beta})$, i.e. there exists a $\delta'$ such that if $\|\triangle \boldsymbol{\beta}\|_1 < \delta'$ then $Q(\lambda, \hat{\boldsymbol{\beta}} + \triangle \boldsymbol{\beta}) \leq Q(\lambda, \hat{\boldsymbol{\beta}})$.

We denote $\triangle \boldsymbol{\beta} = \triangle \boldsymbol{\beta}^{(1)} + \triangle \boldsymbol{\beta}^{(2)}$, where $\triangle \boldsymbol{\beta}_k^{(1)} = 0$ if $\|\hat{\boldsymbol{\beta}}_k\|_1 = 0$ and $\triangle \boldsymbol{\beta}_k^{(2)} = 0$ if $\|\hat{\boldsymbol{\beta}}_k\|_1 \neq 0$. We have $\|\triangle \boldsymbol{\beta}\|_1 = \|\triangle \boldsymbol{\beta}^{(1)}\|_1 + \|\triangle \boldsymbol{\beta}^{(2)}\|_1$.

Now we show $Q(\lambda, \hat{\boldsymbol{\beta}} + \triangle \boldsymbol{\beta}^{(1)}) \leq Q(\lambda, \hat{\boldsymbol{\beta}})$ if $\delta'$ is small enough. By Lemma II.7, we have $\hat{d}_k = \sqrt{\lambda \|\hat{\boldsymbol{\beta}}_k\|_1}$ and $\hat{\boldsymbol{\alpha}}_k = \frac{\hat{\boldsymbol{\beta}}_k}{\sqrt{\lambda \|\hat{\boldsymbol{\beta}}_k\|_1}}$ if $\|\hat{d}_k\|_1 \neq 0$; and $\hat{\boldsymbol{\alpha}}_k = \boldsymbol{0}$ if $\|\hat{d}_k\|_1 = 0$. Furthermore, let $\hat{d}'_k = \sqrt{\lambda \|\hat{\boldsymbol{\beta}}_k + \triangle \boldsymbol{\beta}_k^{(1)}\|_1}, \hat{\boldsymbol{\alpha}}'_k = \frac{\hat{\boldsymbol{\beta}}_k + \triangle \boldsymbol{\beta}_k^{(1)}}{\sqrt{\lambda \|\hat{\boldsymbol{\beta}}_k + \triangle \boldsymbol{\beta}_k^{(1)}\|_1}}$ if $\|\hat{d}_k\|_1 \neq 0$. Let $\hat{d}'_k = 0, \hat{\boldsymbol{\alpha}}'_k = \boldsymbol{0}$ if $\|\hat{d}_k\|_1 = 0$. Then we have $Q^\star(\lambda, \hat{\boldsymbol{d}}', \hat{\boldsymbol{\alpha}}') = Q(\lambda, \hat{\boldsymbol{\beta}} + \triangle \boldsymbol{\beta}^{(1)})$ and

$Q^\star(\lambda, \hat{\boldsymbol{d}}, \hat{\boldsymbol{\alpha}}) = Q(\lambda, \hat{\boldsymbol{\beta}})$. Hence we only need to show that $Q^\star(\lambda, \hat{\boldsymbol{d}}', \hat{\boldsymbol{\alpha}}') \leq Q^\star(\lambda, \hat{\boldsymbol{d}}, \hat{\boldsymbol{\alpha}})$. Note that $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{\alpha}})$ ia a local maximizer of $Q^\star(\lambda, \boldsymbol{d}, \boldsymbol{\alpha})$. Therefore there exists a $\delta$ such that for any $\boldsymbol{d}', \boldsymbol{\alpha}'$ satisfying $\|\boldsymbol{d}' - \hat{\boldsymbol{d}}\|_1 + \|\boldsymbol{\alpha}' - \hat{\boldsymbol{\alpha}}\|_1 < \delta$, we have $Q^\star(\lambda, \boldsymbol{d}', \boldsymbol{\alpha}') \leq Q^\star(\lambda, \hat{\boldsymbol{d}}, \hat{\boldsymbol{\alpha}})$.

Now since

$$\begin{aligned}
|\hat{d}_k' - \hat{d}_k| &= |\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k + \triangle\boldsymbol{\beta}_k^{(1)}\|_1} - \sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k\|_1}| \\
&\leq |\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k\|_1 - \lambda\|\triangle\boldsymbol{\beta}_k^{(1)}\|_1} - \sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k\|_1}| \\
&\leq \frac{1}{2}\frac{\lambda\|\triangle\boldsymbol{\beta}_k^{(1)}\|_1}{\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k\|_1 - \lambda\|\triangle\boldsymbol{\beta}_k^{(1)}\|_1}} \\
&\leq \frac{1}{2}\frac{\lambda\|\triangle\boldsymbol{\beta}_k^{(1)}\|_1}{\sqrt{\lambda a - \lambda\delta'}} \\
&\leq \frac{1}{2}\frac{\lambda\|\triangle\boldsymbol{\beta}_k^{(1)}\|_1}{\sqrt{\lambda a/2}},
\end{aligned}$$

where $a = \min\{\|\hat{\boldsymbol{\beta}}_k\|_1 : \|\hat{\boldsymbol{\beta}}_k\|_1 \neq 0\}$ and $\delta' < a/2$.

Furthermore

$$\begin{aligned}
\|\hat{\boldsymbol{\alpha}}_k' - \hat{\boldsymbol{\alpha}}_k\|_1 &= \left\| \frac{\hat{\boldsymbol{\beta}}_k + \triangle\boldsymbol{\beta}_k^{(1)}}{\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k + \triangle\boldsymbol{\beta}_k^{(1)}\|_1}} - \frac{\hat{\boldsymbol{\beta}}_k}{\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k\|_1}} \right\|_1 \\
&\leq \left\| \frac{\hat{\boldsymbol{\beta}}_k + \triangle\boldsymbol{\beta}_k^{(1)}}{\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k + \triangle\boldsymbol{\beta}_k^{(1)}\|_1}} - \frac{\hat{\boldsymbol{\beta}}_k}{\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k + \triangle\boldsymbol{\beta}_k^{(1)}\|_1}} \right\|_1 \\
&\quad + \left\| \frac{\hat{\boldsymbol{\beta}}_k}{\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k + \triangle\boldsymbol{\beta}_k^{(1)}\|_1}} - \frac{\hat{\boldsymbol{\beta}}_k}{\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k\|_1}} \right\|_1 \\
&\leq \frac{\|\triangle\boldsymbol{\beta}_k^{(1)}\|_1}{\sqrt{\lambda a/2}} \\
&\quad + \frac{\|\hat{\boldsymbol{\beta}}_k\|_1 |\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k + \triangle\boldsymbol{\beta}_k^{(1)}\|_1} - \sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k\|_1}|}{\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k + \triangle\boldsymbol{\beta}_k^{(1)}\|_1}\sqrt{\lambda\|\hat{\boldsymbol{\beta}}_k\|_1}} \\
&\leq \frac{\|\triangle\boldsymbol{\beta}_k^{(1)}\|_1}{\sqrt{\lambda a/2}} + \frac{b}{\sqrt{\lambda a/2}\sqrt{\lambda a}} \left( \frac{1}{2} \frac{\lambda\|\triangle\boldsymbol{\beta}_k^{(1)}\|_1}{\sqrt{\lambda a/2}} \right) \\
&\leq \|\triangle\boldsymbol{\beta}_k^{(1)}\|_1 \left( \frac{1}{\sqrt{\lambda a/2}} + \frac{b}{a\sqrt{\lambda a}} \right),
\end{aligned}$$

where $b = \max\{\|\hat{\boldsymbol{\beta}}_k\|_1 : \|\hat{\boldsymbol{\beta}}_k\|_1 \neq 0\}$.

Therefore, there exists a small enough $\delta'$, if $\|\triangle\boldsymbol{\beta}^{(1)}\|_1 < \delta'$ we have $\|\hat{\boldsymbol{d}}' - \hat{\boldsymbol{d}}\|_1 + \|\hat{\boldsymbol{\alpha}}' - \hat{\boldsymbol{\alpha}}\|_1 < \delta$. Hence $Q^\star(\lambda, \hat{\boldsymbol{d}}', \hat{\boldsymbol{\alpha}}') \leq Q^\star(\lambda, \hat{\boldsymbol{d}}, \hat{\boldsymbol{\alpha}})$ (due to local maximality) and $Q(\lambda, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)}) \leq Q(\lambda, \hat{\boldsymbol{\beta}})$.

Next we show $Q(\lambda, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)} + \triangle\boldsymbol{\beta}^{(2)}) \leq Q(\lambda, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)})$. Note that

$$Q(\lambda, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)} + \triangle\boldsymbol{\beta}^{(2)}) - Q(\lambda, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)}) = \triangle\boldsymbol{\beta}^{(2)^\intercal} \nabla L(\hat{\boldsymbol{\beta}}^*) - \sum_{k=1}^{K} \sqrt{\lambda\|\triangle\boldsymbol{\beta}^{(2)}\|_1},$$

where $\boldsymbol{\beta}^*$ is a vector between $\hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)} + \triangle\boldsymbol{\beta}^{(2)}$ and $\hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)}$. Since $\|\triangle\boldsymbol{\beta}^{(2)}\|_1 < \delta'$ is small enough, the second term dominates the first term, hence we have $Q(\lambda, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)} + \triangle\boldsymbol{\beta}^{(2)}) \leq Q(\lambda, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)})$.

Overall, we have that there exists a small enough $\delta'$, if $\|\triangle\boldsymbol{\beta}\|_1 < \delta'$, then $Q(\lambda, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}) \leq Q(\lambda, \hat{\boldsymbol{\beta}})$, which implies that $\hat{\boldsymbol{\beta}}$ is a local maximizer of $Q(\lambda, \boldsymbol{\beta})$.

Similarly, we can prove that if $\hat{\boldsymbol{\beta}}$ is a local maximizer of $Q(\lambda, \boldsymbol{\beta})$, and if we let $\hat{d}_k = \sqrt{\lambda \|\hat{\boldsymbol{\beta}}_k\|_1}$ and $\hat{\boldsymbol{\alpha}}_k = \frac{\hat{\boldsymbol{\beta}}_k}{\sqrt{\lambda \|\hat{\boldsymbol{\beta}}_k\|_1}}$ for $\|\hat{\boldsymbol{\beta}}_k\|_1 \neq 0$, and let $\hat{d}_k = 0$ and $\hat{\boldsymbol{\alpha}}_k = \mathbf{0}$ for $\|\hat{\boldsymbol{\beta}}_k\|_1 = 0$, then $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{\alpha}})$ is a local maximizer of $Q^{\star}(\lambda, \boldsymbol{d}, \boldsymbol{\alpha})$.

**Regularity Conditions**

Let $S_n$ be the number of non-zero groups, i.e., $\|\boldsymbol{\beta}^0_{nk}\| \neq 0$. Without loss of generality, we assume

$$\begin{aligned} \|\boldsymbol{\beta}^0_{nk}\| &\neq 0, \text{ for } k = 1, \ldots, S_n, \\ \|\boldsymbol{\beta}^0_{nk}\| &= 0, \text{ for } k = S_n + 1, \ldots, K_n. \end{aligned}$$

Let $s_{nk}$ be the number of non-zero coefficients in group $k, 1 \leq k \leq S_n$; again, without loss of generality, we assume

$$\begin{aligned} \beta^0_{n,kj} &\neq 0, \text{ for } k = 1, \ldots, S_n;\ j = 1, \ldots, s_{nk}, \\ \beta^0_{n,kj} &= 0, \text{ for } k = 1, \ldots, S_n;\ j = s_{nk} + 1, \ldots, p_{nk}. \end{aligned}$$

For simplicity, we write $\beta_{n,kj}$, $p_{nk}$ and $s_{nk}$ as $\beta_{kj}$, $p_k$ and $s_k$ in the following.

Since we have diverging number of parameters, to keep the uniform properties of the likelihood function, we need some conditions on the higher-order moment of the likelihood function, as compared to the usual condition in the asymptotic theory of the likelihood estimate under finite parameters [35].

(A1) For every $n$, the observations $\{\boldsymbol{V}_{ni}, i = 1, 2, \ldots, n\}$ are independent and identically distributed, each with a density $f_n(\boldsymbol{V}_{n1}, \boldsymbol{\beta}_n)$. Here $f_n(\boldsymbol{V}_{n1}, \boldsymbol{\beta}_n)$ has a common support and the model is identifiable. Furthermore, the first and second logarithmic derivatives of $f_n$ satisfy the equations

$$\begin{aligned}
\mathrm{E}_{\boldsymbol{\beta}_n}\left[\frac{\partial \log f_n(\boldsymbol{V}_{n1},\boldsymbol{\beta}_n)}{\partial \beta_{kj}}\right] &= 0, \text{ for } k=1,\ldots,K_n;\ j=1,\ldots,p_k \\
\boldsymbol{I}_{k_1j_1k_2j_2}(\boldsymbol{\beta}_n) &= \mathrm{E}_{\boldsymbol{\beta}_n}\left[\frac{\partial}{\partial \beta_{k_1j_1}}\log f_n(\boldsymbol{V}_{n1},\boldsymbol{\beta}_n)\frac{\partial}{\partial \beta_{k_2j_2}}\log f_n(\boldsymbol{V}_{n1},\boldsymbol{\beta}_n)\right] \\
&= \mathrm{E}_{\boldsymbol{\beta}_n}\left[-\frac{\partial^2}{\partial \beta_{k_1j_2}\partial \beta_{k_2j_2}}\log f_n(\boldsymbol{V}_{n1},\boldsymbol{\beta}_n)\right].
\end{aligned}$$

(A2) The Fisher information matrix

$$\boldsymbol{I}(\boldsymbol{\beta}_n) = \mathrm{E}_{\boldsymbol{\beta}_n}\left[\frac{\partial}{\partial \boldsymbol{\beta}_n}\log f_n(\boldsymbol{V}_{n1},\boldsymbol{\beta}_n)\frac{\partial^{\mathsf{T}}}{\partial \boldsymbol{\beta}_n}\log f_n(\boldsymbol{V}_{n1},\boldsymbol{\beta}_n)\right]$$

satisfies the condition

$$0 < C_1 < \lambda_{\min}\{\boldsymbol{I}(\boldsymbol{\beta}_n)\} \le \lambda_{\max}\{\boldsymbol{I}(\boldsymbol{\beta}_n)\} < C_2 < \infty,$$

and for any $k_1, j_1, k_2, j_2$, we have

$$\begin{aligned}
\mathrm{E}_{\boldsymbol{\beta}_n}\left[\frac{\partial}{\partial \beta_{k_1j_1}}\log f_n(\boldsymbol{V}_{n1},\boldsymbol{\beta}_n)\frac{\partial}{\partial \beta_{k_2j_2}}\log f_n(\boldsymbol{V}_{n1},\boldsymbol{\beta}_n)\right]^2 &< C_3 < \infty, \\
\mathrm{E}_{\boldsymbol{\beta}_n}\left[-\frac{\partial^2}{\partial \beta_{k_1j_1}\partial \beta_{k_2j_2}}\log f_n(\boldsymbol{V}_{n1},\boldsymbol{\beta}_n)\right]^2 &< C_4 < \infty.
\end{aligned}$$

(A3) There exists an open subset $\omega_n$ of $\Omega_n \in R^{P_n}$ that contains the true parameter point $\boldsymbol{\beta}_n^0$ such that for almost all $\boldsymbol{V}_{n1}$, the density $f_n(\boldsymbol{V}_{n1},\boldsymbol{\beta}_n)$ admits all third derivatives $\partial^3 f_n(\boldsymbol{V}_{n1},\boldsymbol{\beta}_n)/(\partial\beta_{k_1j_1}\partial\beta_{k_2j_2}\partial\beta_{k_3j_3})$ for all $\boldsymbol{\beta}_n \in \omega_n$. Furthermore, there exist functions $M_{nk_1j_1k_2j_2k_3j_3}$ such that

$$\left|\frac{\partial^3}{\partial \beta_{k_1j_1}\partial \beta_{k_2j_2}\partial \beta_{k_3j_3}}\log f_n(\boldsymbol{V}_{n1},\boldsymbol{\beta}_n)\right| \le M_{nk_1j_1k_2j_2k_3j_3}(\boldsymbol{V}_{n1}) \text{ for all } \boldsymbol{\beta}_n \in \omega_n,$$

and $\mathrm{E}_{\boldsymbol{\beta}_n}[M^2_{nk_1j_1k_2j_2k_3j_3}(\boldsymbol{V}_{n1})] < C_5 < \infty$.

These regularity conditions guarantee the asymptotic normality of the ordinary maximum likelihood estimates for diverging number of parameters.

For expositional simplicity, we will first prove Theorem II.4 and Theorem II.5, then prove Theorem II.3.

**Proof of Theorem II.4**

We will show that for any given $\epsilon > 0$, there exists a constant $C$ such that

$$\Pr\left\{ \sup_{\|\boldsymbol{u}\|=C} Q_n(\boldsymbol{\beta}_n^0 + \alpha_n \boldsymbol{u}) < Q_n(\boldsymbol{\beta}_n^0) \right\} \geq 1 - \epsilon, \tag{2.18}$$

where $\alpha_n = \sqrt{P_n}(n^{-1/2} + \lambda_n\sqrt{a_n}/2\sqrt{c_1})$. This implies that with probability at least $1-\epsilon$, that there exists a local maximum in the ball $\{\boldsymbol{\beta}_n^0 + \alpha_n\boldsymbol{u} : \|\boldsymbol{u}\| \leq C\}$. Hence, there exists a local maximizer such that $\|\hat{\boldsymbol{\beta}}_n - \boldsymbol{\beta}_n^0\| = O_p(\alpha_n)$. Since $1/2\sqrt{c_1}$ is a constant, we have $\|\hat{\boldsymbol{\beta}}_n - \boldsymbol{\beta}_n^0\| = O_p(\sqrt{P_n}(n^{-1/2} + \lambda_n\sqrt{a_n}))$.

Using $p_{\lambda_n,\boldsymbol{w}_n}(0) = 0$, we have

$$\begin{aligned} D_n(\boldsymbol{u}) &= Q_n(\boldsymbol{\beta}_n^0 + \alpha_n\boldsymbol{u}) - Q_n(\boldsymbol{\beta}_n^0) \\ &\leq L_n(\boldsymbol{\beta}_n^0 + \alpha_n\boldsymbol{u}) - L_n(\boldsymbol{\beta}_n^0) \\ &\quad -n\sum_{k=1}^{S_n}(p_{\lambda_n,\boldsymbol{w}_n}(\boldsymbol{\beta}_{nk}^0 + \alpha_n\boldsymbol{u}_k) - p_{\lambda_n,\boldsymbol{w}_n}(\boldsymbol{\beta}_{nk}^0)) \\ &\triangleq (I) + (II). \end{aligned} \tag{2.19}$$

Using the standard argument on the Taylor expansion of the likelihood function, we have

$$\begin{aligned} (I) &= \alpha_n\boldsymbol{u}^{\intercal}\nabla L_n(\boldsymbol{\beta}_n^0) + \frac{1}{2}\boldsymbol{u}^{\intercal}\nabla^2 L_n(\boldsymbol{\beta}_n^0)\boldsymbol{u}\alpha_n^2 + \frac{1}{6}\boldsymbol{u}^{\intercal}\nabla\{\boldsymbol{u}^{\intercal}\nabla^2 L_n(\boldsymbol{\beta}_n^*)\boldsymbol{u}\}\alpha_n^3 \\ &\triangleq I_1 + I_2 + I_3, \end{aligned} \tag{2.20}$$

where $\boldsymbol{\beta}_n^*$ lies between $\boldsymbol{\beta}_n^0$ and $\boldsymbol{\beta}_n^0 + \alpha_n\boldsymbol{u}$. Using the same argument as in the proof of Theorem 1 of [21], we have

$$|I_1| = O_p(\alpha_n^2 n)\|\boldsymbol{u}\|, \tag{2.21}$$

$$I_2 = -\frac{n\alpha_n^2}{2}\boldsymbol{u}^{\intercal}\boldsymbol{I}_n(\boldsymbol{\beta}_n^0)\boldsymbol{u} + o_p(1)n\alpha_n^2\|\boldsymbol{u}\|^2, \tag{2.22}$$

and

$$\begin{aligned}
|I_3| &= \left| \frac{1}{6} \sum_{k_1=1}^{K_n} \sum_{j_1=1}^{p_k} \sum_{k_2=1}^{K_n} \sum_{j_2=1}^{p_k} \sum_{k_3=1}^{K_n} \sum_{j_3=1}^{p_k} \frac{\partial^3 L_n(\beta_n^*)}{\partial \beta_{k_1 j_1} \partial \beta_{k_2 j_2} \partial \beta_{k_3 j_3}} u_{k_1 j_1} u_{k_2 j_2} u_{k_3 j_3} \alpha_n^3 \right| \\
&\leq \frac{1}{6} \sum_{i=1}^{n} \left\{ \sum_{k_1=1}^{K_n} \sum_{j_1=1}^{p_k} \sum_{k_2=1}^{K_n} \sum_{j_2=1}^{p_k} \sum_{k_3=1}^{K_n} \sum_{j_3=1}^{p_k} M^2_{nk_1 j_1 k_2 j_2 k_3 j_3}(V_{ni}) \right\}^{1/2} \|\boldsymbol{u}\|^3 \alpha_n^3 \\
&= O_p(P_n^{3/2} \alpha_n) n \alpha_n^2 \|\boldsymbol{u}\|^3.
\end{aligned}$$

Since $\frac{P_n^4}{n} \to 0$ and $P_n^2 \lambda_n \sqrt{a_n} \to 0$ as $n \to \infty$, we have

$$|I_3| = o_p(n\alpha_n^2)\|\boldsymbol{u}\|^3. \tag{2.23}$$

From (2.21)-(2.23), we can see that, by choosing a sufficiently large $C$, the first term in $I_2$ dominates $I_1$ uniformly on $\|\boldsymbol{u}\| = C$; when $n$ is large enough, $I_2$ also dominates $I_3$ uniformly on $\|\boldsymbol{u}\| = C$.

Now we consider $(II)$. Since $\alpha_n = \sqrt{P_n}(n^{-1/2} + \lambda_n \sqrt{a_n}/2\sqrt{c_1}) \to 0$, for $\|\boldsymbol{u}\| \leq C$ we have

$$|\beta_{kj}^0 + \alpha_n u_{kj}| \geq |\beta_{kj}^0| - |\alpha_n u_{kj}| > 0 \tag{2.24}$$

for $n$ large enough and $\beta_{kj}^0 \neq 0$. Hence, we have

$$\begin{aligned}
& p_{\lambda_n, \boldsymbol{w}_n}(\boldsymbol{\beta}_{nk}^0 + \alpha_n \boldsymbol{u}_k) - p_{\lambda_n, \boldsymbol{w}_n}(\boldsymbol{\beta}_{nk}^0) \\
={}& \lambda_n \Big( \sqrt{w_{n,k1}|\beta_{k1}^0 + \alpha_n u_{k1}| + \ldots + w_{n,kp_k}|\beta_{kp_k}^0 + \alpha_n u_{kp_k}|} \\
& - \sqrt{w_{n,k1}|\beta_{k1}^0| + \ldots + w_{n,kp_k}|\beta_{kp_k}^0|} \Big) \\
\geq{}& \lambda_n \Big( \sqrt{w_{n,k1}|\beta_{k1}^0 + \alpha_n u_{k1}| + \ldots + w_{n,ks_k}|\beta_{ks_k}^0 + \alpha_n u_{ks_k}|} \\
& - \sqrt{w_{n,k1}|\beta_{k1}^0| + \ldots + w_{n,ks_k}|\beta_{ks_k}^0|} \Big) \\
\geq{}& \lambda_n \Big( \sqrt{w_{n,k1}|\beta_{k1}^0| + \ldots + w_{n,ks_k}|\beta_{ks_k}^0| - \alpha_n (w_{n,k1}|u_{k1}| + \ldots + w_{n,ks_k}|u_{ks_k}|)} \\
& - \sqrt{w_{n,k1}|\beta_{k1}^0| + \ldots + w_{n,ks_k}|\beta_{ks_k}^0|} \Big) \text{ (for n large enough, by (2.24))} \\
={}& \lambda_n \sqrt{w_{n,k1}|\beta_{k1}^0| + \ldots + w_{n,ks_k}|\beta_{ks_k}^0|} \left( \sqrt{1 - \gamma_{nk}} - 1 \right),
\end{aligned}$$

where $\gamma_{nk}$ is defined as $\gamma_{nk} = \frac{\alpha_n(w_{n,k1}|u_{k1}|+\ldots+w_{n,ks_k}|u_{ks_k}|)}{w_{n,k1}|\beta^0_{k1}|+\ldots+w_{n,ks_k}|\beta^0_{ks_k}|}$. For $n$ large enough, we have $0 \leq \gamma_{nk} < 1$ and $\gamma_{nk} \leq \frac{\alpha_n\|\boldsymbol{u}_k\|(w_{n,k1}+\ldots+w_{n,ks_k})}{c_1(w_{n,k1}+\ldots+w_{n,ks_k})} = \frac{\alpha_n\|\boldsymbol{u}_k\|}{c_1} \leq \frac{\alpha_n C}{c_1} \to 0$ with probability tending to 1 as $n \to \infty$.

