Structured Sparsity and Precision Matrix Estimation

Cody A. Mazza-Anthony

Department of Electrical & Computer Engineering

McGill University

Montreal, Quebec, Canada

December 8, 2019

A thesis submitted to McGill University in partial fulfillment of the requirements of the degree of Masters of Engineering

Abstract

We address the task of identifying densely connected subsets of multivariate Gaussian random variables within a graphical model framework. We propose two novel estimators based on the Ordered Weighted ℓ_1 (OWL) norm: 1) The Graphical OWL (GOWL) is a penalized likelihood method that applies the OWL norm to the lower triangle components of the precision matrix. 2) The column-by-column Graphical OWL (ccGOWL) estimates the precision matrix by performing OWL regularized linear regressions. Both methods can simultaneously identify highly correlated groups of variables and control the sparsity in the resulting precision matrix. We formulate GOWL such that it solves a composite optimization problem and establish that the estimator has a unique global solution. In addition, we prove sufficient grouping conditions for each column of the ccGOWL precision matrix estimate. For synthetic data where group structure is present, the ccGOWL estimator requires significantly reduced computation and achieves similar or greater accuracy than state-of-the-art estimators. Timing comparisons are presented that demonstrate the superior computational efficiency of the ccGOWL. We illustrate the grouping performance of the ccGOWL method on a cancer gene expression data set and an equities data set.

Sommarie

Nous abordons la tâche d'identification de sous-ensembles densément connectés de variables aléatoires gaussiennes multivariées dans un cadre de modèle graphique. Nous proposons deux nouveaux estimateurs basés sur la norme ordonnée pondérée ℓ_1 (OWL) : 1) La méthode OWL graphique (GOWL) est une méthode de vraisemblance pénalisée qui applique la norme OWL aux composantes du triangle inférieur de la matrice de précision. 2) La méthode OWL graphique colonne par colonne (ccGOWL), permet d'estimer la matrice de précision en effectuant des régressions linéaires régularisées selon le procédé OWL. Ces deux méthodes peuvent identifier simultanément des groupes de variables hautement corrélés et contrôler la rareté dans la matrice de précision résultante. Nous formulons GOWL de manière à résoudre un problème d'optimisation composite et à établir que l'estimateur dispose d'une solution globale unique. De plus, nous démontrons que les conditions de regroupement sont suffisantes pour chaque colonne de l'estimation par matrice de précision ccGOWL. Pour les données synthétiques où la structure de groupe est présente, l'estimateur ccGOWL nécessite une quantité de calcul considérablement réduite et permet d'obtenir une précision similaire ou supérieure à celle des meilleurs estimateurs actuels. Des comparaisons de durée de calcul sont présentées, démontrant l'efficacité de calcul supérieure du ccGOWL. Nous illustrons les performances de regroupement de la méthode ccGOWL sur un ensemble de données d'expression de gènes liés au cancer et d'actions cotées en bourse.

Acknowledgments

This thesis would not have been possible without the guidance and support of several people. I would like to begin by thanking my advisor, Prof. Mark Coates, for his mentor-ship and his support. Your direction has been invaluable to me throughout my research experience. I would also like to thank my collaborator, Bogdan Mazoure, for all his help and for always keeping me motivated. I am grateful to all my professors who have shared their knowledge and inspired me to keep moving forward in my pursuit of knowledge. Thank you to my mentor Jüergen Wendland for your constant encouragement throughout this process. Thank you to my friend, Tudor Manole, for continuing to inspire me through all the frustration and for helping me find my love of mathematics and optimization. Thanks to my family, Mom, Dad, Jarrett, and Jenna, for all the love they have given me through these tough times. I could not have achieved anything without them in my life.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Today, data is more readily available than ever before. We can easily store and organize large amounts of data and access it with ease. As a result, the dimensionality of these datasets has increased dramatically leading to a discipline in the field of statistics termed "high dimensional inference." This discipline, where the data dimension p can be much larger than the sample size n, has been accompanied by new variable selection and dimension reduction techniques for coping with the scale of modern datasets. A method that has received considerable attention is regularization, which includes both standard sparsity and structured sparsity regularization methods. Well-studied models often do not perform well in high dimensions for a variety of reasons and require the aforementioned methods to allow for a more scalable and interpretable model.

In this thesis, we develop new methods for a particular high dimensional inference task – the inference of the structure of Gaussian graphical models. In modern multivariate analysis, Gaussian graphical models (GGMs) have received much attention in recent years because of their ability to compactly capture dependencies between variables. However, learning the structure of GGMs is not well understood and remains a burgeoning area of research.

Identifying dependencies among variables plays an important role in a plethora of scientific fields including analysis of gene expression data and portfolio allocation. Unfortunately, existing structured sparsity methods lack scalability and are not well-equipped for high dimensional problems. The structure learning problem involves the estimation of the inverse covariance matrix also referred to as the precision matrix. GGMs encode their graph through the non-zero pattern of the inverse covariance matrix. An inherent way to estimate the precision matrix is via maximum likelihood [1]. However, this method can encounter problems such as the non-invertibility of the estimate and can yield a fully connected graph providing no structural information. Because of these restrictions, estimating the precision matrix column-by-column has received much attention since these methods can be numerically simpler and more accommodating to theoretical analysis [2], [3]. In recent years, many categories of structured covariance and precision matrices have been introduced. However, we will focus on the categories of sparsity and structured sparsity.

Structure learning in GGMs addresses the connection between precision matrix estimation and structured sparsity. We propose two novel estimators that utilize a specific regularizer known as the Order Weighted ℓ_1 (OWL) regularizer which has many benefits including efficient computation and the ability to cluster highly correlated predictors, referred to in the literature as the "grouping effect." These estimators provide solutions to both the scalability problem while effectively capturing structural dependencies in the data.

1.2 Problem Formulation

In this section, we describe the standard problem formulation for estimating the precision matrix in the Gaussian graphical model. Let $\mathbf{X} = \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n\} \sim N(\mathbf{0}, \boldsymbol{\Sigma})$ represent a collection of n i.i.d. p-dimensional random samples from a zero-mean multivariate Gaussian distribution. Given a random sample of $\mathbf{X}$, the goal is to estimate the precision matrix $\mathbf{\Theta} = \mathbf{\Sigma}^{-1}$. The non-zero elements of the precision matrix are of particular interest, since a non-zero element indicates the conditional dependence of two random variables.

A Gaussian graphical model is represented by an undirected graph $G = (\boldsymbol{V}, \boldsymbol{E})$, where $\boldsymbol{V}$ contains nodes corresponding to p random variables and $\boldsymbol{E}$ is a set of edges where the edge $(i, j) \in \boldsymbol{E}$ exists if and only if $\mathbf{\Theta}_{ij} \neq 0$. Therefore, the graph $G = (\boldsymbol{V}, \boldsymbol{E})$ describes the conditional independence/dependence relationships of variables $\mathbf{X} = \{\boldsymbol{X}_1, \dots, \boldsymbol{X}_n\}$. Thus, GGM estimation is equivalent to estimating the precision matrix and identifying the non-zero pattern of the estimate.

Structured sparsity allows for more complex knowledge to be extracted from the non-zero pattern of the precision matrix. Previous methods refer to this knowledge as *groups* or *blocks* of variables. Structural priors then encourage group (block) sparsity — an edge in the graphical model is modelled as more likely if it connects variables that belong to the same group. Conversely, methods can also introduce a structural prior that encourages grouping of *edges* in the graphical model. In this thesis, our primary interest is recovering the structure of a GGM under the assumption that pairs of variables are estimated to have the same non-zero $\mathbf{\Theta}_{ij}$ entry in the precision matrix estimate.

1.3 Main Contributions

This thesis presents two novel estimators of the precision matrix associated with a Gaussian graphical model and makes four major contributions. Both estimators are based on the Ordered Weighted ℓ_1 (OWL) norm: 1) The Graphical OWL (GOWL) is a penalized likelihood method that applies the OWL norm to the lower triangle components of the precision matrix. 2) The column-by-column Graphical OWL (ccGOWL) estimates the precision matrix by performing OWL regularized linear regressions. First, both estimators require a small amount of *a priori* information and do not require any information about group structure. More specifically, they only require two hyperparameters, one for controlling sparsity and the other for controlling the amount of grouping [4]. This differs from [5] which requires p^2 hyperparameters and GRAB [6] which requires an objective function that imposes significantly more constraints. Second, both estimators can learn overlapping groups and network structure in a single-step proximal descent procedure. This can be interpreted as an advantage over the *cluster graphical lasso* [7], which solves the task with a two-step procedure by alternating between clustering and gradient steps. Likewise, the GRAB algorithm also alternates between learning the overlapping group prior matrix and learning the inverse covariance. Our focus is on estimating the graphical model, rather than identifying groups, but we can apply a Gaussian Mixture Model (GMM) to identify overlapping groups from our precision matrix estimate. Third, we establish the uniqueness of the first proposed estimator GOWL estimator by deriving its dual formulation. Fourth, the ccGOWL framework provides new theoretical guarantees for grouping related entries in the precision matrix and offers a more computationaly efficient algorithm. When comparing ccGOWL to the previously mentioned penalized likelihood methods, it is clear that the ccGOWL is more computationally attractive in the high dimensional setting as it can estimate the precision matrix one column at a time by solving a simple linear regression that can be easily parallelized/distributed.

1.3.1 Uniqueness of the GOWL

In optimization theory, the “duality principle” states that optimization problems can be viewed in their primal or dual forms. The solution to the dual problem yields a lower bound on the solution to the primal problem. However, their solutions need not be equal and their difference is referred to as the “duality gap.” We derive the dual formulation for the GOWL estimator and show that under certain conditions the duality gap is zero showing the equivalence of the primal and dual solutions. Furthermore, using this fact we show that the GOWL estimator has a unique solution.

1.3.2 Sufficient Grouping Conditions for ccGOWL

We prove sufficient grouping conditions when estimating each column of $\boldsymbol{\Theta}$ by drawing on previous work for the OSCAR and OWL regularizers. With this sufficient grouping condition, an explicit relationship between the hyperparameters chosen and sample correlation between covariates can be established. This can be utilized when tuning hyperparameters for the ccGOWL estimator which might normally be a difficult task.

1.4 Notation

Throughout this thesis, we highlight vectors and matrices by lowercase and uppercase boldfaced letters, respectively. Let $\mathcal{X}$ be a Hilbert space with associated inner product $\langle \cdot, \cdot \rangle$ and norm $|| \cdot ||$. For a vector $\mathbf{a} \in \mathbb{R}^p$, let $\mathbf{a}_{-i}$ denote a vector with its i^{th} component removed. For a symmetric matrix $\mathbf{A} \in \mathbb{R}^{p \times p}$: $\mathbf{A}_{i,*}$ represents the i^{th} row of $\mathbf{A}$, $\mathbf{A}_{*,j}$ denotes the j^{th} column of $\mathbf{A}$, $\mathbf{A}_{i,-j}$ denotes the i^{th} row of $\mathbf{A}$ with its j^{th} entry removed, $\mathbf{A}_{-i,j}$ denotes the j^{th} column of $\mathbf{A}$ with its i^{th} entry removed and the matrix $\mathbf{A}_{-i,-j}$ denotes a $(p-1) \times (p-1)$ matrix obtained by removing the i^{th} row and j^{th} column. Moreover, we denote as $\mathbf{vechs}(\mathbf{A})$ the strict column-wise vectorization of the lower triangular component of $\mathbf{A}$:

$$\mathbf{vechs}(\mathbf{A}) = \left[\mathbf{A}_{2,1}, \ldots, \mathbf{A}_{n,1}, \mathbf{A}_{3,2}, \ldots, \mathbf{A}_{n,2}, \mathbf{A}_{n,n-1}\right]. \tag{1.1}$$

For a vector $\mathbf{a} = (a_1, \ldots, a_p)^T \in \mathbb{R}^p$, the ℓ_q norm is $||\mathbf{a}||_q = (\sum_{i=1}^{p} |a_i|^q)^{1/q}$ for $1 \leq q \leq \infty$. The i^{th} largest component in magnitude of $\{|a_1|, |a_2|, \ldots, |a_p|\}$ is denoted $a_{[i]}$. The vector obtained by sorting (in non-increasing order) the components of $\mathbf{a}$ is denoted $\mathbf{a}_{\downarrow} = \{a_{[1]}, a_{[2]}, ..., a_{[p]}\}$. For matrices and vectors, we define $| \cdot |$ to be the element-wise absolute value function.

1.5 Contributions and Acknowledgements

The production of this thesis has benefited greatly from the editorial help and advice given by my supervisor Prof. Mark Coates. In addition, Bogdan Mazoure assisted with the derivation of the dual problem for the GOWL estimator which is described in Chapter 4 and also provided some editorial help with Chapter 1. More specifically, he suggested using the generalized rearrangement inequality in order to extend the results derived for the ℓ_1 norm to the OWL norm. Bogdan also helped with the generation of the figures in Chapter 6 and Chapter 7. All other contents of this thesis including materials such as code used to generated figures and results were completed by me.

1.6 Thesis Overview

In Chapter 2, we detail terminology in convex optimization, proximal methods, and structure learning. We discuss the important regularization techniques in the linear setting and discuss important structure learning estimators. The numerical procedures that achieve optimal points for the GOWL and ccGOWL estimators are also discussed in detail. In Chapter 3, we review relevant literature for structure learning in GGMs. In Chapter 4, we introduce the theoretical results and implementations of the GOWL and ccGOWL, respectively. In Chapter 5 and 6, we showcase the superior performance of ccGOWL over state-of-the-art methods on synthetic and real datasets. In Chapter 7, we summarize our conclusions and outline possible future work.

Chapter 2

Background Material

2.1 Overview

In this chapter, we describe all background work on optimization and numerical procedures presented in the literature. All commonly used terminology in these disciplines are outlined rigorously.

First, we define terminology used in constrained and unconstrained convex optimization, which are pivotal in understanding the theory presented in later chapters. Regularization and the overfitting problem are discussed which provide intuition for the main contributions of this thesis. Then, we provide a brief description of the most commonly studied regularizers applied in the linear regression setting. We summarize and review the OWL and OSCAR regularizers; the latter being a special case of the OWL. We provide background on GGMs including definitions required for understanding precision matrix estimation. At the end of this chapter, we outline the class of algorithms used to minimize the objective functions of both estimators presented.

2.2 Convex Optimization

Definition 1 (Convex Sets) *A set $C \subset \mathbb{R}^n$ is called convex if*

$$\lambda \boldsymbol{x} + (1-\lambda)\boldsymbol{y} \in C \quad (\boldsymbol{x}, \boldsymbol{y} \in C, \lambda \in (0,1))$$

Simply put, a set is convex if a line can be drawn between any two points that will remain in the set.

Definition 2 (Lipschitz Smoothness) *A function $f : \mathcal{X} \to \mathbb{R}$ where $\mathcal{X}$ is a Hilbert space, is called Lipschitz continuous if there exists a positive real constant L such that for any $\boldsymbol{x}_1, \boldsymbol{x_2} \in \mathcal{X}$*

$$||f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)||_2 \leq L||\boldsymbol{x}_1 - \boldsymbol{x}_2||$$

Definition 3 (Convex Constrained Non-Linear Program (NLP)) *A convex NLP is of the form*

$$\begin{aligned} \min f(\boldsymbol{x}) \quad s.t. \quad & g_i(\boldsymbol{x}) \leq 0 \quad (i = 1, \dots, m) \\ & h_j(\boldsymbol{x}) = 0 \quad (j = 1, \dots, p) \end{aligned} \tag{2.1}$$

where $f, g_j, h_j : \mathbb{R}^n \to \mathbb{R}$ are convex which implies that the feasible set:

$$X = \{\boldsymbol{x} \in \mathbb{R}^n : g_i(\boldsymbol{x}) \leq 0 \ (i = 1, \dots, m) \ and \ h_j(\boldsymbol{x}) = 0 \ (j = 1, \dots, p)\}$$

is convex. It's Lagrangian is given by

$$L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^T g(\boldsymbol{x}) + \boldsymbol{\mu}^T h(\boldsymbol{x}) \tag{2.2}$$

We will refer to (2.1) *as the Lagrangian primal problem.*

The dual problem relies on the following observation:

$$\sup_{\boldsymbol{\lambda}\in\mathbb{R}^m_+,\boldsymbol{\mu}\in\mathbb{R}^p} L(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\mu}) = \begin{cases} f(\boldsymbol{x}) & \text{if} \quad \boldsymbol{x} \in X \\ +\infty & \text{else} \end{cases}$$

Definition 4 (Lagrangian Dual Problem) *The Lagrangian dual problem of* (2.2) *is given by*

$$\max d(\boldsymbol{\lambda},\boldsymbol{\mu}) \quad s.t. \quad \boldsymbol{\lambda} \geq 0$$

where

$$d : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R} \cup \{-\infty\}, \quad d(\boldsymbol{\lambda},\boldsymbol{\mu}) := \inf_{\boldsymbol{x}\in\mathbb{R}^n} L(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\mu})$$

is the dual objective function.

Definition 5 (Slater's Constraint Qualification (SCQ)) *We say that Slater constraint qualification (SCQ) holds for* (2.1) *if there exists* $\hat{\mathbf{x}} \in \mathcal{X}$ *such that*

$$g_i(\hat{\mathbf{x}}) < 0 \quad \textit{and} \quad h_j(\hat{\mathbf{x}}) = 0$$

Then strong duality holds at the point $\hat{\mathbf{x}}$. Consequently, without any differentiablility assumption on f, g, h, if SCQ holds and $f, g, \mathcal{X}$ are convex, then no duality gap exists between the primal and dual solutions. This is often referred to as *strong duality.*

Definition 6 (Strong Duality) *Without any differentiability assumptions on* (2.1), *if the SCQ holds, then we can say that there is no duality gap. This means that the primal solution is equal to the dual solution.*

2.3 Regularization

We will be considering objective functions of the form

$$\min_{\boldsymbol{x}\in\mathbb{R}^p} \mathcal{L}(\boldsymbol{x}) + \Omega(\boldsymbol{x})$$

where $\mathcal{L}$ is referred to as the *Loss function* and $\Omega(\boldsymbol{x})$ is the *Regularization term.*

2.3.1 Ordinary Least Square Regression

Let us first begin with defining the Ordinary Least Square regression problem.

Definition 7 *The* ***Ordinary Least Squares (OLS)*** *(also referred to as the linear least squares method (LLS)) estimator is defined as the minimizer of the residual sum of squares (RSS or mean-squared error) objective defined by*

$$RSS(\boldsymbol{\beta}) = \sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2$$

There are a few things to highlight in the above equation. First of all, since the objective function given by $RSS(\boldsymbol{\beta})$ is convex, it is minimized when the gradient is zero. Alternatively, we can define the linear least squares problem in terms of matrices:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{1}{2}||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||_2^2 \tag{2.3}$$

We will primarily be referring to the matrix formulation of problems such as the least squares method above for the remainder of the thesis. In optimization terms, we refer to (2.3) as a finite-dimensional optimization problem of the form

$$\min_{\boldsymbol{\beta}\in\mathcal{X}} f(\boldsymbol{x})$$

where $\mathcal{X} \subset \mathbb{R}^n$ is nonempty and $f : \mathcal{X} \to \mathbb{R}$. Here $\mathcal{X}$ is called the *feasible set* and f can be referred to as the *objective function.* If the feasible set $\mathcal{X} = \mathbb{R}^n$ we refer to the problem as an unconstrained optimization problem and if $\mathcal{X} \subsetneq \mathbb{R}^n$ we refer to the problem as a constrained optimization problem.

Theorem 2.3.1 *Let $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ and $\boldsymbol{y} \in \mathbb{R}^n$. Then the following hold:*

(a) $\hat{\boldsymbol{\beta}}$ solves (2.3) *if and only if $\hat{\boldsymbol{\beta}}$ is the solution to the equation*

$$\boldsymbol{X}^T \boldsymbol{X} \boldsymbol{\beta} = \boldsymbol{X}^T \boldsymbol{y}$$

and (2.3) *always has a solution.*

(b) The solution of (2.3) *is unique if and only if rank $\boldsymbol{X} = p$. In other words, if and only if $\boldsymbol{X}$ has full column rank. $\boldsymbol{X}^T\boldsymbol{X}$ needs to be invertible in order for $\hat{\boldsymbol{\beta}}$ to be unique.*

The main problem with *linear least squares method* is that when $p > n$ the matrix $\boldsymbol{X}^T\boldsymbol{X}$ is not invertible and thus $\hat{\boldsymbol{\beta}}$ is not unique. This fact results in many $\hat{\boldsymbol{\beta}}$ that are solutions to the problem, leading to an *ill-posed* problem which we will formally define in a future section.

2.3.2 Tikhonov Regularization

Tikhonov Regularization, also known as *ridge regression*, is a regularization technique that applies a penalty to the regression coefficients. In order to understand why Tikhonov regularization is so important we must first define a few important concepts.

Definition 8 *A problem is said to be* ***well-posed problem*** *if the following are true:*

(a) a solution exists

(b) the solution is unique

Problems that are not well-posed are termed ***ill-posed problems****.*

Recall, our data matrix $\boldsymbol{X} \in \mathbb{R}^{n \times p}$. The matrix is said to be *underdetermined* if $n < p$. This means there are infinitely many solutions to (2.3) leading to an *ill-posed problem.* The question that arises is how do we reformulate this problem in order for the problem to be *well-posed*?

Definition 9 *Let $\boldsymbol{X} \in \mathbb{R}^{n \times p}, \boldsymbol{y} \in \mathbb{R}^n, \boldsymbol{D} \in \mathbb{R}^{p \times n}$ and $\lambda > 0$, we define the* ***Tikhonov regularization of the OLS*** *problem as*

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^p} \frac{1}{2}||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||_2^2 + \frac{\lambda}{2}||\boldsymbol{D}\boldsymbol{\beta}||_2^2 \tag{2.4}$$

The matrix $\boldsymbol{D}$ is referred to as the Tikhonov matrix and can be chosen to be the identity matrix $\boldsymbol{D} = \boldsymbol{I}$, which is known as ℓ_2 ***regularization****.*

The following properties therefore hold:

(a) $\hat{\boldsymbol{\beta}}$ is the solution to (2.4) if and only if $\hat{\boldsymbol{\beta}}$ solves $(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{D}^T\boldsymbol{D})\boldsymbol{\beta} = \boldsymbol{X}^T\boldsymbol{y}$.

(b) Thus, if $(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{D}^T\boldsymbol{D})$ is invertible then a unique solution does exist. The optimal solution of (2.4) is given by $(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{D}^T\boldsymbol{D})^{-1}\boldsymbol{X}^T\boldsymbol{y}$.

Fact 2.3.2 *Let $\boldsymbol{X} \in \mathbb{R}^{n \times p}, \lambda \in \mathbb{R}$ where $\lambda > 0$, the quantity $\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}$ is always invertible.*

The main problem with the *OLS method* is that if $\boldsymbol{X}^T\boldsymbol{X}$ is not *invertible* than an infinite number of solutions exist. *Ridge regression* fixes this problem when we choose $\boldsymbol{D}$ to be an *orthogonal matrix* by applying Fact 2.3.2. The main advantages of the *Tikhonov Regularization* are its simple closed-form solution and its smooth convex objective function.

2.3.3 LASSO Regularization

LASSO is a sparsity inducing regularization technique that penalizes the sum of the absolute values of the regression coefficients as opposed to the sum of squares such as in Tikhonov regularization. LASSO stands for *least absolute shrinkage and*

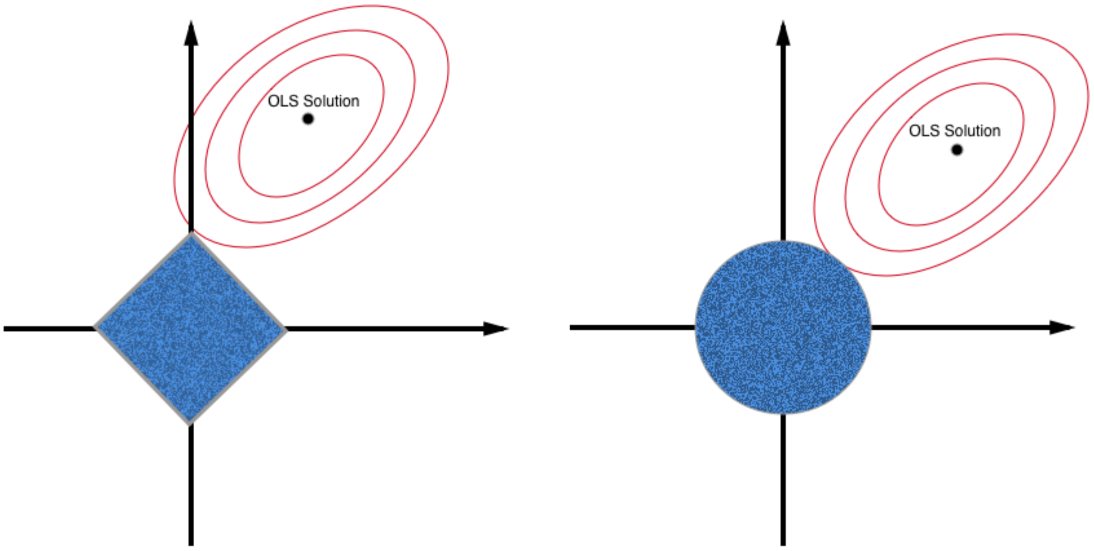

Figure 2.1: Contours centered at OLS estimate with LASSO and Ridge constraint regions.

selection operator and was originally proposed in [8]. LASSO performs both shrinkage and variable selection, thus removing predictors entirely from the model. The main benefit is that it can be used in high dimensional problems and can be easily interpreted.

