Learning structured matrices in high dimensions: Low rank estimation and structured graphical models

Karthik Mohan

A dissertation submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2015

Reading Committee:

Maryam Fazel, Chair

James Burke

Jeff Bilmes

Program Authorized to Offer Degree:
Electrical Engineering

University of Washington

Abstract

Learning structured matrices in high dimensions:
Low rank estimation and structured graphical models

Karthik Mohan

Chair of the Supervisory Committee:
Associate Professor Maryam Fazel
Electrical Engineering

The topic of learning matrix structures in the *high-dimensional statistical setting* has received a lot of attention in machine learning, statistics and signal processing literature. High dimensional setting refers to problems that have more parameters to estimate than samples or measurements. Examples of problems that fall in this area include matrix factorization, matrix rank minimization, and graphical model estimation. These problems arise in many applications including collaborative filtering, system identification, and learning gene-regulatory networks.

The focus of this thesis is on the algorithmic and theoretical aspects of learning matrix structures in the high dimensional setting with emphasis on the two problems of *matrix rank minimization* and *graphical model estimation*.

We first consider the problem of reconstructing a low-rank matrix given a few linear measurements. This problem is known to be NP-hard. We propose a family of *iterative reweighted algorithms* that reconstruct the low-rank matrix efficiently and accurately. We also give recovery guarantees for these algorithms under suitable assumptions on the measurement maps. Finally, we also apply our algorithms to the problems of *system identification* and *matrix completion*. Our extensive numerical experiments illustrate that our algorithms perform better than state of the art algorithms on both synthetic and real data when the rank of the true underlying matrix is unknown.

We next consider the problem of learning *structured* Gaussian graphical models in the high dimensional setting. This problem has a lot of applications ranging from

biological modeling to social networks. While existing literature focuses on learning Gaussian graphical models using a simple sparsity regularizer (e.g. the graphical lasso formulation), applications often present prior knowledge that require more structured regularizers. Examples of networks that occur in applications include *scale-free networks* (where a few nodes have a high degree), *community networks* (where the connections occur in small dense communities), etc. We propose the *structured graphical lasso* formulation, a framework that generalizes graphical lasso to learning graphical models with arbitrary user-specified structures.

We continue by considering a special case of the structured graphical lasso that pertains to learning graphical model with a few *hub nodes*. We propose an ADMM algorithm to solve this formulation. Empirical studies indicate the superiority of hub graphical lasso over graphical lasso and other traditional methods in learning graphs with few hubs. When the number of hubs to be learned is $O(1)$, hub graphical lasso requires far fewer samples than graphical lasso to learn the right graphical model.

We move on to study the problem of estimating *multiple* Gaussian graphical models where information is shared between the graphical models. We propose different formulations that capture different ways in which information is shared between the networks. Our empirical results indicate that our approach does better than existing approaches in estimating gene-regulatory networks with shared information.

We switch gears by focusing on methods to reduce computational time for our models. We note that the proposed ADMM algorithm for hub graphical lasso and other formulations, though effective, involves a singular value decomposition in each iteration whose computational cost can be prohibitive for large problem sizes. To alleviate the computational bottle neck, we first provide algorithm independent approaches (based on screening rules) to speed up our proposed convex formulations. These screening rules are used to break down the problem into independent sub-problems (where possible), yielding significant speed-ups in computation. Typical algorithms for learning sparse graphical models require a Cholesky computation to check for positive definiteness of the iterates. We propose a block-coordinate descent algorithm with step-size selection rules that avoid expensive Cholesky checks and only rely on matrix vector multiplies in each iteration. Our empirical results show significant speed ups over the SVD based ADMM algorithm.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

Mathematical notation

1. $\mathbb{R}^{p\times p}$ - Set of all $p \times p$ matrices with entries being real valued.

2. rank(X) - Matrix rank of the matrix X.

3. card(x) - Cardinality or the number of non-zero entries in the vector x.

4. $\|x\|_0$ - See card(x).