Therefore,

$$\begin{aligned}
& p_{\lambda_n,\boldsymbol{w}_n}(\boldsymbol{\beta}^0_{nk} + \alpha_n\boldsymbol{u}_k) - p_{\lambda_n,\boldsymbol{w}_n}(\boldsymbol{\beta}^0_{nk}) \\
\geq\ & \lambda_n\sqrt{w_{n,k1}|\beta^0_{k1}| + \ldots + w_{n,ks_k}|\beta^0_{ks_k}|}(\sqrt{1-\gamma_{nk}} - 1) \\
\geq\ & \lambda_n\sqrt{w_{n,k1}|\beta^0_{k1}| + \ldots + w_{n,ks_k}|\beta^0_{ks_k}|}\left(\frac{1+|o_p(1)|}{2}(-\gamma_{nk})\right) \\
& \text{(Using } \gamma_{nk} = o_p(1) \text{ and Taylor expansion)} \\
\geq\ & -\lambda_n\frac{\alpha_n(w_{n,k1}|u_{k1}| + \ldots + w_{n,ks_k}|u_{ks_k}|)}{\sqrt{w_{n,k1}|\beta^0_{k1}| + \ldots + w_{n,ks_k}|\beta^0_{ks_k}|}}\left(\frac{1+|o_p(1)|}{2}\right) \\
\geq\ & -\alpha_n\lambda_n\frac{\|\boldsymbol{u}_k\|\sqrt{a_n s_k}}{2\sqrt{c_1}}(1+|o_p(1)|).
\end{aligned}$$

Therefore, the term $(II)$ in (2.19) is bounded by

$$n\alpha_n\lambda_n\left(\sum_{k=1}^{S_n}\frac{\|\boldsymbol{u}_k\|\sqrt{a_n s_k}}{2\sqrt{c_1}}\right)(1+|o_p(1)|),$$

which is further bounded by

$$n\alpha_n\lambda_n\sqrt{a_n}(\|\boldsymbol{u}\| \cdot \frac{\sqrt{P_n}}{2\sqrt{c_1}})(1+|o_p(1)|).$$

Note that $\alpha_n = \sqrt{P_n}(n^{-1/2} + \lambda_n\sqrt{a_n}/2\sqrt{c_1})$, hence the above expression is bounded by

$$\|\boldsymbol{u}\|n\alpha_n^2(1+|o_p(1)|).$$

This term is also dominated by the first term of $I_2$ on $\|\boldsymbol{u}\| = C$ uniformly. Therefore, $D_n(\boldsymbol{u}) < 0$ is satisfied uniformly on $\|\boldsymbol{u}\| = C$. This completes the proof of the theorem.

**Proof of Theorem II.5**

We have proved that if $\lambda_n\sqrt{a_n} = O_p(n^{-1/2})$, there exists a root-$(n/P_n)$ consistent estimate $\hat{\boldsymbol{\beta}}_n$. Now we prove that this root-$(n/P_n)$ consistent estimate has the oracle sparsity under the condition $\frac{P_n^2}{\lambda_n^2 b_n} = o_p(n)$, i.e., $\hat{\beta}_{kj} = 0$ with probability tending to 1 if $\beta^0_{kj} = 0$.

Using Taylor's expansion, we have

$$\begin{aligned}
\frac{\partial Q_n(\boldsymbol{\beta}_n)}{\partial \beta_{kj}} &= \frac{\partial L_n(\boldsymbol{\beta}_n)}{\partial \beta_{kj}} - n\frac{\partial p_{\lambda_n,\boldsymbol{w}_n}(\boldsymbol{\beta}_{nk})}{\partial \beta_{kj}} \\
&= \frac{\partial L_n(\boldsymbol{\beta}^0_n)}{\partial \beta_{kj}} + \sum_{k_1=1}^{K_n}\sum_{j_1=1}^{p_{k_1}} \frac{\partial^2 L_n(\boldsymbol{\beta}^0)}{\partial \beta_{kj}\partial \beta_{k_2 j_2}}(\beta_{k_1 j_1} - \beta^0_{k_1 j_1}) \\
&\quad + \frac{1}{2}\sum_{k_1=1}^{K_n}\sum_{j_1=1}^{p_{k_1}}\sum_{k_2=1}^{K_n}\sum_{j_2=1}^{p_{k_2}} \frac{\partial^3 L_n(\boldsymbol{\beta}^*_n)}{\partial \beta_{kj}\partial \beta_{k_1 j_1}\partial \beta_{k_2 j_2}}(\beta_{k_1 j_1} - \beta^0_{k_1 j_1})(\beta_{k_2 j_2} - \beta^0_{k_2 j_2}) \\
&\quad - \frac{n\lambda_n w_{n,kj}}{2\sqrt{w_{n,k1}|\beta_{k1}| + \ldots + w_{n,kp_k}|\beta_{kp_k}|}}\text{sgn}(\beta_{kj}) \qquad (2.25) \\
&\triangleq I_1 + I_2 + I_3 + I_4,
\end{aligned}$$

where $\boldsymbol{\beta}^*_n$ lies between $\boldsymbol{\beta}_n$ and $\boldsymbol{\beta}^0_n$.

Using the argument in the proof of Lemma 5 of [21], for any $\boldsymbol{\beta}_n$ satisfying $\|\boldsymbol{\beta}_n - \boldsymbol{\beta}^0_n\| = O_p(\sqrt{P_n/n})$, we have

$$\begin{aligned}
I_1 &= O_p(\sqrt{n}) = O_p(\sqrt{nP_n}), \\
I_2 &= O_p(\sqrt{nP_n}), \\
I_3 &= o_p(\sqrt{nP_n}).
\end{aligned}$$

Then, since $\hat{\boldsymbol{\beta}}_n$ is a root-$(n/P_n)$ consistent estimate maximizing $Q_n(\boldsymbol{\beta}_n)$, if $\hat{\beta}_{kj} \neq 0$, we have

$$\begin{aligned}
\left.\frac{\partial Q_n(\boldsymbol{\beta}_n)}{\partial \beta_{kj}}\right|_{\boldsymbol{\beta}_n = \hat{\boldsymbol{\beta}}_n} &= O_p(\sqrt{nP_n}) - \frac{n\lambda_n w_{n,kj}}{2\sqrt{w_{n,k1}|\hat{\beta}_{k1}| + \ldots + w_{n,kp_k}|\hat{\beta}_{kp_k}|}}\text{sgn}(\hat{\beta}_{kj}) \\
&= 0. \qquad (2.26)
\end{aligned}$$

Therefore,

$$\frac{n\lambda_n w_{n,kj}}{\sqrt{w_{n,k1}|\hat{\beta}_{k1}| + \ldots + w_{n,kp_k}|\hat{\beta}_{kp_k}|}} = O_p(\sqrt{nP_n}) \text{ for } \hat{\beta}_{kj} \neq 0.$$

This can be extended to

$$\frac{n\lambda_n w_{n,kj}|\hat{\beta}_{kj}|}{\sqrt{w_{n,k1}|\hat{\beta}_{k1}| + \ldots + w_{n,kp_k}|\hat{\beta}_{kp_k}|}} = |\hat{\beta}_{kj}|O_p(\sqrt{nP_n}),$$

for any $\hat{\beta}_{kj}$ with $\hat{\boldsymbol{\beta}}_{nk} \neq 0$. If we sum this over all $j$ in the $k$th group, we have

$$n\lambda_n\sqrt{w_{n,k1}|\hat{\beta}_{k1}| + \ldots + w_{n,kp_k}|\hat{\beta}_{kp_k}|} = \sum_{j=1}^{p_k}|\hat{\beta}_{kj}|O_p(\sqrt{nP_n}). \tag{2.27}$$

Since $\hat{\boldsymbol{\beta}}_n$ is a root-$(n/P_n)$ consistent estimate of $\boldsymbol{\beta}_n^0$, we have $|\hat{\beta}_{kj}| = O_p(1)$ for $(k,j) \in \mathcal{A}_n$ and $|\hat{\beta}_{kj}| = O_p(\sqrt{P_n/n})$ for $(k,j) \in \mathcal{B}_n \cup \mathcal{C}_n$.

Now for any $k$ and $j$ satisfying $\beta_{kj}^0 = 0$ and $\hat{\beta}_{kj} \neq 0$, equation (2.26) can be written as:

$$\begin{aligned} \left.\frac{\partial Q_n(\boldsymbol{\beta}_n)}{\partial \beta_{kj}}\right|_{\boldsymbol{\beta}_n=\hat{\boldsymbol{\beta}}_n} &= \frac{1}{2\lambda_n\sqrt{w_{n,k1}|\hat{\beta}_{k1}| + \ldots + w_{n,kp_k}|\hat{\beta}_{kp_k}|}} \\ &\quad (O_p(\sqrt{P_n/n})n\lambda_n\sqrt{w_{n,k1}|\hat{\beta}_{k1}| + \ldots + w_{n,kp_k}|\hat{\beta}_{kp_k}|} \\ &\quad -n\lambda_n^2 w_{n,kj}\mathrm{sgn}(\hat{\beta}_{kj})) \\ &= 0. \end{aligned} \tag{2.28}$$

Denote $h_{nk} = O_p(\sqrt{P_n/n})n\lambda_n\sqrt{w_{n,k1}|\hat{\beta}_{k1}| + \ldots + w_{n,kp_k}|\hat{\beta}_{kp_k}|}$. Let $h_n = \sum_{k=1}^{K_n} h_{nk}$. By equation (2.27), we have $h_n = \sum_{k=1}^{K_n} O_p(\sqrt{P_n/n})\sum_{j=1}^{p_k}|\hat{\beta}_{kj}|O_p(\sqrt{nP_n}) = O_p(P_n^2)$. Since $\frac{P_n^2}{\lambda_n^2 b_n} = o_p(n)$ guarantees that $n\lambda_n^2 b_n$ dominates $h_n$ with probability tending to 1 as $n \to \infty$, the first term in (2.28) is dominated by the second term as $n \to \infty$ uniformly for all $k$ and $j$ satisfying $\beta_{kj}^0 = 0$ since $w_{n,kj} \geq b_n$ and $h_n > h_{nk}$. Denote $g_{nk} = 2\lambda_n\sqrt{w_{n,k1}|\hat{\beta}_{k1}| + \ldots + w_{n,kp_k}|\hat{\beta}_{kp_k}|}/(n\lambda_n^2 b_n)$. Let $g_n = \sum_{k=1}^{K_n} g_{nk}$. By equation (2.27), we have $g_n = 2\sum_{k=1}^{K_n}(1/n)\sum_{j=1}^{p_k}|\hat{\beta}_{kj}|O_p(\sqrt{nP_n})/(n\lambda_n^2 b_n) = o_p(1/\sqrt{nP_n})$. The absolute value of the second term in (2.28) is bounded below by $1/g_n$. So with probability uniformly converging to 1 the second term in the derivative $\frac{\partial Q(\boldsymbol{\beta})}{\partial \beta_{kj}}|_{\boldsymbol{\beta}=\hat{\boldsymbol{\beta}}_n}$ will go to $\infty$ as $n \to \infty$, which is a contradiction with equation (2.28). Therefore, for any $k$ and $j$ satisfying $\beta^0_{kj} = 0$, we have $\hat{\beta}_{kj} = 0$ with a probability tending to 1 as $n \to \infty$. We have $\hat{\boldsymbol{\beta}}_{\mathcal{D}_n} = 0$ with probability tending to 1 as well.

Now we prove the second part of Theorem II.5. From the above proof, we know that there exists $(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n}, \mathbf{0})$ with probability tending to 1, which is a root-$(n/P_n)$ consistent local maximizer of $Q(\boldsymbol{\beta}_n)$. With a slight abuse of notation, let $Q_n(\boldsymbol{\beta}_{n,\mathcal{A}_n}) = Q_n(\boldsymbol{\beta}_{n,\mathcal{A}_n}, \mathbf{0})$. Using the Taylor expansion on $\nabla Q_n(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n})$ at point $\boldsymbol{\beta}^0_{n,\mathcal{A}_n}$, we have

$$\begin{aligned}(2.29)\quad & \frac{1}{n}(\nabla^2 L_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{\beta}^0_{n,\mathcal{A}_n}) - \nabla J_n(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n})) \\ = \; & -\frac{1}{n}\left(\nabla L_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n}) + \frac{1}{2}(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{\beta}^0_{n,\mathcal{A}_n})^{\intercal} \nabla^2\{\nabla L_n(\boldsymbol{\beta}^*_{n,\mathcal{A}_n})\}(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{\beta}^0_{n,\mathcal{A}_n})\right),\end{aligned}$$

where $\boldsymbol{\beta}^*_{n,\mathcal{A}_n}$ lies between $\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n}$ and $\boldsymbol{\beta}^0_{n,\mathcal{A}_n}$.

Now we define

$$\mathcal{C}_n \triangleq \frac{1}{2}(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{\beta}^0_{n,\mathcal{A}_n})^{\intercal} \nabla^2\{\nabla L_n(\boldsymbol{\beta}^*_{n,\mathcal{A}_n})\}(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{\beta}^0_{n,\mathcal{A}_n}).$$

Using the Cauchy-Schwarz inequality, we have

$$\begin{aligned}\left\|\frac{1}{n}\mathcal{C}_n\right\|^2 \;&\leq\; \frac{1}{n^2}\sum_{i=1}^{n} n\|\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{\beta}^0_{n,\mathcal{A}_n}\|^4 \sum_{k_1=1}^{S_n}\sum_{j_1=1}^{p_k}\sum_{k_2=1}^{S_n}\sum_{j_2=1}^{p_k}\sum_{k_3=1}^{S_n}\sum_{j_3=1}^{p_k} M^3_{nk_1j_1k_2j_2k_3j_2}(\boldsymbol{V}_{ni}) \\ (2.30)\qquad &=\; O_p(P_n^2/n^2)O_p(P_n^3) = O_p(P_n^5/n^2) = o_p(1/n).\end{aligned}$$

Since $\frac{P_n^5}{n} \to 0$ as $n \to \infty$, by Lemma 8 of [21], we have

$$\left\|\frac{1}{n}\nabla^2 L_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n}) + \boldsymbol{I}_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})\right\| = o_p(1/P_n)$$

and

$$(2.31)\quad \left\|\left(\frac{1}{n}\nabla^2 L_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n}) + \boldsymbol{I}_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})\right)(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{\beta}^0_{n,\mathcal{A}_n})\right\| = o_p(1/\sqrt{nP_n}) = o_p(1/\sqrt{n}).$$

Since

$$
\begin{aligned}
& \sqrt{w_{n,k1}|\hat{\beta}_{k1}| + \ldots + w_{n,ks_k}|\hat{\beta}_{ks_k}|} \\
= \ & \sqrt{w_{n,k1}|\beta^0_{k1}|(1 + O_p(\sqrt{P_n/n})) + \ldots + w_{n,ks_k}|\beta^0_{ks_k}|(1 + O_p(\sqrt{P_n/n}))} \\
= \ & \sqrt{w_{n,k1}|\beta^0_{k1}| + \ldots + w_{n,ks_k}|\beta^0_{ks_k}|}(1 + O_p(\sqrt{P_n/n})),
\end{aligned}
$$

we have

$$
\frac{\lambda_n w_{n,kj}}{\sqrt{w_{n,k1}|\hat{\beta}_{k1}| + \ldots + w_{n,ks_k}|\hat{\beta}_{ks_k}|}} = \frac{\lambda_n w_{n,kj}}{\sqrt{w_{n,k1}|\beta^0_{k1}| + \ldots + w_{n,ks_k}|\beta^0_{ks_k}|}}(1+O_p(\sqrt{P_n/n})).
$$

Furthermore, since

$$
\frac{\lambda_n w_{n,kj}}{\sqrt{w_{n,k1}|\beta^0_{k1}| + \ldots + w_{n,ks_k}|\beta^0_{ks_k}|}} \leq \frac{\lambda_n w_{n,kj}}{\sqrt{w_{n,kj}c_1}} \leq \frac{\lambda_n \sqrt{a_n}}{\sqrt{c_1}} = o_p((nP_n)^{-1/2})
$$

for $(k, j) \in \mathcal{A}_n$, we have

$$
\left(\frac{1}{n}\nabla J_n(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n})\right)_{kj} = \frac{\lambda_n w_{n,kj}}{2\sqrt{w_{n,k1}|\hat{\beta}_{k1}| + \ldots + w_{n,ks_k}|\hat{\beta}_{ks_k}|}} = o_p((nP_n)^{-1/2})
$$

and

$$
\left\|\frac{1}{n}\nabla J_n(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n})\right\| \leq \sqrt{P_n} o_p((nP_n)^{-1/2}) = o_p(1/\sqrt{n}). \tag{2.32}
$$

Together with (2.30), (2.31) and (2.32), from (2.29) we have

$$
\boldsymbol{I}_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{\beta}^0_{n,\mathcal{A}_n}) = \frac{1}{n}\nabla L_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n}) + o_p(1/\sqrt{n}).
$$

Now using the same argument as in the proof of Theorem 2 of [21], we have

$$
\sqrt{n}\boldsymbol{A}_n\boldsymbol{I}_n^{1/2}(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{\beta}^0_{n,\mathcal{A}_n}) \to \sqrt{n}\boldsymbol{A}_n\boldsymbol{I}_n^{-1/2}(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})\left(\frac{1}{n}\nabla L_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})\right) \to \mathcal{N}(\mathbf{0}, \boldsymbol{G}),
$$

where $\boldsymbol{A}_n$ is a $q \times |\mathcal{A}_n|$ matrix such that $\boldsymbol{A}_n\boldsymbol{A}_n{}^\intercal \to \boldsymbol{G}$ and $\boldsymbol{G}$ is a $q \times q$ nonnegative symmetric matrix.

**Proof of Theorem II.3**

Note that when $w_{n,kj} = 1$, we have $a_n = 1$ and $b_n = 1$. The conditions $\lambda_n\sqrt{a_n} = O_p(n^{-1/2})$ and $\frac{P_n^2}{\lambda_n^2 b_n} = o_p(n)$ in Theorem II.5 become $\lambda_n\sqrt{n} = O_p(1)$ and $\frac{P_n}{\lambda_n\sqrt{n}} \to 0$. These two conditions cannot be satisfied simultaneously by adjusting $\lambda_n$, which implies that $\Pr(\hat{\boldsymbol{\beta}}_{\mathcal{D}} = 0) \to 1$ cannot be guaranteed.

We will prove that by choosing $\lambda_n$ satisfying $\sqrt{n}\lambda_n = O_p(1)$ and $P_n n^{-3/4}/\lambda_n \to 0$ as $n \to \infty$, we can have a root-$n$ consistent local maximizer $\hat{\boldsymbol{\beta}}_n = (\hat{\boldsymbol{\beta}}_{\mathcal{A}_n}, \hat{\boldsymbol{\beta}}_{\mathcal{B}_n}, \hat{\boldsymbol{\beta}}_{\mathcal{C}_n})^{\mathsf{T}}$ such that $\Pr(\hat{\boldsymbol{\beta}}_{\mathcal{C}_n} = 0) \to 1$.

Similar as in the proof of Theorem II.5, we let $h_n' = \sum_{k=S_n+1}^{K_n} h_{nk}$. By equation (2.27), we have $h_n' = \sum_{k=S_n+1}^{K_n} O_p(\sqrt{P_n/n}) \sum_{j=1}^{p_k} |\hat{\beta}_{kj}| O_p(\sqrt{nP_n}) = O_p(P_n^2/\sqrt{n})$. Since $P_n n^{-3/4}/\lambda_n \to 0$ guarantees that $n\lambda_n^2$ dominates $h_n'$ with probability tending to 1 as $n \to \infty$, the first term in (2.28) is dominated by the second term as $n \to \infty$ uniformly for any $k$ satisfying $\boldsymbol{\beta}_{nk}^0 = 0$ since $w_{n,kj} = 1$ and $h_n' > h_{nk}$. Similar as in the proof of Theorem II.5, we have $\hat{\boldsymbol{\beta}}_{\mathcal{C}_n} = 0$ with probability tending to 1.

**Proof of Theorem II.6**

Let $N_n = |\mathcal{A}_n|$ be the number of nonzero parameters. Let $\boldsymbol{B}_n$ be an $(N_n - q) \times N_n$ matrix which satisfies $\boldsymbol{B}_n\boldsymbol{B}_n^{\mathsf{T}} = \boldsymbol{I}_{N_n - q}$ and $\boldsymbol{A}_n\boldsymbol{B}_n^{\mathsf{T}} = 0$. As $\boldsymbol{\beta}_{n,\mathcal{A}_n}$ is in the orthogonal complement to the linear space that is spanned by the rows of $\boldsymbol{A}_n$ under the null hypothesis $H_0$, it follows that

$$\boldsymbol{\beta}_{n,\mathcal{A}_n} = \boldsymbol{B}_n^{\mathsf{T}}\boldsymbol{\gamma}_n,$$

where $\boldsymbol{\gamma}_n$ is an $(N_n - q) \times 1$ vector. Then, under $H_0$ the penalized likelihood estimator is also the local maximizer $\hat{\boldsymbol{\gamma}}_n$ of the problem

$$Q_n(\boldsymbol{\beta}_{n,\mathcal{A}_n}) = \max_{\boldsymbol{\gamma}_n} Q_n(\boldsymbol{B}_n^{\mathsf{T}}\boldsymbol{\gamma}_n).$$

To prove Theorem II.6 we need the following two lemmas.

**Lemma II.8.** *Under condition* (*b*) *of Theorem II.5 and the null hypothesis* $H_0$*, we have*

$$\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{\beta}^0_{n,\mathcal{A}_n} = \frac{1}{n}\boldsymbol{I}_n^{-1}(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})\nabla L_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n}) + o_p(n^{-1/2}),$$

$$\boldsymbol{B}_n^{\intercal}(\hat{\boldsymbol{\gamma}}_n - \boldsymbol{\gamma}_n^0) = \frac{1}{n}\boldsymbol{B}_n^{\intercal}\{\boldsymbol{B}_n\boldsymbol{I}_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})\boldsymbol{B}_n^{\intercal}\}^{-1}\boldsymbol{B}_n\nabla L_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n}) + o_p(n^{-1/2}).$$

**Proof of of Lemma II.8**

We need only prove the second equation. The first equation can be shown in the same manner. Following the proof of Theorem II.5, it follows that under $H_0$,

$$\boldsymbol{B}_n\boldsymbol{I}_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})\boldsymbol{B}_n^{\intercal}(\hat{\boldsymbol{\gamma}}_n - \boldsymbol{\gamma}_n^0) = \frac{1}{n}\boldsymbol{B}_n\nabla L_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n}) + o_p(n^{-1/2}).$$

As the eigenvalue $\lambda_i(\boldsymbol{B}_n\boldsymbol{I}_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})\boldsymbol{B}_n^{\intercal})$ is uniformly bounded away from 0 and infinity, we have

$$\boldsymbol{B}_n^{\intercal}(\hat{\boldsymbol{\gamma}}_n - \boldsymbol{\gamma}_n^0) = \frac{1}{n}\boldsymbol{B}_n^{\intercal}\{\boldsymbol{B}_n\boldsymbol{I}_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})\boldsymbol{B}_n^{\intercal}\}^{-1}\boldsymbol{B}_n\nabla L_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n}) + o_p(n^{-1/2}).$$

**Lemma II.9.** *Under condition* (*b*) *of Theorem II.5 and the null hypothesis* $H_0$*, we have*

(2.33)
$$\begin{aligned} & Q_n(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n}) - Q_n(\boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n) \\ = \; & \frac{n}{2}(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n)^{\intercal}\boldsymbol{I}_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n) + o_p(1). \end{aligned}$$

**Proof of Lemma II.9**

A Taylor's expansion of $Q_n(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n}) - Q_n(\boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n)$ at the point $\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n}$ yields

$$Q_n(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n}) - Q_n(\boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n) = T_1 + T_2 + T_3 + T_4,$$

where

$$\begin{aligned}
T_1 &= \nabla^{\intercal} Q_n(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n})(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n), \\
T_2 &= -\frac{1}{2}(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n)^{\intercal}\nabla^2 L_n(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n})(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n), \\
T_3 &= \frac{1}{6}\nabla^{\intercal}\{(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n)^{\intercal}\nabla^2 L_n(\boldsymbol{\beta}^{\star}_{n,\mathcal{A}_n})(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n)\}(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n), \\
T_4 &= \frac{1}{2}(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n)^{\intercal}\nabla^2 J_n(\boldsymbol{\beta}^{*}_{n,\mathcal{A}_n})(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n).
\end{aligned}$$

We have $T_1 = 0$ as $\nabla^{\intercal} Q_n(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n}) = 0$.