Definition 10 *Let $\boldsymbol{X} \in \mathbb{R}^{n \times p}, \boldsymbol{y} \in \mathbb{R}^n, \boldsymbol{\beta} \in \mathbb{R}^p$, and $\lambda > 0$. We define the* ***LASSO regularization of OLS*** *problem as*

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^p} \frac{1}{2}||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||_2^2 + \lambda||\boldsymbol{\beta}||_1 \tag{2.5}$$

Figure 2.1 illustrates a comparison between LASSO (left) and ridge (right) if there were two parameters β_1 and β_2 in your model. The mean-squared error loss function is represented by red elliptical contours which are centered at the OLS solution. The LASSO constraint region is diamond in shape and the regression constraint region is a circle. In the LASSO case, the goal is to arrive at a solution on the corners depicting a sparse solution. Conversely, ridge regression does not offer this benefit.

Similar to Tikhonov Regularization, in the case where the matrix $\boldsymbol{X}$ is *orthonormal* ($n^{-1}\boldsymbol{X}^T\boldsymbol{X}$) equation (2.5) will have an explicit solution. Figure 2.2 shows that both

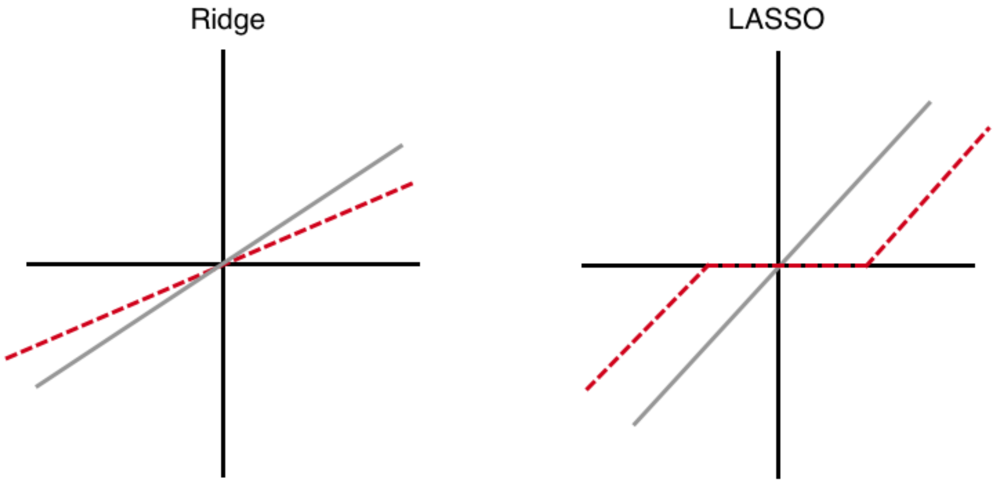

Figure 2.2: Depicts that ridge and LASSO penalty operators induce a transformation on the OLS estimates $\hat{\boldsymbol{\beta}}$.

methods apply transformations to the OLS estimate $\hat{\boldsymbol{\beta}}$. In the LASSO case, it will translate each OLS estimate $\hat{\beta}_j$ by a constant factor λ, truncating it to zero. This is called the "soft-thresholding" operator.

Definition 11 *Let $\boldsymbol{X} \in \mathbb{R}^{n \times p}, \hat{\beta}_j \in \mathbb{R}$. The **soft-thresholding** operator is defined as*

$$S(\hat{\beta}_j|\lambda) = \text{sign}(\hat{\beta}_j)(|\hat{\beta}_j| - \lambda)_+ = \text{sign}(\hat{\beta}_j)\max(0, |\hat{\beta}_j| - \lambda)$$

where sign denotes the sign of the argument (± 1), and x_+ denotes the "positive part" of x. Alternatively, the soft-thresholding operator can be defined as

$$S(\hat{\beta}_j|\lambda) = \begin{cases} \hat{\beta}_j - \lambda & \text{if } \hat{\beta}_j > \lambda \\ 0 & \text{if } |\hat{\beta}_j| \leq \lambda \\ \hat{\beta}_j + \lambda & \text{if } \hat{\beta}_j < -\lambda \end{cases}$$

Thus, the probability of a predictor $\boldsymbol{x}_j$ being rendered as insignificant by our model is $\mathbb{P}(|\hat{\beta}_j| \leq \lambda)$.

Although the LASSO's objective function is non-smooth, its still convex which can lead to computational and theoretical benefits. The LASSO also produces sparse solutions where a subset of predictors are used in the model, potentially reducing overfitting. Computation of the LASSO is quite efficient with algorithms such as proximal algorithms and LARS [9]. Some drawbacks of the LASSO includes its introduction of **bias** into coefficient estimates. *Statistical consistency* means that as the sample size grows, the regression coefficient estimates should converge to the true values that were used to generate $\boldsymbol{y}$. There are a few approaches such as the *relaxed lasso* [10] that are used to address this problem, but we will not go into further detail here. The LASSO depends on the **order of predictors**. If you change the order of predictors in the LASSO you could compute completely different estimates. When minimizing the LASSO objective, regularization paths are usually computed using coordinate descent. As a result, the order of predictors can affect the path taken which can lead to differences in the final estimate. This fact presents an issue since the order of predictors is usually chosen arbitrarily.

2.3.4 Adaptive Lasso

Recall that the LASSO suffers from high bias which is mainly determined by the λ parameter. One way to reduce this bias is by modifying the λ penalty for individual predictors. This can be achieved by introducing weights: $\lambda_j = \lambda w_j$, where w_j is the weight applied to the j^{th} predictor. In simpler terms, the weight w_j scales the λ penalization to be larger or smaller for certain predictors. The choice of $\boldsymbol{w}$ is realized by choosing an initial estimate of $\boldsymbol{\beta}$ denoted by $\tilde{\boldsymbol{\beta}}$. The initial estimate can be chosen to be the OLS or LASSO estimate. This approach is referred to as the *Adaptive LASSO* and was initially proposed in [11].

Definition 12 *Let $\boldsymbol{X} \in \mathbb{R}^{n\times p}, \boldsymbol{y} \in \mathbb{R}^n, \boldsymbol{\beta} \in \mathbb{R}^p$, and $\lambda > 0$, we define the **Adaptive Lasso** problem as the following*

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{1}{2}||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||_2^2 + \lambda \sum_{j}^{p} w_j|\beta_j|$$

where $w_j = |\tilde{\beta}_j|^{-1}$

The above is known as the *two stage approach* because after estimates of $\tilde{\boldsymbol{\beta}}$ are made, we then choose the vector of weights. The *pathwise approach* is to let weights change with λ. There are many different weighting strategies present in the literature.

The main advantages of the Adaptive LASSO includes its convexity and its ability to yield **consistent** estimates of regression coefficients. It satisfies the **Oracle property** and utilizes the *LARS* algorithm. However, it fails to identify groups of correlated predictors.

2.3.5 Group LASSO, Group SCAD, Group MCP

A "group" method focuses on situations where features can be organized into related groups and the goal is to select the important groups. This can lead to more sensible models. The group regularizers that we examine in this section encourage structured/group sparsity. The group LASSO was proposed in [12] as an extension of the LASSO that penalizes groups of coefficients as opposed to single coefficients. The design matrix $\boldsymbol{X}$ will be composed of J groups $\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_J}$. Let K_j denote the size of group j. So $\sum_{j=1}^{J} K_j = p$. $\eta = \boldsymbol{X\beta} = \sum_{j=1}^{p} \boldsymbol{x_j\beta_j}$ where $\boldsymbol{\beta_j}$ is a group of coefficients.

Definition 13 *Let* $\boldsymbol{X} \in \mathbb{R}^{n\times p}, \boldsymbol{y} \in \mathbb{R}^n, \boldsymbol{\beta} \in \mathbb{R}^p$, *and* $\lambda > 0$. *We define the* ***Group Lasso*** *problem as*

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{1}{2}||\boldsymbol{y} - \boldsymbol{X\beta}||_2^2 + \lambda \sum_{j}^{p} \lambda_j ||\boldsymbol{\beta_j}||_2$$

Similar to previous regularizers, we consider the case where the design matrix $\boldsymbol{X}$ is *orthonormal.* An equivalent in the group setting can be defined as $\boldsymbol{x_j^T x_k} = 0$ for $j \neq k$.

Theorem 2.3.3 ([13]) *Suppose $\boldsymbol{X_j^T X_k} = 0$ for $j \neq k$ and $\frac{1}{n}\boldsymbol{X_j^T X_k} = \boldsymbol{I}$ for all j. Let $\hat{\boldsymbol{\beta}}_{\boldsymbol{j}} = \frac{1}{n}\boldsymbol{X_j^T y}$ denote the OLS solution. The value of $\boldsymbol{\beta}$ that minimizes*

$$\frac{1}{2n}(\boldsymbol{y} - \boldsymbol{X\beta})^T(\boldsymbol{y} - \boldsymbol{X\beta}) + \sum_j \lambda_j ||\boldsymbol{\beta_j}||_2$$

is given by

$$\boldsymbol{\beta_j} = S(||\hat{\boldsymbol{\beta}}_{\boldsymbol{j}}||_2, \lambda_j)\frac{\hat{\boldsymbol{\beta}}_{\boldsymbol{j}}}{||\hat{\boldsymbol{\beta}}_{\boldsymbol{j}}||_2}$$

The above intuition can be extended to the group Smoothly Clipped Absolute Deviation (SCAD) [14] and the group Minimax Concave Penalty (MCP) [15]. One of the main drawbacks is that the closed form solutions have a strong assumption that $\boldsymbol{X}_j^T\boldsymbol{X}_j = \boldsymbol{I}$ which is not always the case. However, a really interesting approach is that you can transform any problem to the orthonormal case and then transform it back using the spectral theorem.

In conclusion, the *Group Methods* discussed in this section can be computed efficiently and yield **consistent** estimates, but require **prior specification** of the group structure of our data, which is often unknown.

2.3.6 Elastic Net and Dantzig Selector

The Elastic Net (EN) penalty was first proposed in [16]; this approach involves introducing a ridge penalty in order to reduce the variance at the cost of increasing the bias.

Definition 14 *Let $\boldsymbol{X} \in \mathbb{R}^{n\times p}, \boldsymbol{y} \in \mathbb{R}^n, \boldsymbol{\beta} \in \mathbb{R}^p$, and $\lambda > 0$. We define the **Elastic Net** problem as the following*

$$\min_{\beta\in\mathbb{R}^p} \frac{1}{2}||\boldsymbol{y} - \boldsymbol{X\beta}||_2^2 + \lambda_1||\boldsymbol{\beta}||_1 + \frac{\lambda_2}{2}||\boldsymbol{\beta}||_2^2$$

Remark 1 *A few remarks concerning the Elastic Net penalty [16]* *(a) Recall that the ridge penalty is strictly convex. Thus, the solution will be unique provided that $\lambda_2 > 0$.*

(b) If $\lambda < 1$, the LASSO will produce infinitely many solutions. If we look at an example where $\boldsymbol{\beta} \in \mathbb{R}^2$, the LASSO will admit infinitely many solutions along the line $\beta_1 + \beta_2 = 1 - \lambda$ when $\beta_1 > 0, \beta_2 > 0$.

(c) One beneficial property of EN is whenever $\boldsymbol{x_j} = \boldsymbol{x_k}$, then $\hat{\beta}_j = \hat{\beta}_k$.

(d) EN inherits the sparsity property of the LASSO when $\boldsymbol{\beta} = 0$ if $\lambda_1 > 1$.

(e) EN inherits the ability to always produce a unique solution from ridge regularization.

One of the challenges in practice when using the regularizers we have mentioned is *hyperparameter tuning.* One common approach is that of *cross-validation* for the selection of the optimal hyperparameters. A common reparameterization of the EN is to express the hyperparameters λ_1 and λ_2 using a single hyperparameter λ and a constant α which determines the balance between the LASSO and ridge penalties as follows:

$$\lambda_1 = \alpha\lambda$$

$$\lambda_2 = (1 - \alpha)\lambda$$

This allows us to fix $\alpha > 0$ and select λ using a technique such as cross-validation. It is more computationally expensive to tune λ_1 and λ_2 separately. We will revisit this common issue in further sections. Similar to previous regularizers, Definition 15 shows that in the orthonormal case of EN, the regularization term has a closed form solution. The orthonormal case refers to when the design matrix $\boldsymbol{X}$ is orthonormal: $(n^{-1}\boldsymbol{X}^T\boldsymbol{X} = \boldsymbol{I})$. A closed form solution refers to the fact that there is no iterative optimization procedure required to arrive at a solution.

Definition 15 *Let $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ and $\hat{\beta}_j^{OLS} \in \mathbb{R}$ denote the j^{th} component of the OLS solution. The **EN thresholding operator** is defined by:*

$$S_{EN}(\hat{\beta}_j^{OLS}|\lambda_1, \lambda_2) = \frac{S(\hat{\beta}_j^{OLS}|\lambda_1)}{1+\lambda_2}$$

where S is the soft-thresholding operator defined in Definition 11.

A very important concept which will be a central theme of this thesis is a property called the *grouping effect*. This property states that if predictors are highly correlated they will have similar regression coefficients. More specifically, in a high-dimensional problem predictors can often be "grouped". For instance, in genomics it is often desirable to identify highly correlated genes. Previously, principal component analysis (PCA) has been used to identify such groups, but recently it has been proposed to use a regularization regression procedure to identify such "groups". We define this property as an upper bound on the difference between two predictors.

Theorem 2.3.4 ([16]) *Let $\boldsymbol{y} \in \mathbb{R}^n, \hat{\beta}_j, \hat{\beta}_k \in \mathbb{R}, \rho_{jk} \in [0,1]$. The **grouping effect** states that*

$$|\hat{\beta}_j - \hat{\beta}_k| \leq \frac{||\boldsymbol{y}||_2\sqrt{2(1-\rho_{jk})}}{\lambda_2\sqrt{n}}$$

where $\rho_{jk} = \boldsymbol{x_j^T x_k}$. If $\rho_{jk} \to 1$, the difference between $|\hat{\beta}_j - \hat{\beta}_k| \to 0$.

The Elastic Net balances variance and bias well and can be computed efficiently. It also exhibits the **grouping effect**. We will discuss other regularizers that have been shown to out perform EN in feature grouping [4].

2.4 OSCAR/OWL/SLOPE

2.4.1 OSCAR

One of main interests of this thesis is the idea of structured/group sparsity by means of convex optimization [17]. One such regularizer that promotes variable/feature grouping is the OSCAR [4]. This regularizer is different than previous regularizers discussed in that it requires no previous knowledge of structure from the data and is not tied to the order of the predictors.

Definition 16 ***(OSCAR)*** *Let $\boldsymbol{X} \in \mathbb{R}^{n\times p}, \boldsymbol{y} \in \mathbb{R}^n, \boldsymbol{\beta} \in \mathbb{R}^p$*

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{1}{2}||\boldsymbol{y} - \boldsymbol{X\beta}||_2^2 + \lambda_1||\boldsymbol{\beta}||_1 + \lambda_2 \sum_{i<j} \max\{|\beta_i|, |\beta_j|\}$$

where $\lambda_1, \lambda_2 \geq 0$. The OSCAR formulation can be written alternatively as a constrained optimization problem

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{1}{2}||\boldsymbol{y} - \boldsymbol{X\beta}||_2^2 + \lambda \sum_{j=1}^{p} \{c(j-1)\}|\beta|_{(j)} \tag{2.6}$$

Similar to the Elastic Net, the OSCAR exhibits a "grouping effect" which can be understood in terms of the correlation between predictors. Furthermore, this means that if the correlation between predictors is large, then the regression coefficients will be equal.

Theorem 2.4.1 ([4]) *Set $\lambda_1 \equiv \lambda$ and $\lambda_2 \equiv c\lambda$ in the Lagrangian formulation given by (2.6). Given $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ and $\boldsymbol{y} \in \mathbb{R}^n$, let $\boldsymbol{\beta}(\lambda_1, \lambda_2)$ be the OSCAR estimate using tuning parameters (λ_1, λ_2). Assume that the predictors are signed so that $\hat{\beta}_i(\lambda_1, \lambda_2) \geq 0$ for all i. Let $\rho_{ij} = \boldsymbol{x_i^T x_j}$ be the sample correlation between predictors i and j. For a given pair of predictors $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$, suppose both $\hat{\beta}_i(\lambda_1, \lambda_2) > 0$ and $\hat{\beta}_j(\lambda_1, \lambda_2) > 0$ are distinct from the other $\hat{\beta}_k$. Then there exists $\lambda_0 \geq 0$ such that if $\lambda_2 > \lambda_0$ then*

$$\hat{\beta}_i(\lambda_1, \lambda_2) = \hat{\beta}_j(\lambda_1, \lambda_2), \textit{ for all } \lambda_1 > 0.$$

Furthermore, it must be that

$$\lambda_0 \leq 2||\boldsymbol{y}||_2\sqrt{2(1-\rho_{ij})} \tag{2.7}$$

According to Theorem 2.4.1 λ_2 will control the degree of grouping, and as λ_2 increases the two predictors will eventually group together, meaning that they will have the same regression coefficient estimate. This will be further elaborated on in the next section. In addition, equation (2.7) shows that predictors will be grouped if $\rho = 1$ for any $\lambda_2 > 0$. OSCAR was proposed to be solved as a Quadratic Programming (QP) problem. Unfortunately, OSCAR suffers from a similar disadvantage in the hyperparameter tuning procedure previously stated in the Elastic Net section.

2.4.2 OWL/SLOPE

The OWL was proposed jointly in [18] (referred to as SLOPE) and in [19] of which the OSCAR discussed in the previous section is a particular case.

Definition 17 (OWL) *Let* $\boldsymbol{X} \in \mathbb{R}^{n\times p}, \boldsymbol{y} \in \mathbb{R}^n, \boldsymbol{\beta} \in \mathbb{R}^p$

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{1}{2}||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||_2^2 + \sum_{i=1}^{p} w_i|\beta|_{[i]} \tag{2.8}$$

where $|\beta|_{[i]}$ *is the i-th largest component in magnitude of* $\boldsymbol{\beta} \in \mathbb{R}^p$*, and* $\boldsymbol{w} \in \mathbb{R}_+^p$*.*

The OSCAR is a special case of the OWL obtained by setting $w_i = \lambda_1 + \lambda_2(p - i)$. The OWL norm proposed by [18] sets the weights to $w_i = F^{-1}(1 - iq/(2p))$, where F is the cumulative distribution function of the error terms which are indirectly observed in data, and $q \in (0, 1)$ is the expected proportion of false discoveries. The authors aim to control the *false discovery rate* (FDR) which is utilized as a measure of error in multiple hypothesis testing. The FDR is computed by dividing the mean number of false rejections by the total number of rejections. The solution to (2.8) with an *orthonormal design* matrix $\boldsymbol{X}$ has a FDR for variable selection bounded by $q(p - k)/p$, where k is the number of non-zero coefficients in the true coefficients that generated the response. We will now discuss sufficient conditions for grouping of the OWL.

Theorem 2.4.2 ([20]) *Consider the objective function with a squared-error loss function* (2.8)*. Let* $\boldsymbol{a_i} \in \mathbb{R}^n$ *denote the i-th column (for* $i = 1, \dots, p$*) of matrix* $\boldsymbol{X}$*.*

The columns of the design matrix $\boldsymbol{X}$ *must be normalized to a common norm, that is,* $||\boldsymbol{a_k}||_2 = c$*, for* $k = 1, \ldots, p$*. Let* $\hat{\boldsymbol{\beta}}$ *be any minimizer of the objective function* (2.8)*. Then, for every pair of columns* (i, j) *for which* $||\boldsymbol{y}||_2|| \operatorname{sign}(\hat{\beta}_i)\boldsymbol{a}_i - \operatorname{sign}(\hat{\beta}_j)\boldsymbol{a}_j||_2 < \Delta$ *where* $\Delta := \min\{w_l - w_{l+1}, l = 1, \ldots, p-1\}$ *is the minimum gap between two consecutive components of vector* $\boldsymbol{w}$*, we have* $|\hat{\beta}_i| = |\hat{\beta}_j|$*.*

Corollary 2.4.3 ([20]) *Let the columns of* $\boldsymbol{X}$ *be normalized to zero sample mean and unit norm:* $\mathbb{1}^T\boldsymbol{a_k} = 0$ *and* $||\boldsymbol{a_k}||_2 = 1$*, for* $k = 1, \ldots, p$*. Denote their inner products (ie. sample correlation of the corresponding predictor variables) as* $\rho_{ij} = \frac{\boldsymbol{a_i}^T\boldsymbol{a_j}}{||\boldsymbol{a_i}||_2||\boldsymbol{a_j}||_2} = \boldsymbol{a_i}^T\boldsymbol{a_j}$*. Then, the condition in Theorem 2.1 becomes*
$||\boldsymbol{y}||_2||\sqrt{2 - 2\rho_{ij}\operatorname{sign}(\hat{\beta}_i\hat{\beta}_j)} < \Delta$.

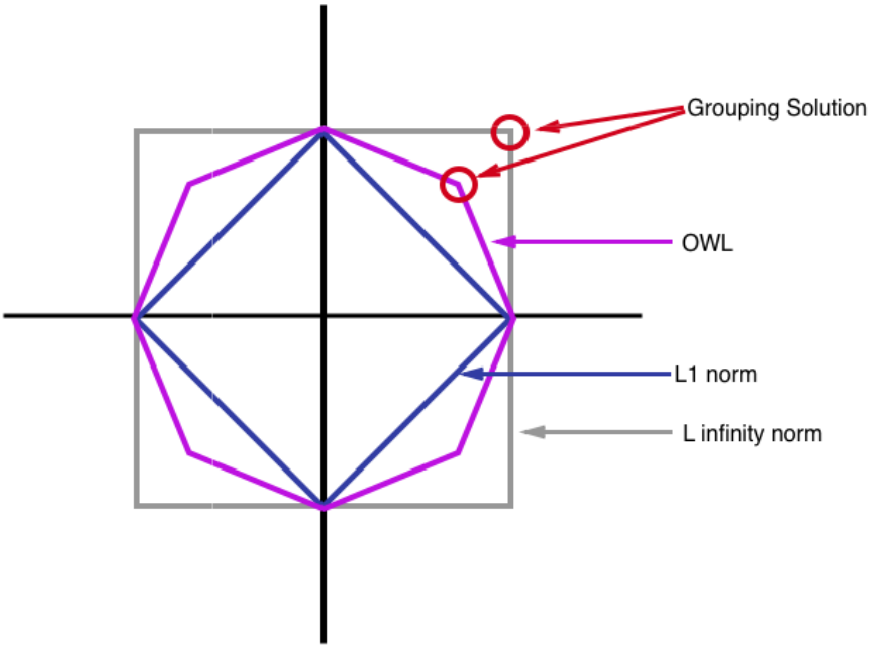

Figure 2.3: Illustrates behaviour of different regularizers in a 2-dimensional setting where the axes represent the values of β_1 and β_2, respectively.

Figure 2.3 provides the geometric intuition for the "grouping effect". The grouping solution is the equality-inducing vertex which is equal distance from each axis. It is important to note that the ℓ_∞ norm also induces grouping of predictors. Due to the high bias of the OWL, [18] proposes a two-stage procedure that involves using the

OWL to identify the correlated predictors and in the second stage fitting an OLS regression.

The OWL/SLOPE regularizer provides many benefits and will be the focus of this thesis. The solution can be computed **efficiently** by applying *proximal methods* and induces the **grouping effect**.

2.5 Gaussian Graphical Models (GGMs)

Probabilistic graphical models determine the conditional independence properties of a set of random variables. Since the structure of graph is unknown, the aim is to estimate it based on a collection $\boldsymbol{X} = \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n\}$ of random variables. Let G be an undirected graph consisting of a vertex set $V = \{1, 2, \ldots, p\}$, and an edge set $E \subset V \times V$.