5. $\|x\|_1$ - The ℓ_1 norm or the sum of absolute values of entries of x.

6. x_i - The ith entry of vector x.

7. x^{-i} - A vector that equals x except in the ith entry where it equals 0. $x_j^{-i} = \begin{cases} x_j & \text{if } j \neq i \\ 0 & \text{otherwise} \end{cases}$

8. $X_{\cdot i}$ - For a matrix X, $X_{\cdot i}$ denotes the ith column of X.

9. $\mathbf{0}$ - Denotes the vector of all zeros.

10. trace(X) - The sum of the diagonal entries of a square matrix X.

11. e_i - A vector that has a 1 in the ith coordinate and is zero elsewhere.

12. $\mathcal{S}^p$ - Set of all symmetric $p \times p$ matrices.

13. $\mathcal{S}^p_{++}$ - Set of all positive definite matrices in $\mathcal{S}^p$.

14. $X \succeq 0$ - Denotes that the symmetric matrix X is positive semi-definite.

15. $X \succ 0$ - Denotes that the symmetric matrix X is positive definite.

16. $\|x\|_q$ - ℓ_q norm of vector x. $\|x\|_q = \left(\sum_{i=1}^{q}\right)^{1/q}$.

17. $\text{vec}(X)$ - Given a matrix $X \in \mathbb{R}^{p\times p}$, $\text{vec}(X) \in \mathbb{R}^{p^2}$ denotes the vectorization of X obtained by stacking the columns of X into a vector.

18. $\langle X, Y\rangle$ - Euclidean inner-product where $\langle X, Y\rangle = \sum_{i=1}^{p}\sum_{j=1}^{p} X_{ij}Y_{ij} = \text{trace}(X^TY)$.

19. $\|X\|_F$ - Frobenius norm of X given by $\|X\|_F = \|\text{vec}(X)\|_2$. Note that $\|X\|_F^2 = \text{trace}(X^TX) = \langle X, X\rangle$.

20. $\sigma(X)$ - The vector of singular values of X.

21. $\sigma_i(X)$ - The ith largest singular value of X.

22. $\|X\|_*$ - Nuclear norm of X given by $\|X\|_* = \sum_{i=1}^{p}\sigma_i(X)$.

23. Schatten-q norm is given by $\|\sigma(x)\|_q$.

24. Spectral norm - The maximum singular value of the matrix X.

25. X^q for symmetric and positive definite X - Let $X = V\Sigma V^T$ be the eigen decomposition of X. Then $X^q = V\Sigma^q V^T$.

26. $(X^TX)^{q/2} := V\sigma^qV^T$ where $X = U\sigma V^T$ denote the singular value decomposition of X.

27. $\log\det(X)$ - $\log\det(X)$ denotes the log of the determinant of X and the domain of the function is the set of all symmetric and positive definite matrices.

28. Adjoint ($\mathcal{A}^*$) of linear operator ($\mathcal{A}$) - Given a linear operator $\mathcal{A}$, its adjoint operator $\mathcal{A}^*$ satisfies $\langle\mathcal{A}(X), Y\rangle = \langle X, \mathcal{A}^*(Y)\rangle$.

29. supp(X) - Support or the indices corresponding to the non-zero entries of X.

30. $\mathcal{N}(\mathcal{A})$: Null-space of the linear operator $\mathcal{A}$.

31. Diag(Z) - Given a matrix $Z \in \mathbb{R}^{p \times p}$, Diag($Z$) is a diagonal matrix whose diagonals are the diagonals of Z.

32. argmin - Refers to the global minimum of an optimization problem.

33. argmax - Refers to the global maximum of an optimization problem.

34. $\|V\|_{u,v}$ - The ℓ_u/ℓ_v norm of a matrix V given by the ℓ_u norm of a vector w, where the ith entry, $w_i = \|V_i\|_v$ or the ℓ_v norm of the ith column of V.