Let $\boldsymbol{\Theta}_n = \boldsymbol{I}_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})$ and $\boldsymbol{\Phi}_n = \frac{1}{n}\nabla L_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})$. By Lemma II.7 we have

$$\begin{aligned}
&(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n) \\
=\ &\boldsymbol{\Theta}_n^{-1/2}\{\boldsymbol{I}_n - \boldsymbol{\Theta}_n^{1/2}\boldsymbol{B}_n^{\intercal}(\boldsymbol{B}_n\boldsymbol{\Theta}_n\boldsymbol{B}_n^{\intercal})^{-1}\boldsymbol{B}_n\boldsymbol{\Theta}_n^{1/2}\}\boldsymbol{\Theta}_n^{-1/2}\boldsymbol{\Phi}_n \\
&+o_p(n^{-1/2}).
\end{aligned}$$

$\boldsymbol{I}_n - \boldsymbol{\Theta}_n^{1/2}\boldsymbol{B}_n^{\intercal}(\boldsymbol{B}_n\boldsymbol{\Theta}_n\boldsymbol{B}_n^{\intercal})^{-1}\boldsymbol{B}_n\boldsymbol{\Theta}_n^{1/2}$ is an idempotent matrix with rank $q$. Hence, by a standard argument and condition (A2),

$$(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n) = O_p(\sqrt{\frac{q}{n}}).$$

We have

$$\left(\frac{1}{n}\nabla^2 J_n(\boldsymbol{\beta}_{n,\mathcal{A}_n})\right)_{kjk_1j_1} = 0, \text{ for } k \neq k_1 \tag{2.34}$$

and

$$\begin{aligned}
&\left(\frac{1}{n}\nabla^2 J_n(\boldsymbol{\beta}^{*}_{n,\mathcal{A}_n})\right)_{kjkj_1} \\
=\ &\frac{\lambda_n w_{n,kj} w_{n,kj_1}}{4(w_{n,k1}|\beta^{*}_{k1}| + \ldots + w_{n,ks_k}|\beta^{*}_{ks_k}|)^{3/2}} \\
=\ &\frac{\lambda_n w_{n,kj} w_{n,kj_1}}{4(w_{n,k1}|\beta^{0}_{k1}| + \ldots + w_{n,ks_k}|\beta^{0}_{ks_k}|)^{3/2}}(1+o_p(1)) \\
\leq\ &\frac{\lambda_n\sqrt{a_n}}{4(c_1)^{3/2}}(1+o_p(1)) \\
=\ &o_p((nP_n)^{-1/2}). \qquad (2.35)
\end{aligned}$$

Combining (2.34), (2.35) and condition $q < P_n$, following the proof of $I_3$ in Theorem II.4, we have

$$T_3 = O_p(nP_n^{3/2}n^{-3/2}q^{3/2}) = o_p(1)$$

and

$$\begin{aligned} T_4 &\leq n\left\|\frac{1}{n}\nabla^2 J_n(\boldsymbol{\beta}^*_{n,\mathcal{A}_n})\right\| \|\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n\|^2 \\ &= nP_n o_p((nP_n)^{-1/2})O_p(\frac{q}{n}) \\ &= o_p(1). \end{aligned}$$

Thus,

$$Q_n(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n}) - Q_n(\boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n) = T_2 + o_p(1). \tag{2.36}$$

It follows from Lemmas 8 and 9 of [21] that

$$\left\|\frac{1}{n}\nabla^2 L_n(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n}) + \boldsymbol{I}_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})\right\| = o_p\left(\frac{1}{\sqrt{P_n}}\right).$$

Hence, we have

$$\begin{aligned} &\frac{1}{2}(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n)^{\intercal}\{\nabla^2 L_n(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n}) + n\boldsymbol{I}_n(\boldsymbol{\beta}^0_{n,\mathcal{A}_n})\}(\hat{\boldsymbol{\beta}}_{n,\mathcal{A}_n} - \boldsymbol{B}_n^{\intercal}\hat{\boldsymbol{\gamma}}_n) \\ \leq\ & o_p\left(n\frac{1}{\sqrt{P_n}}\right)O_p(\frac{q}{n}) = o_p(1). \end{aligned} \tag{2.37}$$

The combination of (2.36) and (2.37) yields (2.33).

**Proof of Theorem II.6**

The proof of the Theorem is the same as the proof of Theorem 4 in [21] given Lemmas II.8 and II.9.

## CHAPTER III

# Partial Correlation Estimation by Joint Sparse Regression Models

In this chapter, a computationally efficient approach —`space`(Sparse PArtial Correlation Estimation)— for selecting non-zero partial correlations under the high-dimension-low-sample-size setting is proposed. This method assumes the overall sparsity of the partial correlation matrix and employs sparse regression techniques for model fitting. The performance of `space` is illustrated by extensive simulation studies. It is shown that `space` performs well in both non-zero partial correlation selection and identification of hub variables, and it also outperforms two existing methods. We then apply `space` to a microarray breast cancer data set and identify a set of *hub genes* which may provide important insights on genetic regulatory networks.

### 3.1 Introduction

There has been a large amount of literature on *covariance selection*: the identification and estimation of non-zero entries in the inverse covariance matrix (a.k.a. *concentration matrix* or *precision matrix*) starting with the seminal paper by Dempster [16]. Covariance selection is very useful in elucidating associations among a set of random variables, as it is well known that non-zero entries of the concentration matrix correspond to non-zero partial correlations. Moreover, under Gaussianity, non-zero entries of the concentration matrix imply conditional dependency between corresponding variable pairs conditional on the rest of the variables [17]. Traditional methods do not work unless the sample size ($n$) is larger than the number of variables ($p$) [17, 68]. Recently, a number of methods have been introduced to perform covariance selection for data sets with $p > n$ [37, 43, 55, 71].

In this chapter, we propose a novel approach using sparse regression techniques for covariance selection. Our work is partly motivated by the construction of *genetic regulatory networks (GRN)* based on high dimensional gene expression data. Denote the expression levels of $p$ genes as $y_1, \cdots, y_p$. A *concentration network* is defined as an undirected graph, in which the $p$ vertices represent the $p$ genes and an edge connects gene $i$ and gene $j$ if and only if the partial correlation $\rho^{ij}$ between $y_i$ and $y_j$ is nonzero. Note that, under the assumption that $y_1, \cdots, y_p$ are jointly normal, the partial correlation $\rho^{ij}$ equals to $\mathrm{Corr}(y_i, y_j | y_{-(i,j)})$, where $y_{-(i,j)} = \{y_k : 1 \leq k \neq i, j \leq p\}$. Therefore, for $\rho^{ij}$ being nonzero is equivalent to $y_i$ and $y_j$ being conditionally dependent given all other variables $y_{-(i,j)}$. The proposed method is specifically designed for the high-dimension-low-sample-size scenario. It relies on the assumption that the partial correlation matrix is sparse (under normality assumption, this means that most variable pairs are conditionally independent), which is reasonable for many real life problems. For instance, it has been shown that most genetic networks are intrinsically sparse [25, 31, 62]. The proposed method is also particularly powerful in the identification of *hubs*: vertices (variables) that are connected to (have nonzero partial correlations with) many other vertices (variables). The existence of hubs is a well known phenomenon for many large networks, such as the internet, citation networks, and protein interaction networks [45]. In particular, it is widely believed that genetic pathways consist of many genes with few interactions and a few hub genes with many interactions [4].

Another contribution of this chapter is to propose `active-shooting`, a novel algorithm for solving penalized optimization problems such as Lasso [63]. This algorithm is computationally more efficient than the original `shooting` algorithm [24]. It enables us to implement the proposed procedure efficiently, such that we can conduct extensive simulation studies involving $\sim 1000$ variables and hundreds of samples. To our knowledge, this is the first set of intensive simulation studies for covariance selection with such high dimensions.

A few methods have also been proposed recently to perform covariance selection in the context of $p \gg n$. Similar to the method proposed in this chapter, they all assume sparsity of the partial correlation matrix. Meinshausen and Buhlmann [43] introduced a variable-by-variable approach for neighborhood selection via the Lasso regression. They proved that neighborhoods can be consistently selected under a set of suitable assumptions. However, as regression models are fitted for each variable separately, this method has two major limitations. First, it does not take into account the intrinsic symmetry of the problem (i.e., $\rho^{ij} = \rho^{ji}$). This could result in a loss of efficiency, as well as contradictory neighborhoods. Secondly, if the same penalty parameter is used for all $p$ Lasso regressions as suggested by their paper, more or less equal effort is placed on building each neighborhood. This does not seem to be the most efficient way to address the problem, unless the degree distribution of the network is nearly uniform. However, most real life networks have skewed degree distributions, such as the *power-law networks*. As observed by Schafer and Strimmer [55], the neighborhood selection approach limits the number of edges connecting to each node. Therefore, it is not very effective in hub detection. On the contrary, the proposed method is based on a joint sparse regression model, which simultaneously performs neighborhood selection for all variables. It also preserves the symmetry of the problem and thus utilizes data more efficiently. We show by intensive simulation studies that our method performs better in both model selection and hub identification. Moreover, as a joint model is used, it is easier to incorporate prior knowledge such as network topology into the model. This is discussed in Section 3.2.1.

Besides the regression approach mentioned above, another class of methods employ the maximum likelihood framework. Yuan and Lin [71] proposed a penalized maximum likelihood approach which performs model selection and estimation simultaneously and ensures the positive definiteness of the estimated concentration matrix. However, their algorithm can not handle high dimensional data. The largest dimension considered by them is $p = 10$ in simulation and $p = 5$ in real data. Friedman et al. [23] proposed an efficient algorithm `glasso` to implement this method, such that it can be applied to problems with high dimensions. We show by simulation studies that, the proposed method performs better than `glasso` in both model selection and hub identification. Rothman et al. [51] proposed another algorithm to implement the method of Yuan and Lin [71]. The computational cost is on the same order of `glasso`, but in general not as efficient as `glasso`. Li and Gui [37] introduced a threshold gradient descent (TGD) regularization procedure. Schafer and Strimmer [55] proposed a shrinkage covariance estimation procedure to overcome the ill-conditioned problem of sample covariance matrix when $p > n$. There are also a large class of methods covering the situation where variables have a natural ordering, e.g., longitudinal data, time series, spatial data, or spectroscopy. These methods are all based on the modified Cholesky decomposition of the concentration matrix

[6, 29, 36, 69]. In this chapter, we, however, focus on the general case where an ordering of the variables is not available.

The rest of this chapter is organized as follows. In Section 3.2, we describe the joint sparse regression model, its implementation and the `active-shooting` algorithm. In Section 3.3, the performance of the proposed method is illustrated through simulation studies and compared with that of the neighborhood selection approach and the likelihood based approach `glasso`. In Section 3.4, the proposed method is applied to a microarray expression data set of $n = 244$ breast cancer tumor samples and $p = 1217$ genes. A summary of the main results are given in Section 3.5. Technique details of the algorithm and more simulation results are provided in Appendix B.

## 3.2 Method

### 3.2.1 Model

In this section, we describe a novel method for detecting pairs of variables having nonzero partial correlations among a large number of random variables based on i.i.d. samples. Suppose that, $(y_1, \cdots, y_p)^T$ has a joint distribution with mean 0 and covariance $\mathbf{\Sigma}$, where $\mathbf{\Sigma}$ is a $p$ by $p$ positive definite matrix. Denote the partial correlation between $y_i$ and $y_j$ by $\rho^{ij}$ $(1 \leq i < j \leq p)$. It is defined as $\mathrm{Corr}(\epsilon_i, \epsilon_j)$, where $\epsilon_i$ and $\epsilon_j$ are the prediction errors of the best linear predictors of $y_i$ and $y_j$ based on $y_{-(i,j)} = \{y_k : 1 \leq k \neq i, j \leq p\}$, respectively. Denote the *concentration matrix* $\mathbf{\Sigma}^{-1}$ by $(\sigma^{ij})_{p \times p}$. It is known that, $\rho^{ij} = -\frac{\sigma^{ij}}{\sqrt{\sigma^{ii}\sigma^{jj}}}$. Let $y_{-i} := \{y_k : 1 \leq k \neq i \leq p\}$. The following well-known result (Lemma III.1) relates the estimation of partial correlations to a regression problem.

**Lemma III.1.** *: For $1 \leq i \leq p$, $y_i$ is expressed as $y_i = \sum_{j \neq i} \beta_{ij} y_j + \epsilon_i$, such that $\epsilon_i$ is uncorrelated with $y_{-i}$ if and only if $\beta_{ij} = -\frac{\sigma^{ij}}{\sigma^{ii}} = \rho^{ij}\sqrt{\frac{\sigma^{jj}}{\sigma^{ii}}}$. Moreover, for such* *defined* $\beta_{ij}$, $\mathrm{Var}(\epsilon_i) = \frac{1}{\sigma^{ii}}, \mathrm{Cov}(\epsilon_i, \epsilon_j) = \frac{\sigma^{ij}}{\sigma^{ii}\sigma^{jj}}$.

Note that, under the normality assumption, $\rho^{ij} = \mathrm{Corr}(y_i, y_j | y_{-(i,j)})$ and in Lemma III.1, we can replace "uncorrelated" by "independent". Since $\rho^{ij} = \mathrm{sign}(\beta_{ij})\sqrt{\beta_{ij}\beta_{ji}}$, the search for non-zero partial correlations can be viewed as a model selection problem under the regression setting. In this chapter, we are mainly interested in the case where the dimension $p$ is larger than the sample size $n$. This is a typical scenario for many real life problems. For example, high throughput genomic experiments usually result in data sets of thousands of genes for tens or at most hundreds of samples. However, many high-dimensional problems are intrinsically sparse. In the case of genetic regulatory networks, it is widely believed that most gene pairs are not directly interacting with each other. Sparsity suggests that even if the number of variables is much larger than the sample size, the effective dimensionality of the problem might still be within a tractable range. Therefore, we propose to employ sparse regression techniques by imposing the $L_1$-norm penalty on a suitable loss function to tackle the high-dimension-low-sample-size problem.

Suppose $\boldsymbol{Y}^k = (y_1^k, \cdots, y_p^k)^T$ are i.i.d. observations from $(0, \boldsymbol{\Sigma})$, for $k = 1, \cdots, n$. Denote the sample of the $i$th variable as $\boldsymbol{Y}_i = (y_i^1, \cdots, y_i^n)^T$. Based on Lemma III.1, we propose the following joint loss function

$$\begin{aligned} L_n(\boldsymbol{\theta}, \boldsymbol{\sigma}, \boldsymbol{Y}) &= \frac{1}{2}\Big(\sum_{i=1}^p w_i ||\boldsymbol{Y}_i - \sum_{j \neq i} \beta_{ij} \boldsymbol{Y}_j||^2\Big) \\ &= \frac{1}{2}\Big(\sum_{i=1}^p w_i ||\boldsymbol{Y}_i - \sum_{j \neq i} \rho^{ij} \sqrt{\frac{\sigma^{jj}}{\sigma^{ii}}} \boldsymbol{Y}_j||^2\Big), \end{aligned} \tag{3.1}$$

where $\boldsymbol{\theta} = (\rho^{12}, \cdots, \rho^{(p-1)p})^T$, $\boldsymbol{\sigma} = \{\sigma^{ii}\}_{i=1}^p$; $\boldsymbol{Y} = \{\boldsymbol{Y}^k\}_{k=1}^n$; and $\boldsymbol{w} = \{w_i\}_{i=1}^p$ are nonnegative weights. For example, we can choose $w_i = 1/\mathrm{Var}(\epsilon_i) = \sigma^{ii}$ to weigh individual regressions in the joint loss function according to their residual variances, as is done in regression with heteroscedastic noise. We propose to estimate the partial correlations $\boldsymbol{\theta}$ by minimizing a penalized loss function

$$\mathcal{L}_n(\boldsymbol{\theta}, \boldsymbol{\sigma}, \boldsymbol{Y}) = L_n(\boldsymbol{\theta}, \boldsymbol{\sigma}, \boldsymbol{Y}) + \mathcal{J}(\boldsymbol{\theta}), \tag{3.2}$$

where the penalty term $\mathcal{J}(\boldsymbol{\theta})$ controls the overall sparsity of the final estimation of $\boldsymbol{\theta}$. In this chapter, we focus on the $L_1$-norm penalty [63]:

$$\mathcal{J}(\boldsymbol{\theta}) = \lambda ||\boldsymbol{\theta}||_1 = \lambda \sum_{1 \leq i < j \leq p} |\rho^{ij}|. \tag{3.3}$$

The proposed joint method is referred as `space` (Sparse PArtial Correlation Estimation) hereafter. It is related to the *neighborhood selection approach* [43] (referred as `MB` hereafter), where a Lasso regression is performed separately for each variable on the rest of the variables. However, `space` has several important advantages.

(i) In `space`, sparsity is utilized for the partial correlations $\theta$ as a whole view. However, in the neighborhood selection approach, sparsity is imposed on each neighborhood. The former treatment is more natural and utilizes the data more efficiently, especially for networks with hubs. A prominent example is the genetic regulatory network, where master regulators are believed to exist and are of great interest.

(ii) According to Lemma III.1, $\beta_{ij}$ and $\beta_{ji}$ have the same sign. The proposed method assures this sign consistency as it estimates $\{\rho^{ij}\}$ directly. However, when fitting $p$ separate (Lasso) regressions, it is possible that $\text{sign}(\widehat{\beta}_{ij})$ is different from $\text{sign}(\widehat{\beta}_{ji})$, which may lead to contradictory neighborhoods.

(iii) Furthermore, the utility of the symmetric nature of the problem allows us to reduce the number of unknown parameters in the model by almost half ($p(p+1)/2$ for `space` vs. $(p-1)^2$ for `MB`), and thus improves the efficiency.

(iv) Finally, prior knowledge of the network structure are often available. The joint model is more flexible in incorporating such prior knowledge. For example, we may assign different weights $w_i$ to different nodes according to their "importance". We have already discussed the residual variance weights, where $w_i = \sigma^{ii}$. We can also consider the weight that is proportional to the (estimated) degree of each variable, i.e., the estimated number of edges connecting with each node in the network. This would result in a preferential attachment effect which explains the cumulative advantage phenomena observed in many real life networks including GRNs [3].

These advantages help enhance the performance of `space`. As illustrated by the simulation study in Section 3.3, the proposed joint method performs better than the neighborhood selection approach in both non-zero partial correlation selection and hub detection.

As compared to the penalized maximum likelihood approach `glasso` [23], the simulation study in Section 3.3 shows that `space` also outperforms `glasso` in both edge detection and hub identification under all settings that we have considered. In addition, `space` has the following advantages.

(i) The complexity of `glasso` is $O(p^3)$, while as discussed in Section 3.2.2, the `space` algorithm has the complexity of $\min(O(np^2), O(p^3))$, which is much faster than the algorithm of Yuan and Lin [71] and in general should also be faster than `glasso` when $n < p$, which is the case in many real studies.

(ii) As discussed in Section 3.5, `space` allows for trivial generalizations to other penalties of the form of $|\rho^{ij}|^q$ rather than simply $|\rho^{ij}|$, which includes ridge and bridge [24] or other more complicated penalties like SCAD [20]. The `glasso` algorithm, on the other hand, is tied to the Lasso formulation and cannot be extended to other penalties in a natural manner.

Note that, in the penalized loss function (3.2), $\boldsymbol{\sigma}$ needs to be specified. We propose to estimate $\boldsymbol{\theta}$ and $\boldsymbol{\sigma}$ by a two-step iterative procedure. Given an initial estimate $\boldsymbol{\sigma}^{(0)}$ of $\boldsymbol{\sigma}$, $\boldsymbol{\theta}$ is estimated by minimizing the penalized loss function (3.2), whose implementation is discussed in Section 3.2.2. Then given the current estimates $\boldsymbol{\theta}^{(c)}$ and $\boldsymbol{\sigma}^{(c)}$, $\boldsymbol{\sigma}$ is updated based on Lemma III.1: $1/\widehat{\sigma}^{ii} = \frac{1}{n}||\boldsymbol{Y}_i - \sum_{j \neq i} \widehat{\beta}_{ij}^{(c)} \boldsymbol{Y}_j||^2$, where $\widehat{\beta}_{ij}^{(c)} = (\rho^{ij})^{(c)} \sqrt{\frac{(\sigma^{jj})^{(c)}}{(\sigma^{ii})^{(c)}}}$. We then iterate between these two steps until convergence. Since $1/\sigma^{ii} \leq \mathrm{Var}(y_i) = \sigma_{ii}$, we can use $1/\widehat{\sigma}_{ii}$ as the initial estimate of $\sigma^{ii}$, where $\widehat{\sigma}_{ii} = \frac{1}{n-1}\sum_{k=1}^{n}(y_i^k - \bar{y}_i)^2$ is the sample variance of $y_i$. Our simulation study shows that, it usually takes no more than three iterations for this procedure to stabilize.

### 3.2.2 Implementation

In this section, we discuss the implementation of the `space` procedure: that is, minimizing (3.2) under the $L_1$-norm penalty (3.3). We first re-formulate the problem, such that the loss function (3.1) corresponds to the $L_2$-norm loss of a "regression problem." We then use the `active-shooting` algorithm proposed in Section 3.2.3 to solve this Lasso regression problem efficiently.

Given $\boldsymbol{\sigma}$ and positive weights $w$, let $\boldsymbol{\mathcal{Y}} = (\tilde{\boldsymbol{Y}}_1^T, ..., \tilde{\boldsymbol{Y}}_p^T)^T$ be a $np \times 1$ column vector, where $\tilde{\boldsymbol{Y}}_i = \sqrt{w_i}\boldsymbol{Y}_i$ $(i = 1, \cdots, p)$; and let $\boldsymbol{\mathcal{X}} = (\tilde{\boldsymbol{\mathcal{X}}}_{(1,2)}, \cdots, \tilde{\boldsymbol{\mathcal{X}}}_{(p-1,p)})$ be a $np$ by $p(p-1)/2$ matrix, with

$$\tilde{\boldsymbol{\mathcal{X}}}_{(i,j)} = (0, ..., 0, \underset{\substack{\uparrow \\ i^{th}\text{block}}}{\sqrt{\frac{\tilde{\sigma}^{jj}}{\tilde{\sigma}^{ii}}}\tilde{\boldsymbol{Y}}_j^T}, 0, ..., 0, \underset{\substack{\uparrow \\ j^{th}\text{block}}}{\sqrt{\frac{\tilde{\sigma}^{ii}}{\tilde{\sigma}^{jj}}}\tilde{\boldsymbol{Y}}_i^T}, 0, ..., 0)^T \quad ,$$

where $\tilde{\sigma}^{ii} = \sigma^{ii}/w_i$ $(i = 1, \cdots, p)$. Then it is easy to see that the loss function

(3.1) equals to $\frac{1}{2}||\boldsymbol{\mathcal{Y}} - \boldsymbol{\mathcal{X}}\boldsymbol{\theta}||_2^2$, and the corresponding $L_1$-norm minimization problem is equivalent to: $\min_{\boldsymbol{\theta}} \frac{1}{2}||\boldsymbol{\mathcal{Y}} - \boldsymbol{\mathcal{X}}\boldsymbol{\theta}||_2^2 + \lambda||\boldsymbol{\theta}||_1$. Note that, the current dimension $\tilde{n} = np$ and $\tilde{p} = p(p-1)/2$ are of a much higher order than the original $n$ and $p$. This could cause serious computational problems. Fortunately, $\boldsymbol{\mathcal{X}}$ is a block matrix with many zero blocks. Thus, algorithms for Lasso regressions can be efficiently implemented by taking into consideration this structure (see Part I of Appendix B for the detailed implementation). To further decrease the computational cost, we develop a new algorithm `active-shooting` (Section 3.2.3) for the `space` model fitting. `Active-shooting` is a modification of the `shooting` algorithm, which was first proposed by Fu [24] and then extended by many others including Genkin et al. [26] and Friedman et al. [22]. `Active-shooting` exploits the sparse nature of sparse penalization problems in a more efficient way, and is therefore computationally much faster. This is crucial for applying `space` for large $p$ and/or $n$. It can be shown that the computational cost of `space` is $\min(O(np^2), O(p^3))$, which is the same as applying $p$ individual Lasso regressions as in the neighborhood selection approach. We want to point out that, the proposed method can also be implemented by `lars` [19]. However, unless the exact whole solution path is needed, compared with `shooting` type algorithms, `lars` is computationally less appealing [22].

Finally, note that the concentration matrix should be positive definite. In principle, the proposed method (or more generally, the regression based methods) does not guarantee the positive definiteness of the resulting estimator, while the likelihood based method by Yuan and Lin [71] and Friedman et al. [23] assures the positive definiteness. While admitting that this is one limitation of the proposed method, we argue that, since we are more interested in model selection than parameter estimation, we are less concerned with this issue. Indeed, the `space` estimators are rarely non-positive-definite under the high dimensional sparse settings that we are interested in. More discussions on this issue can be found in Section 3.3.

### 3.2.3 Active Shooting

In this section, we propose a computationally very efficient algorithm `active-shooting` for solving Lasso regression problems. `Active-shooting` is motivated by the `shooting` algorithm [24], which solves the Lasso regression by updating each coordinate iteratively until convergence. `Shooting` is computationally very competitive compared with the well known `lars` procedure [19]. Suppose that we want to minimize an $L_1$-norm penalized loss function with respect to $\beta$

$$f(\boldsymbol{\beta}) = \frac{1}{2}||\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}||_2^2 + \gamma \sum_j |\beta_j|,$$

where $\boldsymbol{Y} = (y_1, \cdots, y_n)^T$, $\boldsymbol{X} = (x_{ij})_{n\times p} = (\boldsymbol{X}_1 : \cdots : \boldsymbol{X}_p)$ and $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_p)^T$. The `shooting` algorithm proceeds as follows:

1. Initial step: for $j = 1, \cdots, p$,

$$\begin{aligned} \beta_j^{(0)} &= \arg\min_{\beta_j} \{\tfrac{1}{2}||\boldsymbol{Y} - \beta_j \boldsymbol{X}_j||^2 + \gamma|\beta_j|\} \\ &= \text{sign}(\boldsymbol{Y}^T\boldsymbol{X}_j)\tfrac{(|\boldsymbol{Y}^T\boldsymbol{X}_j| - \gamma)_+}{\boldsymbol{X}_j^T\boldsymbol{X}_j}, \end{aligned} \tag{3.4}$$

   where $(x)_+ = x\mathbb{I}(x > 0)$.

2. For $j = 1, ..., p$, update $\boldsymbol{\beta}^{(old)} \longrightarrow \boldsymbol{\beta}^{(new)}$ :

$$\begin{aligned} \beta_i^{(new)} &= \beta_i^{(old)}, i \neq j; \\ \beta_j^{(new)} &= \arg\min_{\beta_j} \tfrac{1}{2}\left\| \boldsymbol{Y} - \textstyle\sum_{i\neq j} \beta_i^{(old)}\boldsymbol{X}_i - \beta_j\boldsymbol{X}_j \right\|^2 + \gamma|\beta_j| \\ &= \text{sign}\left(\tfrac{(\boldsymbol{\epsilon}^{(old)})^T\boldsymbol{X}_j}{\boldsymbol{X}_j^T\boldsymbol{X}_j} + \beta_j^{(old)}\right)\left(\left|\tfrac{(\boldsymbol{\epsilon}^{(old)})^T\boldsymbol{X}_j}{\boldsymbol{X}_j^T\boldsymbol{X}_j} + \beta_j^{(old)}\right| - \tfrac{\gamma}{\boldsymbol{X}_j^T\boldsymbol{X}_j}\right)_+, \end{aligned} \tag{3.5}$$

   where $\boldsymbol{\epsilon}^{(old)} = \boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}^{(old)}$.