Definition 18 (Covariance Matrix) *The covariance matrix $\boldsymbol{\Sigma}$ is defined to be:*

$$\boldsymbol{\Sigma} = \begin{bmatrix} \mathbb{V}(\boldsymbol{X}_1) & Cov(\boldsymbol{X}_1, \boldsymbol{X}_2) & \ldots & Cov(\boldsymbol{X}_1, \boldsymbol{X}_p) \\ Cov(\boldsymbol{X}_2, \boldsymbol{X}_1) & \mathbb{V}(\boldsymbol{X}_2) & \ldots & Cov(\boldsymbol{X}_2, \boldsymbol{X}_p) \\ \vdots & \vdots & \vdots & \vdots \\ Cov(\boldsymbol{X}_p, \boldsymbol{X}_1) & Cov(\boldsymbol{X}_p, \boldsymbol{X}_2) & \ldots & \mathbb{V}(\boldsymbol{X}_p) \end{bmatrix}$$

A few properties of $\boldsymbol{\Sigma}$ are:

(a) $\boldsymbol{\Sigma}$ is positive-semidefinite, i.e. $\boldsymbol{v}^T\boldsymbol{\Sigma}\boldsymbol{v} \geq 0$ for all $\boldsymbol{a} \in \mathbb{R}^n$

(b) $\boldsymbol{\Sigma}$ is symmetric.

Given an i.i.d random sample $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n\}$ we define the **sample covariance** to be:

$$\boldsymbol{S} = \frac{1}{n}\sum_{k=1}^{n}(\boldsymbol{X}_k - \hat{\boldsymbol{X}})(\boldsymbol{X}_k - \hat{\boldsymbol{X}})^T$$

where $\hat{\boldsymbol{X}} = \frac{1}{n}\sum_{k=1}^{n} \boldsymbol{X}_k$. The main issue with this estimator is the fact that for $p > n$, the matrix $\boldsymbol{S}$ is singular and thus not invertible.

Definition 19 (Multivariate Gaussian Probability density function) *Let* $\boldsymbol{X} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ *be a Gaussian distribution in p dimensions, with mean vector* $\boldsymbol{\mu} \in \mathbb{R}^p$ *and covariance matrix* $\boldsymbol{\Sigma} \in \mathbb{R}^{p\times p}$*:*

$$\mathbb{P}_{\boldsymbol{\mu},\boldsymbol{\Sigma}}(\boldsymbol{X}) = \frac{1}{(2\pi)^{\frac{p}{2}}\, det[\boldsymbol{\Sigma}]} e^{-\frac{1}{2}(\boldsymbol{X}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{X}-\boldsymbol{\mu})}$$

We consider the Gaussian distribution in terms of its canonical parameters in order to represent it as a Gaussian graphical model. This leads us to only consider the symmetric precision matrix $\boldsymbol{\Theta} \in \mathbb{R}^{p\times p}$. Let $\mathbf{X} = \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n\} \sim N(\mathbf{0}, \boldsymbol{\Sigma})$ represent samples from a zero-mean multivariate Gaussian with precision matrix $\boldsymbol{\Theta} = \boldsymbol{\Sigma}^{-1}$. The zero pattern of $\boldsymbol{\Theta}$ can then be used to determine the graph structure. For instance, if the entry $\boldsymbol{\Theta}_{ij}$ is zero, then variables i and j are considered conditionally independent, given all other variables. We can re-scale the log-likelihood $\mathcal{L}(\boldsymbol{\Theta}; \boldsymbol{X})$ of $\boldsymbol{\Theta}$ (Definition 19) to be defined in the following way:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\Theta}; \boldsymbol{X}) &= \frac{1}{N}\sum_{i=1}^{N} \log \mathbb{P}_{\boldsymbol{\Theta}}(\boldsymbol{X}_i) \\ &= \log \det\boldsymbol{\Theta} - \operatorname{tr}(\boldsymbol{S}\boldsymbol{\Theta})\end{aligned}$$

and

$$\mathbb{P}_{\boldsymbol{\Theta}}(\boldsymbol{X}) = \exp\left\{ -\frac{1}{2}\sum_{s,t=1}^{p} \boldsymbol{\Theta}_{st}\boldsymbol{X}_s\boldsymbol{X}_t + \frac{1}{2}\log\det[\boldsymbol{\Theta}/(2\pi)]\right\}$$

where $\boldsymbol{S} = \frac{1}{n}\boldsymbol{X}\boldsymbol{X}^T$ and $\mathbb{P}_{\boldsymbol{\Theta}}$ is the multivariate Gaussian function parameterized in terms of its canonical parameters: $\boldsymbol{\Theta} \in \mathbb{R}^{p\times p}$ and $\gamma \in \mathbb{R}^p$. The γ term does not appear since we consider a zero-mean multivariate Gaussian function when estimating GGMs. We can consider constrained or regularized forms of the MLE. For instance, we may want to constrain the estimation of the precision matrix $\boldsymbol{\Theta}$ to allow for a model that is easily interpreted. It is well understood that the sparsity pattern of $\boldsymbol{\Theta}$ determines the conditional dependence/independence between two variables [21], [22]. More specifically, if an entry $\boldsymbol{\Theta}_{ij}$ in $\boldsymbol{\Theta}$ is non-zero then an edge exists between the i^{th} and j^{th} variables. It is natural to consider sparsity inducing regularizers discussed in the previous sections to impose less connections on the resulting graph. This creates a graph that is easier to interpret. However, some problems may arise with the MLE problem formulation. For example, the number of nodes p may be larger than the number of observations n causing $\boldsymbol{S}$ to be non-invertible. This would cause the MLE to not exist. This fact serves as further motivation to consider regularized forms of the MLE.

2.6 Proximal Methods

In this section, we will define the *proximal operator* and explain how it is used to solve unconstrained nonsmooth minimization problems. The subproblems that need to be solved in proximal algorithms involve evaluating the proximal operator. Many important problems in the fields of machine learning, signal processing, and statistics can be formulated in *composite form*:

$$\min_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x}) := g(\boldsymbol{x}) + h(\boldsymbol{x}) \tag{2.9}$$

where we assume the following

(a) $h : \mathbb{R}^n \to (-\infty, \infty]$ is proper closed and convex.

(b) $g : \mathbb{R}^n \to (-\infty, \infty]$ is proper and closed, $\mathrm{dom}(g)$ is convex, $\mathrm{dom}(g) \subseteq \mathrm{int}(\mathrm{dom}(g))$, and g is Lipschitz smooth.

(c) The optimal set of problem (3.4) is nonempty and denoted by $\boldsymbol{X}^*$. The optimal value of the problem is denoted by $\hat{\boldsymbol{\beta}}$.

The *proximal operator* of the function h must be possible to evaluate efficiently. In addition, we assume that the optimal value of f, f^*, is achieved at optimal solution $\hat{\boldsymbol{x}}$ which is not necessarily unique.

Definition 20 (Proximal Mapping) *Let $f : \mathcal{X} \to \mathcal{X}$ be a closed convex function. This function can also be referred to as an extended real-value function. The **proximal operator** $prox_f : \mathcal{X} \to \mathcal{X}$ of f is defined by:*

$$prox_f(\boldsymbol{v}) = \arg\min_{\boldsymbol{x}} \left(f(\boldsymbol{x}) + \frac{1}{2}||\boldsymbol{x} - \boldsymbol{v}||_2^2 \right)$$

The function being minimized is a strongly convex function and is not infinite everywhere, thus it has a unique minimizer for every $\boldsymbol{v} \in \mathcal{X}$ [23].

For the remainder of this section we will refer to the "proximal operator" as the "prox operator". The prox operator takes a single vector $\boldsymbol{v} \in \mathcal{X}$ and maps it into a subset of $\mathcal{X}$, which can be empty, a singleton, or a set of multiple vectors in $\mathbb{R}^n$.

Remark 2 *Prox operators can be understood as a projections. For instance, if we let h be the indicator function of a convex set, then $prox_h(\boldsymbol{x})$ is the projection of $\boldsymbol{x}$ onto the set.*

Definition 21 *We will outline a few well studied example of proximal mappings.*

1. *If $f = c$ for $c \in \mathbb{R}$, then*

$$prox_f(\boldsymbol{x}) = \boldsymbol{x}$$

2. *If $f(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{x} \rangle + \boldsymbol{b}$, where $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}$, then*

$$prox_f(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{a}$$

3. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be given by $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T A\boldsymbol{x} + \boldsymbol{b}^T\boldsymbol{x} + c$. The vector $prox_f(\boldsymbol{x})$ is the minimizer of the problem*

$$\min_{u \in \mathbb{R}} \left\{ \frac{1}{2}\boldsymbol{x}^T A\boldsymbol{x} + \boldsymbol{b}^T\boldsymbol{x} + c + \frac{1}{2}||\boldsymbol{u} - \boldsymbol{x}||^2 \right\}$$

The optimal solution is

$$prox_f(\boldsymbol{x}) = (\boldsymbol{A} + \boldsymbol{I})^{-1}(\boldsymbol{x} - \boldsymbol{b})$$

2.6.1 First-order Proximal Methods

One of the most relevant methods for solving the composite model are *first-order methods* and the *accelerated first-order methods* that evaluate the *prox operator* to handle the non-differentiable component of the objective. The former include methods such as spaRSA [24] and TRIP [25]. The latter include the Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) [26].

The proximal method can be thought of as a two-step procedure which minimizes the nonsmooth function h plus a quadratic approximation of the smooth function g every iteration:

$$\begin{aligned}\boldsymbol{x}^{k+1} &= \text{prox}_{t_k h}(\boldsymbol{x}_k - t_k \nabla g(\boldsymbol{x}^k)) \\ \boldsymbol{x}^{k+1} &= \underset{\boldsymbol{y}\in\mathbb{R}^n}{\arg\min}\, \nabla g(\boldsymbol{x}^k)^T(\boldsymbol{y} - \boldsymbol{x}^k) + \frac{1}{L_k}||\boldsymbol{y} - \boldsymbol{x}_k||^2 + h(\boldsymbol{y})\end{aligned}$$

where $L_k \in \mathbb{R}$ is the *step size* at k-th iteration. It can be written more generally as follows:

Algorithm 1 Proximal Gradient Method

```
Require: pick $\boldsymbol{x}^0$ int(dom($f$)), $L_k > 0$
1: repeat
2:     Set $\boldsymbol{x}^{k+1} = \text{prox}_{\frac{1}{L_k}g}(\boldsymbol{x}^k - \frac{1}{L_k}\nabla f(\boldsymbol{x}^k))$
3: until stopping conditions are satisfied
```

Definition 22 *The general update step can be written more compactly and referred to as prox-grad operator defined by*

$$T_L^{f,g}(\boldsymbol{x}) \equiv prox_{\frac{1}{L_k}g}\Big(\boldsymbol{x} - \frac{1}{L}\nabla f(\boldsymbol{x})\Big)$$

where L_k is the step-size.

Theorem 2.6.1 *Suppose the assumptions from* (2.9) *hold. Let $\{\boldsymbol{x}^k\}_{k\geq 0}$ be the sequence generated by the proximal gradient method for solving problem* (2.9) *with a constant stepsize rule in which $L_k \equiv L_f$ for all $k \geq 0$. Then for any $k \geq 0$,*

$$f(\boldsymbol{x}^k) - f(\boldsymbol{x}^*) \leq \frac{\alpha L_f ||\boldsymbol{x}^0 - \boldsymbol{x}^*||_2^2}{2k}$$

where $\boldsymbol{x}^*$ is the solution to (2.9) *and $\alpha = 1$. Thus, the proximal gradient method has an $O(1/k)$ rate of convergence.*

2.6.2 Graphical ISTA (G-ISTA)

In [27], Rolfs et al. proposed a proximal gradient method for precision matrix estimation which possesses attractive theoretical properties as well as a $O(\log \varepsilon)$ iteration complexity to reach a tolerance of ε. The methods described in this section are adapted by defining $g, h : \mathbb{S}^p_{++} \to \mathbb{R}$ which are both continuous convex functions defined over the positive definite cone $\mathbb{S}_{++}$. G-ISTA uses a backtracking line search for determining step size and using the duality gap as a stopping criterion for the algorithm. The details are outlined in Algorithm 2. In the algorithm, $Q_{t_k}(\boldsymbol{\Theta}_{k+1}, \boldsymbol{\Theta}_k)$ is a quadratic approximation of the smooth component of the regularized MLE objective for GGM estimation defined by:

$$\begin{aligned} Q_{t_k}(\boldsymbol{\Theta}_{k+1}, \boldsymbol{\Theta}_k) = & - \log\det \boldsymbol{\Theta} + \operatorname{tr}(\boldsymbol{S}\boldsymbol{\Theta}) + \operatorname{tr}(\boldsymbol{\Theta}_{k+1} - \boldsymbol{\Theta}_k, \mathbf{S} - \boldsymbol{\Theta}_k^{-1}) \\ & + \frac{1}{2t_k} ||\boldsymbol{\Theta}_{k+1} - \boldsymbol{\Theta}_k||_F^2 \, . \end{aligned}$$

In addition, the authors propose an initial estimate of $\boldsymbol{\Theta_0}$ satisfying $[\boldsymbol{\Theta_0}]_{ii} = (\mathbf{S}_{ii} + \lambda)^{-1}$.

Algorithm 2 G-ISTA for Problem (3.1)

```
Require: $\mathbf{S}$, tolerance $\varepsilon$, $\boldsymbol{\lambda}$, $t_0 > 0$, $\boldsymbol{\Theta}_0$, $c \in (0, 1)$
  while $\Delta > \varepsilon$ do
    (1) Let $t_k$ be the largest element of $\{c^j t_{k,0}\}$ so that for
    $\boldsymbol{\Theta}_{k+1} = \text{prox}(\boldsymbol{\Theta}_k - t_k(\mathbf{S} - (\boldsymbol{\Theta}_k^{-1}))$,
    the following are satisfied:
        $\boldsymbol{\Theta}_{k+1} \succ 0 \quad \text{and} \quad -\log\det \boldsymbol{\Theta}_{k+1} + \text{tr}(\boldsymbol{S}\boldsymbol{\Theta}_{k+1}) \leq Q_{t_k}(\boldsymbol{\Theta}_{k+1}, \boldsymbol{\Theta}_k)\,,$
    (2) Compute $\nabla g(\boldsymbol{\Theta}_k) := \boldsymbol{\Theta}_k - t_k(\boldsymbol{S} - \boldsymbol{\Theta}_k^{-1})$.
    (3) Set $\boldsymbol{\Theta}_{k+1} := \text{prox}(\nabla g(\boldsymbol{\Theta}_k))$.
    (4) Compute the duality gap $\Delta$.
Ensure: $\varepsilon$-optimal solution $\hat{\boldsymbol{\Theta}}$.
```

2.7 Summary

In this chapter, we reviewed all terminology necessary to understand our key findings and provided background into related regularization techniques. Well-studied structured sparsity techniques were discussed in detail and strengths and weaknesses were outlined. In the next chapter, we discuss all related work and outline the key state-of-the-art methods found in the literature.

Chapter 3

Literature Review

3.1 Overview

In this thesis, we introduce *variable selection structure learning* methods for Gaussian Graphical models. The methods proposed identify groups of important predictors while also promoting sparsity in the set of edges. In this chapter, we summarize and review important literature by organizing state-of-the-art methods into two categories. The first being the penalized maximum likelihood approaches which penalize the negative Gaussian log-likelihood function with an added regularization term. The second are column-by-column estimation methods which solve several problems introduced by the former likelihood approaches. Keep in mind that there is a vast literature that explores structure learning in Gaussian Graphical models. As a result, since we are only interested in methods that aim to identify related groups amongst variables, we only review methods that directly address this problem. See [28] for a more detailed review of variable selection structure learning methods.

3.2 Penalized Likelihood Methods

Penalized Likelihood methods can be organized into different subclasses. The first subclass are methods which learn a graph given groups *a priori*. This includes [5] which applies a group ℓ_1 penalty to encourage structured sparsity but requires a set of pre-defined hyperparameters to control the topology of the network. The second class consist of methods which find similar groups and then learn the structure of each group. This class includes [7] which proposes a two-step approach to the problem: first applying hierarchical clustering to identify groups and then using the graphical lasso within each group. Devijver et al. (2018) detect groups in the covariance matrix using thresholding and then apply graphical lasso to each group [29]. The last class are methods which learn both group and graph structures simultaneously. Defazio et al. (2012) propose using a non-decreasing, concave function in the penalty to enforce the submodularity of the objective [30]. Hosseini et al. (2016) propose the GRAB estimator that solves a joint optimization problem which alternates between estimating overlapping groups encoded into a Laplacian prior and learning the precision matrix [6].

Throughout this section, let $\mathbf{X}$ be a $n \times p$ design matrix where the rows of $\mathbf{X}$ are denoted as $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n \sim N(\mathbf{0}, \boldsymbol{\Sigma})$ and are assumed to be a collection of i.i.d. p-dimensional random samples. Assume that the columns of the design matrix $\mathbf{X}$ have been standardized to have zero mean and unit variance. Let $\boldsymbol{S}$ denote the sample covariance matrix, defined as $\boldsymbol{S} = n^{-1} \sum_{k=1}^{n} \boldsymbol{X}_k \boldsymbol{X}_k^T$ and let $\boldsymbol{\Theta}$ denote the precision matrix, defined as $\boldsymbol{\Theta} = \boldsymbol{\Sigma}^{-1}$.

3.2.1 Graphical Lasso

One of the most thoroughly studied likelihood methods is the Graphical Lasso which solves a convex ℓ_1 penalized objective function. This work has motivated much of the research into variable selection in Gaussian Graphical models.

Definition 23 (Graphical Lasso [31]) *The graphical lasso has been proposed to estimate $\boldsymbol{\Theta}$ which yields a maximum likelihood estimator given by the following*

$$\min_{\boldsymbol{\Theta} \succ 0} \quad -\log \det \boldsymbol{\Theta} + \operatorname{tr}(\boldsymbol{S}\boldsymbol{\Theta}) + \lambda ||\boldsymbol{\Theta}||_1 \tag{3.1}$$

where $||\boldsymbol{\Theta}||_1 = \sum_{i \neq j} |\boldsymbol{\Theta}_{ij}|$ *and* λ *is a non-negative tuning parameter that controls the level of sparsity in* $\boldsymbol{\Theta}$*.*

The LASSO penalty penalizes off-diagonal entries of $\boldsymbol{\Theta}$. Note that the graphical lasso problem is a non-differentiable convex optimization problem which can be written in the following *composite form*:

$$f(\boldsymbol{\Theta}) = g(\boldsymbol{\Theta}) + h(\boldsymbol{\Theta}) \tag{3.2}$$

where

$$g(\boldsymbol{\Theta}) = -\log\det\boldsymbol{\Theta} - \mathrm{tr}(\boldsymbol{S}\boldsymbol{\Theta}) \tag{3.3}$$

and

$$h(\boldsymbol{\Theta}) = \lambda \sum_{s \neq t} |\boldsymbol{\Theta}_{st}| \tag{3.4}$$

where $\lambda > 0$.

Dual vs. Primal Problem

Primal methods directly solve Problem (3.1), yielding a precision matrix estimate. The dual of this problem is given by:

$$\min_{\boldsymbol{\Sigma}} \quad -\log\det\boldsymbol{\Sigma} \quad s.t. \quad ||\boldsymbol{\Sigma} - \boldsymbol{S}||_\infty \leq \lambda \tag{3.5}$$

Many methods we briefly highlight in Section 3.2 either solve the primal problem or the dual problem. Convergence rates for the dual problem do not necessarily imply the same convergence for the primal problem. Generally, the dual problem is easier to solve than the primal problem. The primal problem begins with a feasible point and seeks to achieve first-order optimality. However, the dual problem begins with an infeasible point that satisfies first-order optimality and then seeks to make it feasible.

3.2.2 The Positive Definite Problem

In the high dimensional setting where $n < p$ the sample covariance of observed data is singular (not invertible) for large sample sizes. However, for many estimators, the positive definiteness of the precision or covariance matrix must be guaranteed empirically. The estimator of interest in our work is the penalized log-likelihood estimator. In addition, if we compare $\boldsymbol{\Theta}$ and $\boldsymbol{\Sigma}$ there are situations where $\boldsymbol{\Theta}$ is sparse but $\boldsymbol{\Sigma}$ is dense indicating their differences in encoding relationships. Although there is an overlap between estimating $\boldsymbol{\Theta}$ and $\boldsymbol{\Sigma}$, the methods outlined in the literature might differ in their goal. Three general techniques can be applied if at some point in an algorithm the matrix estimate is is no longer positive definite:

(a) Incorporating a projection of the estimator onto the set of positive semidefinite matrices, thus formulating the problem as a constrained optimization problem. However, this method may become inefficient as the dimensionality of the problem increases.

(b) Performing an eigenvalue decomposition of the matrix and setting negative eigenvalues to zero. This method continues to be feasible even if the dimension of the problem is high. This is especially true when we assume the matrix is sparse by construction. In this case, many fast methods can be used to perform this decomposition. This method ensures the positive definiteness of the estimator. However, this method is not sparsity preserving and thus may not be viable.

(c) Setting the next iteration $k+1$ of the estimator to be $\hat{\boldsymbol{\Theta}}_{k+1} = \hat{\boldsymbol{\Theta}}_k - \lambda_p(\hat{\boldsymbol{\Theta}}_k)I_p$, where $\lambda_p(\hat{\boldsymbol{\Theta}}_k)$ is the smallest eigenvalue of $\hat{\boldsymbol{\Theta}}_k$. This does not change the sparsity of the matrix because we only add to the diagonal entries of matrix.

Nearest Positive Definite Matrix: Constrained Numerical Optimization

In [32], Higham first proposed a method for estimating the precision matrix by transforming the problem into a constrained optimization problem:

$$\begin{aligned} \hat{\boldsymbol{\Theta}}_{k+1} &= \arg\min_{\boldsymbol{A}} ||\hat{\boldsymbol{\Theta}}_k - \boldsymbol{A}||_F^2 \\ s.t. \quad & \boldsymbol{A} \succ 0, \\ & \text{diag}(\boldsymbol{A}) = \boldsymbol{I}_p \end{aligned} \tag{3.6}$$

then, we can write the following:

$$\hat{\boldsymbol{\Sigma}} = \text{diag}(\boldsymbol{S})^{1/2} \cdot \hat{\boldsymbol{\Theta}}_{k+1} \cdot \text{diag}(\boldsymbol{S})^{1/2}. \tag{3.7}$$

This problem can also be formulated as:

$$\begin{aligned} \hat{\boldsymbol{\Theta}}_{k+1} &= \arg\min_{\boldsymbol{A}} ||\hat{\boldsymbol{\Theta}}_k - \boldsymbol{A}||_F^2 \\ s.t. \quad & \boldsymbol{A} + \boldsymbol{A}^T \succ 0. \end{aligned} \tag{3.8}$$

The problem is convex, meaning that there is a unique global minimizer to the problem. The alternating projections algorithm proposed in [32] only guarantees a linear convergence rate. [33] introduces a Newton-based algorithm with global quadratic convergence. In the literature, this procedure is usually referred to as "nearest correlation matrix projection". The main issue with this method is that the formulation of the problem does not necessarily result in a sparse estimate $\hat{\boldsymbol{\Theta}}$, removing the efficacy of a penalized log-likelihood estimator. To address this issue, [34] introduces a covariance matrix estimation method that preserves sparsity:

$$\hat{\boldsymbol{\Sigma}} = \arg\min_{\lambda_p(\Sigma) \geq \tau} \left\{ ||\boldsymbol{\Sigma} - \boldsymbol{S}||_F^2 + \sum_{i \neq j} P_\lambda(\boldsymbol{\Sigma}_{ij}) \right\} \tag{3.9}$$

where $\tau \geq 0$ is a tuning parameter that controls the smallest eigenvalue of $\boldsymbol{\Sigma}$.

Nearest Positive Definite Matrix: Without Numerical Optimization

In [35], Chan and Wood propose a method to compute the closest positive definite matrix. This method involves the eigendecomposition of $\hat{\mathbf{\Theta}}$. Suppose $\hat{\mathbf{\Theta}}$ is a symmetric non-positive definite matrix. We can write $\hat{\mathbf{\Theta}}$ as follows:

$$\begin{aligned}\hat{\mathbf{\Theta}} &= Q\Lambda Q^T \\ &= Q\begin{bmatrix}\lambda_1 & & \\ & \ddots & \\ & & \lambda_p\end{bmatrix}Q^T \\ &= \sum_{i=1}^{p}\lambda_i q_i q_i^T\end{aligned} \tag{3.10}$$

where $\lambda_1, \ldots, \lambda_p$ are the eigenvalues of $\hat{\mathbf{\Theta}}$ and q_i are the corresponding eigenvectors. Next, choose a matrix $\rho\rho^T$ where:

$$\rho = tr(\Lambda)/tr(\Lambda_+) \tag{3.11}$$

or

$$\rho = \sqrt{tr(\Lambda)/tr(\Lambda_+)} \tag{3.12}$$

where $[\Lambda_+] = \max\{\Lambda_{ii}, 0\}$ or simply setting any negative eigenvalue to 0. Then the positive definite $\hat{\mathbf{\Theta}}_{PD}$ matrix can be written as:

$$\hat{\mathbf{\Theta}}_{PD} = (\rho\rho^T)\hat{\mathbf{\Theta}} \tag{3.13}$$

In a similar way, [36] defines $\hat{\mathbf{\Theta}}_{PD}$ to be:

$$\hat{\mathbf{\Theta}}_{PD} = \hat{\mathbf{\Theta}} - \lambda_{\min}(\hat{\mathbf{\Theta}})I_p \cdot I\{\lambda_p < 0\} \tag{3.14}$$

where $\lambda_{\min}(\hat{\mathbf{\Theta}})$ is the smallest eigenvalue of $\hat{\mathbf{\Theta}}$ and $I\{\lambda_p < 0\}$ is the indicator function.