35. $\|V\|_\infty$ - For a matrix V, $\|V\|_\infty = \max_{ij} |V_{ij}|$ denotes the infinity norm, given by the maximum absolute entry in V.

36. A_T - For a matrix A and a set of indices T, A_T refers to the restriction of A to the support defined by T. I.e. $(A_T)_{ij} = A_{ij}$ if $(i,j) \in T$ and $(A_T)_{ij} = 0$ otherwise.

37. T^c - Complement of the set T.

Acronyms and other definitions

1. SVD - Singular value decomposition

2. IRLS - Iterative reweighted least squares

3. IRLS-GP - The gradient projection implementation for IRLS

4. sIRLS - A simpler version of the IRLS algorithm

5. FR - Degrees of freedom ratio

6. RIP - Restricted isometry property.

7. NSP - Null space property.
8. GMRF - Gaussian MRF, where the random variables are Gaussian.
9. High dimensional setting - A setting where the problem dimension, p is greater than the number of samples, n.
10. MLE - Maximum likelihood estimator.
11. GL - Graphical lasso
12. RCON - Row column overlap norm
13. HGL - Hub graphical lasso
14. SGL - Structured graphical lasso
15. ADMM - Alternating direction method of multipliers algorithm
16. RCON - Row-column overlap norm
17. PNJGL - Perturbed node joint graphical lasso
18. CNJGL - Common hub node joint graphical lasso
19. FGL - Fused graphical lasso
20. SBCD - SVD-free block coordinate descent
21. Glasso - An algorithm to optimize the graphical lasso formulation
22. HGlasso - Hub Glasso, i.e. an algorithm to optimize the HGL formulation

ACKNOWLEDGMENTS

There is a popular piece of wisdom that goes something like this - "You can't necessarily change circumstances or people's behavior to your liking. All you can change for sure is how you respond to circumstances and to people". Phd for me has been that phase of my life where I have experienced this truth repeatedly and have had many opportunities to see situations and people in a different *light* and thus completely transform my experience to a more positive one. Indeed, I have come to believe and experience that happiness is more of an internal state than something dependent on external circumstances. There are many people I have to thank for and there are also many situations and phases of my Phd I am very grateful to have experienced, even if only in retrospect.

First and foremost, I would like to thank my advisor, Maryam Fazel for supporting me and guiding me through the ups and downs of my phd. She has taught me both the importance of listening critically and critiquing ideas carefully. I would often come up with complicated ideas to work on, only to come back from a meeting appreciating a simpler way to do the same. I would like to thank my committee members - Maryam Fazel, Jim Burke, Jeff Bilmes, Marina Meila and Don Percival for their support and willingness to listen to my ideas and discuss my research with them. In particular, I would like to thank Jim for being willing to brainstorm, collaborate and advise on algorithms for different optimization problems. I would like to thank Jeff for being a big moral support and inspiration during testing times and for being always willing to take few minutes off his schedule to address any concerns I have. I have also enjoyed sitting in and being a TA for Jeff's classes. My big thanks to Radha for boosting my confidence during critical times and believing in me. I would also like to thank a previous advisor of mine for reflecting to me my own boundaries and thus enabling me to change them for the better. Thanks also to Daniela Witten and Su-In Lee for collaborating on the graphical models project.

My deep gratitude goes to my family for their love and support throughout life. I

would like to thank my friends including Dj, Bharath, Rishabh, Ankit, Raghu, Vamsi and Kannan for interesting discussions and fun times without which the Phd would have been a monotonous experience. I would like to thank my lab mates Dennis, Amin, Reza and Palma for brainstorming ideas during group meetings and giving valuable feedback. Special thanks to Ting-kei for proof reading one of my papers and to Jiashan for working with me on the last (but exciting nonetheless) project of my Phd.

Life is a balancing act and that's something Phd students can find tough to achieve due to the pressure to make progress in research. Fortunately squash and running have helped me on that front, giving me a break at the end of the day and keeping me healthy. I have many of my squash friends to thank for including Tor, Christian, Bela, Abid and my coach Joel who have pushed me and taken the journey with me to become good squash players. I would also like to thank everyone I have met during my Phd who have had an impact on me or vice-versa - It has been my great honor to be that person for you.