3. Repeat step 2 until convergence.

At each updating step of the `shooting` algorithm, we define the set of currently non-zero coefficients as the *active set.* Since under sparse models, the active set should remain small, we propose to first update the coefficients within the active set until convergence is achieved before moving on to update other coefficients. The `active-shooting` algorithm proceeds as follows:

1. Initial step: same as the initial step of `shooting`.
2. Define the current active set $\Lambda = \{k : \text{current } \beta_k \neq 0\}$.
   - (2.1) For each $k \in \Lambda$, update $\beta_k$ with all other coefficients fixed at the current value as in equation (3.5);
   - (2.2) Repeat (2.1) until convergence is achieved on the active set.
3. For $j = 1$ to $p$, update $\beta_j$ with all other coefficients fixed at the current value as in equation (3.5). If no $\beta_j$ changes during this process, return the current $\beta$ as the final estimate. Otherwise, go back to step 2.

Table 3.1: The numbers of iterations required by the `shooting` algorithm and the `active-shooting` algorithm to achieve convergence ($n = 100$, $\lambda = 2$). "coef. #" is the number of non-zero coefficients

| $p$ | coef. # | `shooting` | `active-shooting` |
|---|---|---|---|
| 200 | 14 | 29600 | 4216 |
| 500 | 25 | 154000 | 10570 |
| 1000 | 28 | 291000 | 17029 |

The idea of `active-shooting` is to focus on the set of variables that is more likely to be in the model, and thus it improves the computational efficiency by achieving a faster convergence. We illustrate the improvement of the `active-shooting` over the `shooting` algorithm by a small simulation study of the Lasso regression (generated in the same way as in Section 5.1 of [22]). The two algorithms result in exact same solutions. However, as can be seen from Table 3.1, `active-shooting` takes much fewer iterations to converge (where one iteration is counted whenever an attempt to update a $\beta_j$ is made). In particular, it takes less than 30 seconds (on average) to fit the `space` model by `active-shooting` (implemented in `c` code) for cases with 1000 variables, 200 samples and when the resulting model has around 1000 non-zero partial correlations on a server with two Dual/Core, CPU 3 GHz and 4 GB RAM. This great computational advantage enables us to conduct large scale simulation studies to examine the performance of the proposed method (Section 3.3).

### 3.2.4 Tuning

The choice of the tuning parameter $\lambda$ is of great importance. Since the `space` method uses a Lasso criterion, methods that have been developed for selecting the tuning parameter for Lasso can also be applied to `space`, such as the GCV [63], the CV [20], the AIC [11] and the BIC [78]. Several methods have also been proposed for selecting the tuning parameter in the setting of covariance estimation, for example, the MSE based criterion [55], the likelihood based method [29] and the cross-validation and bootstrap methods [37]. In this chapter, we propose to use a "BIC-type" criterion for selecting the tuning parameter mainly due to its simplicity and computational easiness. For a given $\lambda$, denote the `space` estimator by $\widehat{\boldsymbol{\theta}}_\lambda = \{\widehat{\rho}^{ij}_\lambda : 1 \leq i < j \leq p\}$ and $\widehat{\boldsymbol{\sigma}}_\lambda = \{\widehat{\sigma}^{ii}_\lambda : 1 \leq i \leq p\}$. The corresponding residual sum of squares for the $i$-th regression: $y_i = \sum_{j \neq i} \beta_{ij} y_j + \epsilon_i$ is

$$RSS_i(\lambda) = \sum_{k=1}^{n} \left( y_i^k - \sum_{j \neq i} \widehat{\rho}^{ij}_\lambda \sqrt{\frac{\widehat{\sigma}^{jj}_\lambda}{\widehat{\sigma}^{ii}_\lambda}} y_j^k \right)^2 .$$

We then define a "BIC-type" criterion for the $i$-th regression as

$$BIC_i(\lambda) = n \times \log(RSS_i(\lambda)) + \log n \times \#\{j : j \neq i, \widehat{\rho}^{ij}_\lambda \neq 0\}. \tag{3.6}$$

Finally, we define $BIC(\lambda) := \sum_{i=1}^{p} BIC_i(\lambda)$ and select $\lambda$ by minimizing $BIC(\lambda)$. This method is referred as `space.joint` hereafter.

In [71], a BIC criterion is proposed for the penalized maximum likelihood approach. Namely

$$BIC(\lambda) := n \times \Big[ -\log|\widehat{\boldsymbol{\Sigma}}_{\lambda}^{-1}| + \mathrm{trace}(\widehat{\boldsymbol{\Sigma}}_{\lambda}^{-1}\boldsymbol{S}) \Big] + \log n \times \#\{(i,j) : 1 \le i \le j \le p, \widehat{\sigma}_{\lambda}^{ij} \neq 0\}, \tag{3.7}$$

where $\boldsymbol{S}$ is the sample covariance matrix, and $\widehat{\boldsymbol{\Sigma}}_{\lambda}^{-1} = (\widehat{\boldsymbol{\sigma}}_{\lambda}^{ij})$ is the estimator under $\lambda$. In this chapter, we refer this method as `glasso.like`. For the purpose of comparison, we also consider the selection of the tuning parameter for `MB`. Since `MB` essentially performs $p$ individual Lasso regressions, the tuning parameter can be selected for each of them separately. Specifically, we use criterion (3.6) (evaluated at the corresponding `MB` estimators) to select the tuning parameter $\lambda_i$ for the $i$-th regression. We denote this method as `MB.sep`. Alternatively, as suggested by Meinshausen and Buhlmann [43], when all $\boldsymbol{Y}_i$ are standardized to have sample standard deviation one, the same $\lambda(\alpha) = \sqrt{n}\Phi^{-1}(1 - \frac{\alpha}{2p^2})$ is applied to all regressions. Here, $\Phi$ is the standard normal c.d.f.; $\alpha$ is used to control the false discovery rate and is usually taken as 0.05 or 0.1. We denote this method as `MB.alpha`. These methods are examined by the simulation studies in the next section.

## 3.3 Simulation

In this section, we conduct a series of simulation experiments to examine the performance of the proposed method `space` and compare it with the neighborhood selection approach `MB` as well as the penalized likelihood method `glasso`. For all three methods, variables are first standardized to have sample mean zero and sample standard deviation one before model fitting. For `space`, we consider three different types of weights: (1) uniform weights: $w_i = 1$; (2) residual variance based weights: $w_i = \widehat{\sigma}^{ii}$; and (3) degree based weights: $w_i$ is proportional to the estimated degree of $y_i$, i.e., $\#\{j : \widehat{\rho}^{ij} \neq 0, j \neq i\}$. The corresponding methods are referred as `space`, `space.sw` and `space.dew`, respectively. For all three `space` methods, the initial value of $\sigma^{ii}$ is set to be one. Iterations are used for these `space` methods as discussed in Section 3.2.1. For `space.dew` and `space.sw`, the initial weights are taken to be one (i.e., equal weights). In each subsequent iteration, new weights are calculated based on the estimated residual variances (for `space.sw`) or the estimated degrees (for `space.dew`) of the previous iteration. For all three `space` methods, three iterations (that is updating between $\{\sigma^{ii}\}$ and $\{\rho^{ij}\}$) are used since the procedure converges very fast and more iterations result in essentially the same estimator. For `glasso`, the diagonal of the concentration matrix is not penalized.

We simulate networks consisting of disjointed modules. This is done because many real life large networks exhibit a modular structure comprised of many disjointed or loosely connected components of relatively small size. For example, experiments on model organisms like yeast or bacteria suggest that the transcriptional regulatory networks have modular structures [34]. Each of our network modules is set to have 100 nodes and generated according to a given degree distribution, where the *degree* of a node is defined as the number of edges connecting to it. We mainly consider two different types of degree distributions and denote their corresponding networks by `Hub network` and `Power-law network` (details are given later). Given an undirected network with $p$ nodes, the initial "concentration matrix" $(\tilde{\sigma}^{ij})_{p \times p}$ is generated by

$$\tilde{\sigma}^{ij} = \begin{cases} 1, & i = j; \\ 0, & i \neq j \text{ and no edge between nodes } i \text{ and } j; \\ \sim Uniform([-1,-0.5] \cup [0.5,1]), & i \neq j \text{ and an edge connecting nodes } i \text{ and } j. \end{cases}$$

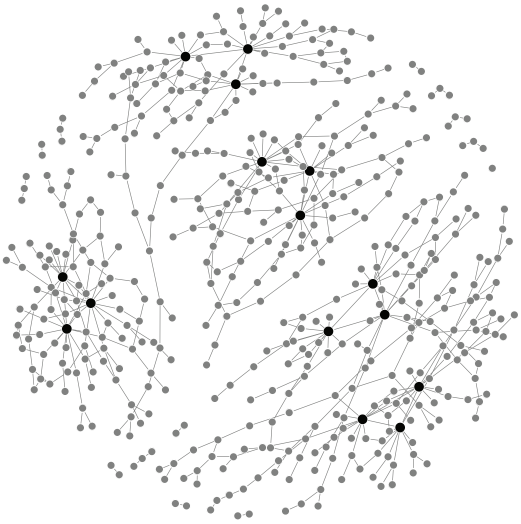
We then rescale the non-zero elements in the above matrix to assure positive definiteness. Specifically, for each row, we first sum the absolute values of the off-diagonal entries, and then divide each off-diagonal entry by 1.5 fold of the sum. We then average this re-scaled matrix with its transpose to ensure symmetry. Finally the diagonal entries are all set to be one. This process results in diagonal dominance. Denote the final matrix as $\boldsymbol{A}$. The covariance matrix $\boldsymbol{\Sigma}$ is then determined by

$$\boldsymbol{\Sigma}(i,j) = \boldsymbol{A}^{-1}(i,j)/\sqrt{\boldsymbol{A}^{-1}(i,i)\boldsymbol{A}^{-1}(j,j)}.$$

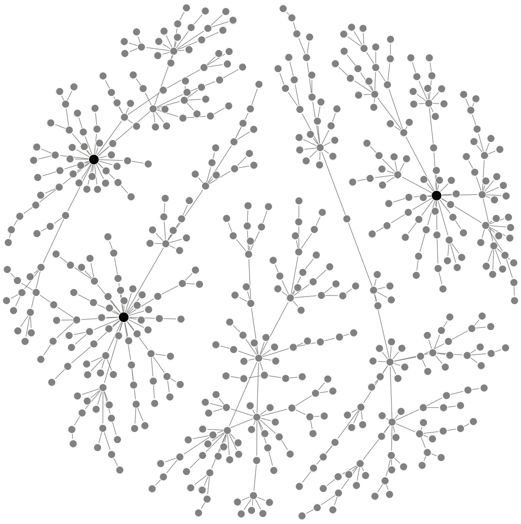
Finally, i.i.d. samples $\{\boldsymbol{Y}^k\}_{k=1}^n$ are generated from Normal$(\mathbf{0}, \boldsymbol{\Sigma})$. Note that, $\boldsymbol{\Sigma}(i,i) = 1$, and $\boldsymbol{\Sigma}^{-1}(i,i) = \sigma^{ii} \geq 1$.

**Hub networks** In the first set of simulations, module networks are generated by inserting a few hub nodes into a very sparse graph. Specifically, each module consists of three hubs with degrees around 15, and the other 97 nodes with degrees at most four. This setting is designed to mimic the genetic regulatory networks where there exist a few hub genes, and most other genes have only a few edges. A network consisting of such modules is shown in Figure 3.1(a). In this network, there are $p = 500$ nodes and 568 edges. The simulated non-zero partial correlations fall in $(-0.67, -0.1] \cup [0.1, 0.67)$, with two modes around -0.28 and 0.28. Based on this network and the partial correlation matrix, we generate 50 independent data sets each consisting of $n = 250$ i.i.d. samples.

We then evaluate each method at a series of different values of the tuning parameter $\lambda$. The number of total detected edges ($N_t$) decreases as $\lambda$ increases. Figure 3.2(a) shows the number of correctly detected edges ($N_c$) vs. the number of total detected edges ($N_t$) averaged across the 50 independent data sets for each method. We observe that all three `space` methods (`space`, `space.sw` and `space.dew`) consistently detect more correct edges than the neighborhood selection method `MB` (except for `space.sw` when $N_t < 470$) and the likelihood based method `glasso`. `MB` performs favorably over `glasso` when $N_t$ is relatively small (say less than 530), but performs



(a) Hub network: 500 nodes and 568 edges. 15 nodes (in black) have degrees of around 15.



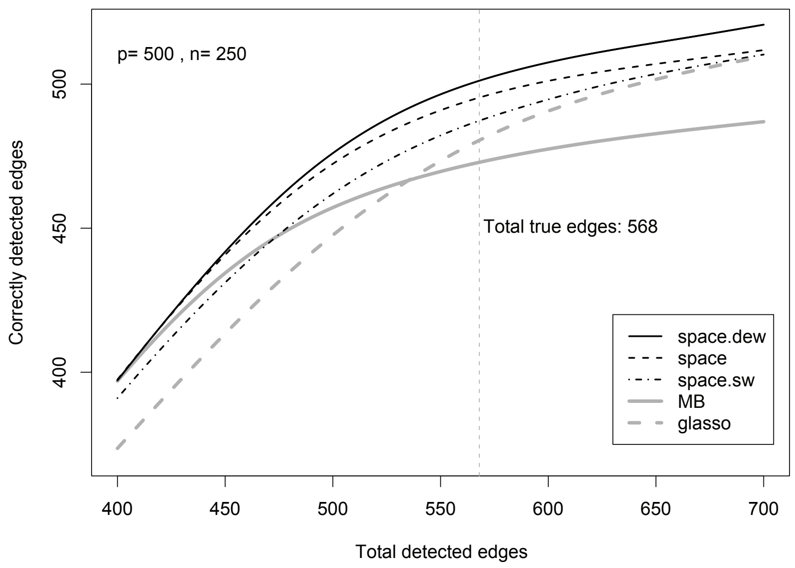
(b) Power-law network: 500 nodes and 495 edges. 3 nodes (in black) have degrees at least 20.

Figure 3.1: Topology of simulated networks.

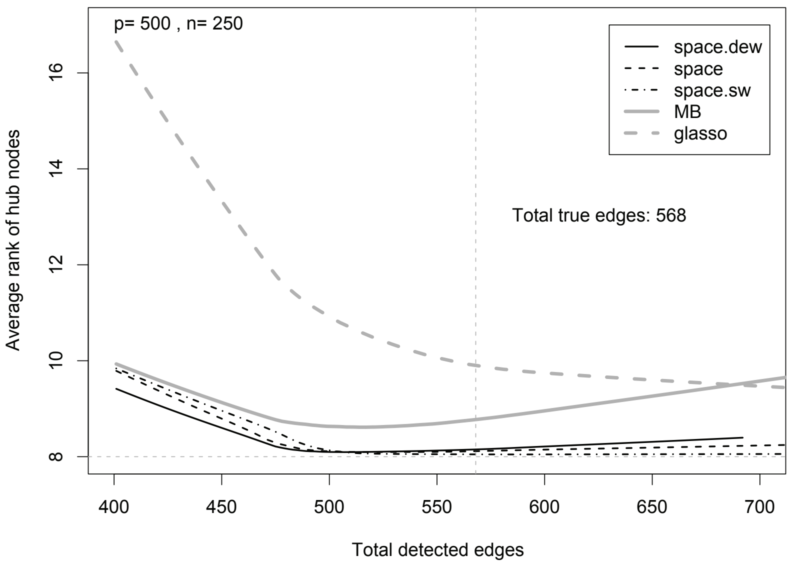
worse than glasso when $N_t$ is large. Overall, space.dew is the best among all methods. Specifically, when $N_t = 568$ (which is the number of true edges), space.dew detects 501 correct edges on average with a standard deviation 4.5 edges. The corresponding sensitivity and specificity are both 88%, where through out this section sensitivity is defined as number of correctly detected edges/total number of true edges, and specificity is defined as number of correctly detected edges/number of total detected edges. On the other hand, MB and glasso detect 472 and 480 correct edges on average, respectively, when the number of total detected edges $N_t$ being 568.

In terms of hub detection, for a given $N_t$, a rank is assigned to each variable $y_i$ based on its estimated degree (the larger the estimated degree, the smaller the rank value). We then calculate the average rank of the 15 true hub nodes for each method. The results are shown in Figure 3.2(b). This average rank would achieve the minimum value 8 (indicated by the grey horizontal line), if the 15 true hubs have larger estimated degrees than all other non-hub nodes. As can be seen from the figure, the average rank curves (as a function of $N_t$) for the three space methods are very close to the optimal minimum value 8 for a large range of $N_t$. This suggests that these methods can successfully identify most of the true hubs. Indeed, for space.dew, when $N_t$ equals to the number of true edges (568), the top 15 nodes with the highest estimated degrees contain at least 14 out of the 15 true hub nodes in all replicates. On the other hand, both MB and glasso identify far fewer hub nodes, as their corresponding average rank curves are much higher than the grey horizontal line.

To investigate the impact of dimensionality $p$ and sample size $n$, we perform simulation studies for a larger dimension with $p = 1000$ and various sample sizes



(a) *x-axis*: the number of total detected edges(i.e., the total number of pairs $(i, j)$ with $\widehat{\rho}^{ij} \neq 0$); *y-axis*: the number of correctly identified edges. The vertical grey line corresponds to the number of true edges.



(b) *x-axis*: the number of total detected edges; *y-axis*: the average rank of the estimated degrees of the 15 true hub nodes.

Figure 3.2: Simulation results for Hub network.

Table 3.2: Power (sensitivity) of `space.dew` , `MB` and `glasso` in identifying correct edges when FDR is controlled at 0.05.

| Network | $p$ | $n$ | `space.dew` | `MB` | `glasso` |
|---|---|---|---|---|---|
| Hub-network | 500 | 250 | 0.844 | 0.784 | 0.655 |
| Hub-network | 1000 | 200 | 0.707 | 0.656 | 0.559 |
|  |  | 300 | 0.856 | 0.790 | 0.690 |
|  |  | 500 | 0.963 | 0.894 | 0.826 |
| Power-law network | 500 | 250 | 0.704 | 0.667 | 0.580 |

with $n = 200, 300$ and 500. The simulated network includes ten disjointed modules of size 100 each and has 1163 edges in total. Non-zero partial correlations form a similar distribution as that of the $p = 500$ network discussed above. The ROC curves for `space.dew`, `MB` and `glasso` resulted from these simulations are shown in Figure 3.3. When false discovery rate (=1-specificity) is controlled at 0.05, the power (=sensitivity) for detecting correct edges is given in Table 3.2. From the figure and the table, we observe that the sample size has a big impact on the performance of all methods. For $p = 1000$, when the sample size increases from 200 to 300, the power of `space.dew` increases more than 20%; when the sample size is 500, `space.dew` achieves an impressive power of 96%. On the other hand, the dimensionality seems to have relatively less influence. When the total number of variables is doubled from 500 to 1000, with only 20% more samples (that is $p = 500, n = 250$ vs. $p = 1000, n = 300$), all three methods achieve similar powers. This is presumably because the larger network ($p = 1000$) is sparser than the smaller network ($p = 500$) and also the complexity of the modules remains unchanged. Finally, it is obvious from Figure 3.3 that, `space.dew` performs best among the three methods.

We then investigate the performance of these methods at the selected tuning parameters (see Section 3.2.4 for details). For the above Hub network with $p = 1000$ nodes and $n = 200, 300, 500$, the results are reported in Table 3.3. As can be seen from the table, BIC based approaches tend to select large models (compared to the

Table 3.3: Edge detection under the selected tuning parameter $\lambda$. For average rank, the optimal value is 15.5. For `MB.alpha`, $\alpha = 0.05$ is used.

| Sample size | Method | Total edge detected | Sensitivity | Specificity | Average rank |
|---|---|---|---|---|---|
| $n = 200$ | `space.joint` | 1357 | 0.821 | 0.703 | 28.6 |
| | `MB.sep` | 1240 | 0.751 | 0.703 | 57.5 |
| | `MB.alpha` | 404 | 0.347 | 1.00 | 175.8 |
| | `glasso.like` | 1542 | 0.821 | 0.619 | 35.4 |
| $n = 300$ | `space.joint` | 1481 | 0.921 | 0.724 | 18.2 |
| | `MB.sep` | 1456 | 0.867 | 0.692 | 30.4 |
| | `MB.alpha` | 562 | 0.483 | 1.00 | 128.9 |
| | `glasso.like` | 1743 | 0.920 | 0.614 | 21 |
| $n = 500$ | `space.joint` | 1525 | 0.980 | 0.747 | 16.0 |
| | `MB.sep` | 1555 | 0.940 | 0.706 | 16.9 |
| | `MB.alpha` | 788 | 0.678 | 1.00 | 52.1 |
| | `glasso.like` | 1942 | 0.978 | 0.586 | 16.5 |

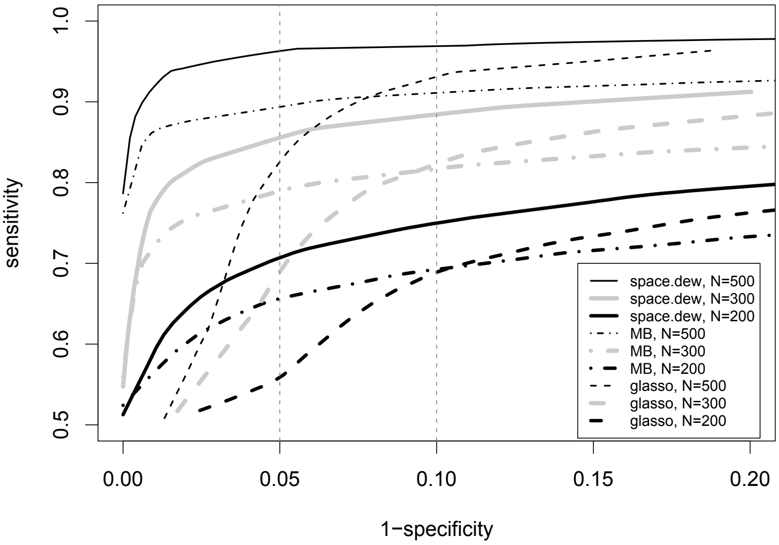


Figure 3.3: Hub network: ROC curves for different samples sizes ($p = 1000$).

true model which has 1163 edges). `space.joint` and `MB.sep` perform similarly in terms of specificity, and `glasso.like` works considerably worse than the other two in this regard. On the other hand, `space.joint` and `glasso.like` performs similarly in terms of sensitivity, and are better than `MB.sep` on this aspect. In contrast, `MB.alpha` selects very small models and thus results in very high specificity, but very low sensitivity. In terms of hub identification, `space.joint` apparently performs better than other methods (indicated by a smaller average rank over 30 true hub nodes). Moreover, the performances of all methods improve with sample size.

**Power-law networks** Many real world networks have a *power-law (also a.k.a scale-free)* degree distribution with an estimated power parameter $\alpha = 2 \sim 3$ [45]. Thus, in the second set of simulations, the module networks are generated according to a power-law degree distribution with the power-law parameter $\alpha = 2.3$, as this value is close to the estimated power parameters for biological networks [45]. Figure 3.1(b) illustrates a network formed by five such modules with each having 100 nodes. It can be seen that there are three obvious hub nodes in this network with degrees of at least 20. The simulated non-zero partial correlations fall in the range $(-0.51, -0.08] \cup [0.08, 0.51)$, with two modes around -0.22 and 0.22. Similar to the simulation done for Hub networks, we generate 50 independent data sets each consisting of $n = 250$ i.i.d. samples. We then compare the number of correctly detected edges by various methods. The result is shown in Figure 3.4. On average, when the number of total detected edges equals to the number of true edges which is 495, `space.dew` detects 406 correct edges, while `MB` detects only 378 and `glasso` detects only 381 edges. In terms of hub detection, all methods can correctly identify the three hub nodes for this network.

These simulation results suggest that when the (concentration) networks are rea-

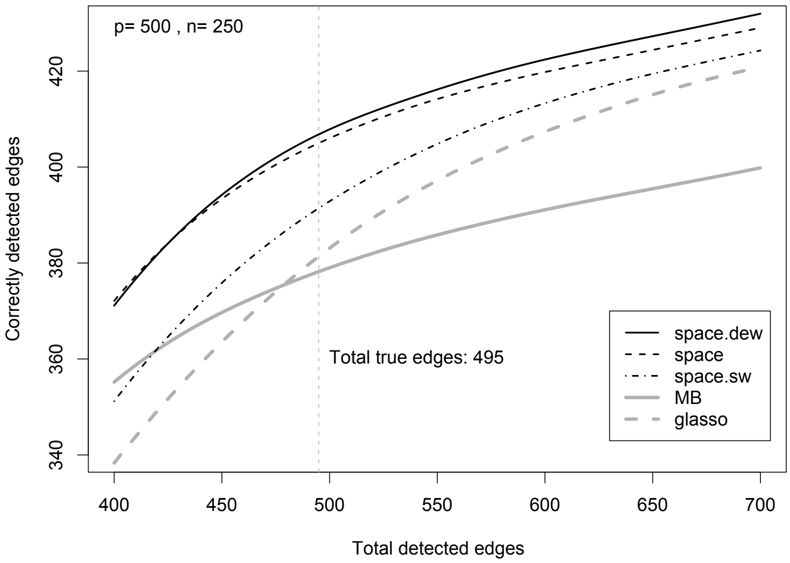


Figure 3.4: Simulation results for Power-law network. *x-axis*: the number of total detected edges; *y-axis*: the number of correctly identified edges. The vertical grey line corresponds to the number of true edges.

sonably sparse, we should be able to characterize their structures with only a couple-of-hundreds of samples when there are a couple of thousands of nodes. In addition, `space.dew` outperforms `MB` by at least 6% on the power of edge detection under all simulation settings above when FDR is controlled at 0.05, and the improvements are even larger when FDR is controlled at a higher level say 0.1 (see Figure 3.3). Also, compared to `glasso`, the improvement of `space.dew` is at least 15% when FDR is controlled at 0.05, and the advantages become smaller when FDR is controlled at a higher level (see Figure 3.3). Moreover, the `space` methods perform much better in hub identification than both `MB` and `glasso`. We have also applied `space` methods, `MB` and `glasso` on networks with nearly uniform degree distributions generated by following the simulation procedures in the paper of Meinshausen and Buhlmann [43], as well as the AR network discussed by Friedman et al. [23], Yuan and Lin [71]. For these cases, the space methods perform comparably, if not better than, the other two methods. However, for these networks without hubs, the advantages of space become smaller compared to the results on the networks with hubs. The detailed results are shown in Part II of Appendix B.