3.2.3 Projected subgradient methods for learning sparse Gaussian Graphical Models

In [5], Duchi et al. propose a projected subgradient method for imposing sparsity on entire blocks of the precision matrix. The problem is formulated as a penalized likelihood problem and considers a regularizer that penalizes groups of edges. Duchi et al. consider a popular problem in computational biology that involves penalizing interactions between genes in multiple pathways.

Definition 24 *The projected subgradient estimator is defined by the following:*

$$\underset{\mathbf{\Theta} \succ 0}{\arg\min} \quad -\log\det\mathbf{\Theta} + \operatorname{tr}(\boldsymbol{S}\mathbf{\Theta}) + \sum_{i,j} \lambda_{ij} |\mathbf{\Theta}_{ij}| \tag{3.15}$$

where λ_{ij} are a collection of tuning parameters that control the level of sparsity in $\mathbf{\Theta}$.

Duchi et al. prove that as long as $\lambda_{ij} > 0 \quad \forall \quad i \neq j$ and $\lambda_{ii} \geq 0$ the solution to (3.15) is unique and positive definite. The dual formulation of the problem is:

$$\begin{aligned} \max_{\mathbf{W}} \quad & \log\det(\mathbf{S} + \mathbf{W}) \\ \text{s.t.} \quad & |\mathbf{W}_{ij}| \leq \lambda_{ij} \quad \forall i, j \\ & \mathbf{S} + \mathbf{W} \succ 0 \end{aligned} \tag{3.16}$$

By analyzing the duality gap, it can be shown that strong duality holds between the dual and primal problems [5]. Duchi et al. then show that dual problem has a unique solution implying the primal problem does as well. The results presented in [5] demonstrate the efficacy of the approach for synthetic and real datasets and suggest that the algorithm is more efficient than previous state-of-the-art methods.

3.2.4 A Convex Formulation for Learning Scale-Free Networks via Submodular Relaxation

Defazio et al. considers a structured sparsity inducing prior to determine the structure of a network [30]. A regularizer was proposed that behaves like an ℓ_1 norm but imposes additional penalization for larger edge weights.

Definition 25 *The submodular relaxation estimator is defined by the following:*

$$\min_{\boldsymbol{\Theta} \succ 0} \quad -\log\det\boldsymbol{\Theta} + \operatorname{tr}(\boldsymbol{S}\boldsymbol{\Theta}) + \sum_{i=0}^{p}\sum_{k=0}^{n-1}(h(k+1) - h(k))|\boldsymbol{\Theta}_{i,(k)}| \tag{3.17}$$

where $\boldsymbol{\Theta}_{i,(j)}$ denotes the weight of the jth edge connected to i, under a decreasing ordering by absolute value (i.e. $|\boldsymbol{\Theta}_{i,(0)}| \geq \cdots \geq |\boldsymbol{\Theta}_{i,(p)}|$) and h is non-decreasing, concave and $h(0) = 0$.

Defazio et al. use the alternating direction of multipliers (ADMM) algorithm to solve their optimization problem instead of using standard proximal methods. The results presented in [30] show the superior recovery of synthetic and real networks over the standard graphical lasso algorithm.

3.2.5 A Unified Framework for Structured Graph Learning via Spectral Constraints

Kumar et al. introduce a learning framework that combines GGMs and spectral graph theory by imposing spectral constraints on the precision matrix [37]. As we have seen in previous sections GGMs can be represented by a matrix where nonzero entries denote edges.

Definition 26 *A matrix $\boldsymbol{\Theta} \in \mathbb{R}^{p \times p}$ is called Laplacian if it belongs to $\mathcal{S}_{\boldsymbol{\Theta}}$, the set of matrices defined as*

$$\mathcal{S}_{\boldsymbol{\Theta}} = \Big\{\boldsymbol{\Theta} | \boldsymbol{\Theta}_{ij} = \boldsymbol{\Theta}_{ji} \leq 0 \textit{ for } i \neq j; \boldsymbol{\Theta}_{ii} = -\sum_{j \neq i} \boldsymbol{\Theta}_{ij}\Big\}. \tag{3.18}$$

Thus, a Laplacian matrix is diagonally dominant, positive semidefinite, has zero row sum and zero column sum, and has non-positive off-diagonal entries.

Definition 27 *The spectral constraint estimator is defined by the following:*

$$\begin{aligned}\max_{\boldsymbol{\Theta},\boldsymbol{\lambda},\boldsymbol{U}} \quad & \log\det(\boldsymbol{\Theta}) - \operatorname{tr}(\boldsymbol{\Theta}\mathbf{S}) - \Omega(\boldsymbol{\Theta}) \\ s.t.\ & \boldsymbol{\Theta} \in \mathcal{S}_{\boldsymbol{\Theta}}, \boldsymbol{\Theta} = \mathbf{U}\operatorname{diag}(\boldsymbol{\lambda})\mathbf{U}^T, \boldsymbol{\lambda}^T\boldsymbol{U} = \boldsymbol{I}\end{aligned} \tag{3.19}$$

where $\boldsymbol{\lambda} \in \{\lambda_i\}_{i=1}^p$ *and* $\boldsymbol{U} \in \mathbb{R}^{p\times p}$.

Various choices for $\boldsymbol{\lambda}$ are proposed and a custom algorithm based on quadratic methods was employed to solve (3.19).

3.2.6 Cluster Graphical Lasso (CGL)

In [7], Tan et al. propose an algorithm that connects the graphical lasso algorithm to single linkage clustering (SLC). The algorithm CGL involves clustering features using SLC, and then performing the graphical lasso on the subset of variables in each cluster. Let $\tilde{\boldsymbol{S}}$ denote a $p \times p$ matrix whose elements take the form $\tilde{S}_{jj'} = |\mathbf{X}_j^T\mathbf{X}_{j'}|/n = |S_{jj'}|$ where X_j is the j column of $\mathbf{X}$. Let $C_1, \ldots, C_K$ be the clusters obtained by performing a clustering method of choice based on the similarity matrix $\tilde{\boldsymbol{S}}$. The k^{th} cluster contains $|C_k|$ features. Then the graphical lasso is performed on each feature. The algorithm can be interpreted as the following penalized log-likelihood problem:

$$\hat{\boldsymbol{\Theta}} = \arg\min_{\boldsymbol{\Theta}\succ 0} \quad \operatorname{tr}(\boldsymbol{S}\boldsymbol{\Theta}) - \log\det(\boldsymbol{\Theta}) + \sum_{j\neq j'} w_{jj'}|\boldsymbol{\Theta}_{jj'}| \tag{3.20}$$

where

$$w_{jj'} = \begin{cases} \lambda_k & \text{if } j, j' \in C_K \\ \infty & \text{if } j \in C_k, j' \in C_{k'}, k \neq k' \end{cases} \tag{3.21}$$

where $\lambda_k > 0$ for $k = 1, 2, \ldots, K$. This indicates that this method imposes a large penalty if the jth and j'th features are in different clusters. The method is applied to synthetic data where block diagonal precision matrices containing two blocks are generated. The variables are standardized to have mean zero and variance one. The method is applied to equities data, web page data, and gene expression data. The task for the equities dataset is to identify the Global Industry Classification Standard (GICS) sectors. The CGL and Graphical Lasso methods are compared to conclude that the CGL clusters more effectively.

3.2.7 Graphical models with overlapping blocks (GRAB)

In [6], Hosseini et al. propose an algorithm (GRAB) to learn the densely connected components of a graph which are referred to as "blocks". GRAB encourages the nodes in the network to be densely connected within a block and at the same time, sparse between blocks. GRAB requires no prior information about the blocks and as well, blocks can also overlap. The GRAB estimator is compared to three state-of-the-art competitors. The results in [6] clearly outline the graphical lasso's failure to detect such "blocks" demonstrating how this structure is extremely prevalent in gene expression data. In addition, the results illustrate how GRAB outperforms CGL in identifying the block structure.

The GRAB regularizer is formulated as $\mathrm{tr}(\boldsymbol{Z}\boldsymbol{Z}^T||\boldsymbol{\Theta}||_1)$ which encourages subsets of variables to be densely connected in the network estimate. Let $\boldsymbol{Z}$ be a real matrix of size $p \times K$, where K is the total number of blocks. Each element i, j in $\boldsymbol{Z}\boldsymbol{Z}^T$ measures the similarity of the i^{th} and j^{th} variables as a score of how likely the variables belong to the kth block. The GRAB algorithm jointly learns both $\boldsymbol{Z}$ and $\boldsymbol{\Theta}$. The objective function for GRAB is defined as follows:

$$\hat{\boldsymbol{\Theta}} = \underset{\boldsymbol{\Theta}\succeq 0, Z\in\mathcal{D}}{\arg\min}\, \mathrm{tr}(\boldsymbol{S}\boldsymbol{\Theta}) - \log\det(\boldsymbol{\Theta}) + \sum_{i,j} \lambda\Big(1 - (\boldsymbol{Z}\boldsymbol{Z}^T)_{ij}\Big)|\Theta_{ij}| \tag{3.22}$$

where λ is a non-negative tuning parameter and $\mathcal{D} \subset [-1,1]^{p\times K}$ contains the matrices $\boldsymbol{Z}$ satisfying the following constraints:

(a) $||Z_i||_2 \leq 1$ where Z_i denotes the ith row of $\boldsymbol{Z}$.

(b) $||\boldsymbol{Z}||_F \leq \beta$, where $\beta > 0$

(c) $||\boldsymbol{Z}||_2 \leq \tau$, where $\tau > 0$

Three state-of-the-art methods were compared to GRAB: UGL1 - unknown group ℓ_1 regularization [38] , CGL - cluster graphical lasso [7] , and GLASSO - standard graphical lasso [31]. For the synthetic dataset, K overlapping blocks were generated forming a chain, a random tree or lattice. Every two neighbouring blocks overlap each other by a ratio o. The GRAB estimator was applied to the MILE dataset [39] that measures the mRNA expression levels of 16,853 genes in 541 patients with acute myeloid leukemia (AML), an aggressive blood cancer. It was concluded that GRAB was highly effective in identifying overlapping gene pathways that are connected to the progression of both diseases.

3.3 Column-by-Column Estimation Methods

Estimating the precision matrix column-by-column has received much attention since this approach can lead to methods that are numerically simpler and more accommodating to theoretical analysis [2], [3]. Under the same assumptions outlined in Definition 23, [3] showed that $\boldsymbol{\Theta}$ can be estimated through column-by-column linear regressions.

3.3.1 The column-by-column Graphical LASSO

[3] takes a simpler approach than [31] and fits a lasso model for each variable using the remaining variables as predictors. Let $\mathbf{x_i} \sim N(\mathbf{0}, \boldsymbol{\Sigma})$ for $i = 1, \ldots, n$, the

conditional distribution of x_i given $\mathbf{x}_{-i}$ satisfies:

$$x_j|\mathbf{x}_{-j} \sim N(\boldsymbol{\beta}_j^T\mathbf{x}_{-j}, \sigma_j^2)\,, \tag{3.23}$$

where $\boldsymbol{\beta} = (\boldsymbol{\Sigma}_{-j,-j})^{-1}\boldsymbol{\Sigma}_{-j,j} \in \mathbb{R}^{p-1}$ and $\sigma_j^2 = \boldsymbol{\Sigma}_{j,j} - \boldsymbol{\Sigma}_{j,-j}(\boldsymbol{\Sigma}_{-j,-j})\boldsymbol{\Sigma}_{-j,j}$. Thus, we can write the following:

$$x_j = \boldsymbol{\beta}_j^T\mathbf{x}_{-j} + \varepsilon_j\,, \tag{3.24}$$

where $\varepsilon_j \sim N(0, \sigma_j^2)$. By the block matrix inversion formula we define:

$$\boldsymbol{\Theta}_{jj} = \sigma_j^{-2}, \quad \text{and} \quad \boldsymbol{\Theta}_{-j,j} = -\sigma_j^{-2}\boldsymbol{\beta}_j\,. \tag{3.25}$$

Thus, we can estimate the precision matrix column-by-column by regressing x_j on $\mathbf{x}_{-j}$, and a LASSO procedure can be adopted by solving:

$$\hat{\boldsymbol{\beta}}_j = \underset{\boldsymbol{\beta}_j \in \mathbb{R}^{p-1}}{\arg\min}\, ||\mathbf{X}_{*,j} - \mathbf{X}_{*,-j}\boldsymbol{\beta}_j||_2^2 + \lambda||\boldsymbol{\beta}_j||_1\,. \tag{3.26}$$

The variance $\hat{\sigma}^2$ can be estimated in a similar way:

$$\hat{\sigma}^2 = ||\mathbf{X}_{*,j} - \mathbf{X}_{*,-j}\boldsymbol{\beta}_j||_2^2 \tag{3.27}$$

3.3.2 CLIME and SCIO

Other column-by-column estimators include: constrained ℓ_1-minimization for inverse matrix estimation (CLIME) and sparse column-wise inverse operator (SCIO). The CLIME method obtains the precision matrix, $\boldsymbol{\Theta}$, one column at a time by solving a linear program and then combines all column vectors to obtain a final estimate of $\boldsymbol{\Theta}$.

Definition 28 (CLIME) *For an i.i.d. sample of n, p-dimensional random variables $\{\boldsymbol{X_1}, \ldots, \boldsymbol{X_n}\}$ each with covariance matrix $\boldsymbol{\Sigma}$, the covariance matrix may be* *estimated as follows:*

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{n}\sum_{k=1}^{n}(\boldsymbol{X_k} - \bar{\boldsymbol{X}})(\boldsymbol{X_k} - \bar{\boldsymbol{X}})^T \tag{3.28}$$

where $\bar{\boldsymbol{X}} = \frac{1}{n}\sum_{k=1}^{n}\boldsymbol{X_k}$*. The CLIME estimator is defined as follows:*

$$\begin{aligned} &\min_{\hat{\boldsymbol{\Theta}} \in \mathbb{R}^{p \times p}} ||\hat{\boldsymbol{\Theta}}||_1 \\ &s.t. |\hat{\boldsymbol{\Sigma}}\hat{\boldsymbol{\Theta}} - \boldsymbol{I}|_\infty \leq \lambda_n \end{aligned} \tag{3.29}$$

Numerical performance of CLIME was compared to the graphical lasso and the graphical smoothly clipped absolute deviation (SCAD) estimators, using a synthetic dataset. The results demonstrate superior convergence, both theoretically and empirically. In [40], Lui and Luo propose the SCIO estimator that adds an ℓ_1 penalty to a column loss function, which is similar to the CLIME estimator. By doing this, it decomposes the problem into the following smaller problems.

Definition 29 (SCIO) *For precison matrix estimate* $\hat{\boldsymbol{\Theta}}$ *and sample covariance matrix* $\boldsymbol{S}$*. The SCIO is defined as follows:*

$$\min_{\hat{\boldsymbol{\Theta}}_{*i} \in \mathbb{R}^p} \{\frac{1}{2}\hat{\boldsymbol{\Theta}}_{*i}^T \boldsymbol{S}\hat{\boldsymbol{\Theta}}_{*i} - \boldsymbol{e_i}^T\hat{\boldsymbol{\Theta}}_{*i} + \lambda_i||\hat{\boldsymbol{\Theta}}_{*i}||_1\} \quad i = 1, 2, \ldots, p \tag{3.30}$$

where $|| \cdot ||_1$ *is the element-wise* ℓ_1 *norm,* $\boldsymbol{e_i}$ *is the i-th column of the* $p \times p$ *identity matrix and the non-negative tuning parameter,* λ_i*, can be different from column to column.*

The precision matrix estimate $\hat{\boldsymbol{\Theta}}$ can then be formulated where each column corresponds to $\hat{\boldsymbol{\Theta}}_{*i}$. This then requires an additional symmetrization step.

3.4 Discussion

Each of the various techniques presented in this chapter offer different advantages depending on whether your aim is speed, flexibility, or handling group structure.

Before we address where each technique excels, we will review traditional methods such as the Graphical Lasso. Most methods discussed in this chapter are based on the Graphical Lasso. The GLASSO was one of the first techniques to estimate sparse graphs using ℓ_1 penalization and proposed a coordinate descent procedure which was considered computationally efficient at the time it was introduced. It is well known that the GLASSO does not identify group structure as well as other techniques, especially techniques discussed in this chapter.

The techniques that offer the greatest speed when computing the precision matrix estimate are the column-by-column estimation techniques. This increase in speed is attributed to the absence of several matrix inversions required by likelihood methods. In addition, another advantage is that each column can be estimated individually by solving a simple linear program which can be easily scaled for larger datasets. Finally, once all columns have been estimated, an efficient symmetrization step is required to arrive at the final estimate. Another disadvantage of likelihood methods is that the feasible set of the log-determinant objective required by all likelihood methods is very complicated as a result of positive definiteness being required at every iteration. In terms of handling group structure, to our knowledge there has been no column-by-column techniques whose primary aim was to identify such structure.

The technique that is most effective at identifying group structure is the GRAB estimator. The GRAB estimator offers simple encoding for the groups encoded in the sparsity pattern of $\mathbf{Z}\mathbf{Z}^T$ and jointly learns $\mathbf{Z}$ and $\mathbf{\Theta}$. The main downfall of GRAB is speed since it requires many inversions, an alternating minimization procedure, and requires a k-means clustering algorithm in order to obtain the blocks from $\mathbf{Z}$. As illustrated in the chapter, there does not exist a technique which offers speed when identifying group structure in Gaussian Graphical models.

3.5 Summary

In this chapter, we reviewed methods that seek to identify structure in GGMs. We pointed out two main methodologies for formulating the problem providing summaries of state-of-the-art methods in each of them. We also provided an overview of the problem formulations and algorithms for achieving an optimal solution.

Chapter 4

Graphical Order-Weighted ℓ_1 Estimators

4.1 Overview

In this chapter, we present two novel estimators for variable selection structure learning in Gaussian Graphical models. We will present two different approaches for discovering the topology of a Gaussian Graphical model under the assumption that the data exhibits additional structure other than sparsity among edges. Common challenges for this problem are the large number of possible structures present in the data and finding appropriate regularization techniques to reduce over-fitting. First, we describe the GOWL estimator: a penalized likelihood method that takes inspiration from previous methods such as the GLASSO and can be solved efficiently using proximal methods. Next, we describe the ccGOWL estimator that uses a column-by-column estimation approach and applies the OWL penalizer to the columns of the precision matrix estimate. We outline the differences between these methods and show desirable theoretical properties for both methods.

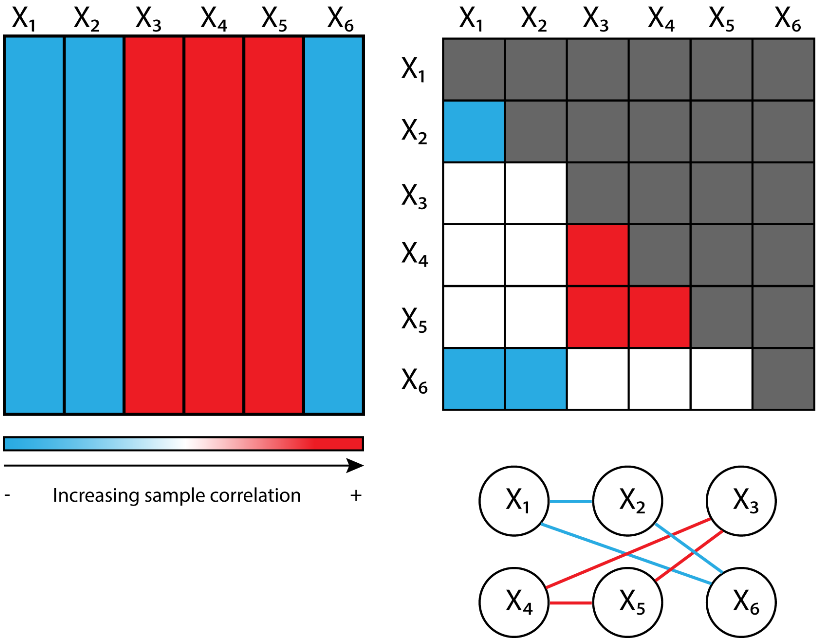
X_1 X_2 X_3 X_4 X_5 X_6
X_1 X_2 X_3 X_4 X_5 X_6
X_1 X_2 X_3 X_4 X_5 X_6
-
Increasing sample correlation
+
X_1 X_2 X_3 X_4 X_5 X_6

Figure 4.1: A design matrix, precision matrix, and network with two groups of edges that share similar sample partial correlation.

4.2 Motivation

The proposed GOWL and ccGOWL estimators are based on a different notion of groups than the algorithms previously discussed. Prior methods consider groups or blocks of *variables*. Structural priors then encourage group (block) sparsity — an edge in the graphical model (a non-zero precision matrix entry) is considered as more likely if it connects variables that belong to the same group. In contrast, the GOWL estimator introduces a structural prior that encourages grouping of *edges* in the graphical model. The GOWL estimator favours precision matrix estimates where multiple entries have the same value, i.e., the pairs of variables are estimated to have the same partial correlation. The GOWL structural prior thus models scenarios where the relationships between multiple pairs of variables are likely to have the same strength, with the relationships arising from a common cause or being impacted by a common factor. As an example, consider three mining companies which focus on different metals; we might a priori expect the partial correlations between the stock prices and the specific metal prices to be of the same magnitude. The ccGOWL estimator is similar, but only encourages grouping of edges that have one variable in common. For this case, consider an example of a single company that devotes equal resources to mining three metals; we might model the partial correlations between the company's stock price and the three metals to be equal.

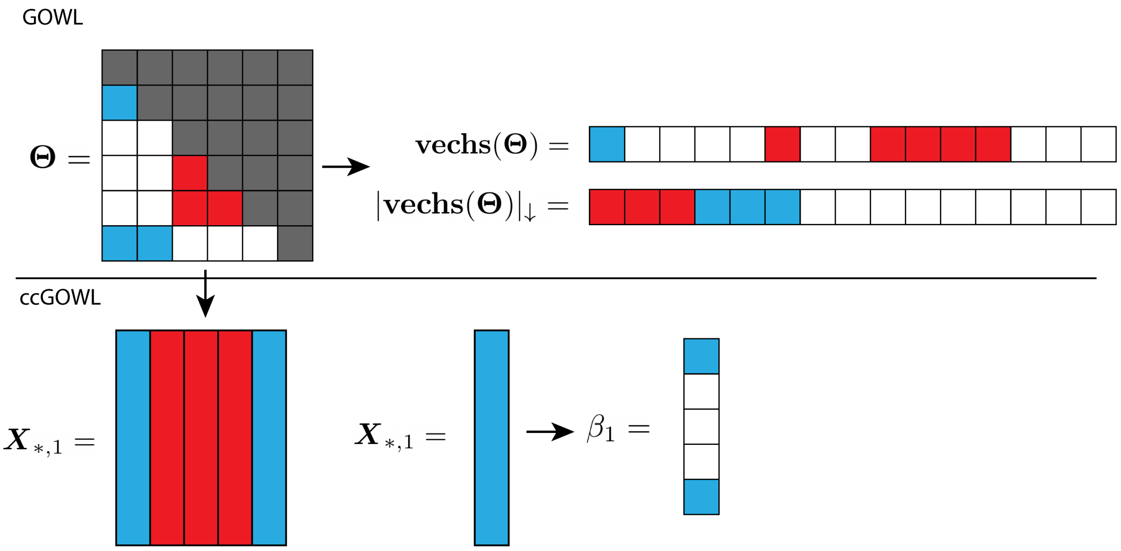

Figure 4.2: The GOWL and ccGOWL frameworks—likelihood and column-by-column OWL penalized estimation procedures.

Figure 4.1 illustrates the relationship between the design matrix, precision matrix, and the resulting penalized GGM. GOWL penalizes certain groups of edges all together while ccGOWL penalizes edges for each node separately. The goal is to promote groups of related edges taking the same value while penalizing less relevant relationships between variables.