Now, I come to my favorite part of this acknowledgement section. There are some experiences in life that could shake you to the core, and make you question everything you have taken for granted. Experiences that can bring you to a realization that nothing really matters in life unless you decide it does. I had one such experience during my transition from civil engineering to Electrical engineering. I had to deal with the fear and uncertainity of where my future research direction lay, how I would be supported and would it all come together, if at all. Overcoming these fears, and forging ahead successfully has given me a perspective that's invaluable and I am deeply grateful to life for that.

DEDICATION

To my family and to the divine

Chapter 1

INTRODUCTION

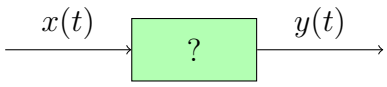
Big data has revolutionized machine learning by giving practitioners the ability to learn complex non-linear models without the curse of over-fitting. A recent success story is the use of deep learning to more accurately solve problems in computer vision and speech processing using massive amounts of data. While big data is the scenario where n, the number of measurements or samples is significantly greater than p, the dimensionality of the problem, in many applications the orthogonal problem is also of interest: Can we learn meaningful models in the *high-dimensional statistical setting* where $n << p$? A few years back, Netflix announced a million dollar prize for improving the accuracy in estimating movies-users ratings given only partial information on what movies each users like. The 100 million ratings data set for example provided 100 million data points to estimate a total of 1 billion ratings, clearly a problem in the high-dimensional setting. Another example that arises in biology is the problem of learning correlations between genes in gene-regulatory networks. In this scenario the gene-expression data obtained from patient records only yield a few hundred or thousand data points while the number of possible genes to consider numbers in tens of thousands. A third example that arises in systems theory is the problem of identifying a linear, time-invariant (LTI) system from input/output data, which could be limited due to measurement costs or other factors. The common theme in the above three examples is that the number of parameters to learn is significantly higher than the limited number of measurements or samples available. Thus learning any meaningful model in these settings is futile unless we have, and can leverage *prior information* into the model, therefore alleviating overfitting that these settings are prone to.

1.1 Prior information

Prior information can show up in the above examples in myriad ways. We explore this in the following problems.

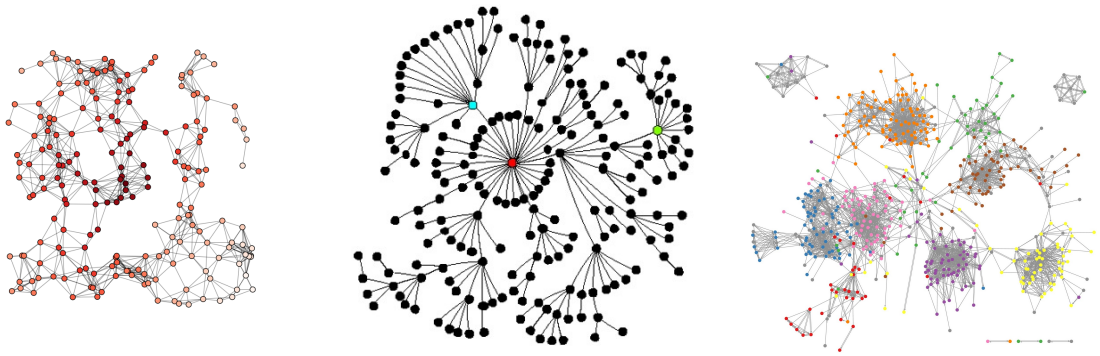
- The Netflix prize problem can be posed as a matrix completion problem, where we are interested in *completing* a partial users-movies ratings matrix. It is intuitive that users can be categorized into reasonably small clusters based on the kind of movies they like. Similarly, movies can be categorized into a small number of clusters based on their genres and year of release, etc. Though there are infinite number of ways to complete a matrix, we would like to do so in a way that accounts for dependencies between users and between movies. The notion of dependency therefore acts as a prior and can be captured through the matrix rank of the ratings matrix. Thus a low matrix rank implies that the users and movies can be categorized into a small number of clusters, therefore matching our prior.