We conjecture that, under the sparse and high dimensional setting, the superior performance in model selection of the regression based method space over the penalized likelihood method is partly due to its simpler quadratic loss function. Moreover, since space ignores the correlation structure of the regression residuals, it amounts to a greater degree of regularization, which may render additional benefits under the sparse and high dimensional setting.

In terms of parameter estimation, we compare the entropy loss of the three methods. We find that, they perform similarly when the estimated models are of small or moderate size. When the estimated models are large, glasso generally performs better in this regard than the other two methods. Since the interest of this chapter lies in model selection, detailed results of parameter estimation are not reported here.

As discussed earlier, one limitation of space is its lack of assurance of positive definiteness. However, for simulations reported above, the corresponding estimators we have examined (over 3000 in total) are all positive definite. To further investigate this issue, we design a few additional simulations. We first consider a case with a similar network structure as the Hub network, however having a nearly singular concentration matrix (the condition number is $16,240$; as a comparison, the condition number for the original Hub network is 62). For this case, the estimate of space remains positive definite until the number of total detected edges increases to $50,000$; while the estimate of MB remains positive definite until the number of total detected edges is more than $23,000$. Note that, the total number of true edges of this model is only 568, and the model selected by `space.joint` has 791 edges. In the second simulation, we consider a denser network ($p = 500$ and the number of true edges is $6,188$) with a nearly singular concentration matrix (condition number is $3,669$). Again, we observe that, the `space` estimate only becomes non-positive-definite when the estimated models are huge (the number of detected edges is more than $45,000$). This suggests that, for the regime we are interested in in this chapter (the sparse and high dimensional setting), non-positive-definiteness does not seem to be a big issue for the proposed method, as it only occurs when the resulting model is huge and thus very far away from the true model. As long as the estimated models are reasonably sparse, the corresponding estimators by `space` remain positive definite. We believe that this is partly due to the heavy shrinkage imposed on the off-diagonal entries in order to ensure sparsity.

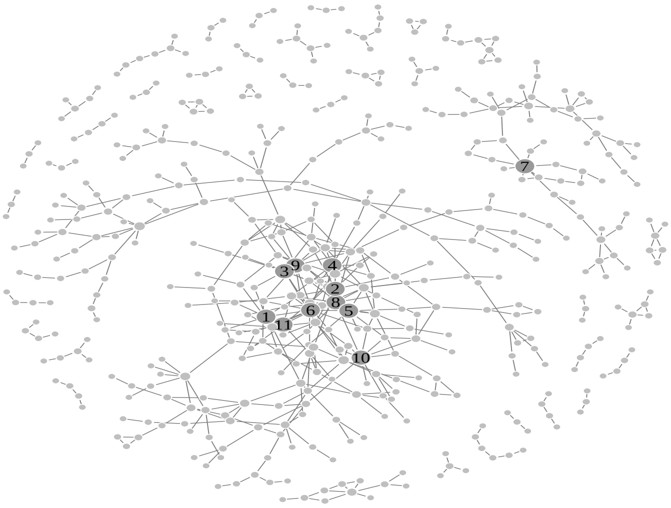
Finally, we investigate the performance of these methods when the observations come from a non-normal distribution. Particularly, we consider the multivariate $t_{df}$-distribution with $df = 3, 6, 10$. The performances of all three methods deteriorate compared to the normal case, however the overall picture in terms of relative performance among these methods remains essentially unchanged (detailed results not shown).

## 3.4 Application

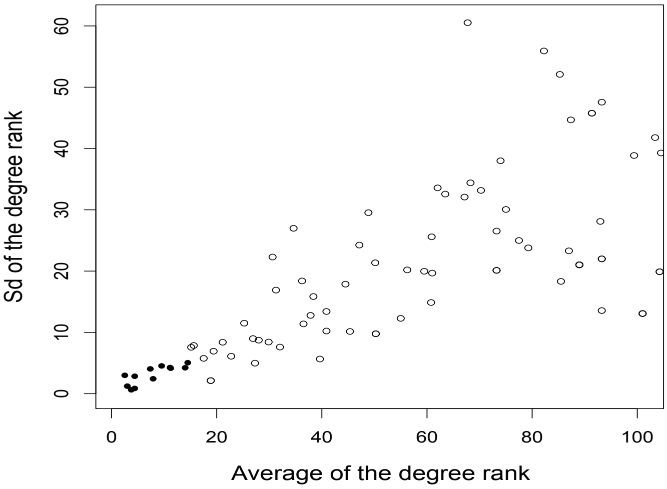
More than 500,000 women die annually of breast cancer world wide. Great efforts are being made to improve the prevention, diagnosis and treatment for breast cancer. Specifically, in the past couple of years, molecular diagnostics of breast cancer have been revolutionized by high throughput genomics technologies. A large number of gene expression signatures have been identified (or even validated) to have potential clinical usage. However, since breast cancer is a complex disease, the tumor process cannot be understood by only analyzing individual genes. There is a pressing need to study the interactions between genes, which may well lead to better understanding of the disease pathologies.

In a recent breast cancer study, microarray expression experiments were conducted for 295 primary invasive breast carcinoma samples [14, 66]. Raw array results and patient clinical outcomes for 244 of these samples are available on-line and are used in this chapter. Data can be downloaded at `http://microarray-pubs.stanford.edu/wound_NKI/explore.html`. To globally characterize the association among thousands of mRNA expression levels in this group of patients, we apply the `space` method on this data set as follows. First, for each expression array, we perform the global normalization by centering the mean to zero and scaling the median absolute deviation to one. Then we focus on a subset of $p = 1217$ genes/clones whose expression levels are significantly associated with tumor progression ($p$-values from univariate Cox models $< 0.0008$, corresponding FDR $= 0.01$). We estimate the partial correlation matrix of these 1217 genes with `space.dew` for a series of $\lambda$ values. The degree distribution of the inferred network is heavily skewed to the right. Specifically, when 629 edges are detected, 598 out of the 1217 genes do not connect to any other genes, while five genes have degrees of at least 10. The power-law parameter of this degree distribution is $\alpha = 2.56$ , which is consistent with the findings in the literature for GRNs [45]. The topology of the inferred network is shown in Figure 3.5(a), which supports the statement that genetic pathways consist of many genes with few interactions and a few hub genes with many interactions.

We then search for potential hub genes by ranking nodes according to their degrees. There are 11 candidate hub genes whose degrees consistently rank the highest under various $\lambda$ (see Figure 3.5(b)). Among these 11 genes, five are important known regulators in breast cancer. For example, *HNF3A* (also known as *FOXA1*) is a transcription factor expressed predominantly in a subtype of breast cancer, which regulates the expression of the cell cycle inhibitor $p27kip1$ and the cell adhesion molecule E-cadherin. This gene is essential for the expression of approximately 50% of estrogene-regulated genes and has the potential to serve as a therapeutic target [44]. Except for *HNF3A*, all the other 10 hub genes fall in the same big network component related to cell cycle/proliferation. This is not surprising as it is well-agreed that cell cycle/proliferation signature is prognostic for breast cancer. Specifically, *KNSL6, STK12, RAD54L* and *BUB1* have been previously reported to play a role in breast cancer: *KNSL6* (also known as *KIF2C*) is important for anaphase chromosome segregation and centromere separation, which is overexpressed in breast cancer cells but expressed undetectably in other human tissues except testis [58]; *STK12* (also known as *AURKB*) regulates chromosomal segregation during mitosis as well as meiosis, whose LOH contributes to an increased breast cancer risk and may influence the therapy outcome [61]; RAD54L is a recombinational repair protein associated with tumor suppressors BRCA1 and BRCA2, whose mutation leads to defect in repair processes involving homologous recombination and triggers the tumor development [41]; in the end, BUB1 is a spindle checkpoint gene and belongs to the BML-1 oncogene-driven pathway, whose activation contributes to the survival life cycle of cancer stem cells and promotes tumor progression. The roles of the other six hub genes in breast cancer are worth of further investigation. The functions of all hub genes are briefly summarized in Table 3.4.



(a) Network inferred from the real data (only showing components with at least three nodes). The gene annotation of the hub nodes (numbered) are given in Table 3.4.



(b) Degree ranks (for the 100 genes with highest degrees). Different circles represent different genes. *Solid circles*: the 11 genes with highest degrees. *Circles*: the other genes. The sd(rank) of the top 11 genes are all smaller than 4.62 (4.62 is the 1% quantile of sd(rank) among all the 1217 genes), and thus are identified as hub nodes.

Figure 3.5: Results for the breast cancer expression data set.

Table 3.4: Annotation of hub genes

| Index | Gene Symbol | Summary Function (GO) |
|---|---|---|
| 1 | CENPA | Encodes a centromere protein (nucleosome assembly) |
| 2 | *NA.* | *Annotation not available* |
| 3 | KNSL6 | Anaphase chromosome segregation (cell proliferation) |
| 4 | STK12 | Regulation of chromosomal segregation (cell cycle) |
| 5 | *NA.* | *Annotation not available* |
| 6 | URLC9 | *Annotation not available* (up-regulated in lung cancer) |
| 7 | HNF3A | Transcriptional factor activity (epithelial cell differentiation) |
| 8 | TPX2 | Spindle formation (cell proliferation) |
| 9 | RAD54L | Homologous recombination related DNA repair (meiosis) |
| 10 | ID-GAP | Stimulate GTP hydrolysis (cell cycle) |
| 11 | BUB1 | Spindle checkpoint (cell cycle) |

## 3.5 Summary

In this chapter, we propose a joint sparse regression model – `space` – for selecting non-zero partial correlations under the high-dimension-low-sample-size setting. By controlling the overall sparsity of the partial correlation matrix, `space` is able to automatically adjust for different neighborhood sizes and thus to utilize data more effectively. The proposed method also explicitly employs the symmetry among the partial correlations, which also helps to improve efficiency. Moreover, this joint model makes it easy to incorporate prior knowledge about network structure. We develop a fast algorithm `active-shooting` to implement the proposed procedure, which can be readily extended to solve some other penalized optimization problems. We also propose a “BIC-type” criterion for the selection of the tuning parameter. With extensive simulation studies, we demonstrate that this method achieves good power in non-zero partial correlation selection as well as hub identification, and also performs favorably compared to two existing methods. The impact of the sample size and dimensionality has been examined on simulation examples as well. We then apply this method on a microarray data set of 1217 genes from 244 breast cancer tumor samples, and find 11 candidate hubs, of which five are known breast cancer related regulators.

## 3.6 Appendix B

### Part I

In this section, we provide details for the implementation of `space` which takes advantage of the sparse structure of $\boldsymbol{\mathcal{X}}$. Denote the target loss function as

$$f(\boldsymbol{\theta}) = \frac{1}{2}\left\|\boldsymbol{\mathcal{Y}} - \boldsymbol{\mathcal{X}}\boldsymbol{\theta}\right\|^2 + \lambda_1 \sum_{i<j} |\rho^{ij}|. \tag{3.8}$$

Our goal is to find $\widehat{\boldsymbol{\theta}} = \text{argmin}_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$ for a given $\lambda_1$. We will employ `active-shooting` algorithm (Section 2.3) to solve this optimization problem.

Without loss of generality, we assume $\text{mean}(\boldsymbol{Y}_i) = 1/n \sum_{k=1}^n y_i^k = 0$ for $i = 1, \ldots, p$. Denote $\xi_i = \boldsymbol{Y}_i^T \boldsymbol{Y}_i$. We have

$$\boldsymbol{\mathcal{X}}_{(i,j)}^T \boldsymbol{\mathcal{X}}_{(i,j)} = \xi_j \frac{\sigma^{jj}}{\sigma^{ii}} + \xi_i \frac{\sigma^{ii}}{\sigma^{jj}};$$

$$\boldsymbol{\mathcal{Y}}^T \boldsymbol{\mathcal{X}}_{(i,j)} = \sqrt{\frac{\sigma^{jj}}{\sigma^{ii}}} \boldsymbol{Y}_i^T \boldsymbol{Y}_j + \sqrt{\frac{\sigma^{ii}}{\sigma^{jj}}} \boldsymbol{Y}_j^T \boldsymbol{Y}_i.$$

Denote $\rho^{ij} = \rho_{(i,j)}$. We now present details of the initialization step and the updating steps in the `active-shooting` algorithm.

### 1. Initialization

Let

$$\begin{aligned} \rho_{(i,j)}^{(0)} &= \frac{\left(|\boldsymbol{\mathcal{Y}}^T \boldsymbol{\mathcal{X}}_{(i,j)}| - \lambda_1\right)_+ \cdot \text{sign}(\boldsymbol{\mathcal{Y}}^T \boldsymbol{\mathcal{X}}_{(i,j)})}{\boldsymbol{\mathcal{X}}_{(i,j)}^T \boldsymbol{\mathcal{X}}_{(i,j)}} \\ &= \frac{\left(\left|\sqrt{\frac{\sigma^{jj}}{\sigma^{ii}}} \boldsymbol{Y}_i^T \boldsymbol{Y}_j + \sqrt{\frac{\sigma^{ii}}{\sigma^{jj}}} \boldsymbol{Y}_j^T \boldsymbol{Y}_i\right| - \lambda_1\right)_+ \cdot \text{sign}(\boldsymbol{Y}_i^T \boldsymbol{Y}_j)}{\xi_j \frac{\sigma^{jj}}{\sigma^{ii}} + \xi_i \frac{\sigma^{ii}}{\sigma^{jj}}}. \end{aligned} \tag{3.9}$$

For $j = 1, \ldots, p$, compute

$$\widehat{\boldsymbol{Y}}_j^{(0)} = \left(\sqrt{\frac{\sigma^{11}}{\sigma^{jj}}} \boldsymbol{Y}_1, ..., \sqrt{\frac{\sigma^{pp}}{\sigma^{jj}}} \boldsymbol{Y}_p\right) \cdot \begin{pmatrix} \rho_{(1,j)}^{(0)} \\ \vdots \\ \rho_{(p,j)}^{(0)} \end{pmatrix}, \tag{3.10}$$

and

$$\boldsymbol{E}^{(0)} = \boldsymbol{\mathcal{Y}} - \widehat{\boldsymbol{\mathcal{Y}}}^{(0)} = \left((\boldsymbol{E}_1^{(0)})^T, ..., (\boldsymbol{E}_p^{(0)})^T\right), \tag{3.11}$$

where $\boldsymbol{E}_j^{(0)} = \boldsymbol{Y}_j - \widehat{\boldsymbol{Y}}_j^{(0)}$, for $1 \leq j \leq p$.

**2. Update** $\rho_{(i,j)}^{(0)} \longrightarrow \rho_{(i,j)}^{(1)}$

Let

$$A_{(i,j)} = (\boldsymbol{E}_j^{(0)})^T \cdot \sqrt{\frac{\sigma^{ii}}{\sigma^{jj}}} \boldsymbol{Y}_i, \tag{3.12}$$

$$A_{(j,i)} = (\boldsymbol{E}_i^{(0)})^T \cdot \sqrt{\frac{\sigma^{jj}}{\sigma^{ii}}} \boldsymbol{Y}_j. \tag{3.13}$$

We have

$$\begin{aligned} (\boldsymbol{E}^{(0)})^T \boldsymbol{\mathcal{X}}_{(i,j)} &= (\boldsymbol{E}_i^{(0)})^T \cdot \sqrt{\tfrac{\sigma^{jj}}{\sigma^{ii}}} \boldsymbol{Y}_j + (\boldsymbol{E}_j^{(0)})^T \cdot \sqrt{\tfrac{\sigma^{ii}}{\sigma^{jj}}} \boldsymbol{Y}_i \\ &= A_{(j,i)} + A_{(i,j)}. \end{aligned} \tag{3.14}$$

It follows

$$\begin{aligned} \rho_{(i,j)}^{(1)} &= \operatorname{sign}\left( \frac{(\boldsymbol{E}^{(0)})^T \boldsymbol{\mathcal{X}}_{(i,j)}}{\boldsymbol{\mathcal{X}}_{(i,j)}^T \boldsymbol{\mathcal{X}}_{(i,j)}} + \rho_{(i,j)}^{(0)} \right) \left( \left| \frac{(\boldsymbol{E}^{(0)})^T \boldsymbol{\mathcal{X}}_{(i,j)}}{\boldsymbol{\mathcal{X}}_{(i,j)}^T \boldsymbol{\mathcal{X}}_{(i,j)}} + \rho_{(i,j)}^{(0)} \right| - \frac{\lambda_1}{\boldsymbol{\mathcal{X}}_{(i,j)}^T \boldsymbol{\mathcal{X}}_{(i,j)}} \right)_+ \\ &= \operatorname{sign}\left( \frac{A_{(j,i)} + A_{(i,j)}}{\xi_j \frac{\sigma^{jj}}{\sigma^{ii}} + \xi_i \frac{\sigma^{ii}}{\sigma^{jj}}} + \rho_{(i,j)}^{(0)} \right) \left( \left| \frac{A_{(j,i)} + A_{(i,j)}}{\xi_j \frac{\sigma^{jj}}{\sigma^{ii}} + \xi_i \frac{\sigma^{ii}}{\sigma^{jj}}} + \rho_{(i,j)}^{(0)} \right| - \frac{\lambda_1}{\xi_j \frac{\sigma^{jj}}{\sigma^{ii}} + \xi_i \frac{\sigma^{ii}}{\sigma^{jj}}} \right)_+ . \end{aligned} \tag{3.15}$$

**3. Update** $\rho^{(t)} \longrightarrow \rho^{(t+1)}$

From the previous iteration, we have

- $\boldsymbol{E}^{(t-1)}$: residual in the previous iteration ($np \times 1$ vector).
- $(i_0, j_0)$: index of coefficient that is updated in the previous iteration.
- $\rho_{(i,j)}^{(t)} = \begin{cases} \rho_{(i,j)}^{(t-1)} & \text{if } (i,j) \neq (i_0, j_0), \text{ nor } (j_0, i_0) \\ \rho_{(i,j)}^{(t-1)} - \Delta & \text{if } (i,j) = (i_0, j_0), \text{ or } (j_0, i_0) \end{cases}$

Then,

$$\begin{aligned}
\boldsymbol{E}_k^{(t)} &= \boldsymbol{E}_k^{(t-1)} \text{ for } k \neq i_0, j_0; \\
\boldsymbol{E}_{j_0}^{(t)} &= \boldsymbol{E}_{j_0}^{(t-1)} + \widehat{\boldsymbol{Y}}_{j_0}^{(t-1)} - \widehat{\boldsymbol{Y}}_{j_0}^{(t)} \\
&= \boldsymbol{E}_{j_0}^{(t-1)} + \textstyle\sum_{i=1}^{p} \sqrt{\frac{\sigma^{ii}}{\sigma^{j_0 j_0}}}\boldsymbol{Y}_i(\rho_{(i,j_0)}^{(t-1)} - \rho_{(i,j_0)}^{(t)}) \\
&= \boldsymbol{E}_{j_0}^{(t-1)} + \sqrt{\frac{\sigma^{i_0 i_0}}{\sigma^{j_0 j_0}}}\boldsymbol{Y}_{i_0} \cdot \Delta; \\
\boldsymbol{E}_{i_0}^{(t)} &= \boldsymbol{E}_{i_0}^{(t-1)} + \sqrt{\frac{\sigma^{j_0 j_0}}{\sigma^{i_0 i_0}}}\boldsymbol{Y}_{j_0} \cdot \Delta.
\end{aligned} \tag{3.16}$$

Suppose the index of the coefficient we would like to update in this iteration is $(i_1, j_1)$, then let

$$A_{(i_1,j_1)} = (\boldsymbol{E}_{j_1}^{(t)})^T \cdot \sqrt{\frac{\sigma^{i_1 i_1}}{\sigma^{j_1 j_1}}}\boldsymbol{Y}_{i_1},$$

$$A_{(j_1,i_1)} = (\boldsymbol{E}_{i_1}^{(t)})^T \cdot \sqrt{\frac{\sigma^{j_1 j_1}}{\sigma^{i_1 i_1}}}\boldsymbol{Y}_{j_1}.$$

We have

$$\begin{aligned}
\rho_{(i,j)}^{(t+1)} = \quad & \operatorname{sign}\left(\frac{A_{(j_1,i_1)} + A_{(i_1,j_1)}}{\xi_j \frac{\sigma^{j_1 j_1}}{\sigma^{i_1 i_1}} + \xi_{i_1}\frac{\sigma^{i_1 i_1}}{\sigma^{j_1 j_1}}} + \rho_{(i_1,j_1)}^{(t)}\right) \\
& \times \left(\left|\frac{A_{(j_1,i_1)} + A_{(i_1,j_1)}}{\xi_j \frac{\sigma^{j_1 j_1}}{\sigma^{i_1 i_1}} + \xi_{i_1}\frac{\sigma^{i_1 i_1}}{\sigma^{j_1 j_1}}} + \rho_{(i_1,j_1)}^{(t)}\right| - \frac{\lambda_1}{\xi_j \frac{\sigma^{jj}}{\sigma^{ii}} + \xi_i \frac{\sigma^{ii}}{\sigma^{jj}}}\right)_+ .
\end{aligned} \tag{3.17}$$

Using the above steps 1–3, we have implemented the `active-shooting` algorithm in `c`, and the corresponding `R` package `space` to fit the `space` model is available on `cran`.

**Part II**

In this section we apply `space`, `MB` and `glasso` on several examples used by Yuan and Lin [71], Meinshausen and Buhlmann [43], and Friedman et al. [23].

(a) Chain network AR(1) [23, 71]

We consider an AR(1) model with $p = 500$, $n = 250$. The concentration matrix is as follows: $\sigma^{ii} = 1$; $\sigma^{ij}$ is 0.25 for $|i-j| = 1$ and 0 otherwise. We then calculate the covariance matrix and re-scale it to have diagonal 1. The condition number of the resulting concentration matrix is 3. The number of true edges in the corresponding network is 499 (a chain shape). We apply `space`, `MB` and `glasso` for a large range of tuning parameters, such that the sizes of the estimated models vary from 300 to 550. All the estimated concentration matrices by `space` and `MB` are positive definite. The results are shown in Figure 3.6.

(b) Circle network

We consider a Circle network with $p = 500$, $n = 250$. The concentration matrix is as follows: $\sigma^{ii} = 1$; $\sigma^{i,i-1} = \sigma^{i-1,i} = 0.3$; and $\sigma^{1,n} = \sigma^{n,1} = 0.3$. We then calculate the covariance matrix and re-scale it to diagonal 1. The condition number of the resulting matrix is 4. The number of true edges in the corresponding network is 500. We apply `space`, `MB` and `glasso` for a large range of tuning parameters, such that the sizes of the estimated models vary from 300 to 650. All the estimated concentration matrices by `space` and `MB` are positive definite. The results are shown in Figure 3.7.

(c) Uniform network [23, 43]

We consider a network similarly as the one used by Meinshausen and Buhlmann [43] with $p = 500$, $n = 250$. We first generate $p$ points uniformly on the two-dimensional unit square $[0, 1]^2$. Then we draw an edge between each pair of nodes with a probability of $\varphi(d/\sqrt{p})$, where $d$ is the Euclidean distance between the pair of variables and $\varphi$ is the density of the standard normal distribution. Then, for each node, if its degree $k$ is larger than 4, we randomly remove $k - 4$ edges connecting to this node. We repeat this process until the maximum degree of the network is 4. Then we set $\sigma^{ii} = 1$, $\sigma^{ij} = 0.245$ if there is an edge between

$(i, j)$, and $\sigma^{ij} = 0$ otherwise. In the end, we re-scale the covariance matrix such that the diagonal elements are 1. The condition number of the resulting matrix is 6.6. The number of true edges in the corresponding network is 447. We apply `space`, `MB` and `glasso` for a large range of tuning parameters, such that the sizes of the estimated models vary from 300 to 550. All the estimated concentration matrices by `space` and `MB` are positive definite. The results are shown in Figure 3.8.

As expected, in all the above simulations, the estimates of `space` are always positive definite (at least within the range that we have examined), as these examples are all well conditioned. Moreover, for these networks without hubs, the performance of `space` is at least comparable to, if not better than, the performance of `MB` and `glasso`.

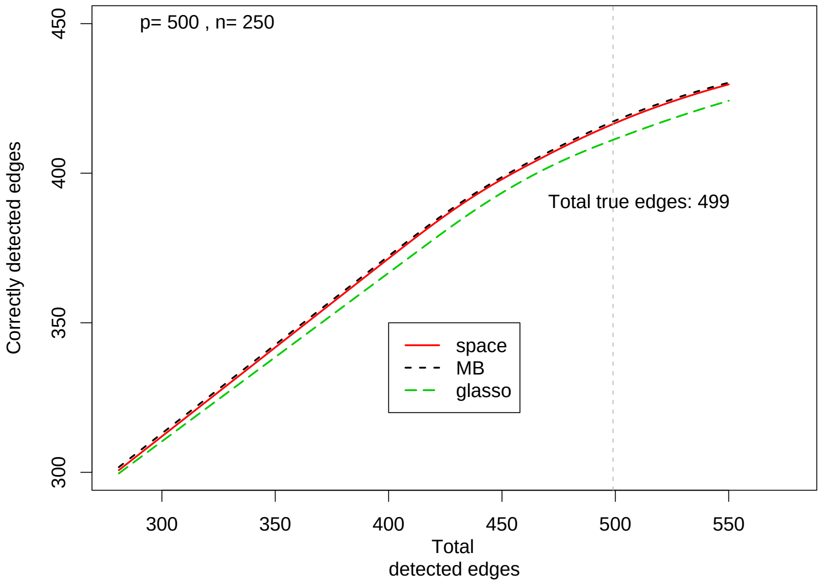


Figure 3.6: Chain network (AR(1)). *x-axis*: the number of total detected edges; *y-axis*: the number of correctly identified edges. The vertical grey line corresponds to the number of true edges.