Figure 4.2 depicts the different ways that each estimator applies the OWL penalty. The GOWL estimator applies the penalty to vectorized lower triangle of the precision matrix and the ccGOWL estimator applies the OWL to the shorter columns of the precision matrix separately.

4.3 Graphical OWL

In this section, we describe the theoretical formulation of the graphical OWL (GOWL) estimator, prove some of its theoretical properties and describe the estimation algorithm.

4.3.1 Formulation of GOWL

The GOWL is defined by the following:

$$\min_{\boldsymbol{\Theta} \succ 0} \quad -\log\det \boldsymbol{\Theta} + \operatorname{tr}(\boldsymbol{S}\boldsymbol{\Theta}) + \Omega_{\text{OWL}}(\boldsymbol{\Theta}) \tag{4.1}$$

where

$$\Omega_{\text{OWL}}(\boldsymbol{\Theta}) = \boldsymbol{\lambda}^T |\mathbf{vechs}(\boldsymbol{\Theta})|_{\downarrow} = \sum_{i=1}^{K} \lambda_i |\mathbf{vechs}(\boldsymbol{\Theta})|_{[i]} \,. \tag{4.2}$$

Here $\boldsymbol{\Theta} \in \mathbb{R}^{p \times p}$, $\mathbf{vechs}(\boldsymbol{\Theta})_{[i]}$ is the i^{th} largest off-diagonal component in magnitude of $\boldsymbol{\Theta}$, $K = (p^2 - p)/2$, and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p \geq 0$. The proximal mapping of Ω_{OWL} denoted by $\text{prox}_{\Omega_{\text{OWL}}} : \mathbb{R}^{p \times p} \mapsto \mathbb{R}^{p \times p}$ can be efficiently computed with $O(n \log n)$ complexity using the *pool adjacent violators* (PAV) algorithm for isotonic regression. A detailed explanation of the derivation can be found in [41].

4.3.2 Dual Problem

In this section, we derive a dual formulation of GOWL. The dual formulation plays an important role for implementing efficient algorithms and allows us to establish uniqueness of the solution of GOWL. The following formulation uses a similar approach introduced in [5]. However, [5] applies a group LASSO penalty whereas the OWL penalty is a sorted ℓ_1 norm and required the use of the generalized rearrangement inequality [42] in order to derive the dual formulation of the problem. Let

$\mathbf{Z} = \mathbf{\Theta}$ and its associated dual variable be $\mathbf{W} \in \mathbb{R}^{p \times p}$, which gives the Lagrangian:

$$\mathcal{L}(\mathbf{\Theta}, \mathbf{Z}, \mathbf{W}) = -\log\det\, \mathbf{\Theta} + \operatorname{tr}(\boldsymbol{S}\mathbf{\Theta}) + \sum_{i=1}^{K} \lambda_i |\mathbf{vechs}(\mathbf{Z})|_{[i]} + \operatorname{tr}(\mathbf{W}(\mathbf{\Theta} - \mathbf{Z})), \tag{4.3}$$

Note that the above quantity is separable into terms involving $\mathbf{\Theta}$ and $\mathbf{Z}$. We first consider the terms involving $\mathbf{Z}$ or more precisely, its vectorized form $\mathbf{z} = \mathbf{vechs}(\mathbf{Z})$:

$$g(\mathbf{z}) = \lambda_{\downarrow}^T \mathbf{z}_{\downarrow} - \operatorname{tr}(\mathbf{W}\mathbf{Z})\,. \tag{4.4}$$

By the generalized rearrangement inequality [42], we know that for arbitrary vectors $\lambda, \mathbf{w}$, $\lambda_{\downarrow}^T \mathbf{w}_{\uparrow} \leq \lambda^T \mathbf{w} \leq \lambda_{\downarrow}^T \mathbf{w}_{\downarrow}$ and hence if $\lambda_i \geq |\mathbf{w}_i|$ for $i = 1, ..., K$ and $\mathbf{w} = \mathbf{vechs}(\mathbf{W})$, then $\inf_{\mathbf{z}} g(\mathbf{z}) = 0$. Otherwise, the problem is unbounded and the infimum is attained at $g(\mathbf{z}) = -\infty$. Therefore

$$\inf_{\mathbf{Z}} \sum_{i=1}^{K} \lambda_i |\mathbf{Z}_{[i]}| - \operatorname{tr}(\boldsymbol{W}\mathbf{Z}) = \begin{cases} 0 & \text{if } \lambda_i \geq |\mathbf{w}_i| \text{ for } i = 1, ..., K, \\ -\infty & \text{otherwise}\,. \end{cases} \tag{4.5}$$

Recall that the gradient of the log determinant is $\nabla_{\mathbf{\Theta}} \log\det(\mathbf{\Theta}) = \mathbf{\Theta}^{-1}$. The infimum over terms involving $\mathbf{\Theta}$ can be computed by setting $\nabla_{\mathbf{\Theta}} \log\det(\mathbf{\Theta})$ to zero and solving for $\mathbf{\Theta}$ under the assumption that $\mathbf{S} + \mathbf{W} \succ 0$:

$$-\mathbf{\Theta}^{-1} + (\mathbf{S} + \mathbf{W}) = 0 \tag{4.6}$$

$$\mathbf{\Theta}^{-1} = (\mathbf{S} + \mathbf{W}) \tag{4.7}$$

This yields the following

$$\inf_{\mathbf{\Theta}} [-\log\det\mathbf{\Theta} + \operatorname{tr}((\mathbf{S} + \mathbf{W})\mathbf{\Theta})] = \log\det\,(\mathbf{S} + \mathbf{W}) + p\,. \tag{4.8}$$

Combining (4.3) with (4.5) allows us to write the dual as:

$$\begin{aligned} \max_{\mathbf{W} \succ 0} \quad & \log\det\,(\mathbf{S}+\mathbf{W}) \\ \text{s.t.} \quad & |\mathbf{w}_i| \leq \lambda_i, \\ & (\mathbf{S}+\mathbf{W}) \succ 0\,. \end{aligned} \tag{4.9}$$

We proceed to solve the problem shown in (4.9) using G-ISTA, which requires the computation of the duality gap $\Delta = p^* - d^*$ for primal p^* and dual d^*. If we define the feasible set $B_\lambda = \{\mathbf{W} : |\mathbf{w}_i| \leq \lambda_i, \forall i, \mathbf{W} \in \mathbb{S}^p_{++}\}$, then for any feasible dual point $\mathbf{W} \in B_\lambda$ and corresponding primal point $\mathbf{\Theta} = (\mathbf{S}+\mathbf{W})^{-1}$, the duality gap Δ is

$$\begin{aligned} \Delta &= p^* - d^* \qquad (4.10)\\ &= \operatorname{tr}(\mathbf{S\Theta}) + \sum_{i=1}^{K} \lambda_i |\mathbf{\Theta}|_{[i]} - p\,. \end{aligned}$$

In practice, Δ is estimated using the difference between the primal and the dual and acts as a stopping criterion for the iterative procedure.

4.3.3 Uniqueness of GOWL

It is often desirable to prove that the solution to an optimization problem is unique. In other words, showing that an objective function is strictly convex shows that the function has one global minimum. We show that the GOWL problem has a unique solution under certain assumptions and build on the approach used in [5]. We aim to prove the following:

Theorem 4.3.1 *If $\lambda_i > 0$ for all i and the diagonal entries of* $\mathbf{S}$ *are greater than zero, then problem* (4.1) *has a unique optimal point* $\mathbf{\Theta}^*$. *Note that the diagonal entries of* $\mathbf{S}$ *are greater than 0 with probability 1.*

We prove Theorem 4.3.1 by building upon Slater's constraint qualification, as stated in Definition 5. First, assume that $c = \max_{i,j} |\mathbf{S}_{ij}|$ is known. In the case where $\mathbf{S}$ is standardized, $c = 1$. The negative log-likelihood is convex in the precision matrix and is defined over the set of all positive-definite matrices. On the other hand, the GOWL estimator minimizes an objective function which is also convex in $\mathbf{\Theta}$ over the same set. Since the sum of two convex functions over the same convex set is convex, we conclude that the main objective is convex.

Using SCQ, we can say that the duality gap is zero and write the primal-dual optimal pair in the following way:

$$\mathbf{\Theta}^* = (\mathbf{S} + \mathbf{W}^*)^{-1} . \tag{4.11}$$

For SCQ to hold, it remains to show that there exists a point $\mathbf{W}$ in the interior of the feasible set given by interior$(B_\lambda) = \{\mathbf{W} : |\mathbf{w}_i| < \lambda_i, \forall i, \mathbf{W} \in \mathbb{R}^p\}$ such that it is the solution of (4.5).

First recall that $\mathbf{S}$ is a symmetric positive semi-definite matrix. Since the sample covariance $\mathbf{S}$ is estimated from data, its diagonal entries are greater than zero with probability one. In other words, any diagonal entry of $\mathbf{S}$ will be zero if the predictor takes on a single value for all observations, that is, the predictor has zero variance. Since predictors are assumed to be distributed according to a continuous probability density having all samples to be the same for a given variable occurs with probability zero. Let $\mathbf{A} = \text{diag}(\mathbf{S}) \succ 0$ since the determinant of a diagonal matrix with all positive entries will be positive. Consequently, by Sylvester's criterion $\mathbf{A}$ is positive definite (PD). We can then write the convex combination of $\mathbf{S}$ and $\mathbf{A}$ as

$$\alpha \mathbf{S} + (1 - \alpha)\mathbf{A} \succ 0 . \tag{4.12}$$

where $\alpha \in [0, 1)$. Thus, we can write

$$\mathbf{S} + \mathbf{W} = \alpha \mathbf{S} + (1 - \alpha)\mathbf{A} \succ 0, \tag{4.13}$$

for non-negative α strictly smaller than 1. For a given matrix of hyperparameters $\mathbf{\Lambda}$, pick $\alpha > 1 - \frac{1}{c}\min_{kl}\mathbf{\Lambda}_{kl}$. In practice this can be achieved by setting $\tilde{\alpha} = 1 - 1/c\min_{kl}\mathbf{\Lambda}_{kl}$ and putting $\alpha = \tilde{\alpha} + \varepsilon$ for some $\varepsilon > 0$. Then,

$$\begin{aligned}\mathbf{W} &= \alpha\mathbf{S} + (1-\alpha)\mathbf{A} - \mathbf{S}\\ &= (1-\alpha)(\mathbf{A} - \mathbf{S}),\end{aligned} \tag{4.14}$$

and hence

$$\begin{aligned}|\mathbf{W}|_{ij} &= (1-\alpha)|\mathbf{S}|_{ij} \qquad (i,j = 1,\dots,p),\\ &< \frac{1}{c}\min_k \lambda_k|\mathbf{S}_{ij}|,\\ &\leq \min_k \lambda_k,\\ &\leq \lambda_{ij},\end{aligned} \tag{4.15}$$

where we used the fact that the values in the empirical estimate of the covariance matrix $\mathbf{S}$ cannot be infinite. By the convexity of the primal objective and SCQ, we conclude that $\mathbf{W}^*$ is unique. Furthermore, since the duality gap is zero at the point $\mathbf{W}^*$, the uniqueness of $\mathbf{W}^*$ implies the uniqueness of $\mathbf{\Theta}^*$. ■

4.3.4 GOWL Algorithm

In this section, we present an algorithm for solving the GOWL optimization problem of (4.1). Our work builds on G-ISTA discussed in Section 2.6.2. The detailed algorithm is presented in Algorithm 3. In this algorithm, $Q_{t_k}(\mathbf{\Theta}_{k+1}, \mathbf{\Theta}_k)$ is defined by the following equation:

$$\begin{aligned}Q_{t_k}(\mathbf{\Theta}_{k+1}, \mathbf{\Theta}_k) = &-\log\det\,\mathbf{\Theta} + \operatorname{tr}(\boldsymbol{S}\mathbf{\Theta}) + \operatorname{tr}(\mathbf{\Theta}_{k+1} - \mathbf{\Theta}_k, \mathbf{S} - \mathbf{\Theta}_k^{-1})\\ &+ \frac{1}{2t_k}||\mathbf{\Theta}_{k+1} - \mathbf{\Theta}_k||_F^2,\end{aligned} \tag{4.16}$$

where t_k is the step size at iteration k. Algorithm 3 uses a backtracking step size defined by (4.17) for the choice of step size. A suitable initial step size t_0 is $\lambda_{\min}(\mathbf{\Theta}_0)^2$ and is an accepted line search criteria in Step 1 of Algorithm 3. The algorithm terminates when the duality gap Δ is less than or equal to a pre-specified tolerance ε or when the algorithm exceeds 100000 iterations.

Algorithm 3 Algorithm for GOWL

```
Require: $\mathbf{S}$, tolerance $\varepsilon$, $\boldsymbol{\lambda}$ (OWL weights), $t_0 > 0$, $\boldsymbol{\Theta}_0$, $c \in (0,1)$
  while $\Delta > \varepsilon$ do
      (1) Let $t_k$ be the largest element of $\{c^j t_{k,0}\}_{j=0,1,\ldots}$ so that for $\boldsymbol{\Theta}_{k+1} = \mathrm{prox}_{\Omega_{\mathrm{OWL}}}(\boldsymbol{\Theta}_k - t_k(\mathbf{S} - (\boldsymbol{\Theta}_k^{-1}))$,
      the following are satisfied:
          $\boldsymbol{\Theta}_{k+1} \succ 0 \quad \text{and} \quad -\log\det \boldsymbol{\Theta}_{k+1} + \mathrm{tr}(\boldsymbol{S}\boldsymbol{\Theta}_{k+1}) \leq Q_{t_k}(\boldsymbol{\Theta}_{k+1}, \boldsymbol{\Theta}_k)\,, \qquad (4.17)$
      (2) Compute $\nabla g(\boldsymbol{\Theta}_k) := \boldsymbol{\Theta}_k - t_k(\boldsymbol{S} - \boldsymbol{\Theta}_k^{-1})$.
      (3) Set $\boldsymbol{\Theta}_{k+1} := \mathrm{prox}_{\mathrm{OWL}}(\nabla g(\boldsymbol{\Theta}_k))$
      (4) Compute the duality gap $\Delta$.
Ensure: $\varepsilon$-optimal solution $\hat{\boldsymbol{\Theta}} = \boldsymbol{\Theta}_{k+1}$.
```

4.4 Column-by-column Graphical OWL: Theoretical Results

In this section, we describe the theoretical formulation of the column-by column graphical OWL (ccGOWL) estimator, prove sufficient grouping conditions for the column estimates and describe the ccGOWL algorithm.

4.4.1 Formulation of ccGOWL

We define the column-by-column Graphical Order Weighted ℓ_1 (ccGOWL) estimator to be the solution to the following unconstrained optimization problem:

$$\hat{\boldsymbol{\beta}}_j = \underset{\boldsymbol{\beta}_j \in \mathbb{R}^{p-1}}{\arg\min} \, ||\mathbf{X}_{*,j} - \mathbf{X}_{*,-j}\boldsymbol{\beta}_j||_2^2 + \boldsymbol{\Omega}_{\mathrm{OWL}}(\boldsymbol{\beta}_j)\,, \qquad (4.18)$$

where $\boldsymbol{\beta}_j \in \mathbb{R}^{p-1}$. We assume that the columns of the design matrix $\mathbf{X}$ have been standardized to have zero mean and unit variance. The final estimate $\hat{\boldsymbol{\Theta}}$ is obtained in the following way:

$$\hat{\mathbf{\Theta}} = \begin{bmatrix} | & | & \dots & | \\ \hat{\boldsymbol{\beta}}_1 & \hat{\boldsymbol{\beta}}_2 & \dots & \hat{\boldsymbol{\beta}}_p \\ | & | & \dots & | \end{bmatrix}, \tag{4.19}$$

where a value of one is inserted in each $\hat{\boldsymbol{\beta}}_j$ at position j to account for the diagonal entries of the matrix.

4.4.2 Sufficient Grouping Conditions for ccGOWL

We can establish sufficient grouping conditions when estimating each column of $\mathbf{\Theta}$ by drawing on previous work for the OSCAR and OWL regularizers. The term "grouping" refers to components of each column estimate being equal.

Theorem 4.4.1 *Let $\hat{\beta}_k, \hat{\beta}_l > 0$ be elements in the column estimate $\hat{\boldsymbol{\beta}}_j$ of $\hat{\mathbf{\Theta}}_j$ generated with hyperparameter λ_2, and let them be unique from other entries in $\hat{\boldsymbol{\beta}}_j$. Then there exists a $\lambda_2^{'}$ such that*

$$0 < |\lambda_2^{'}(p-1)| \leq 2||\mathbf{X}_{*,j}||_2^2\sqrt{2(1-\rho_{kl})}\,, \tag{4.20}$$

so that for all

$$|\lambda_2| > |\lambda_2^{'}| \tag{4.21}$$

we have

$$\hat{\beta}_k = \hat{\beta}_l \tag{4.22}$$

for $j = 1, \dots, (p-1)$ where $\rho_{kl} = \mathbf{x}_k^T\mathbf{x}_l$ is the sample correlation between columns of $\mathbf{X}_{*,-j}$.

Depending on the sample correlation between covariates in $\mathbf{X}_{*,-j} \in \mathbb{R}^{n\times(p-1)}$ the sufficient grouping property can be quantified according to Theorem 4.4.1.

Proof: The following proof follows the approach used in [4]. Suppose $\hat{\beta}_k \neq \hat{\beta}_l$, then by differentiating (26) we obtain

$$-2\mathbf{x}_k^T(\mathbf{X}_{*,j} - \mathbf{X}_{*,-j}\boldsymbol{\beta}_j) - \lambda_k = 0\,, \tag{4.23}$$

and

$$-2\mathbf{x}_l^T(\mathbf{X}_{*,j} - \mathbf{X}_{*,-j}\boldsymbol{\beta}_j) - \lambda_l = 0\,, \tag{4.24}$$

By subtracting the two equations we can write:

$$-2(\mathbf{x}_l^T - \mathbf{x}_k^T)(\mathbf{X}_{*,j} - \mathbf{X}_{*,-j}\boldsymbol{\beta}_j) + (\lambda_l - \lambda_k)\,. \tag{4.25}$$

We can say that $||\mathbf{X}_{*,j} - \mathbf{X}_{*,-j}\boldsymbol{\beta}_j||_2^2 \leq ||\mathbf{X}_{*,j}||_2^2$ and $||\mathbf{x}_l^T - \mathbf{x}_k^T||_2^2 = 2(1-\rho_{kl})$ as a result of $\mathbf{X}$ being standardized. So we can write

$$|\lambda_l - \lambda_k| \leq 2||\mathbf{X}_{*,j}||_2^2\sqrt{2(1-\rho_{kl})}\,. \tag{4.26}$$

We know when the weights are constructed that $|\lambda_l - \lambda_k| \geq c = |\lambda_1 - (\lambda_1 + \lambda_2(p-1))| = |\lambda_2(p-1)|$. So we arrive at a contradiction if $c > 2||\mathbf{X}_{*,j}||_2^2\sqrt{2(1-\rho_{kl})}$. ■

4.4.3 ccGOWL Algorithm

We now present an algorithm (Algorithm 4) to solve the ccGOWL problem expressed in (4.18). The algorithm is based on the proximal method for solving the OWL regularized linear regression proposed in [41] and has a convergence rate of $O(1/k)$. Algorithm 4 applies a proximal method p times and combines each column to arrive at the final precision matrix estimate as illustrated in (4.19). In Algorithm 4, $Q_{t_k}(\boldsymbol{\beta}_j^k, \boldsymbol{\beta}_j^{k+1})$ is a quadratic approximation of the smooth component of the objective defined by:

$$\begin{aligned} Q_{t_k}(\boldsymbol{\beta}_j^k, \boldsymbol{\beta}_j^{k+1}) =& ||\mathbf{X}_{*,j} - \mathbf{X}_{*,-j}\boldsymbol{\beta}_j^{k+1}||_2^2 + 2(\boldsymbol{\beta}_j^k - \boldsymbol{\beta}_j^{k+1})^T \mathbf{X}_{*,-j}^T(\boldsymbol{\beta}_j^k - \boldsymbol{\beta}_j^{k+1}) \\ &+ \frac{t_k}{2}||\boldsymbol{\beta}_j^k - \boldsymbol{\beta}_j^{k+1}||_2^2 \,. \end{aligned} \tag{4.27}$$

A suitable starting step size is one and during the backtracking line search (Step 1), multiplying t_k by some $\varepsilon > 0$ at every iteration allows for a reasonable line search convergence. We found that $\varepsilon = 2$ worked well in practice.

Algorithm 4 Algorithm for ccGOWL

```
**Require:** $\mathbf{X}$, $\boldsymbol{\lambda}$ (OWL weights), $t_0 > 0$
  **for** $j \to p$ **do**
    (1) Determine $t_k > 0$ such that the following is satisfied:
          $$||\mathbf{X}_{*,j} - \mathbf{X}_{*,-j}\boldsymbol{\beta}_j^{k+1}||_2^2 \leq Q_{t_k}(\boldsymbol{\beta}_j^k, \boldsymbol{\beta}_j^{k+1})\,, \tag{4.28}$$
    (2) Compute $\nabla g(\boldsymbol{\beta}_j^k) := n^{-1}\mathbf{X}_{*,-j}^T(\mathbf{X}_{*,j} - \mathbf{X}_{*,-j}\boldsymbol{\beta}_j^k)\,.$
    (3) Set $\boldsymbol{\beta}_j^{k+1} \leftarrow \mathrm{prox}_{\mathrm{OWL}}\Big(\boldsymbol{\beta}_j^k - t_k\nabla g(\boldsymbol{\beta}_j^k)\Big)\,.$
  Combine all $\hat{\boldsymbol{\beta}}_j = \hat{\boldsymbol{\beta}}_j^{k+1}$ to form $\hat{\boldsymbol{\Theta}}$.
```

4.5 Summary

This chapter describes the steps needed to achieve the unique properties of the GOWL estimator and the grouping conditions required for the ccGOWL estimator. We propose two robust proximal algorithms, one for each estimator. In the next chapter we examine the results of both estimators on synthetic data.

Chapter 5

Synthetic Data Analysis

5.1 Overview

To assess the performance of our algorithms (GOWL and ccGOWL), we conducted a series of experiments on a synthetic dataset with controlled sparsity and grouping, using the GRAB [6] and GLASSO [31] algorithms as references. The CGL method was not considered since the GRAB was shown in [6] to more successfully recover group structure. Firstly, we compared the estimation accuracy of these algorithms. We use cross-validation to select hyperparameters and the F_1 score to evaluate the identification of groups in the network estimate. Secondly, we compare the computational efficiency on a subset of the synthetic dataset.

5.2 Data

Synthetic data was generated by first choosing a proportion of groups $\kappa \in \{0.1, 0.2\}$ for a p-dimensional matrix. The κ chosen determines the total number of groups by multiplying κ by p. For instance, if $\kappa = 0.2$ and $p = 10$, then there would be a total of two groups present in the precision matrix. We randomly chose each group size to be between a minimum size of 0.1 of p and a maximum size of 0.4 of p. Furthermore, group values were determined by uniformly sampling their mean between $(0.9, 1.0)$ and $(-0.9, -1.0)$ respectively. After setting all values of a given group to its mean, we randomly insert each block value into $\mathbf{\Theta}^*$. In order to add noise to the true group values, we randomly generated a positive semi-definite matrix with entries set to zero with a fixed probability of 0.5 and remaining values sampled between $(-0.1, 0.1)$. We then added the grouped matrix to the aforementioned noise matrix to create a $\mathbf{\Theta}^* + \boldsymbol{\epsilon}$ matrix. Each grouped matrix was generated 5 times and random noise matrices were added to each of the 5 grouped matrices 20 times. A dataset was then generated by drawing i.i.d. from an $\mathcal{N}_p(0, (\mathbf{\Theta}^* + \boldsymbol{\epsilon})^{-1})$ distribution, from which the empirical covariance matrix can be estimated after standardization of predictors.