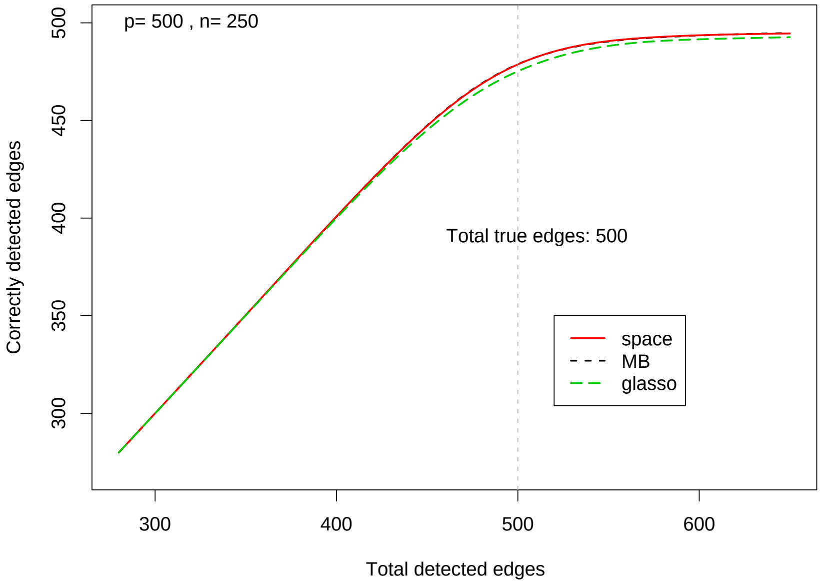


Figure 3.7: Circle network . *x-axis*: the number of total detected edges; *y-axis*: the number of correctly identified edges. The vertical grey line corresponds to the number of true edges.

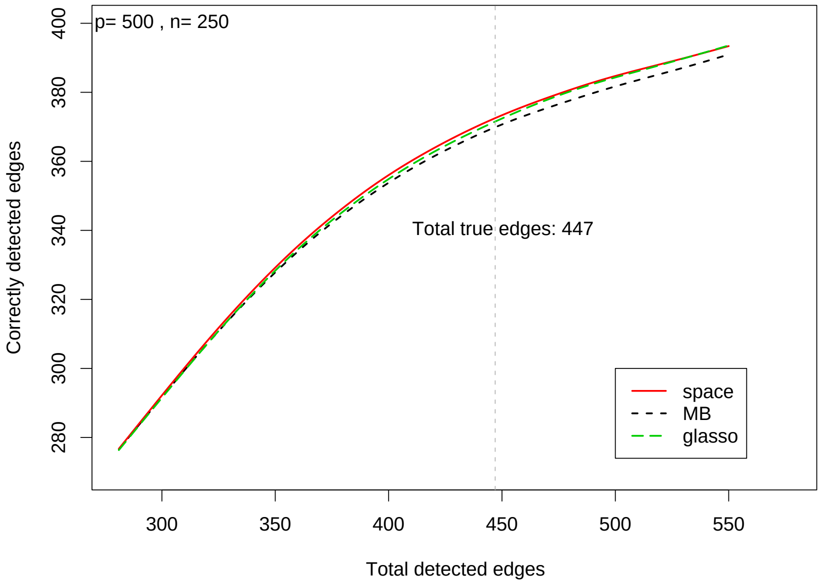


Figure 3.8: Uniform network . *x-axis*: the number of total detected edges; *y-axis*: the number of correctly identified edges. The vertical grey line corresponds to the number of true edges.

# CHAPTER IV

## Sparse Regulation Networks

In many organisms the expression levels of each gene are controlled by the activation levels of known "Transcription Factors" (TF). A problem of considerable interest is that of estimating the "Transcription Regulation Networks" (TRN) relating the TFs and genes. While the expression levels of genes can be observed, the activation levels of the corresponding TFs are usually unknown; greatly increasing the difficulty of the problem. Based on previous experimental work it is often the case that partial information about the TRN is available. For example, certain TFs may be known to regulate a given gene or in other cases a connection may be predicted with a certain probability. In general the biology of the problem indicates there will be very few connections between TFs and genes. Several methods have been proposed for estimating TRNs, however, they all suffer from problems such as unrealistic assumptions about prior knowledge of the network structure or computational limitations. We propose a new approach that can directly utilize prior information about the network structure in conjunction with observed gene expression data to estimate the TRN. Our approach uses $L_1$-norm penalties on the network to ensure a sparse structure. This has the advantage of being computationally efficient as well as making many fewer assumptions about the network structure. We use our methodology to construct the TRN for *E. coli* and show that the estimate is biologically sensible and compares favorably with previous estimates.

## 4.1 Introduction

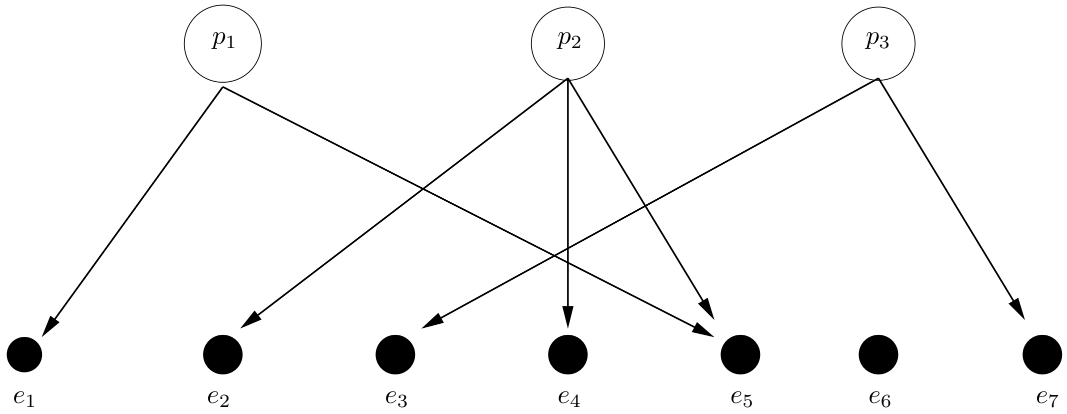


Figure 4.1: A general network with $L = 3$ transcription factors and $n = 7$ genes.

Recent progress in genomic technology allows scientists to gather vast and detailed information on DNA sequences, their variability, the timing and modality of their translation into proteins, and their abundance and interacting partners. The fields of system and computational biology have been redefined by the scale and resolution of these datasets and the necessity to interpret this data deluge. One theme that has clearly emerged is the importance of discovering, modeling, and exploiting interactions among different biological molecules. In some cases, these interactions can be measured directly, in others they can be inferred from data on the interacting partners. In this context, reconstructing networks, analyzing their behavior and modeling their characteristics have become fundamental problems in computational biology.

Depending on the type of biological process considered, and the type of data available, different network structures and different graph properties are relevant. In this work we focus on one type of bipartite network that has been used to model transcription regulation, among other processes, and is illustrated in Figure 4.1. One distinguishes input ($p_1, p_2, p_3$ in Figure 4.1) and output nodes ($e_1, \ldots, e_7$ in Figure 4.1); directed edges connect input nodes to one or more output nodes and indicate control. Furthermore, we can associate a numerical value with each edge, which indicates the nature and strength of the control. The topology of this network can be described with a 0-1 matrix $\boldsymbol{Z}$ with as many rows as the output nodes and as many columns as the input nodes, and where $z_{ij} = 1$ if there is a direct edge from input node $j$ to output node $i$. An analogous matrix $\boldsymbol{A}$ can be used to store the numerical information on the strength of the control, when this is available.

Bipartite networks such as the one illustrated in Figure 4.1 have been successfully used to describe and analyze transcription regulation [39]. Transcription is the initial step of the process where by the information stored in genes is used by the cell to assemble proteins. To adapt to different cell functions and different environmental conditions, only a small number of the genes in the DNA are transcribed at any given time. Understanding this selective process is the first step towards understanding how the information statically coded in DNA dynamically governs all cell life. One critical role in the regulation of this process is played by transcription factors. These molecules bind in the promoter region of the genes, facilitating or making it impossible for the transcription machinery to access the relevant portion of the DNA. To respond to different environments, transcription factors have multiple chemical configurations, typically existing both in “active” and “inactive” forms. Their binding affinity to the DNA regulatory regions vary depending on the particular chemical configuration, allowing for a dynamic regulation of transcription. Depending on the complexity of the organism at hand, the total number of transcription factors (TFs) varies, as well as the number of TFs participating to the regulation of each gene. In bipartite networks such as the one in Figure 4.1, input nodes can be taken to represent the variable concentrations in the active form of transcription factors, and output nodes as the transcript amounts of different genes. An edge connecting a TF to a gene indicates that the TF participates in the control of the gene transcription. As usual, mathematical stylization only captures a simplified version of reality. Bipartite graphs overlook some specific mechanisms of transcription regulation, such as self-regulation of TF expression or feed-back loops connecting genes to transcription factors. Despite these limitations, networks such as the one in Figure 4.1 provide a useful representation of a substantial share of the biological process.

Researchers interested in reconstructing transcription regulation have at their disposal a variety of measurement types, which in turn motivate diverse estimation strategies. The data set that motivated the development of our methodology consisted of measurements of gene transcription levels for *E. coli*, obtained from a collection of 35 gene expression arrays. These experiments, relatively cheap and fairly common, allow one to quantify transcription amounts for all the genes in the *E. coli* genome, under diverse cell conditions. While our data consists of measurements on the the output nodes i.e. the gene expression levels, we also have access to some information on the topology of the network: DNA sequence analysis or ChIP-chip experiments can be used to evaluate the likelihood of each possible edge. However, we have no direct measurements of the input nodes i.e. the concentrations of the active form of the TFs. While, in theory, it is possible to obtain these measurements, they are extremely expensive and are typically unavailable. Changes in transcription of TF are measured with gene expression arrays, but these are usually very limited and responsible for a portion of gene expression variation that we do not aim to analyze (one classical example is cell-cycle induced differences).

Our *E. coli* data consist of spotted array experiments with two dyes, which measure the changes in expression from a baseline level for the queried genes (taking the logarithm of the ratio of intensities, typically reported as raw data). These percentage changes can be related linearly to variations in the concentrations of the active form of the transcription factors [39]. Coupling this linearity assumption, with the bipartite network structure, we model the log-transformed expressions of gene $i$ in experiment $t$, $e_{it}$, as

$$e_{it} = \sum_{j=1}^{L} a_{ij} p_{jt} + \epsilon_{it}, \quad i = 1, \ldots, n, \quad t = 1, \ldots, T$$

where $n$, $L$ and $T$ denote the number of genes, TFs, and experiments respectively; $a_{ij}$ represents the control strength of transcription factor $j$ on gene $i$; $p_{jt}$ the concentration of transcription factor $j$ in experiment $t$; and $\epsilon_{it}$ captures i.i.d. measurement errors and biological variability. A value of $a_{ij} = 0$ indicates that there is no network connection, or equivalently no relationship, between gene $i$ and TF $j$ while non-zero values imply that changes in the TF affect the gene's expression level. It is convenient to formulate the model in matrix notation,

$$\boldsymbol{E} = \boldsymbol{AP} + \boldsymbol{\epsilon}, \tag{4.1}$$

where $\boldsymbol{E}$ is an $n \times T$ matrix of $e_{it}$'s, $\boldsymbol{A}$ is an $n \times L$ matrix of $a_{ij}$'s and $\boldsymbol{P}$ is an $L \times T$ matrix of $p_{jt}$'s. $\boldsymbol{A}$ and $\boldsymbol{P}$ are both unknown quantities.

A number of variants of model (4.1) have been applied to the study of gene expression data. The first attempts utilized dimension reduction techniques such as principal component analysis or singular value decomposition [1]. Using this approach a unique solution to simultaneously estimate the $p_j$'s and the strength of the network connections is obtained by assuming orthogonality of the $p_j$'s–an assumption that does not have biological motivations. An interesting development is the use of Independent Component Analysis, where the orthogonality assumption is substituted by stochastic independence [33]. These models can be quite effective in providing a dimensionality reduction, but the resulting $\boldsymbol{P}$'s often lack interpretability.

When the gene expression data refers to a series of experiments in a meaningful order (temporal, by degree of exposure, etc), model (4.1) can be considered as the emission component of a state space model, where hidden states can be meaningfully connected to transcription factors [5, 38, 54]. Depending on the amount of knowledge assumed on the $\boldsymbol{A}$ matrix, state space models can deal with networks of different size and complexities.

Values of the factors, $\boldsymbol{P}$, that are clearly interpretable as changes in concentration of transcription factors together with the identifiability of model (4.1) can be achieved by imposing restrictions on $\boldsymbol{A}$ that reflect available knowledge on the topology of the network. Liao et al. [39] assume the entire network structure is known *a priori* and gives conditions for identifiability of $\boldsymbol{A}$ and $\boldsymbol{P}$ based on the pattern of zeros in $\boldsymbol{A}$, reflecting the natural sparsity of the system. A simple iterative least squares procedure is proposed for estimation, and the bootstrap used to asses variability.

This approach has two substantial limitations. First, it assumes that the entire network structure is known, while in practice it is most common for only parts of the structure to have been thoroughly studied. Second, not all known transcription networks satisfy the identifiability conditions. A number of subsequent contributions have addressed some of these limitations. Tran et al. [64] introduce other, more general, identifiability conditions; Yu and Li [70] propose an alternative estimation procedure for the factor model; Brynildsen et al. [10] explore the effect of inaccurate specification of the network structure; Chang et al. [13] propose a faster algorithm. Particularly relevant to this chapter is the work of Sabatti and James [52], which removes both limitations of the method of Liao et al. [39] by using a Bayesian approach. The authors obtain a prior probability on the network structure using sequence analysis, and then use a Gibbs sampler to produce posterior estimates of the TRN. In theory, this approach can be applied to any network structure, even when only part of the structure is known. However, a significant limitation is that the computational effort required to implement the Gibbs sampler grows exponentially with the number of potential connections between a particular gene and the transcription factors. As a result, one is forced to choose a prior on the network where the probability of most edges is set to zero, thereby fixing a priori a large portion of the topology. While sparsity in the connections is biologically reasonable, it would obviously be more desirable to allow the gene expression data to directly identify the connections.

To overcome these limitations, we take a somewhat different approach in this chapter that builds in the same advantages as the Bayesian approach in terms of utilizing partial network information and working on any structure. However, our approach is more computationally efficient, which allows increased flexibility in determining the final network topology. We treat the estimation of both the connection strengths, $\boldsymbol{A}$, and the transcription factors concentrations, $\boldsymbol{P}$, as a variable selection problem. In this context, our data has an extremely large number of variables, i.e. potential connections, but is sparse in terms of the number of “true” variables, i.e. connections that actually exist. There have recently been important methodological innovations for this type of variable selection problem. A number of these methods involve the use of an $L_1$-norm penalty on the regression coefficients which has the effect of performing automatic variable selection. A few examples include the Lasso

[63], the adaptive Lasso [76], SCAD [20], the Elastic Net [77], the Dantzig selector [12], the Relaxed Lasso [42], VISA [50], and the Double Dantzig [30]. The most well known of these approaches is the Lasso, which performs variable selection by imposing an $L_1$-norm penalty on the regression coefficients. In analogy with the Lasso, our method also utilizes $L_1$-norm penalties on the connection strengths, $\boldsymbol{A}$, as well as the transcription factor concentrations, $\boldsymbol{P}$. This allows us to automatically produce a sparse network structure, which incorporates the prior information. We show that, given the same prior network, our approach produces similar results to the Bayesian formulation, but is considerably more computationally efficient, which allows us to assume a less restrictive prior.

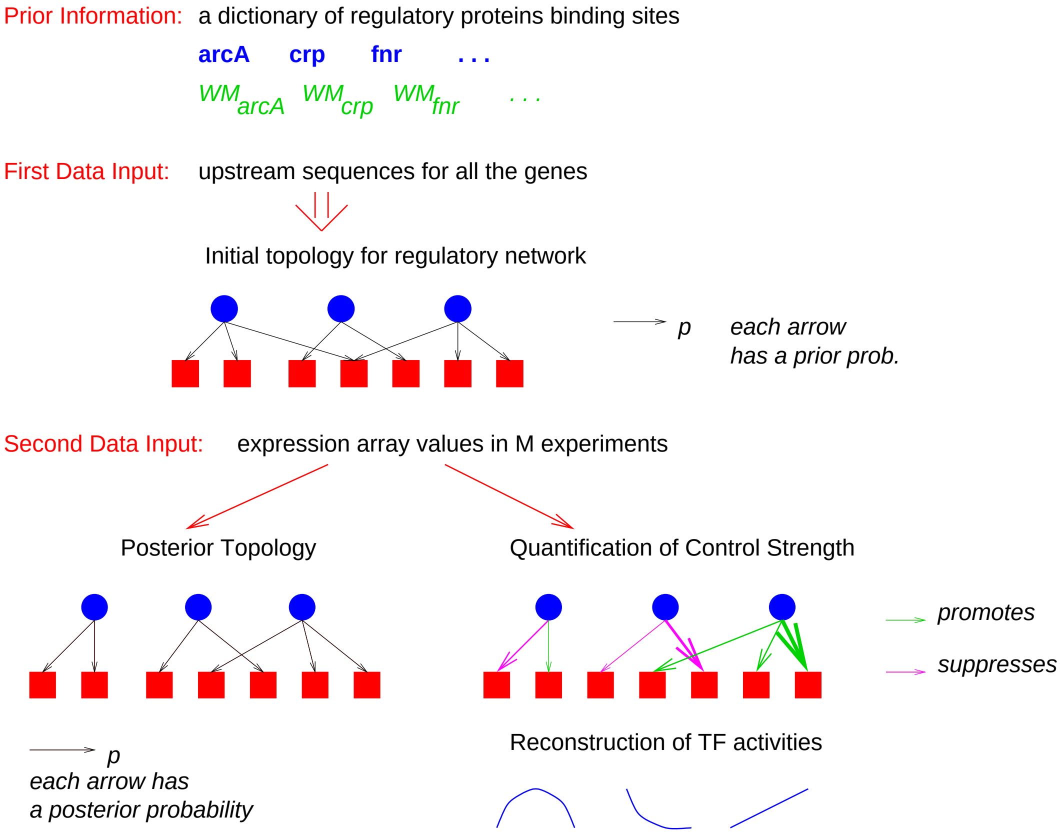

Prior Information:
a dictionary of regulatory proteins binding sites
arcA
crp
fnr
. . .
WM_arcA
WM_crp
WM_fnr
. . .
First Data Input:
upstream sequences for all the genes
Initial topology for regulatory network
p
each arrow
has a prior prob.
Second Data Input:
expression array values in M experiments
Posterior Topology
Quantification of Control Strength
promotes
suppresses
p
each arrow has
a posterior probability
Reconstruction of TF activities

Figure 4.2: Transcription network reconstruction integrating DNA sequence and gene expression information. Circles represent regulatory proteins and Squares genes. An arrow connecting a circle to a square indicates that the transcription factor controls the expression of the gene. Varying arrow thickness signifies different control strengths.

Figure 4.2 gives a schematic illustration of our approach. First, we identify a group of transcription factors that are believed to regulate the gene expression levels. Second, we compute an initial topology for the network using both documented experimental evidence, as well as an analysis of the DNA sequence up-stream of a given gene. Finally, we use the initial topology, as well as the gene expression levels from multiple experiments, as inputs to our $L_1$-norm penalized regression approach to produce an updated final network topology, a quantification of the connection strengths and an estimation of the transcription factor levels.

This chapter is structured as follows. In Section 4.2 we provide a detailed description of the data that we are analyzing and the available prior information. Section 4.3 develops the methodological approach we use to fit the transcription regulation network. Our analysis of the *E. coli* data is presented in Section 4.4. We also include a comparison with the results using the Bayesian approach [52]. A simulation study is provided in Section 4.5 followed by a discussion in Section 4.6.

## 4.2 Data and prior information on network structure

The data set that motivated the development of our methodology included 35 microarray experiments of *Escherichia coli* that were either publicly available or were carried out in the laboratory of Professor James C. Liao at UCLA. The experiments consisted of Tryptophan timecourse data (1-12) [32], glucose acetate transition data (13-19) [46, 48], UV exposure data (20-24) [15] and a protein overexpression timecourse dataset (25-35) [47]. To reduce spurious effects due to the inhomogeneity of the data collection, we standardized the values of each experiment, so that the mean across all genes in each experiment was zero and the variance one. Merging these different datasets resulted in expression measurements on 1433 genes across 35 experiments. In general terms, biological knowledge of the nature of the microarray experiments suggested that the TrpR regulon should be activated in the Tryptophan timecourse data, the LexA regulon should be activated in the UV experiments, and the RpoH regulon in the protein overexpression data.

We also were able to identify partial information about the network structure connecting the transcription factors and genes. We first identified a set of transcription factors that previous literature suggested were important in this system: this resulted in 37 transcription factors. Recall that our bipartite network structure can be represented using an $n \times L$, 0-1 matrix, $\boldsymbol{Z}$, where $n = 1433$ is the number of genes under consideration and $L = 37$ is the number of transcription factors. The element $z_{ij}$ is one if TF $j$ regulates gene $i$, and zero otherwise. For a number of well studied TF experimental data is available that clearly indicates their binding in the upstream region of regulated genes (in other words $z_{ij} = 1$). However, for many of the elements of $\boldsymbol{Z}$, only partial information is available.

To summarize the prior evidence on the network structure, we introduce $\pi_{ij} = P(z_{ij} = 1)$. If there was documented experimental evidence of a binding site for transcription factor $j$ in the promoter region of gene $i$, we set $\pi_{ij} = 1$. We assigned values to the remaining elements of $\boldsymbol{\pi}$ using the same strategy as in Sabatti and James [52]. Briefly, we calibrated $\pi_{ij}$ on the basis of an analysis of the DNA sequence upstream of the studied genes. We used available information on the characteristics of the DNA sequence motif recognized by the TF to inform the sequence analysis, carried out with *Vocabulon* [53]. This algorithm is particularly well suited for this genomewide investigation, but other methodologies could also be applied. We hence identified all the putative binding sites for these transcription factors in the portion of the genome sequence that was likely to have a regulatory function. We categorized a location as a potential binding site if the Vocabulon algorithm assigned it a probability higher than 0.5. In this case we set the corresponding $\pi_{ij} = 0.5$. All remaining entries of $\boldsymbol{\pi}$ were set to zero.

Two qualifications are in order. First, resorting to Vocabulon and sequence analysis is only but one venue to gather knowledge on the network structure. In particular, it is worth noting that results from ChIP-Chip experiments are an important source of information that could be used for this purpose [7, 60]. Secondly, the degree of sparsity of the initial network can be substantially varied, as documented in Section 4.4.3. Indeed, one can use different thresholds to decide when a binding site is detected; moreover putative sites may have a varying degree of certainty that could be reflected in the choice of $\pi_{ij}$. However, we have found that using the value $\pi_{ij} = 0.5$ seemed to work well in our study.

## 4.3 Methodology

### 4.3.1 The Model

As anticipated in the introduction, we couple the bipartite network structure with the assumption of a linear relation between variations in the concentration of the active form of the TF and variations in the gene expression levels, obtaining the following model,

$$\boldsymbol{E} = \boldsymbol{A}\boldsymbol{P} + \boldsymbol{\epsilon},$$

where $\boldsymbol{E}$ is an $n \times T$ matrix of $e_{it}$'s, $\boldsymbol{A}$ is an $n \times L$ matrix of $a_{ij}$'s and $\boldsymbol{P}$ is an $L \times T$ matrix of $p_{jt}$'s. $\boldsymbol{A}$ and $\boldsymbol{P}$ are both unknown quantities. Respectively, $n$, $L$ and $T$ denote the number of genes, TFs and experiments, $a_{ij}$ represents the control strength of transcription factor $j$ on gene $i$, $p_{jt}$ the concentration of transcription factor $j$ in experiment $t$, and $\epsilon_{it}$ captures i.i.d. measurement errors and biological variability. As mentioned previously, this model is unidentifiable. However, we also have available an $n \times L$ matrix of $\pi_{ij}$'s which provide $P(a_{ij} \neq 0)$. This extra information, along with our penalized fitting procedure, ensure identifiable estimates for $\boldsymbol{A}$ and $\boldsymbol{P}$.

### 4.3.2 Fitting the Model

In Section 4.3.2 we begin by examining a simple application of the Lasso optimization approach to fit (4.1). Then, in Section 4.3.2 we extend this approach to provide our final fitting procedure.