5.3 Experimental Methodology

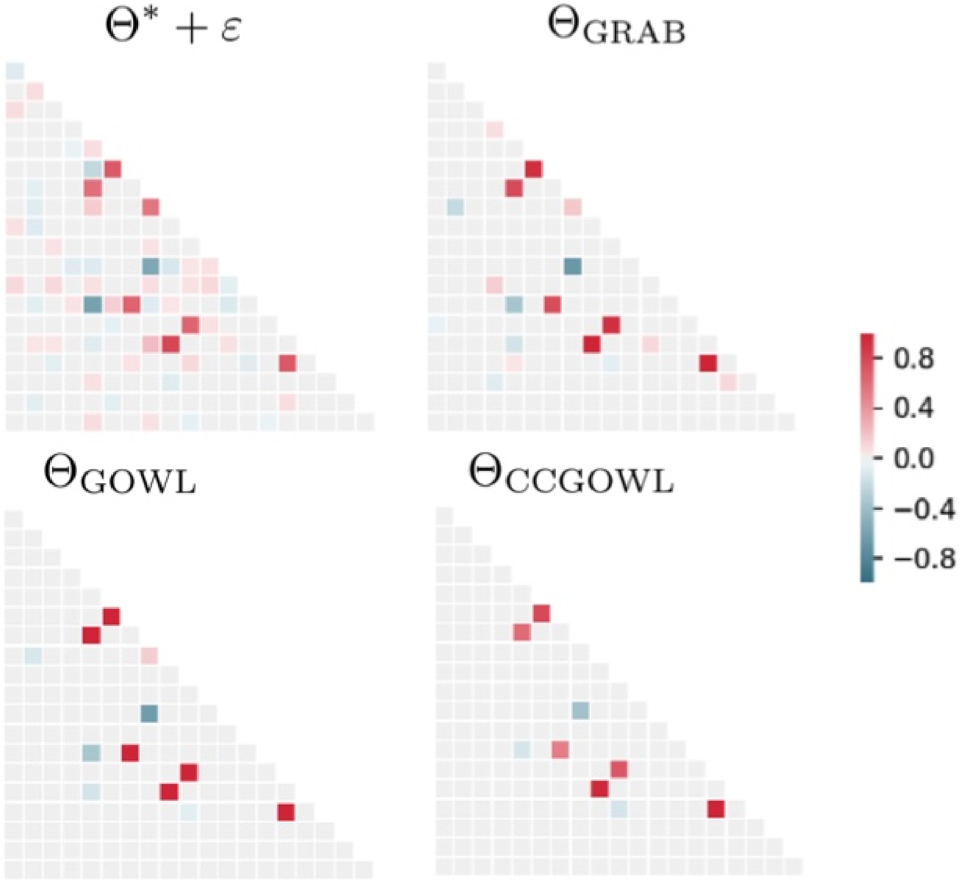

Figure 5.1: Example of synthetic data and estimation results for GRAB, GOWL and ccGOWL with $p = 20$ and $\kappa = 0.1$.

5.3.1 Estimation Accuracy

Figure 5.1 shows an example of a synthetic precision matrix with 2 groups with $\mathbf{\Theta}^* + \varepsilon$ as the ground truth. This ground truth was then used to sample a dataset of $n = 100$, from which we estimated the empirical covariance matrix and provided it as input to the algorithms. We assessed the performance of all four methods using the weighted F_1 classification score (harmonic mean of precision and recall) since we are interested in multi-group classification. The goal was to identify the number of correctly predicted group entries. The F_1 score is defined as $F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$; the weighted F_1 score is obtained by calculating the F_1 score for each label and then evaluating the weighted average, where the weights are proportional to the number of true instances of each group. For each value of $(p, \kappa) \in \{10, 20, 50\} \times \{0.1, 0.2\}$, we generated 5 randomly grouped matrices using the procedure outlined in the previous section and fit GLASSO [31], GRAB [6], GOWL, ccGOWL to each of them. The estimates were then clustered using a Gaussian Mixture Model (GMM) with the same number of groups as originally set. Applying a GMM was needed to identify the number of overlapping groups present in the graphical model. The goodness-of-fit of the clusters was compared to the original group labeling using the weighted F_1 metric, from which we report the permutation of labels giving the highest weighted F_1 score.

Cross-Validation for Hyperparameter Selection

Each of the algorithms evaluated requires hyperparameters to be selected. A well known method for estimating the optimal value of a tuning parameter is k-fold cross-validation [43]. K-fold cross-validation is a statistical technique that indicates how well a model will generalize to independent data. This involves partitioning the data into known and unknown datasets and then training the model on the former and testing the model on the latter. The performance associated with a specific set of hyperparameters is determined by averaging across folds.

Hyperparameters λ_1 and λ_2 for the OSCAR weight generation procedure defined in Section 2.4.2 were selected using a 2-fold standard cross-validation procedure. A grid search was conducted with a subset of 10 evenly spaced values in interval (0, 0.1) for λ_1 and λ_2. Similar to [6], the regularization parameter for the GRAB estimator was also selected using a 2-fold cross-validation procedure. A 2-fold cross-validation procedure was chosen due to limited computational resources. Tables 5.1 to 5.3 show the hyperparameters selected for each algorithm for each dimensionality (p) and group proportion (κ) combination.

Table 5.1: GOWL hyperparameter (λ_1 and λ_2) values selected using cross-validation for 6 p and κ combinations. In all cases, the range of hyperparameters considered were the 10 evenly spaced values in interval (0, 0.1).

p	κ	λ_1	λ_2
10	0.1	0.0368	0.0105
10	0.2	0.0684	0.0105
20	0.1	0.0655	0.00345
20	0.2	0.0241	0.00345
50	0.1	0.008	0.00010
50	0.2	0.006	0.00009

Table 5.2: ccGOWL hyperparameter (λ_1 and λ_2) values selected using cross-validation for 6 p and κ combinations. In all cases, the range of hyperparameters considered were the 10 evenly spaced values in interval (0, 0.1).

p	κ	λ_1	λ_2
10	0.1	0.105	0.05263
10	0.2	0.105	0.05263
20	0.1	0.237	0.00793
20	0.2	0.237	0.00793
50	0.1	0.1	0.00513
50	0.2	0.1	0.00513

Table 5.3: GRAB hyperparameter (λ_1 and λ_2) values selected using cross-validation for 6 p and κ combinations. In all cases, the range of hyperparameters considered were the 10 evenly spaced values in interval (0, 0.1).

p	κ	λ
10	0.1	0.1
10	0.2	0.1
20	0.1	0.7
20	0.2	0.5
50	0.1	0.5
50	0.2	0.4

Figures 5.2 to 5.4 show the cross validation error rate for the ccGOWL for $p = \{10, 20, 50\}$. These figures illustrate the importance of this hyperparameter selection procedure. In particular, Figures 5.2 and 5.4 show that only a small fraction of the hyperparameter values yield a low cross-validation error rate. Most combinations yield a very high error rate and would not be expected to achieve useful results on the test data.

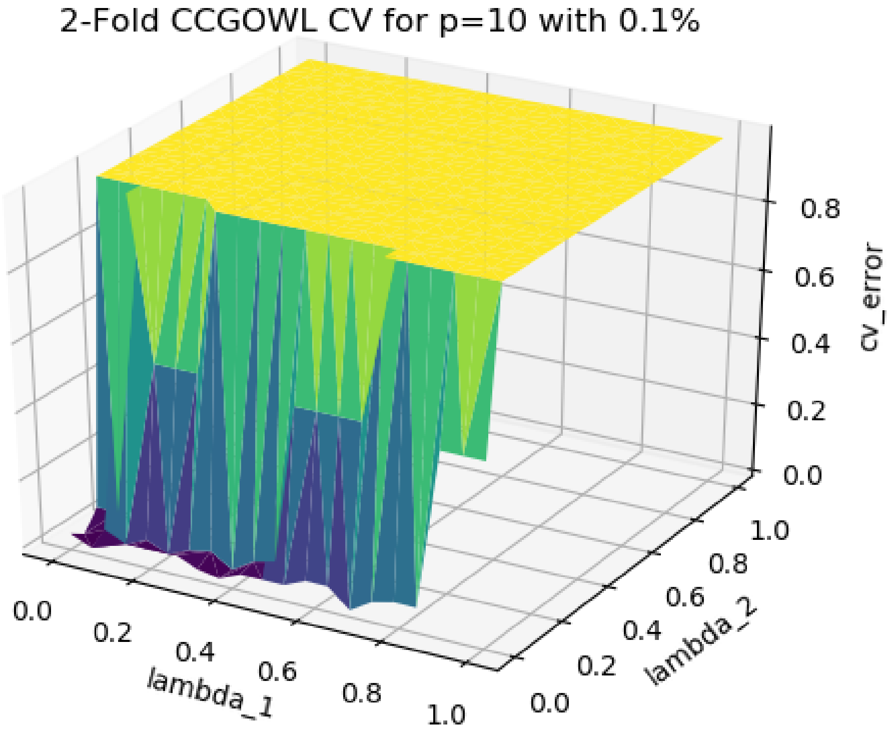

Figure 5.2: The cross-validation error rate over all hyperparameter combinations considered for ccGOWL with $p = 10$ and $\kappa = 0.1$.

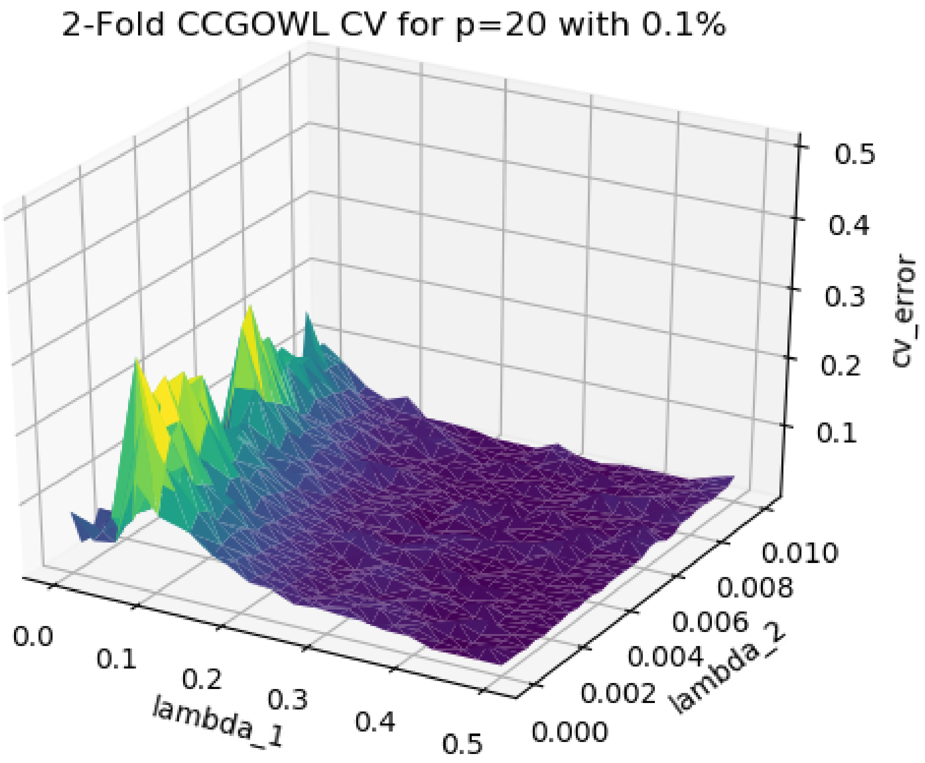

Figure 5.3: The cross-validation error rate over all hyperparameter combinations considered for ccGOWL with $p = 20$ and $\kappa = 0.1$.

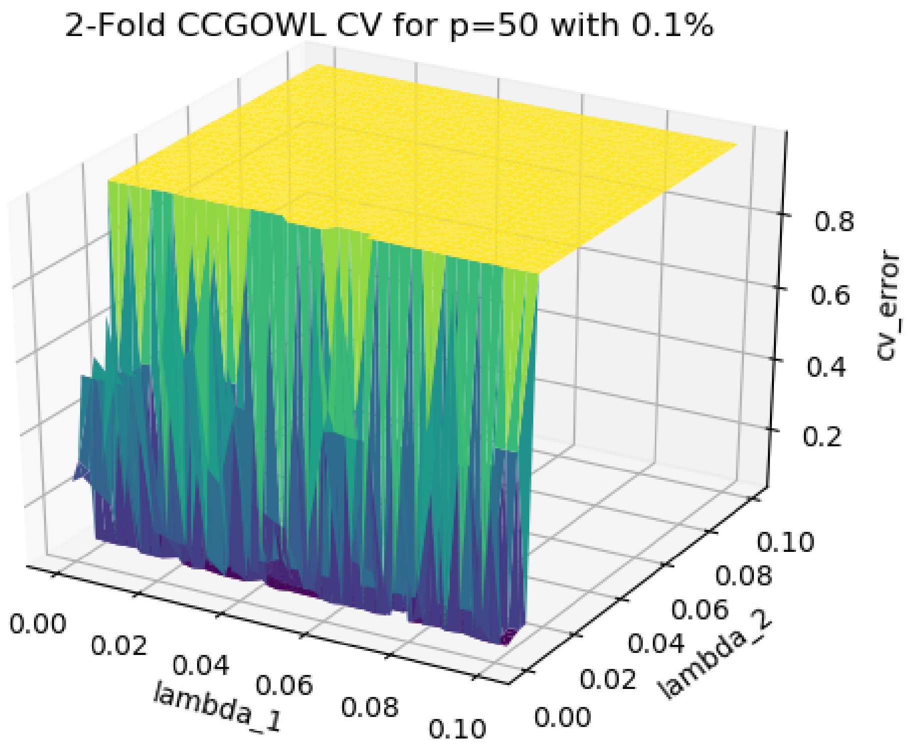

Figure 5.4: The cross-validation error rate over all hyperparameter combinations considered for ccGOWL with $p = 50$ and $\kappa = 0.1$.

5.3.2 Timing Comparisons

The GLASSO, GRAB and ccGOWL algorithms were run on synthetic datasets generated as described in Section 5.2 with varying p, $n = 100$, $\kappa = 0.2$ and different levels of regularization. The GRAB algorithm had a fixed duality gap of 10^{-4} and the ccGOWL had a fixed precision of 10^{-5}. The GRAB algorithm was implemented in `python` and utilizes the `R` package `QUIC` implemented in `C++` [44] to find the optimum of the log-likelihood function. The ccGOWL algorithm is implemented completely in `python` and requires solving p linear regressions with most of the computational complexity attributed to evaluating the proximal mapping. The GLASSO algorithm utilized the `python` package `sklearn` [45]. All algorithms were executed on an Intel i7-8700k 3.20 GHz and 32 GB of RAM.

5.4 Results

5.4.1 Estimation Accuracy

Figure 5.1 shows that the GOWL and ccGOWL precision matrix estimates almost fully recover the 6 red entries from group 1 and the two blue entries from group two. The GLASSO was not included as a result of its poor performance in recovering the true group values. Figure 5.5 shows the distribution of the scores for each algorithm, for each class of matrices. Absolute Squared Error and Mean Squared Error Measurements were recorded during synthetic data simulations. Figures 5.6 and 5.7 show, respectively, the absolute error and mean squared error for the synthetic data experiments.

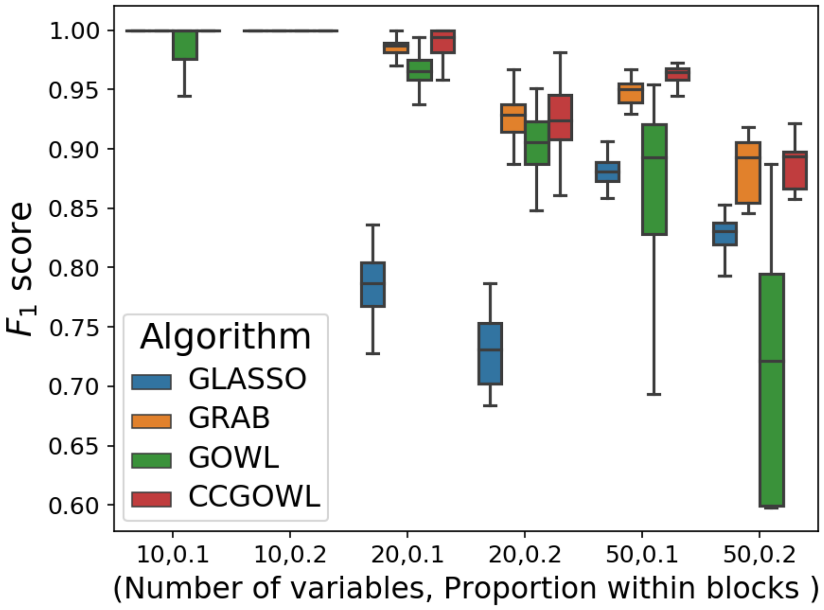

Figure 5.5: Quantiles of weighted F_1 score metric for GLASSO, GRAB, GOWL, CCGOWL. Each box shows the quantiles of the MSE values and whiskers show the remaining values of the distribution. The center line is the median and the vertical lines from the top and bottom of the box are the maximum and minimum respectively.

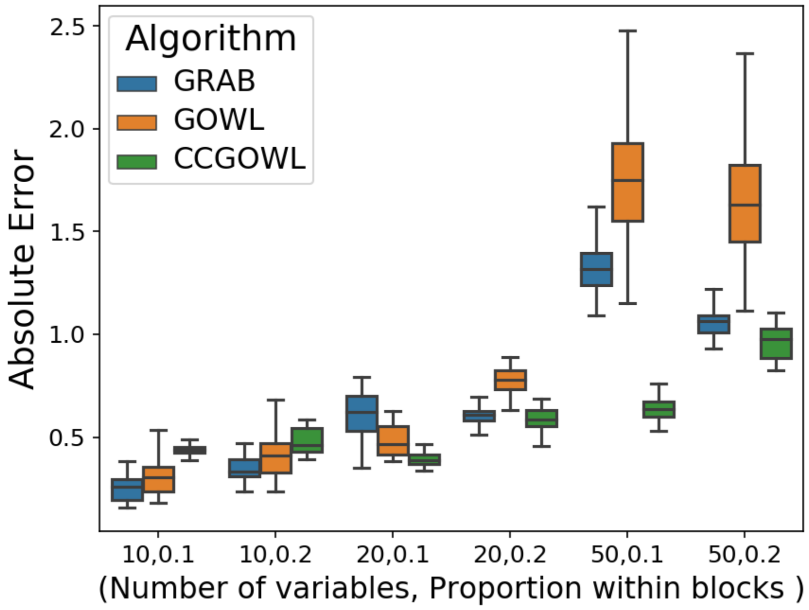

Figure 5.6: Absolute Error Values for GRAB, GOWL, and CCGOWL. Each box shows the quartiles of the MSE values and whiskers show the remaining values of the distribution. The center line is the median and the vertical lines from the top and bottom of the box are the maximum and minimum respectively.

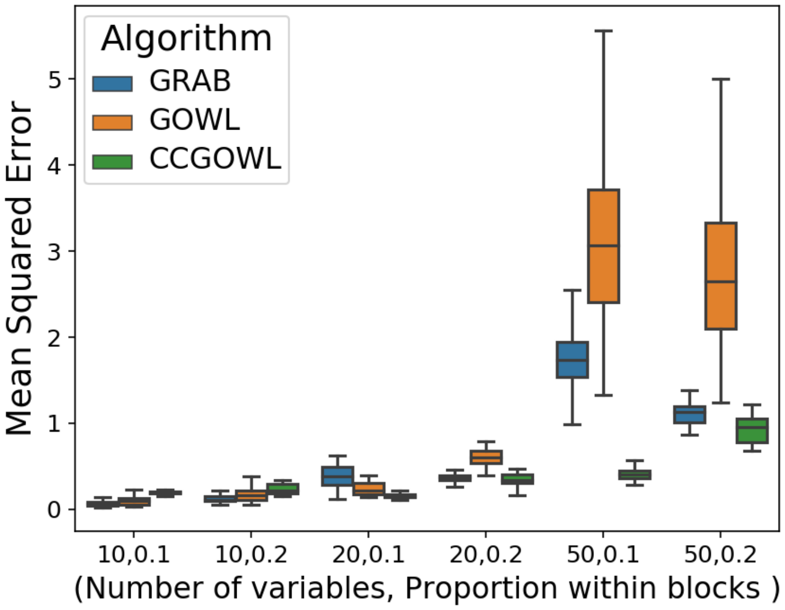

Figure 5.7: Mean Squared Error (MSE) Values for GRAB, GOWL, and CCGOWL. Each box shows the quantiles of the MSE values and whiskers show the remaining values of the distribution. The center line is the median and the vertical lines from the top and bottom of the box are the maximum and minimum respectively.

5.4.2 Timing Comparison

Running times are presented in Table 5.4.

Table 5.4: Timing Comparisons in Seconds

Method	$p = 10$	$p = 20$	$p = 50$	$p = 100$	$p = 500$	$p = 1000$
GLASSO	0.006	0.012	0.017	0.023	0.118	0.456
GRAB	0.071	0.096	0.466	1.243	51.225	499.225
ccGOWL	0.003	0.012	0.034	0.095	0.305	4.126
GOWL	0.003	0.160	2.945	10.280	160.256	> 1000.00

5.5 Discussion

5.5.1 Estimation Accuracy

Figure 5.5 shows that, overall, the ccGOWL outperforms GOWL and GLASSO in terms of variance and mean of F_1 scores. Compared to GRAB, ccGOWL achieves similar or slightly better performance for all scenarios. Figures 5.7 and 5.6 show that, overall, the ccGOWL estimator has a lower mean squared or absolute error and absolute error for $p = 20$ and $p = 50$ but has higher errors for $p = 10$. The GOWL and GRAB have the ability to identify groups across multiple columns, thus when there is low number of variables, they are able to more reliably recover these groups than the ccGOWL. This is why we observe a better performance of the GOWL and GRAB over the ccGOWL for $p = 10$, but not for higher values of p. For larger p such as $p = 20$ and $p = 50$, the optimization for the GOWL and GRAB is in a higher dimensional space making it harder to identify a good solution. Furthermore, GRAB requires an initial clustering step when learning $\mathbf{Z}$ which becomes more difficult in higher dimensions. Our results show that as p increases identifying the underlying structure in the graph becomes very challenging, resulting in lower accuracies even on a tailored synthetic data set.

Figures 5.6 and 5.7 show that ccGOWL has a larger error for $p = 50, \kappa = 0.2$ compared to $p = 50, \kappa = 0.1$. This is expected behaviour since recovering a larger amount of groups is generally a more difficult problem. Conversely, GRAB and GOWL exhibit opposite behavior and have a lower error for $p = 50, \kappa = 0.2$ compared to $p = 50, \kappa = 0.1$. The likelihood methods focus more on identifying groups and less on recovering $\mathbf{\Theta}^* + \varepsilon$ which includes identifying the correct zero entries. Moreover, the error values take into account the difference between the exact value in each group and the estimate, which is not the case when using the F_1 score to evaluation grouping ability of the estimators. This could be another contributing factor in the different behaviour observed when examining the error values in Figures 5.6 and 5.7.

5.5.2 Timing Comparison

Table 5.4 shows a significant advantage of ccGOWL over GRAB. This difference for large p is due to GRAB requiring matrix inversions within `QUIC` ($O(p^2)$) and applying the k-means clustering algorithm ($O(pK)$) on the rows of the block matrix $\mathbf{Z}$. GOWL does not apply the `QUIC` algorithm and instead uses G-ISTA which makes the matrix inversions during optimization even more expensive. The ccGOWL requires no matrix inversions and it's simple loss function is very easy to optimize resulting in a fast convergence. Although we do not take advantage of parallelism in our implementation of ccGOWL due to the algorithm being implemented in `python`, future implementations in a multi-threaded programming language could greatly benefit from the scalability of the ccGOWL. More specifically, a multi-threaded language such as `C++` could compute each column regression in a separate thread.

5.6 Summary

Using the F_1 measure to assess performance, we can observe that the GOWL estimator has lower F_1 accuracies when compared with the GRAB estimator. However, the ccGOWL estimates achieve comparable results. When we observe the overall fit, we determine that the ccGOWL achieves a better fit, in most cases, in terms of mean squared or absolute error. Timing comparisons between methods clearly indicate the benefits of ccGOWL over other methods. In the next chapter, we will see how these methods translate to real datasets.

Chapter 6

Real Data Analysis

6.1 Overview

As previously stated, identifying high correlated components in a graphical model has many real-world applications. In this chapter, we will explore a few of these applications such as in genomics and in finance.

6.2 Cancer Gene Expression Data

6.2.1 Data

We consider a dataset that uses expression monitoring of genes using DNA microarrays in 38 patients that have been diagnosed with either acute myeloid leukemia (AML) or acute lymphoblastic leukemia (ALL). This dataset was initially investigated in [46]. In order to allow for a model that is easier to interpret we selected the 50 most highly correlated genes associated with ALL-AML diseases, as identified in [46]. Of the 50 genes selected, the first 25 genes are known to be highly expressed in ALL and the remaining 25 genes are known to be highly expressed in AML. It is important to note that no single gene is consistently expressed across both AML and ALL patients. This fact illustrates the need for an estimation method that takes into account multiple genes when diagnosing patients. In addition, we use the

Reactome Pathway Database [47] to identify each gene's main biological pathway.

6.2.2 Experimental Methodology

The design matrix $\mathbf{X} \in \mathbb{R}^{38 \times 50}$ and consists of 38 patient bone marrow and peripheral blood samples and 50 of the most highly expressed genes for each disease. The matrix $\mathbf{X}$ was then standardized to have zero mean and unit variance. Genes that are highly expressed in AML should appear in the same group and likewise for the ALL disease.

6.2.3 Results

Figures 6.1 to 6.4 show the association networks derived from the precision matrix estimates produced by GLASSO, GOWL, ccGOWL, and GRAB. The hyperparameters used for the gene dataset analysis were chosen through 2-fold cross-validation. Due to lack of computational resources, the hyperparameters used to analyze the stock data were chosen using the synthetic dataset results as a starting point and then were chosen arbitrarily.

6.2.4 Discussion

Figure 6.3 illustrates that ccGOWL groups genes according to their disease. In fact, ccGOWL correctly identifies all 25 genes associated with the AML disease in one group and 24 of 25 genes associated with the ALL disease in the other group. We compare the network identified by ccGOWL with the commonly-used baseline GLASSO method, to illustrate the importance of employing an estimator that uses grouping (Figure 6.1). In addition, examining the connections within each group can also lead to useful insights. The AML group (top) contains highly connected genes beginning with "M" (blue nodes) which are associated with the neutrophil granulation process in the cell. AML is a disorder in the production of neutrophils [48]. Neutrophils are normal white blood cells with granules inside the cell that

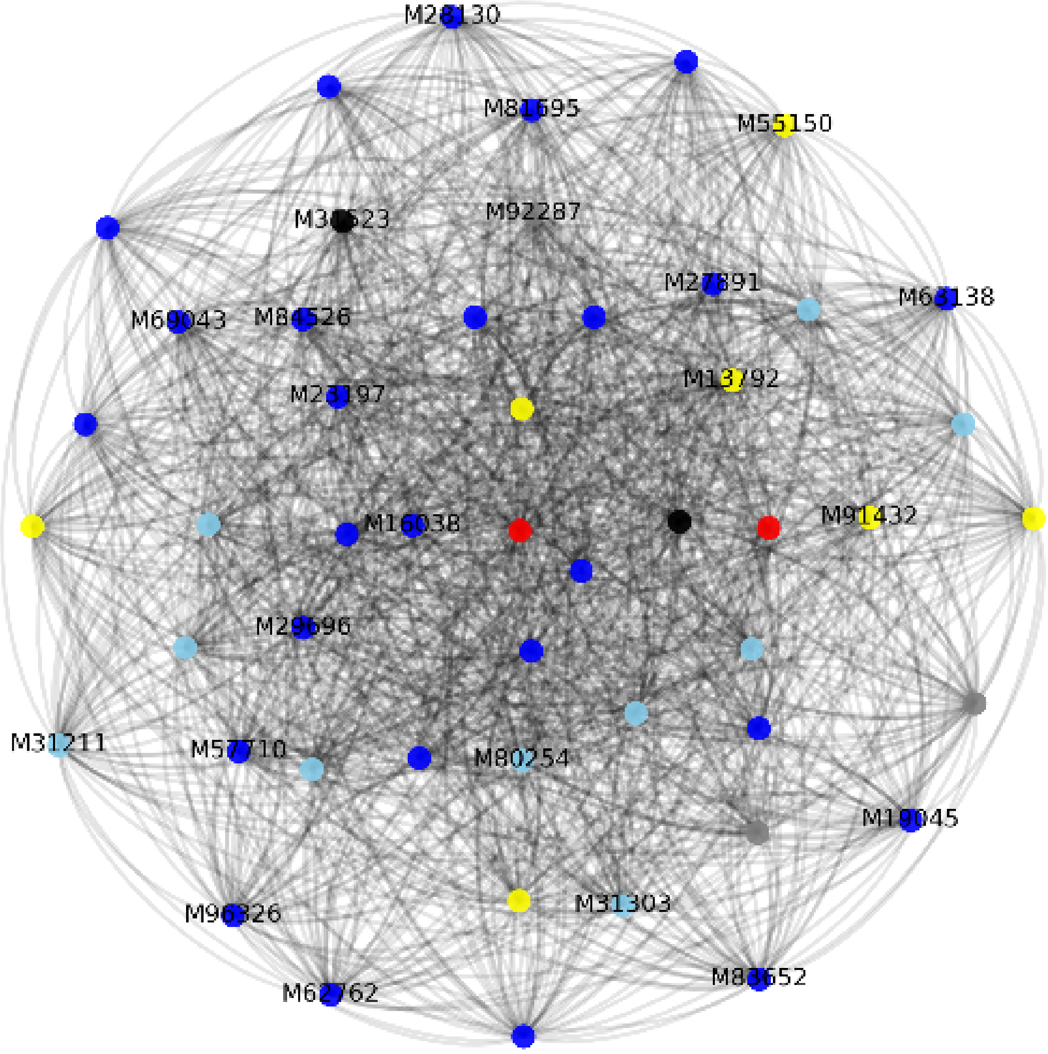

Figure 6.1: Network constructed by GLASSO ($\lambda = 0.4$) on gene expression data. Each color represents a biological pathway: Signal Transduction (red), Immune System (blue), Cell Cycle (gray), Metabolism (yellow), Gene Expression (black), Uncategorized (skyblue).

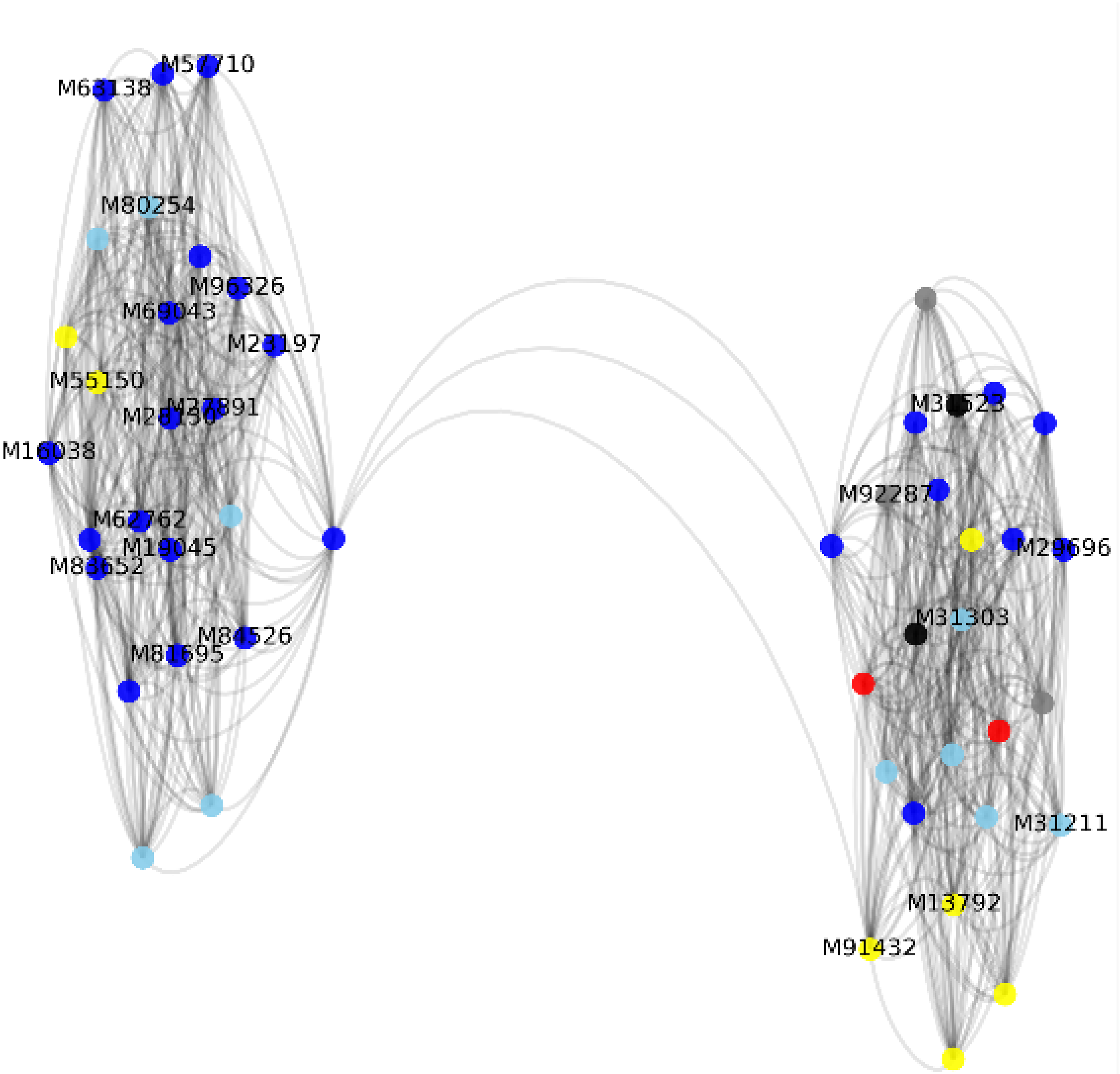

Figure 6.2: Network constructed by GOWL ($\lambda_1 = 0.006$, $\lambda_2 = 0.00009$) on gene expression data. Each color represents a biological pathway: Signal Transduction (red), Immune System (blue), Cell Cycle (gray), Metabolism (yellow), Gene Expression (black), Uncategorized (skyblue).

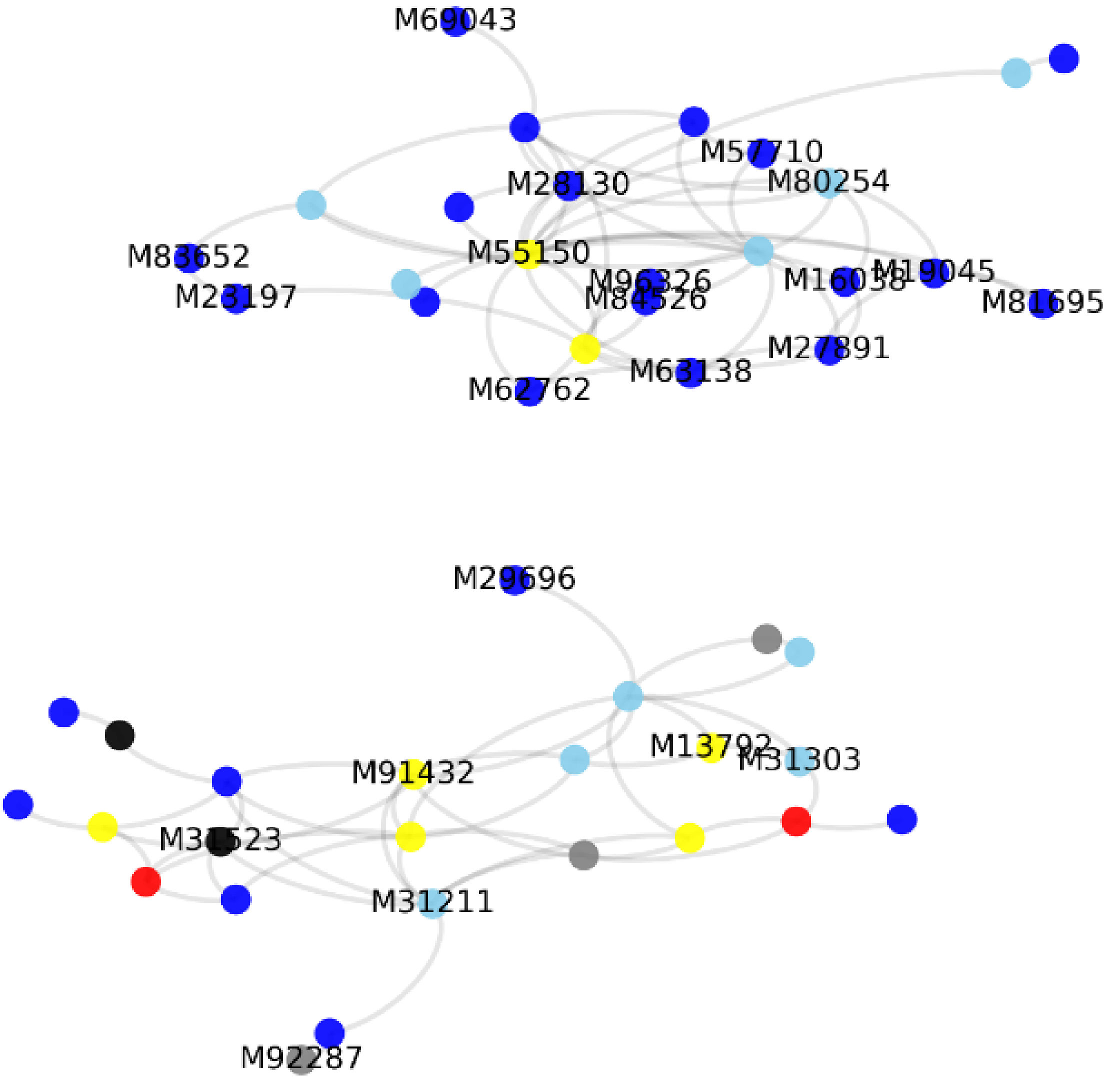

Figure 6.3: Network constructed by ccGOWL ($\lambda_1 = 0.3$, $\lambda_2 = 0.00612821$) on gene expression data. The ccGOWL network estimate clearly groups genes associated with AML (top) and ALL (bottom). Each color represents a biological pathway: Signal Transduction (red), Immune System (blue), Cell Cycle (gray), Metabolism (yellow), Gene Expression (black), Uncategorized (skyblue).

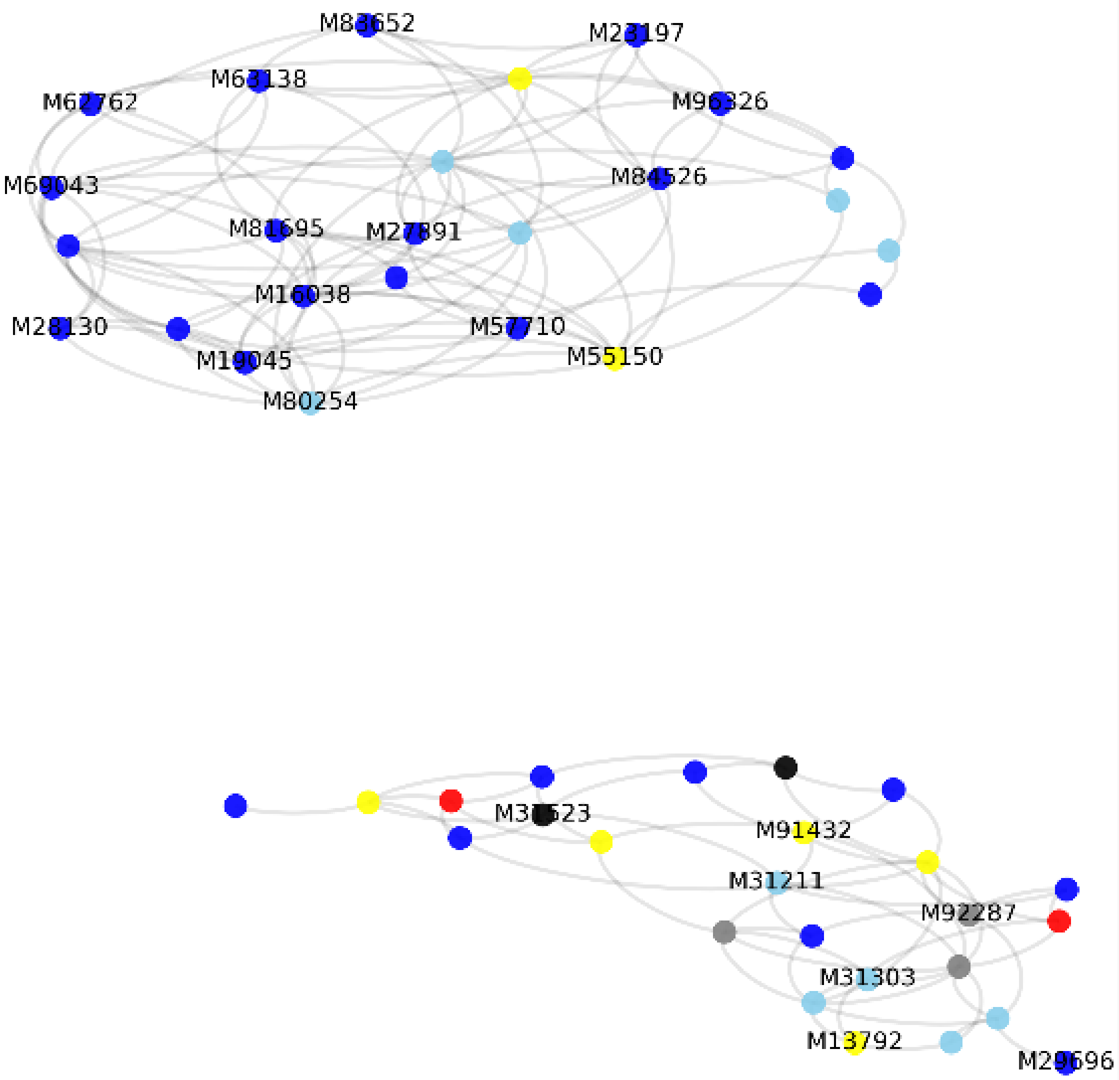

Figure 6.4: Network constructed by GRAB ($\lambda = 0.6$, overlap ratio of 0.3) on gene expression data. Each color represents a biological pathway: Signal Transduction (red), Immune System (blue), Cell Cycle (gray), Metabolism (yellow), Gene Expression (black), Uncategorized (skyblue).

fight infections. AML leads to the production of immature neutrophils (referred to as blasts), leading to large infections. We also include networks generated by the GOWL (Figure 6.2) and GRAB (Figure 6.4). The ccGOWL algorithm demonstrates the same ability as GRAB to identify genes that belong to each disease at a much reduced computational cost. The GOWL estimate includes a large amount of noise and it would be difficult to identify any meaning within each of the two groups.

6.3 Equities Data

6.3.1 Data

We consider the stock price dataset described in [49], which is available in the `huge` package on `CRAN` [50]. The dataset consists of the closing prices of stocks in the S&P 500 between January 1, 2003 and January 1, 2008. The collection of stocks can be categorized into 10 Global Industry Classification Standard (GICS) sectors [51]. Stocks that were not consistently in S&P 500 index or were missing too many closing prices were removed from the dataset.

6.3.2 Experimental Methodology

The design matrix $\mathbf{X} \in \mathbb{R}^{1257 \times 452}$ contains the log-ratio of the price at time t to the price at time $t-1$ for each of the 452 stocks for 1257 trading days. More formally we write that the (i, j)-th entry of $\mathbf{X}$ is defined as $x_{ij} = \log(S_{(i+1)j}/S_{ij})$ where S_{ij} is the closing price of the jth stock on the ith day. The matrix $\mathbf{X}$ was then standardized so each stock has a mean of zero and unit variance. The GICS sector for each stock is known, but this information was not used when estimating the precision matrix based on the matrix $\mathbf{X}$.

6.3.3 Results

Figures 6.5 to 6.8 show the association networks derived from the precision matrix estimates produced by each algorithm.

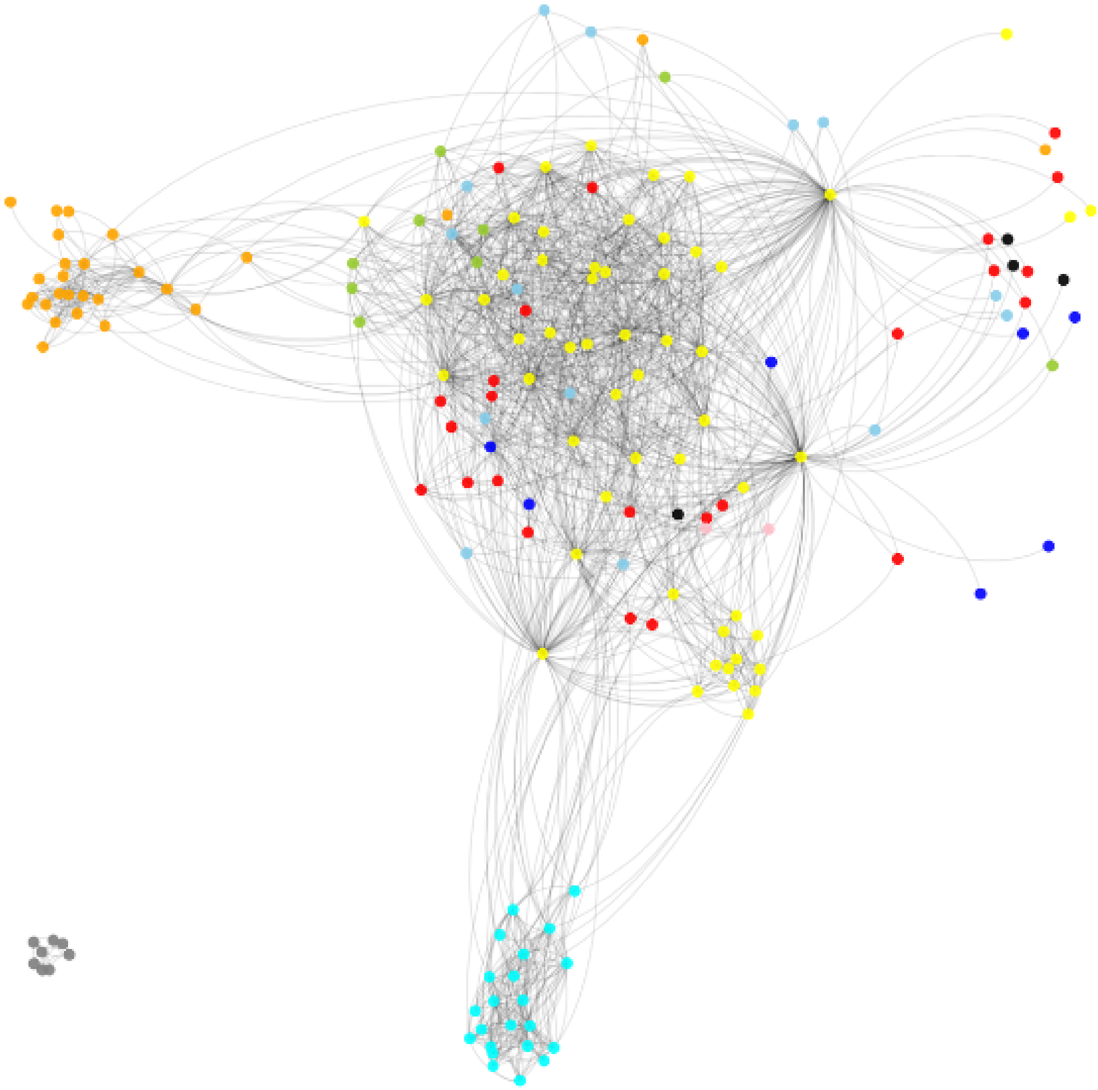

Figure 6.5: Network constructed by GLASSO ($\lambda = 0.1$) on equities expression data. Each colour represents a GICs sector: Consumer Discretionary (red), Consumer Staples (blue), Energy (gray), Financials (yellow), Health Care (black), Industrials (skyblue), Information Technology (orange), Materials (yellow-green), Telecommunications Services (pink), Utilities (cyan).