#### A Preliminary Approach

A natural way to extend the Lasso procedure to estimate $\boldsymbol{A}$ and $\boldsymbol{P}$ is to minimize the penalized squared loss function:

$$\| \boldsymbol{E} - \boldsymbol{AP} \|_2^2 + \lambda_1 \| \boldsymbol{A} \|_1 + \lambda_2 \| \boldsymbol{P} \|_1 \tag{4.2}$$

where $\lambda_1$ and $\lambda_2$ are two tuning parameters and $\| \cdot \|_1$ is the sum of the absolute values of the given matrix. Note that $\| \cdot \|_2^2$ corresponds to the sum of squares of all components of the corresponding matrix with any missing values ignored. While this objective function appears to require the selection of two tuning parameters, (4.2) can be reformulated as

$$\| \boldsymbol{E} - \boldsymbol{A}^* \boldsymbol{P}^* \|_2^2 + \lambda_1 \lambda_2 \| \boldsymbol{A}^* \|_1 + \| \boldsymbol{P}^* \|_1$$

where $\boldsymbol{A}^* = \boldsymbol{A}/\lambda_2$ and $\boldsymbol{P}^* = \lambda_2 \boldsymbol{P}$. Hence, it is clear that a single tuning parameter suffices and $\boldsymbol{A}$ and $\boldsymbol{P}$ can be computed as the minimizers of

$$\| \boldsymbol{E} - \boldsymbol{AP} \|_2^2 + \lambda \| \boldsymbol{A} \|_1 + \| \boldsymbol{P} \|_1 \, . \tag{4.3}$$

Optimizing (4.3) for different values of $\lambda$ controls the level of sparsity of the estimates for $\boldsymbol{A}$ and $\boldsymbol{P}$.

A simple iterative algorithm can be used to solve (4.3). Namely:

- Step 1: Choose initial values for $\boldsymbol{A}$ and $\boldsymbol{P}$ denoted by $\boldsymbol{A}^{(0)}$ and $\boldsymbol{P}^{(0)}$. Let $k = 1$.
- Step 2: Fix $\boldsymbol{A} = \boldsymbol{A}^{(k-1)}$, find the $\boldsymbol{P} = \boldsymbol{P}^{(k)}$ minimizing $\| \boldsymbol{E} - \boldsymbol{A}^{(k-1)}\boldsymbol{P} \|_2^2 + \| \boldsymbol{P} \|_1$
- Step 3: Fix $\boldsymbol{P} = \boldsymbol{P}^{(k)}$, find the $\boldsymbol{A} = \boldsymbol{A}^{(k)}$ minimizing $\| \boldsymbol{E} - \boldsymbol{A}\boldsymbol{P}^{(k)} \|_2^2 + \lambda \| \boldsymbol{A} \|_1$
- Step 4: If $\| \boldsymbol{P}^{(k)} - \boldsymbol{P}^{(k-1)} \|$ or $\| \boldsymbol{A}^{(k)} - \boldsymbol{A}^{(k-1)} \|$ are large, let $k \leftarrow k + 1$ and return to Step 2.

Steps 2 and 3 in this algorithm can be easily achieved using a standard application of the LARS algorithm used for fitting the Lasso.

**Incorporating the Prior Information**

The fitting procedure outlined in the previous section is simple to implement and often quite effective. It can be utilized in situations where no prior information is available about the network structure because minimizing (4.3) is, *a priori*, equally likely to cause any particular element of $\boldsymbol{A}$ to be zero, or not to be zero.

However, in practice, for our data, we know that many elements of $\boldsymbol{A}$ must be zero, i.e. where $\pi_{ij} = 0$, and others can not be zero, i.e. where $\pi_{ij} = 1$. Of the remaining elements, some are highly likely to be zero while others are most likely non-zero, depending on their $\pi_{ij}$. Hence it is important that our fitting procedure directly takes the prior information into account. This limitation is removed by minimizing (4.4),

$$\| \boldsymbol{E} - \boldsymbol{A}\boldsymbol{P} \|_2^2 - \lambda_1 \sum_{ij} \log(\pi_{ij})|a_{ij}| + \lambda_2 \| \boldsymbol{A} \|_2^2 + \| \boldsymbol{P} \|_1 . \tag{4.4}$$

The key changes between (4.3) and (4.4) are the addition of $-\log(\pi_{ij})$ and a square of $L_2$-norm penalty on $\boldsymbol{A}$. The incorporation of the prior information has several effects on the fit. First, $a_{ij}$ is automatically set to zero if $\pi_{ij} = 0$. Second, $a_{ij}$ can not be set to zero if $\pi_{ij} = 1$. Finally, $a_{ij}$'s for which the corresponding $\pi_{ij}$ is small are likely to be set to zero while those for which $\pi_{ij}$ is large are unlikely to be set to zero. Optimizing (4.4) is achieved using a similar iterative approach to that used for (4.3).

- Step 1: Choose initial values for $\boldsymbol{A}$ and $\boldsymbol{P}$ denoted by $\boldsymbol{A}^{(0)}$ and $\boldsymbol{P}^{(0)}$. Let $k = 1$.
- Step 2: Fix $\boldsymbol{A} = \boldsymbol{A}^{(k-1)}$, find the $\boldsymbol{P} = \boldsymbol{P}^{(k)}$ minimizing $\| \boldsymbol{E} - \boldsymbol{A}^{(k-1)}\boldsymbol{P} \|_2^2 + \| \boldsymbol{P} \|_1$.
- Step 3: Fix $\boldsymbol{P} = \boldsymbol{P}^{(k)}$, find the $\boldsymbol{A} = \boldsymbol{A}^{(k)}$ minimizing $\| \boldsymbol{E} - \boldsymbol{A}\boldsymbol{P}^{(k)} \|_2^2$ $-\lambda_1 \sum_{ij} \log(\pi_{ij})|a_{ij}| + \lambda_2 \| \boldsymbol{A} \|_2$.
- Step 4: If $\| \boldsymbol{P}^{(k)} - \boldsymbol{P}^{(k-1)} \|$ or $\| \boldsymbol{A}^{(k)} - \boldsymbol{A}^{(k-1)} \|$ are large, let $k \leftarrow k + 1$ and return to Step 2.

Step 2 can be again be implemented using the LARS algorithm. Step 3 utilizes the shooting algorithm [22, 24].

Equation (4.4) treats all elements of $\boldsymbol{P}$ equally. However, in practice there is often a grouping structure in the experiments, or correspondingly the columns of $\boldsymbol{P}$. For example, in the *E. coli* data columns 1 through 12 of $\boldsymbol{P}$ correspond to the Tryptophan timecourse experiments while columns 13 through 19 represent the glucose acetate transition experiments. To examine any possible advantages from modeling these natural groupings we implemented a second fitting procedure. Let $\mathcal{G}_k$ be the index of the experiments in the $k$th group assuming all the experiments are divided into $K$ groups. Then our second approach involved minimizing,

$$\| \boldsymbol{E} - \boldsymbol{A}\boldsymbol{P} \|_2^2 - \lambda_1 \sum_{ij} \log(\pi_{ij})|a_{ij}| + \lambda_2 \| \boldsymbol{A} \|_2^2 + \| \boldsymbol{P} \|_2 \tag{4.5}$$

where $\| \boldsymbol{P} \|_2 = \sum_{j=1}^{L} \sum_{k=1}^{K} \sqrt{\sum_{t \in \mathcal{G}_k} p_{jt}^2}$. Replacing $\| \boldsymbol{P} \|_1$ with $\| \boldsymbol{P} \|_2$ has the effect of forcing the $p_{jt}$'s within the same group to either all be zero or all non-zero. In other words either all of the experiments or none of the experiments within a group are selected. Minimizing (4.5) uses the same algorithm as for (4.4) except that in Step 2 the shooting algorithm is used rather than LARS. We show results from both methods. To differentiate between the two approaches we call (4.4) the "ungrouped" method and (4.5) the "grouped" approach.

#### Normalizing the Estimators

The use of penalties on $\boldsymbol{A}$ and $\boldsymbol{P}$ allows us to produce unique estimates for the parameters up to an indeterminacy in the signs of $\boldsymbol{A}$ and $\boldsymbol{P}$ i.e. one can obtain identical results by flipping the sign on the $j$th column of $\boldsymbol{A}$ and the $j$th row of $\boldsymbol{P}$. There are a number of potential approaches to deal with the sign. Sabatti and James [52] defined two new quantities that are independent from rescaling and changes of signs and have interesting biological interpretations:

$$\tilde{p}_{jt} = \frac{\sum_i a_{ij} p_{jt}}{\sum_i 1(a_{ij} \neq 0)} \quad \text{and} \quad \tilde{a}_{ij} = \frac{\sum_t a_{ij} p_{jt}}{T}$$

$\tilde{p}_{jt}$ is the average effect of each transcription factor on the genes it regulates (regulon expression), and $\tilde{a}_{ij}$ is the average control strength over all experiments. These quantities are directly related to the expression values of genes in a regulon. We have opted to use $\tilde{p}_{jt}$ and $\tilde{a}_{ij}$ to report our results. This also has the advantage of allowing easy comparison with the analysis of Sabatti and James [52].

## 4.4 Case Study

In this section we give a detailed examination of the results from applying the grouped and ungrouped methods to the *E. coli* data. Section 4.4.1 outlines the construction of our initial network structure while Section 4.4.2 discusses our procedure for choosing the tuning parameters. The main results are provided in Section 4.4.3. Finally, Section 4.4.4 provides the results from a sensitivity analysis performed by adjusting the sparsity level on the initial network structure.

### 4.4.1 The Initial Network Structure

The first step in constructing the Transcription Regulation Network is to develop an initial guess for $\boldsymbol{\pi}$ i.e. the probability distribution of the network structure. As discussed in Section 4.2, $\boldsymbol{\pi}$ was computed using various sources. Where there was experimental evidence of a link between transcription factor $j$ and gene $i$ we set $\pi_{ij} = 1$. For the remaining elements we used the *Vocabulon* [53] algorithm to estimate $\pi_{ij}$. Initially we set $\pi_{ij} = 0.5$ for any probability estimated to be at least 0.5. For all other transcription factor-gene combinations we set $\pi_{ij} = 0$. This was the approach taken in Sabatti and James [52] and allowed us to directly compare the two sets of results. With the Bayesian approach of Sabatti and James [52] this high level of sparsity in the network structure was necessary for computational reasons. However, using our Lasso based methodology this level of sparsity is not required. Hence, in Section 4.4.4 we examine how our results change as we reduce the level of sparsity in the initial structure.

By merging the potential binding sites with the known sites from the literature, and with the expression data, we obtained a set of 1433 genes, potentially regulated by at least one of 37 transcription factors and on which expression measurements were available (missing values in the array data were allowed). Our estimate for $\boldsymbol{\pi}$ suggested a great deal of sparsity with only 2073 non-zero entries, 291 of which corresponded to $\pi_{ij} = 1$ and the remaining 1782 to $\pi_{ij} = 0.5$. Figure 4.3(a) shows the distribution of the number of genes thought to be regulated by a singe transcription factor in our initial network. The figure shows that 14 of the transcription factors

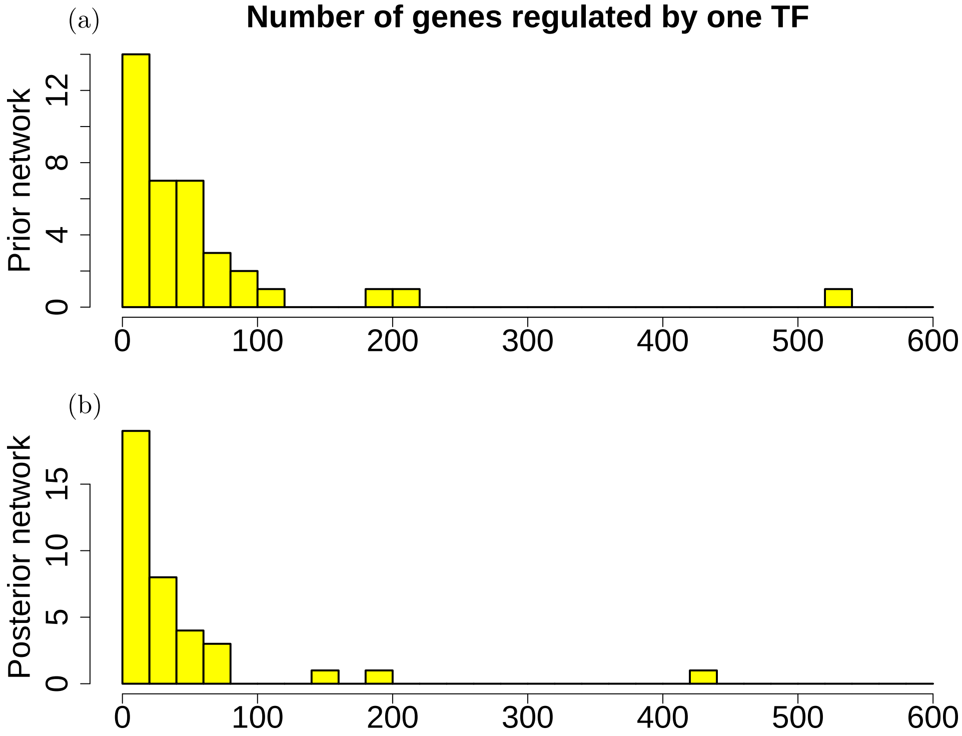


Figure 4.3: Histograms for the number of genes that are regulated by each TF. (a) Prior network. (b) Posterior network after using the ungrouped method.

were only expected to regulate 20 or fewer genes and 34 of the 37 TFs were expected to regulate at most 120 genes. The notable exception was CRP, which potentially regulated over 500 genes. It is worth noting that without adopting our penalized regression framework, we would not be able to study this transcription network, simply because the number of experiments (35) is smaller than the number of TFs considered (37): the use of penalty terms regularizes the problem.

### 4.4.2 Selecting the Tuning Parameters

The first step in estimating $\boldsymbol{A}$ and $\boldsymbol{P}$ requires the selection of the tuning parameters, $\lambda_1$ and $\lambda_2$. These could be chosen subjectively but we experimented with several more objective automated approaches. We first attempted to select the tuning parameters corresponding to the lowest values of BIC or AIC. However, BIC produced models that were biologically too sparse i.e. the number of zero entries in $\boldsymbol{A}$ was too large. It appears that the $\log(n)$ factor used by BIC is too large if one uses the number of non-missing values in the $\boldsymbol{E}$ matrix as "$n$" ($n = 40,000$) because they are not really independent. Conversely, AIC resulted in networks being selected that had too many connections.

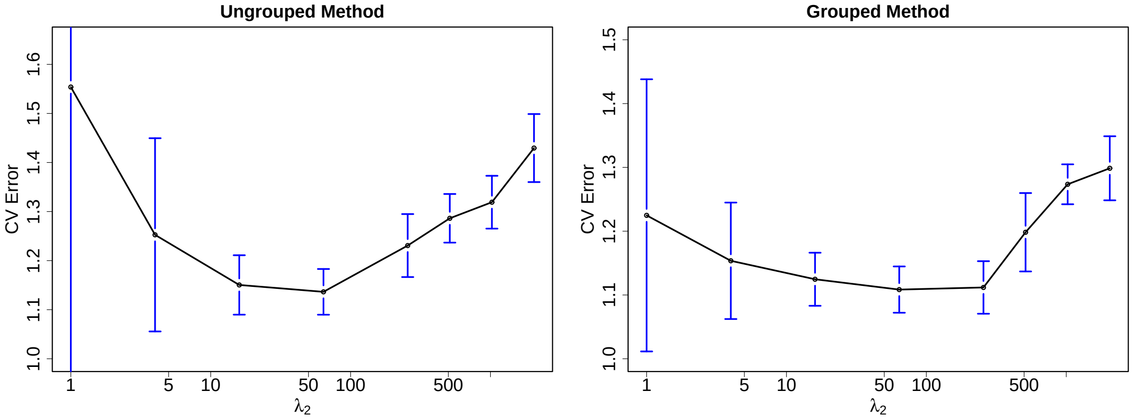


Figure 4.4: Cross validated error rates as a function of $\lambda_2$ for the ungrouped and grouped methods. The vertical lines indicate variability in the cross validated error.

Instead we opted to use a two stage approach. We first computed the cross validated error over a grid of $\lambda_1$'s and $\lambda_2$'s and selected the tuning parameters corresponding to the minimum. Figure 4.4 show the cross validated error rates for different values of $\lambda_2$ with $\lambda_1 = 64$. For both the ungrouped and grouped methods the minimum was achieved with $\lambda_1 = \lambda_2 = 64$. Second, we used the bootstrap to determine whether there was significant evidence that an element in $\boldsymbol{A}$ was non-zero. We ran our method on 100 bootstrap samples. For each element of $\boldsymbol{A}$, we computed a corresponding p-value based on the 100 bootstrap results, thus we had approximately 2000 p-values. Since this constituted a significant multiple testing problem we used False Discovery Rate (FDR) methods to set a cutoff such that the FDR was no more than 0.05. Elements in $\boldsymbol{A}$ with p-values smaller than the cutoff were left as is while the remainder were set to zero. All the results that follow are based on this bootstrap analysis.

### 4.4.3 Results

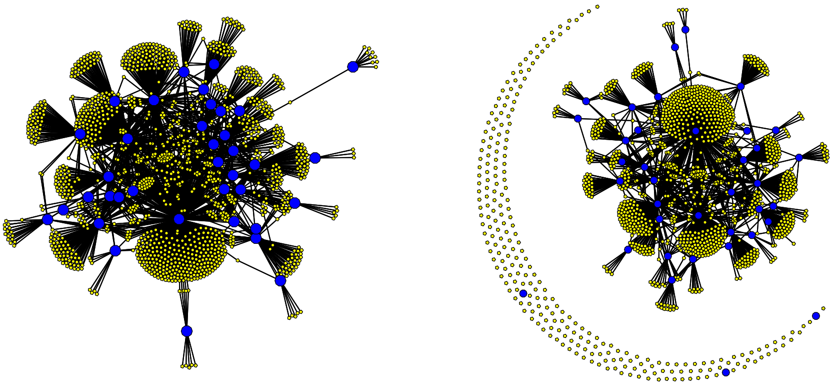


Figure 4.5: Prior network (left) and posterior estimate produced using the ungrouped method (right). The large circles correspond to the 37 transcription factors while the small circles represent the 1433 genes. The lines joining large and small circles indicate network connections.

The results from our analysis of the 35 experiments suggested that a significant portion of the potential binding sites should be discarded. Figure 4.3(b) shows the distribution of the number of genes regulated by a singe transcription factor in our final network structure. We see that 19 TFs are expected to regulate 20 or fewer genes and 30 of the 37 TFs were expected to regulate at most 50 genes. Even CRP, went from over 500 potential binding sites in the prior to approximately 400 in the posterior. The posterior estimate for $\boldsymbol{A}$ contained 1586 non-zero entries, approximately a 25% reduction in the number of connections in our prior guess for the network. Figure 4.5 provides graphical representations for the prior and posterior networks. Note that in the posterior estimate there are many fewer connections and as a result there are numerous genes and three TFs that are no longer connected to the rest of the network, suggesting there is no evidence that these particular genes are regulated by any of the 37 TFs we examined.

Sabatti and James [52] discuss several possible reasons for the changes between the initial and final network structure. In brief, Vocabulon works entirely using the sequence information. Hence, it is quite possible for a portion of the *E. coli* genome sequence to look just like a binding site for a TF, resulting in a high probability as estimated by Vocabulon, when in reality it is not used by the protein in question. In addition, Vocabulon searches for binding sites in the regulatory region of each gene by inspecting 600 base pairs upstream of the start codon which often causes Vocabulon to investigate the same region for multiple genes. If a binding site is located in such a sequence portion, it will be recorded for all of the genes whose "transcription region" covers it.

Figure 4.6 illustrates the estimated transcription factor activation levels using both the ungrouped and grouped methods. We have several ways to validate these results. First, we note that the estimated activation levels show very strong similarities to the results of Sabatti and James [52]. Both their results and ours show the following characteristics. First, there are a number of transcription factors that are not activated in any of the experiments. Focusing on the regulons that are activated in some of the experiments, we note that our method produces results that correspond to the underlying biology. For example, the first 8 experiments [32]—represented in the lower portion of the displays from bottom up—are two 4-point time courses of tryptophan starvation. The absence of tryptophan induces the derepression of the genes regulated by trpR. Correspondingly, our results indicate a clear increase in expression for trpR. In arrays 9-12, the cells were provided with extra tryptophan. Hence, for these experiments we would expect lowered expression. Our results show a negative effect, though the magnitude is small. Additionally, the argR and fliA regulons can be seen to move in the opposite direction to trpR which

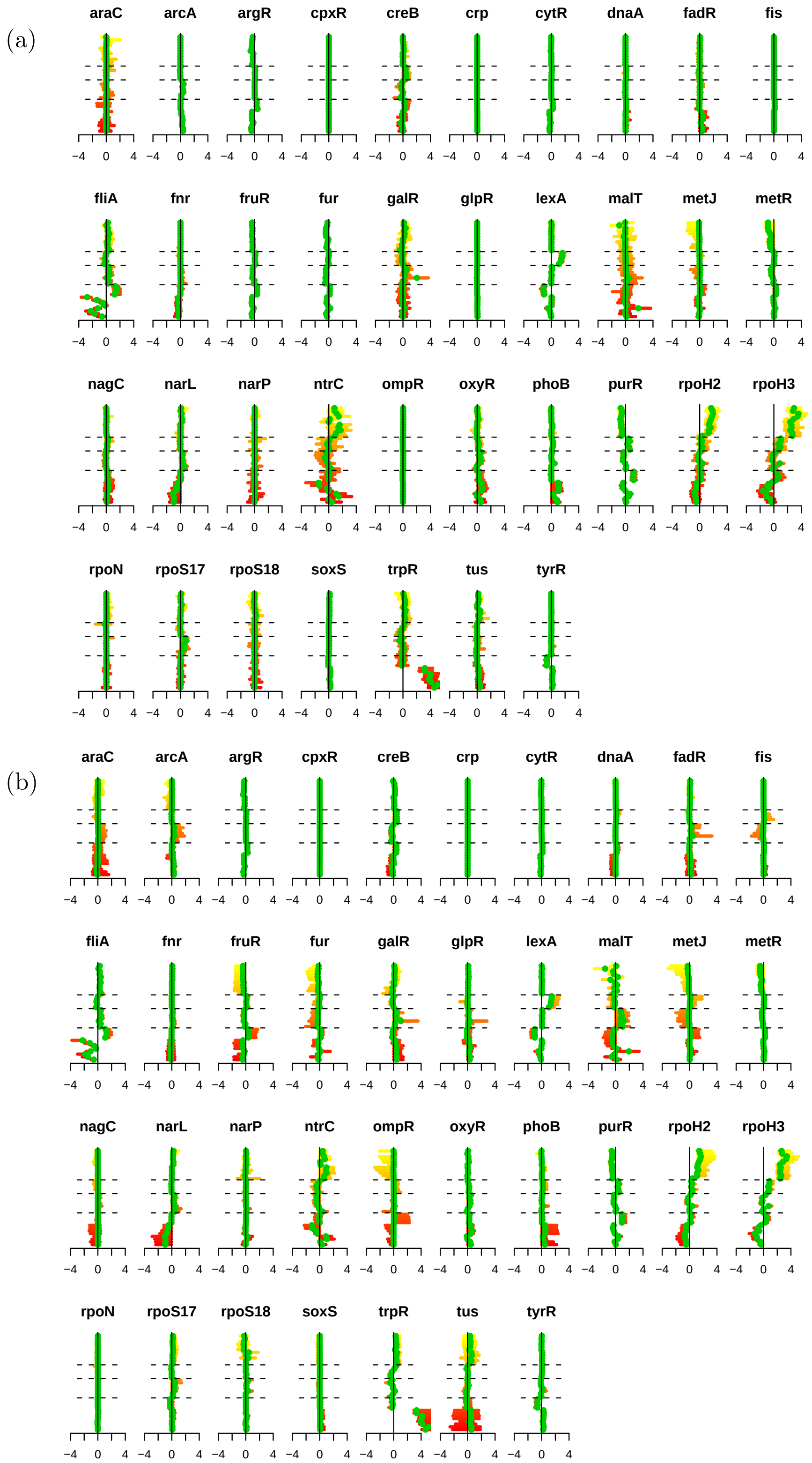


Figure 4.6: (a) Ungrouped and (b) grouped methods. Each plot corresponds to the experiments for one transcription factor. Experiments are organized along the vertical axis, from bottom to top, with dashed lines separating the experiment groups. Green dots indicate the estimates for $\tilde{p}_{jt}$ and the horizontal bars provide bootstrap confidence intervals.

corresponds to what has been documented in the literature [32]. Figure 4.6 also suggests that the rpoH2, rpoH3 and narL regulons are all moving in the opposite direction to trpR.

Experiments 20-24, which correspond to the results between the second and third horizontal dashed lines, are a comparison of wild type *E. coli* cells with cells that were irradiated with ultraviolet light, which results in DNA damage. Note that lexA shows a high expression level in these experiments, as one would predict since many of the DNA damaged-genes are known to be regularly repressed by lexA [15]. Finally, metR, ntrC, purR, rpoH2, and rpoH3 all show strong activations in the protein overexpression data, the final 11 experiments. In particular, notice that rpoH2 and rpoH3 present the same profile across all experiments. This provides further validation of our procedure since these two really represent the same protein, and are listed separately because they correspond to two different types of binding sites of the TF. Overall, these results conform to the known biology, but also suggest some additional areas for exploration.

The main differences between our results and those of Sabatti and James [52] are that our penalties on $\boldsymbol{P}$ tend to generate more exact zero estimates than the Bayesian approach, providing somewhat easier interpretation. The grouped and ungrouped results are also similar, but the grouping structure is more prominent in the grouped method, for example, in metJ and ompR.

Next, we examine the estimates for $\boldsymbol{A}$. Since a number of TFs showed no activation in these experiments we would not expect to be able to accurately estimate their control strengths on the genes. Hence, we will concentrate our analysis here on trpR because this was the most strongly activated TF. Figure 4.7 presents our estimates of $\tilde{a}$ for seven genes associated with the trpR. Each boxplot illustrates the 100 bootstrap

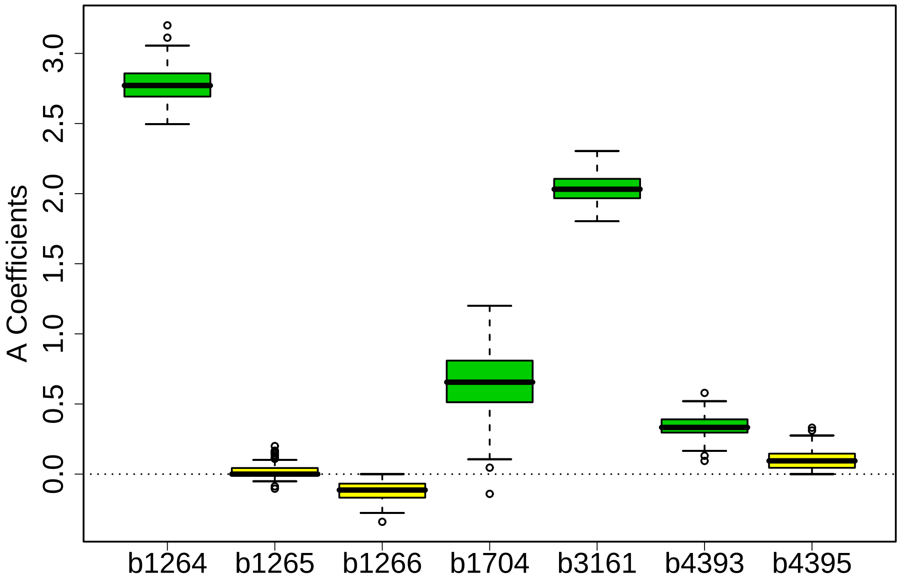


Figure 4.7: Boxplots of the bootstrap estimates for $\tilde{a}$ for seven different genes.

estimates of $\tilde{a}$ for a particular gene. The first three boxplots correspond to genes b1264, b1265, b1266. The b-numbers, that identify the genes, roughly correspond to their genomic location, so it is clear that the genes are adjacent to each other. Gene b1264 is known to be regulated by trpR, so it's $\pi_{ij}$ was set to 1. The other two genes were chosen by Vocabulon as potential candidates because the binding site for b1264 was also in the search regions for b1265 and b1266 i.e. these were cases of the overlapping regulatory regions described previously. While Vocabulon was unable to determine whether a connection existed between b1265, b1266 and trpR, using our approach we can see that, while $\tilde{a}$ for b1264 is large, the estimates for b1265 and b1266 are essentially zero. Thus it is possible to use our model to rule out the regulation of two genes by trpR that are within a reasonable distance from a trpR real binding site. Among the remaining four genes b1704, b3161 and b4393 are all known to be regulated by trpR. Correspondingly, they all have moderate to large estimated activation strengths. b4395 again has an overlapping regulatory region to b4393. The results suggest this is not regulated by trpR.

### 4.4.4 Relaxing Zero Coefficients

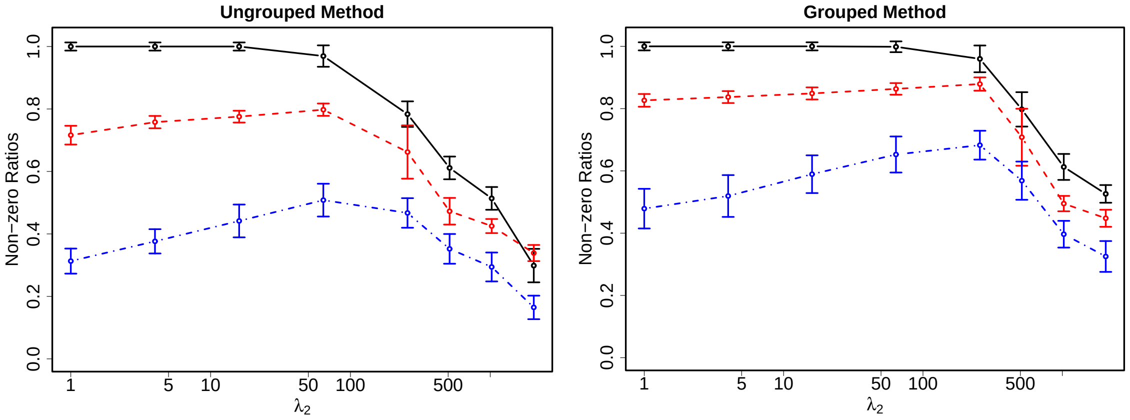


Figure 4.8: Fraction of non-zero $\tilde{a}_{ij}$'s as a function of $\lambda_2$ for the ungrouped and grouped methods. The solid line corresponds to those connections where there was documented evidence of a relationship, the dashed line to where the Vocabulon algorithm suggested there was a relationship and the dash-dot line to where there was no evidence of a relationship.

The results from Section 4.4.3 use the same relatively sparse initial network structure as that of Sabatti and James [52]. Recall, the structure we have assumed so far contained only three possible values for $\boldsymbol{\pi}$ i.e. $\pi_{ij} = 0$, $\pi_{ij} = 0.5$ or $\pi_{ij} = 1$. All connections with $\pi_{ij} = 0$ are forced to remain at zero whatever the gene expression data may suggest. However, as discussed previously, our methodology is able to handle far less sparse structures. Hence, we next investigated the sensitivity of our results to the initial structure by randomly adjusting certain TF-gene connections. In particular we randomly selected 200 of the connections where $\pi_{ij} = 0$ and reset them to $\pi_{ij} = 0.5$. We also reset all connections where $\pi_{ij} = 1$ to $\pi_{ij} = 0.5$ so that all connections were treated equivalently. We then reran the ungrouped and grouped methods using the new values for $\boldsymbol{\pi}$.

Figure 4.8 provides plots of the resulting fractions of non-zero estimates for $\tilde{a}_{ij}$, as a function of $\lambda_2$ with $\lambda_1$ set to 64. A clear pattern emerges with the fraction of non-zero's where there was documented evidence very high (solid line). Somewhat lower is the fraction of non-zero's for the connections suggested by Vocabulon (dashed line). Finally, the lowest level of non-zero's is exhibited where there was no significant evidence of a connection (dash-dot line). These results are comforting because they suggest that our methodology is able to differentiate between the clear, possible and unlikely connections even when $\pi_{ij}$ is equal for all three groups. In addition, there appears to be evidence that the Vocabulon algorithm is doing a good job of separating potential from unlikely connections. Finally, these results illustrate that, unlike the Bayesian approach, it is quite computationally feasible for out methodology to work on relatively dense initial network structures.

## 4.5 Simulation Study

After fitting the *E. coli* data we conducted a simulation study to assess how well our methodology could be expected to reconstruct transcription regulation networks with characteristics similar to those for our data set. We used the estimated matrices, $\hat{\boldsymbol{A}}$ and $\hat{\boldsymbol{P}}$, from Section 4.4 as the starting point for generating the gene expression levels. In particular we first let $\tilde{\boldsymbol{A}} = \hat{\boldsymbol{A}} + \boldsymbol{\varepsilon}_A$, $\tilde{\boldsymbol{P}} = \hat{\boldsymbol{P}} + \boldsymbol{\varepsilon}_P$, where $\varepsilon_{Aij} \sim s_A \times N(0, \sigma^2(\hat{\boldsymbol{A}}))$ and $\varepsilon_{Pij} \sim s_P \times N(0, \sigma^2(\hat{\boldsymbol{P}}_i))$ are noise terms. Depending on the simulation run, $s_A$ was set to either 0.2 or 0.4 while $s_P$ was set to either 0.1 or 0.3. Next, all elements of $\tilde{\boldsymbol{A}}$ corresponding to $\pi_{ij} = 0$ were set to zero. In addition, among elements were $\pi_{ij} = 0.5$, we randomly set $\rho$ of the $\tilde{\boldsymbol{A}}'s$ to zero where $\rho$ was set to either 60% or 80%. The expression levels were then generated using

$$\boldsymbol{E} = \tilde{\boldsymbol{A}}\tilde{\boldsymbol{P}} + s_N \times \tilde{\boldsymbol{\Gamma}},$$

where $\tilde{\mathbf{\Gamma}}$ is a matrix of error term with $\tilde{\mathbf{\Gamma}}_{ij} \sim N(0,1)$ and $s_N$ was set to either 0.2 or 0.4. We produced one simulation run for each combination of $s_A, s_P, \rho$, and $s_N$, resulting in a total of 16 simulations.

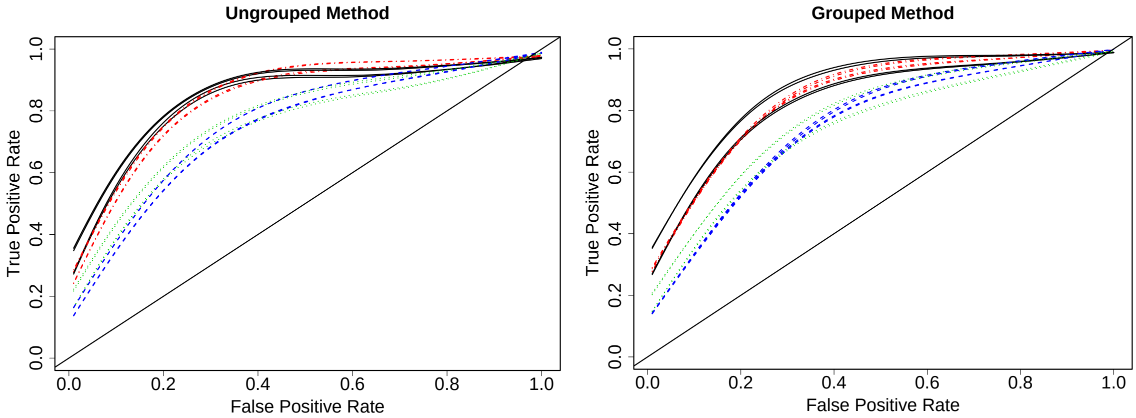


Figure 4.9: False Positive Rates versus True Positive Rates for 16 simulations. Black solid lines correspond to $s_N = 0.2, \rho = 0.8$, dash-dot lines to $s_N = 0.2, \rho = 0.6$, green dotted lines to $s_N = 0.4, \rho = 0.8$ and dashed lines to $s_N = 0.4, \rho = 0.6$. Different values of $s_A$ and $s_P$ had little effect on the results so they have not been individually identified.

For each simulation run we computed the False Positive Rate (FPR) and the True Positive Rate (TPR) for different possible tuning parameters. The FPR is defined as the fraction of estimated non-zero coefficients, $a_{ij}$, among all elements of $\tilde{\boldsymbol{A}}$ where $\tilde{a}_{ij} = 0$ and $\pi_{ij} = 0.5$. The TPR is defined as the fraction of estimated non-zero coefficients, $a_{ij}$, among all elements of $\tilde{\boldsymbol{A}}$ where $\tilde{a}_{ij} \neq 0$ and $\pi_{ij} = 0.5$. Figure 4.9 provides a summary of the results from running the ungrouped and grouped approaches on the sixteen simulations. Each curve corresponds to the FPR vs TPR for one simulation run using different tuning parameters. The results suggest that a reasonable level of accuracy can be produced for this data. For example, with $s_N = 0.2$ both methods can achieve an 80% TPR at the expense of a 20% FPR. To lower the FPR to 10% decreases the TPR to approximately 60%. Even with $s_N = 0.4$, a relatively high level, we can achieve a 60% TPR at the expense of a 20% FPR.

## 4.6 Discussion

We have introduced a new methodology for estimating the parameters of model (4.1) associated with a bipartite network, as illustrated in Figure 4.1. Our approach is based on introducing $L_1$-norm penalties to the regression framework, and using prior information about the network structure.

We have focused on the application of this model to reconstruction of *E. coli* transcription network, as this allows easy comparison with previously proposed models. Our approach has the advantage, over the work of Liao et al. [39] and Sabatti and James [52], that it does not require assuming prior knowledge of a large fraction of the network. When we utilize the same prior structure as used by Sabatti and James [52] we get similar, and biologically sensible, results. However, by relaxing the prior assumptions on the sparsity of the network structure we gain additional insights such as independent validation both of the experimentally derived network connections and also the connections suggested by the *Vocabulon* algorithm.

While we tested our methodology on the *E. coli* data, our approach is potentially applicable to many other organisms. In particular there are many organisms for which far less of the TRN structure is known *a priori*, making it impossible to use the algorithms by Liao et al. [39] and Sabatti and James [52]. In these cases our $L_1$-penalization approach could still be applied, allowing researchers to start to explore the transcription network of these organisms.

Finally, it is worth recalling that, while we describe how to set the $\boldsymbol{\pi}$ values with specific reference to TRN, the $L_1$-penalized regression approach, can be used to estimate parameters of bipartite networks arising in other scientific context.

# BIBLIOGRAPHY

# BIBLIOGRAPHY

[1] O. Alter, P. Brown, and D. Botstein. Singular value decomposition for genome-wide expression data processing and modeling. *Proceedings of the National Academy of Sciences, USA*, 97: 10101–10106, 2000.

[2] A. Antoniadis and J. Fan. Regularization of wavelet approximations. *Journal of the American Statistical Association*, 96:939–967, 2001.

[3] A. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[4] A. L. Barabasi and Z. N. Oltvai. Network biology: understanding the cells functional organization. *Nature Reviews Genetics*, 5:101–113, 2004.

[5] M. Beal, F. Falciani, Z. Ghahramani, C. Rangel, and D. Wild. A bayesian approach to reconstructing genetic regulatory networks with hidden factors. *Bioinformatics*, 21:349–356, 2005.

[6] P. Bickel and E. Levina. Regularized estimation of large covariance matrices. *Annals of Statistics*, 36:199–227, 2008.

[7] A. Boulesteix and K. Strimmer. Predicting transcription factor activities from combined analysis of microarray and chip data: a partial least squares approach. *Theoretical Biology and Medical Modelling*, 2:23, 2005.

[8] L. Breiman. Better subset regression using the nonnegative garrote. *Technometrics*, 37:373–384, 1995.

[9] L. Breiman and J. Friedman. Predicting multivariate responses in multiple linear regression (with discussion). *Journal of the Royal Statistical Society, Series B*, 59(1):3–54, 1997.

[10] M. Brynildsen, L. Tran, and J. Liao. A gibbs sampler for the identification of gene expression and network connectivity consistency. *Bioinformatics*, 22:3040–3046, 2006.

[11] P. Buhlmann. Boosting for high-dimensional linear models. *Annals of Statistics*, 34:559–583, 2006.

[12] E. Candes and T. Tao. The dantzig selector: Statistical estimation when p is much larger than n (with discussion). *Annals of Statistics*, 35(6):2313–2351, 2007.

[13] C. Chang, Z. Ding, Y. Hung, and P. Fung. Fast network component analysis (fastnca) for gene regulatory network reconstruction from microarray data. *Bioinformatics*, 24:1349–1358, 2008.

[14] H. Y. Chang, D. S. A. Nuyten, J. B. Sneddon, T. Hastie, R. Tibshirani, T. Sorlie, H. Dai, Y. He, L. van't Veer, H. Bartelink, and et al. Robustness, scalability, and integration of a wound response gene expression signature in predicting survival of human breast cancer patients. *Proceedings of the National Academy of Sciences*, 102(10):3738–3743, 2005.

[15] J. Courcelle, A. Khodursky, B. Peter, P. O. Brown, and P. C. Hanawalt. Comparative gene expression profiles following uv exposure in wild-type and sos-deficient escherichia coli. *Genetics*, 158:41–64, 2001.

[16] A. Dempster. Covariance selection. *Biometrics*, 28:157–175, 1972.

[17] D. Edward. Introduction to graphical modelling. *New York: Springer*, (2nd edition.), 2000.

[18] B. Efron and R. Tibshirani. On testing the significance of sets of genes. *The Annals of Applied Statistics*, 1:107–129, 2007.

[19] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *The Annals of Statistics*, 32:407–499, 2004.

[20] J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96:1348–1360, 2001.

[21] J. Fan and H. Peng. Nonconcave penalized likelihood with a diverging number of parameters. *Journal of the American Statistical Association*, 32(3):928–961, 2004.

[22] J. Friedman, T. Hastie, H. Hofling, and R. Tibshirani. Pathwise coordinate optimization. *Annals of Applied Statistics*, 1(2):302–332, 2007.

[23] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

[24] W. Fu. Penalized regressions: the bridge vs the lasso. *Journal of Computational and Graphical Statistics*, 7(3):397–416, 1998.

[25] T. S. Gardner, D. di Bernardo, D. Lorenz, and J. J. Collins. Inferring genetic networks and identifying compound mode of action via expression profiling. *Science*, 301:102–105, 2003.

[26] A. Genkin, D. D. Lewis, and D. Madigan. Large-scale bayesian logistic regression for text categorization. *Technometrics*, 49:291–304, 2007.

[27] E. George and D. Foster. Calibration and empirical bayes variable selection. *Biometrika*, 87: 731–747, 2000.

[28] E. George and R. McCulloch. Variable selection via gibbs sampling. *Journal of the American Statistical Association*, 88:881–889, 1993.

[29] J. Huang, N. Liu, M. Pourahmadi, and L. Liu. Covariance matrix selection and estimation via penalised normal likelihood. *Biometrika*, 93:85–98, 2007.

[30] G. M. James and P. Radchenko. A generalized dantzig selector with shrinkage tuning. *Biometrika*, To appear, 2009.

[31] H. Jeong, S. P. Mason, A. L. Barabasi, and Z. N. Oltvai. Lethality and centrality in protein networks. *Nature*, 411:41–42, 2001.

[32] A. B. Khodursky, B. J. Peter, N. R. Cozzarelli, D. Botstein, P. O. Brown, and C. Yanofsky. Dna microarray analysis of gene expression in response to physiological and genetic changes that affect tryptophan metabolism in escherichia coli. *Proceedings of the National Academy of Sciences, USA*, 97:12170–5, 2000.

[33] S. Lee and S. Batzoglou. Application of independent component analysis to microarrays. *Genome Biology*, 4(11):76, 2003.

[34] T. I. Lee, N. J. Rinaldi, F. Robert, D. T. Odom, Z. Bar-Joseph, G. Gerber, N. Hannett, C. T. Harbison, C. M. Thompson, I. Simon, J. Zeitlinger, E. G. Jennings, H. L. Murray, D. B. Gordon, B. Ren, J. J. Wyrick, J.-B. Tagne, T. L. Volkert, E. Fraenkel, D. K. Gifford, and R. A. Young. Transcriptional regulatory networks in saccharomyces cerevisiae. *Science*, 298: 799–804, 2002.

[35] E. Lehmann and G. Casella. *Theory of Point Estimation.* Springer-Verlag, New York, 2nd edition, 1998.

[36] E. Levina, A. J. Rothman, and J. Zhu. Sparse estimation of large covariance matrices via a nested Lasso penalty. *Biometrika*, 90:831–844, 2006.

[37] H. Li and J. Gui. Gradient directed regularization for sparse gaussian concentration graphs, with applications to inference of genetic networks. *Biostatitics*, 7(2):302–317, 2006.

[38] Z. Li, S. Shaw, M. Yedwabnick, and C. Chan. Using a state-space model with hidden variables to infer transcription factor activities. *Bioinformatics*, 22:747–754, 2006.

[39] J. C. Liao, R. Boscolo, Y. Yang, L. Tran, C. Sabatti, and V. Roychowdhury. Network component analysis: reconstruction of regulatory signals in biological systems. *Proceedings of the National Academy of Sciences, USA*, 100:15522–15527, 2003.

[40] Y. Lin and H. Zhang. Component selection and smoothing in smoothing spline analysis of variance models. *The Annals of Statistics*, 34(5):2272–2297, 2006.

[41] M. Matsuda, K. Miyagawa, M. Takahashi, T. Fukuda, T. Kataoka, T. Asahara, H. Inui, M. Watatani, M. Yasutomi, N. Kamada, K. Dohi, and K. Kamiya. Mutations in the rad54 recombination gene in primary cancers. *Oncogene*, 18(22):3427–3430, 1999.

[42] N. Meinshausen. Relaxed lasso. *Computational Statistics and Data Analysis*, 52:374–393, 2007.

[43] N. Meinshausen and P. Buhlmann. Consistent neighbourhood selection for high-dimensional graphs with the lasso. *Annals of Statistics*, 34:1436–1462, 2006.

[44] H. Nakshatri and S. Badve. Foxa1 as a therapeutic target for breast cancer. *Expert Opinion on Therapeutic Targets*, 11(4):507–514, 2007.

[45] M. Newman. The structure and function of complex networks. *Society for Industrial and Applied Mathematics*, 45(2):167–256, 2003.

[46] M. K. Oh and J. C. Liao. Gene expression profiling by dna microarrays and metabolic fluxes in escherichia coli. *Biotechnol. Prog.*, 16:278–286, 2000.

[47] M. K. Oh and J. C. Liao. Dna microarray detection of metabolic responses to protein overproduction in escherichia coli. *Metabolic Engineering*, 2:201–209, 2000.

[48] M. K. Oh, L. Rohlin, and J. C. Liao. Global expression profiling of acetate-grown escherichia coli. *Journal of Biological Chemistry*, 277:13175–13183, 2002.

[49] M. Olivier, R. Eeles, M. Hollstein, M. A. Khan, C. C. Harris, and P. Hainaut. The iarc tp53 database: New online mutation analysis and recommendations to users. *Human Mutation*, 19: 607–614, 2002.

[50] P. Radchenko and G. M. James. Variable inclusion and shrinkage algorithms. *Journal of the American Statistical Association*, 103:1304–1315, 2008.

[51] A. J. Rothman, P. J. Bickel, E. Levina, and J. Zhu. Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics*, 2:494–515, 2008.

[52] C. Sabatti and G. James. Bayesian sparse hidden components analysis for transcription regulation networks. *Bioinformatics*, 22:737–744, 2006.

[53] C. Sabatti and K. Lange. Genomewisemotif identification using a dictionary model. *IEEE Proceedings*, 90:1803–1810, 2002.

[54] G. Sanguinetti, N. Lawrence, and M. Rattray. Probabilistic inference of transcription factor concentrations and gene-specific regulatory activities. *Bioinformatics*, 22:2775–2781, 2006.

[55] J. Schafer and K. Strimmer. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, 4(1):Article 32, 2007.

[56] J. Schafer and K. Strimmer. An empirical bayes approach to inferring large-scale gene association networks. *Bioinformatics*, 21:754–764, 2005.

[57] X. Shen and J. Ye. Adaptive model selection. *Journal of the American Statistical Association*, 97:210–221, 2002.

[58] A. Shimo, C. Tanikawa, T. Nishidate, M. Lin, K. Matsuda, J. Park, T. Ueki, T. Ohta, K. Hirata, M. Fukuda, Y. Nakamura, and T. Katagiri. Involvement of kinesin family member 2c/mitotic centromere-associated kinesin overexpression in mammary carcinogenesis. *Cancer Science*, 99 (1):62–70, 2007.

[59] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, and J. P. Mesirov. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences, USA*, 102:15545–15550, 2005.

[60] N. Sun, R. Carroll, and H. Zhao. Bayesian error analysis model for reconstructing transcriptional regulatory networks. *Proceedings of the National Academy of Sciences, USA*, 103: 7988–7993, 2006.

[61] S. Tchatchou, M. Wirtenberger, K. Hemminki, C. Sutter, A. Meindl, B. Wappenschmidt, M. Kiechle, P. Bugert, R. Schmutzler, C. Bartram, and B. Burwinkel. Aurora kinases a and b and familial breast cancer risk. *Cancer Letters*, 247(2):266–272, 2007.

[62] J. Tegner, M. K. Yeung, J. Hasty, and J. J. Collins. Reverse engineering gene networks: integrating genetic perturbations with dynamical modeling. *Proceedings of the National Academy of Sciences USA*, 100:5944–5949, 2003.

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

[64] L. Tran, M. Brynildsen, K. Kao, J. Suen, and J. Liao. gnca: a framework for determining transcription factor activity based on transcriptome: identifiability and numerical implementation. *Metabolic Engineering*, 7:128–141, 2005.

[65] B. Turlach, W. Venables, and S. Wright. Simultaneous variable selection. *Technometrics*, 47 (3):349–363, 2005.

[66] M. J. van de Vijver, Y. D. He, L. J. van 't Veer, H. Dai, A. A. Hart, D. W. Voskuil, G. J. Schreiber, J. L. Peterse, C. Roberts, M. J. Marton, and et al. A gene-expression signature as a predictor of survival in breast cancer. *New England Journal of Medicine*, 347:1999–2009, 2002.

[67] H. Wang, G. Li, and C. L. Tsai. Regression coefficient and autoregressive order shrinkage and selection via lasso. *Journal of the Royal Statistical Society, Series B*, 69:63–78, 2006.

[68] J. Whittaker. Graphical models in applied mathematical multivariate statistics. *Wiley*, 1990.

[69] W. B. Wu and M. Pourahmadi. Nonparametric estimation of large covariance matrices of longitudinal data. *Annals of Applied Statistics*, 2:245–263, 2003.

[70] T. Yu and K. Li. Inference of transcriptional regulatory network by two-stage constrained space factor analysis. *Bioinformatics*, 21:4033–4038, 2005.

[71] M. Yuan and Y. Lin. Model selection and estimation in the gaussian graphical model. *Biometrika*, 94(1):19–35, 2007.

[72] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variable. *Journal of the Royal Statistical Society, Series B.*, 68(1):49–67, 2006.

[73] H. Zhang and W. Lu. Adaptive-lasso for cox's proportional hazards model. *Biometrika*, 94(3): 691–703, 2007.

[74] P. Zhao and B. Yu. On model selection consistency of lasso. *Journal of Machine Learning Research*, 7:2541–2567, 2006.

[75] P. Zhao, G. Rocha, and B. Yu. Grouped and hierarchical model selection through composite absolute penalties. *The Annals of Statistics*, in press.

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

[77] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 67(2):301–320, 2005.

[78] H. Zou, T. Hasite, and R. Tibshirani. On the degrees of freedom of the lasso. *Annals of Statistics*, 35:2173–2192, 2007.