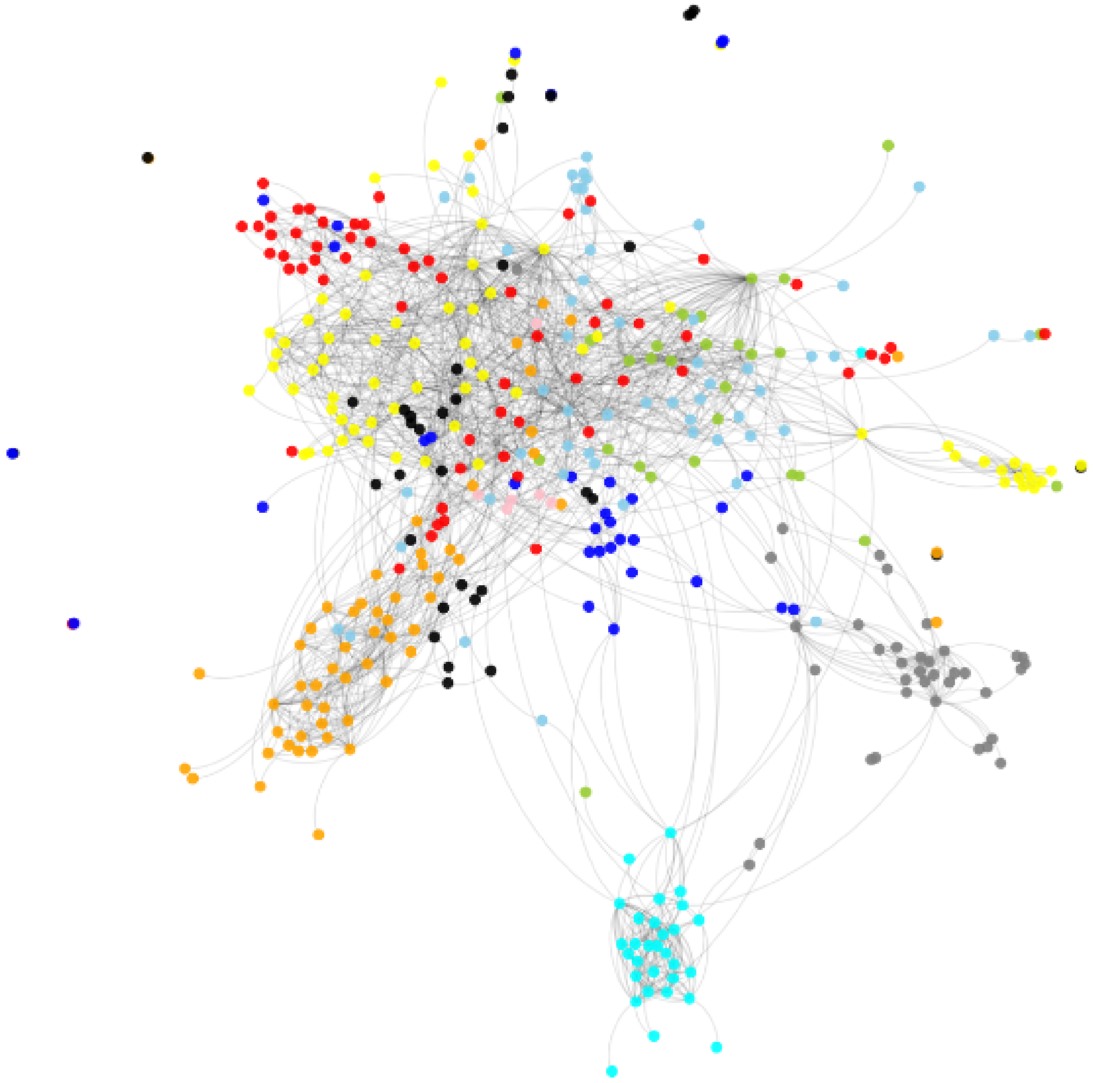

Figure 6.6: Network constructed by ccGOWL ($\lambda_1 = 0.2$, $\lambda_2 = 0.0001$) on equities expression data. Each colour represents a GICs sector: Consumer Discretionary (red), Consumer Staples (blue), Energy (gray), Financials (yellow), Health Care (black), Industrials (skyblue), Information Technology (orange), Materials (yellow-green), Telecommunications Services (pink), Utilities (cyan).

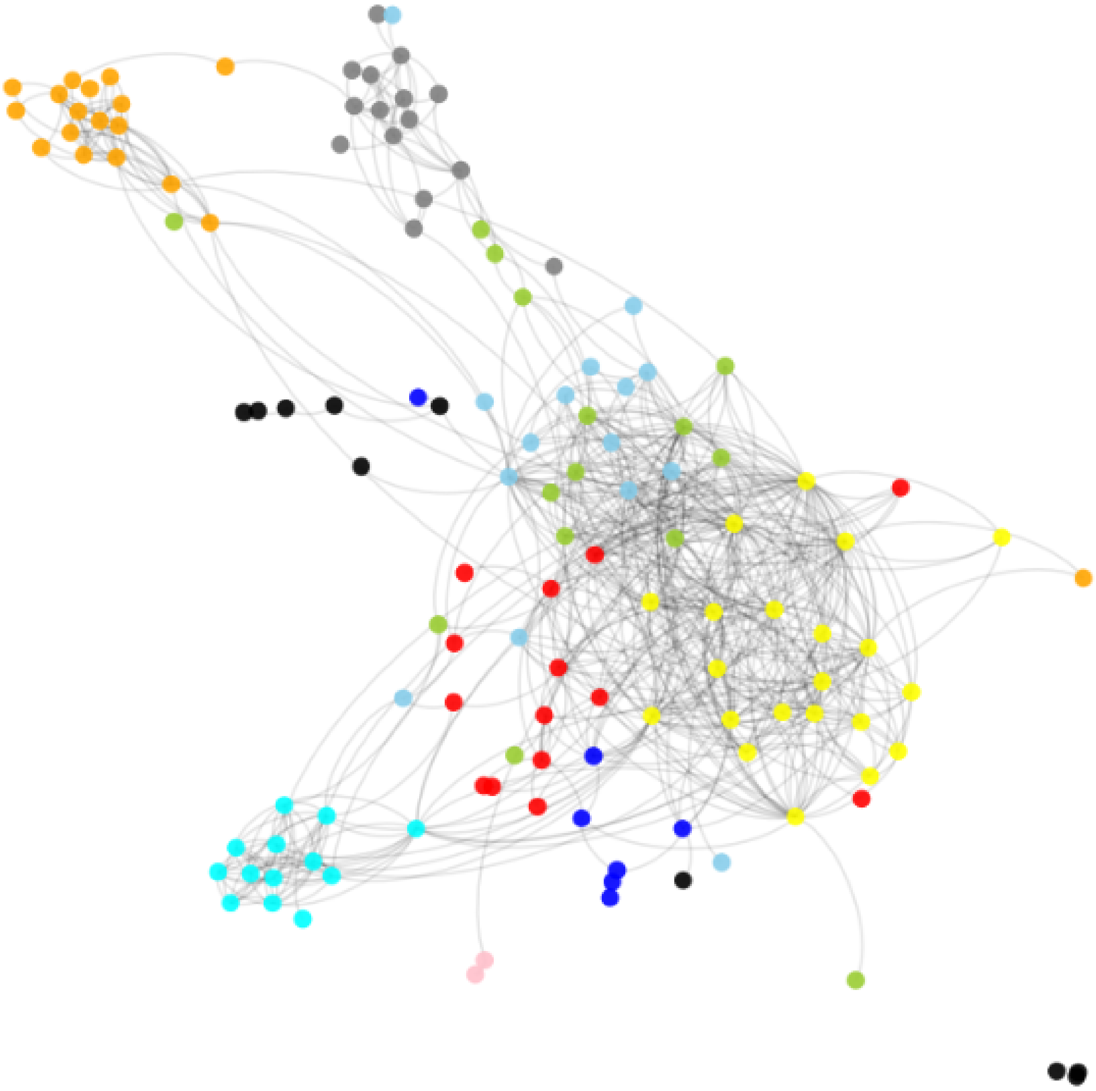

Figure 6.7: Network constructed by GOWL ($\lambda_1 = 1.0\times10^{-6}$, $\lambda_2 = 1.0\times10^{-6}$) on equities expression data. Each colour represents a GICs sector: Consumer Discretionary (red), Consumer Staples (blue), Energy (gray), Financials (yellow), Health Care (black), Industrials (skyblue), Information Technology (orange), Materials (yellow-green), Telecommunications Services (pink), Utilities (cyan).

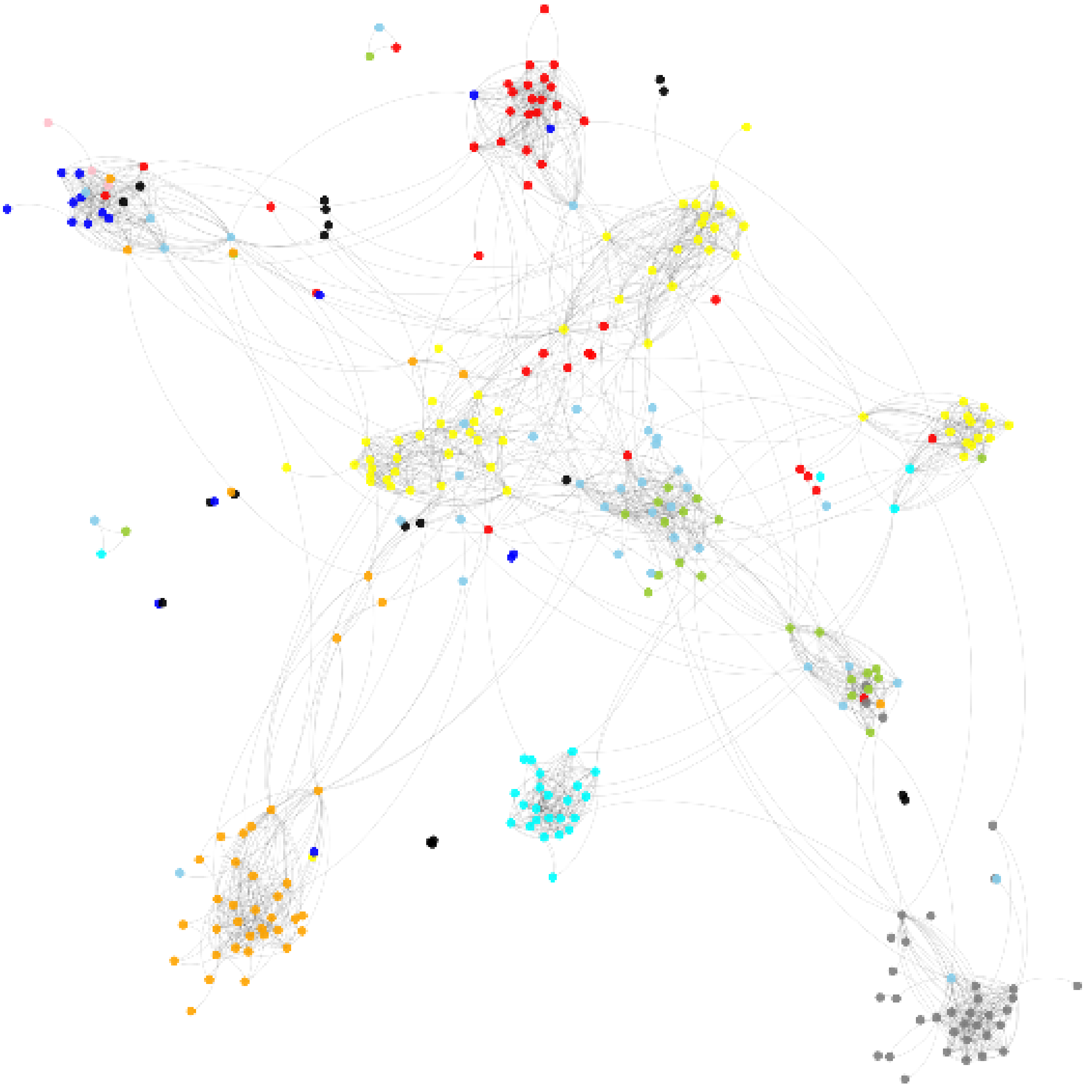

Figure 6.8: Network constructed by GRAB ($\lambda = 0.4$, overlap ratio of 0.3) on equities expression data. Each colour represents a GICs sector: Consumer Discretionary (red), Consumer Staples (blue), Energy (gray), Financials (yellow), Health Care (black), Industrials (skyblue), Information Technology (orange), Materials (yellow-green), Telecommunications Services (pink), Utilities (cyan).

6.3.4 Discussion

Figure 6.6 illustrates the ability of ccGOWL to group stocks that belong to the same sector and is more interpretable than the network constructed by GLASSO (Figure 6.5). Information Technology and Utilities stocks largely exhibit conditional independence of stocks in other GICS sectors. This means that there appear to be few edges between Information Technology stocks and Utilities stocks. On the other hand, there appears to be a dependence between stocks in the Materials and Industrials sectors, probably as a result of multiple customer-supplier relationships between the companies in these sectors. Both sectors are sensitive to economic cycles and the equities offer exposure to global infrastructure replacement. We also include networks generated by the GOWL (Figure 6.7) and GRAB (Figure 6.8). We can observe that all algorithms isolate the majority of companies in the energies, utilities, and IT sectors. Conversely, the algorithms tend to struggle in identifying the Financials, Consumer Discretionary, and Healthcare sectors.

When visually examining Figure 6.5-6.8, each algorithm's graph topologies appear similar with respect to grouping GICS classes. However, in comparing the ccGOWL and GRAB graphs, we consider the average amount of edges per group. We find that ccGOWL and GRAB share about 51% of the same edges. In contrast, only 30% of edges in the GOWL graph are contained in the GRAB graph. This is due to the fact that the GOWL graph contains less edges than the ccGOWL and GRAB corresponding to greater amount of zero entries in it's precision matrix estimate. The GLASSO graph contains the most edges per group on average leading to a topological graph structure that can be difficult to interpret. Both ccGOWL and GRAB demonstrate a similar ability to identify GIC sectors for the group of stocks considered with the caveat that GRAB exhibits more edges per group on average leading to a noisier graph topology.

6.4 Timing Comparisons

In the same way as our synthetic data experiments, we compare the runtimes of different methods on the real datasets considered. The results are displayed in Table 6.1. For both datasets, ccGOWL requires much less time to execute than GRAB and GOWL (at least an order of magnitude). For the equities data, with a larger p, GRAB takes over 100 seconds to execute, whereas ccGOWL provides an estimate after only a few seconds.

Table 6.1: Timing Comparisons in Seconds

Method	Gene Expression Data	Equities Data
GLASSO	0.605	1.539
GRAB	1.165	121.449
ccGOWL	0.053	3.618
GOWL	1.564	80.914

6.5 Summary

We provide a comprehensive comparison between our method and two state-of-the-art methods on two real datasets. GRAB and ccGOWL display a similar capability to identify structure that is known (from other sources) to be present in the datasets. However, we have shown that the ccGOWL significantly out-performs the GRAB with regards to efficiency and provides a much more scalable solution to the problems described in this chapter.

Chapter 7

Conclusions and Future Work

7.1 Summary and Concluding Remarks

Learning structure in Gaussian graphical models is an important problem that requires combining several disciplines such as optimization and high dimensional statistics. These methods have a myriad of applications in fields such as finance and computational biology. Such applications can include identifying economic trends with asset prices or identifying overlapping gene pathways that encode proteins for pivotal processes in the human body. These applications are examined in this thesis, but represent only a fraction of the range of applications that can be considered. These pursuits are further supported by the need for solutions to large computational problems that are pragmatic and easy to use. However, there is still insufficient work bridging the gap between theory and practice.

In this thesis we have developed two novel estimators for Gaussian graphical models by incorporating the Order Weighted ℓ_1 norm as a regularizer. In Chapter 3, we demonstrated the costs and benefits of column-by-column estimation techniques over full precision matrix estimation techniques. In Chapter 4, we developed theory for both estimators by showing uniqueness of the GOWL estimator and sufficient grouping conditions for the ccGOWL. Both are benefits that state-of-the-art competitors fail to provide. We also developed algorithms to solve both estimators based on the well-studied proximal methods. These algorithms are accompanied by open-source `python` implementation readily available to the public[1]. In Chapter 5, we presented the performance of GOWL, ccGOWL and two state-of-the-art estimators, (GRAB and GLASSO), on synthetic datasets. These results indicated that the ccGOWL achieves comparable estimation accuracy to the state-of-the-art, with superior computational efficiency. In Chapter 6, we present the performance of these methods on gene-expression and equities datasets. In both cases, the ccGOWL yielded results with comparable interpretability, but significantly better utility, than other methods. Although penalized likelihood methods for learning structure in precision matrix estimation are widely used and perform well, it is also important to consider column-by-column estimators that accomplish similar feats while reducing computational complexity.

7.2 Future Work

Future work can include the development of a convergence theory around the ccGOWL proximal algorithm. This would provide a more detailed account of why ccGOWL requires significantly less execution time than penalized likelihood methods such as GRAB and GOWL. This could provide a better illustration of how GOWL and ccGOWL behave for higher dimensional problems. It would also be interesting to investigate how other penalizers discussed in Chapter 2 can be extended to the problems presented in this thesis. A comprehensive comparison of popular penalizers in the Gaussian graphical model setting could be very valuable in identifying the correct application setting for specific penalizers.

[1] https://github.com/cmazzaanthony/ccgowl

References

[1] O. Banerjee, L. E. Ghaoui, and A. d'Aspremont, "Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data," *Journal of Machine Learning Research*, vol. 9, no. Mar, pp. 485–516, 2008.

[2] T. Cai, W. Liu, and X. Luo, "A constrained ℓ_1 minimization approach to sparse precision matrix estimation," *Journal of the American Statistical Association*, vol. 106, no. 494, pp. 594–607, 2011.

[3] N. Meinshausen, P. Bühlmann, *et al.*, "High-dimensional graphs and variable selection with the lasso," *The Annals of Statistics*, vol. 34, no. 3, pp. 1436–1462, 2006.

[4] H. D. Bondell and B. J. Reich, "Simultaneous regression shrinkage, variable selection, and supervised clustering of predictors with OSCAR," *Biometrics*, vol. 64, no. 1, pp. 115–123, 2008.

[5] J. Duchi, S. Gould, and D. Koller, "Projected subgradient methods for learning sparse Gaussians," in *Proc. Int. Conf. Artificial Intelligence and Statistics*, 2008, pp. 153–160.

[6] M. J. Hosseini and S.-I. Lee, "Learning sparse Gaussian graphical models with overlapping blocks," in *Advances in Neural Information Processing Systems*, 2016, pp. 3808–3816.

[7] K. M. Tan, D. Witten, and A. Shojaie, “The cluster graphical lasso for improved estimation of Gaussian graphical models,” *Computational Statistics & Data Analysis*, vol. 85, no. 1, pp. 23–36, 2015.

[8] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *Journal of the Royal Statistical Society. Series B*, vol. 58, no. 1, pp. 267–288, 1996.

[9] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani, *et al.*, “Least angle regression,” *The Annals of statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[10] N. Meinshausen, “Relaxed lasso,” *Computational Statistics & Data Analysis*, vol. 52, no. 1, pp. 374–393, 2007.

[11] H. Zou, “The adaptive lasso and its oracle properties,” *Journal of the American Statistical Association*, vol. 101, no. 476, pp. 1418–1429, 2006.

[12] L. Meier, S. Van De Geer, and P. Bühlmann, “The group lasso for logistic regression,” *Journal of the Royal Statistical Society: Series B*, vol. 70, no. 1, pp. 53–71, 2008.

[13] N. Simon, J. Friedman, T. Hastie, and R. Tibshirani, “A sparse-group lasso,” *Journal of Computational and Graphical Statistics*, vol. 22, no. 2, pp. 231–245, 2013.

[14] L. Wang, G. Chen, and H. Li, “Group SCAD regression analysis for microarray time course gene expression data,” *Bioinformatics*, vol. 23, no. 12, pp. 1486–1494, 2007.

[15] J. Huang, P. Breheny, and S. Ma, “A selective review of group selection in high-dimensional models,” *Statistical Science: a Review Journal of the Institute of Mathematical Statistics*, vol. 27, no. 4, pp. 481–499, 2012.

[16] H. Zou and T. Hastie, “Regularization and variable selection via the elastic net,” *Journal of the Royal Statistical Society: Series B*, vol. 67, no. 2, pp. 301–320, 2005.

[17] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski, “Structured sparsity through convex optimization,” *Statistical Science*, vol. 27, no. 4, pp. 450–468, 2012.

[18] M. Bogdan, E. Van Den Berg, C. Sabatti, W. Su, and E. J. Candès, "SLOPE—adaptive variable selection via convex optimization," *The Annals of Applied Statistics*, vol. 9, no. 3, pp. 1103–1140, 2015.

[19] X. Zeng and M. A. Figueiredo, "Decreasing weighted sorted regularization," *IEEE Signal Processing Letters*, vol. 21, no. 10, pp. 1240–1244, 2014.

[20] M. A. Figueiredo and R. D. Nowak, "Sparse estimation with strongly correlated variables using ordered weighted ℓ_1 regularization," *arXiv:1409.4005v1 [stat.ML]*, 2014.

[21] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT Press, 2009.

[22] M. Wytock and Z. Kolter, "Sparse gaussian conditional random fields: Algorithms, theory, and application to energy forecasting," in *Proc. Int. Conf. Machine Learning Research*, 2013, pp. 1265–1273.

[23] N. Parikh, S. Boyd, *et al.*, "Proximal algorithms," *Foundations and Trends in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[24] S. J. Wright, R. D. Nowak, and M. A. Figueiredo, "Sparse reconstruction by separable approximation," *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2479–2493, 2009.

[25] D. Kim, S. Sra, and I. S. Dhillon, "A scalable trust-region algorithm with application to mixed-norm regression," in *Proc. Int. Conf. Machine Learning*, 2010, pp. 519–526.

[26] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[27] B. Rolfs, B. Rajaratnam, D. Guillot, I. Wong, and A. Maleki, "Iterative thresholding algorithm for sparse inverse covariance estimation," in *Advances in Neural Information Processing Systems*, 2012, pp. 1574–1582.

[28] J. Fan, Y. Liao, and H. Liu, "An overview of the estimation of large covariance and precision matrices," *The Econometrics Journal*, vol. 19, no. 1, pp. C1–C32, 2016.

[29] E. Devijver and M. Gallopin, "Block-diagonal covariance selection for high-dimensional Gaussian graphical models," *Journal of the American Statistical Association*, vol. 113, no. 521, pp. 306–314, 2018.

[30] A. Defazio and T. S. Caetano, "A convex formulation for learning scale-free networks via submodular relaxation," in *Advances in Neural Information Processing Systems*, 2012, pp. 1250–1258.

[31] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

[32] N. J. Higham, "Computing the nearest correlation matrix—a problem from finance," *IMA Journal of Numerical Analysis*, vol. 22, no. 3, pp. 329–343, 2002.

[33] H. Qi and D. Sun, "A quadratically convergent newton method for computing the nearest correlation matrix," *SIAM Journal on Matrix Analysis and Applications*, vol. 28, no. 2, pp. 360–385, 2006.

[34] H. Liu, L. Wang, and T. Zhao, "Sparse covariance matrix estimation with eigenvalue constraints," *Journal of Computational and Graphical Statistics*, vol. 23, no. 2, pp. 439–459, 2014.

[35] G. Chan and A. T. Wood, "Algorithm AS 312: An algorithm for simulating stationary Gaussian random fields," *Journal of the Royal Statistical Society: Series C*, vol. 46, no. 1, pp. 171–181, 1997.

[36] N. El Karoui, "Operator norm consistent estimation of large-dimensional sparse covariance matrices," *The Annals of Statistics*, vol. 36, no. 6, pp. 2717–2756, 2008.

[37] S. Kumar, J. Ying, J. V. d. M. Cardoso, and D. Palomar, “A unified framework for structured graph learning via spectral constraints,” *arXiv:1904.09792 [stat.ML]*, 2019.

[38] B. M. Marlin and K. P. Murphy, “Sparse Gaussian graphical models with unknown block structure,” in *Proc. Int. Conf. Machine Learning*, 2009, pp. 705–712.

[39] T. Haferlach, A. Kohlmann, L. Wieczorek, *et al.*, “Clinical utility of microarray-based gene expression profiling in the diagnosis and subclassification of leukemia: Report from the international microarray innovations in leukemia study group,” *Journal of Clinical Oncology*, vol. 28, no. 15, p. 2529, 2010.

[40] W. Liu and X. Luo, “Fast and adaptive sparse precision matrix estimation in high dimensions,” *Journal of Multivariate Analysis*, vol. 135, no. 7524, pp. 153–162, 2015.

[41] X. Zeng and M. A. Figueiredo, “The ordered weighted ℓ_1 norm: Atomic formulation, projections, and algorithms,” *arXiv preprint arXiv:1409.4271*, 2014.

[42] G. Hardy, J. Littlewood, and G. Pólya, *Inequalities. Cambridge Mathematical Library Series.* Cambridge University Press, 1967.

[43] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*, 10. Springer series in statistics, 2001, vol. 1.

[44] C.-J. Hsieh, M. A. Sustik, I. S. Dhillon, and P. K. Ravikumar, “Sparse inverse covariance matrix estimation using quadratic approximation,” in *Advances in Neural Information Processing Systems*, 2011, pp. 2330–2338.

[45] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[46] T. R. Golub, D. K. Slonim, P. Tamayo, *et al.*, “Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring,” *Science*, vol. 286, no. 5439, pp. 531–537, 1999.

[47] D. Croft, A. F. Mundo, R. Haw, *et al.*, “The reactome pathway knowledgebase,” *Nucleic Acids Research*, vol. 42, no. D1, pp. D472–D477, 2013.

[48] F. R. Davey, W. N. Erber, K. C. Gatter, and D. Y. Mason, “Abnormal neutrophils in acute myeloid leukemia and myelodysplastic syndrome,” *Human Pathology*, vol. 19, no. 4, pp. 454–459, 1988.

[49] H. Liu, F. Han, M. Yuan, J. Lafferty, and L. Wasserman, “High-dimensional semiparametric Gaussian copula graphical models,” *The Annals of Statistics*, vol. 40, no. 4, pp. 2293–2326, 2012.

[50] T. Zhao, H. Liu, K. Roeder, J. Lafferty, and L. Wasserman, “The huge package for high-dimensional undirected graph estimation in R,” *Journal of Machine Learning Research*, vol. 13, pp. 1059–1062, Apr. 2012.

[51] MSCI, “The Global Industry Classification Standard (GICS),” 2019. [Online]. Available: https://www.msci.com/gics, Accessed on May. 31, 2019.