- In the system identification problem, it is of interest to estimate an LTI system from input-output data pairs $(x(t), y(t))$.



  When we have few data points, we could possibly fit many LTI systems that match the input-output data. A prior that we could use is the Occam's razor principle: Among all possible models that fit the given data, pick the one that is the simplest (this also helps avoid overfitting). The notion of simplicity for LTI systems can be captured through the model order of the system. Thus, we would want to fit an LTI system that has a low model order. This in turn can be reformulated [Fazel et al., 2001, Mohan and Fazel, 2010a] as minimizing the rank of an appropriate Hankel matrix subject to affine constraints.

- In biological and social network applications, it is often of interest to learn a graphical model that is interpretable. *Sparse* graphs (or graphs with few edges) are more interpretable than dense graphs, and thus sparsity becomes a prior for the problem of learning the structure of graphical models. However, these applications often have more structure than just sparsity (see Figure 1.1).



(a) Random sparse network (b) Scale free network (c) Community structured network

Figure 1.1: Examples of graphs with structure. From left to right: Random sparse graph, scale free graph and community structured graph.

For example, in the problem of learning the structure of gene-regulatory networks from gene-expression data, it is known that certain genes play a regulatory role in controlling the expression of many other genes. These regulatory genes thus tend to be highly connected to other genes, while other genes may have sparse connections. Thus a more structured prior than just sparsity is to estimate a sparse graphical model with a few *hub nodes* (that correspond to the regulatory genes).

Similarly, it is well known [Gopalan et al., 2013, Yang and Leskovec, 2014] that social networks tend to exhibit a community structure - That is there are many clusters of nodes with dense connections *within clusters* but sparse connections *between clusters*. The overall social network could still be sparse but structured through the presence of communities.

In the next few sections, we detail the problems that will be the focus of this thesis and discuss our contributions.

1.2 Matrix rank minimization

The affine matrix rank minimization problem (or ARMP, discussed in Chapter 2) is as follows:

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \text{rank}(X) \\ \text{subject to} & \mathcal{A}(X) = b, \end{array} \tag{1.1}$$

where $X \in \mathbb{R}^{m \times n}$ is the optimization variable, $\mathcal{A} : \mathbb{R}^{m \times n} \to \mathbb{R}^q$ is a linear map, and $b \in \mathbb{R}^q$ denotes the measurements. Also $\text{rank}(X)$ denotes the matrix rank of the matrix X. Matrix completion is a special case of the above problem that arises in recommender systems and can be posed as the following optimization problem:

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \text{rank}(X) \\ \text{subject to} & X_{ij} = M_{ij} \; \forall (i,j) \in C, \end{array} \tag{1.2}$$

where C is a subset of user-movie indices (i, j) for which the ratings M_{ij} are known. Problem (1.1) is known to be NP-hard. In Chapter 2, we present a family of *reweighted algorithms* for ARMP that perform better than the traditional convex heuristic for this problem and discuss the properties of these algorithms. It is also of interest to know what is the minimum size of the set C for which (1.2) can be solved using an appropriate heuristic? These kind of results are referred to as recovery results as they guarantee the recovery of the optimal solution to (1.2) through a suitable optimization method when appropriate conditions hold on the measurements. We also give recovery results in Chapter 2, some of which indicate that our reweighted algorithms are better than the standard convex heuristic and we also verify this to be true in empirical experiments (see Chapter 2).

1.3 Learning structured graphical models

While a lot of literature on graphical models has focused on the problem of inference [Wainwright and Jordan, 2008], learning the structure of graphical models has become a recent topic of interest especially in biological and social network applications [Filkov, 2005, Robins et al., 2007]. In Chapter 3, we introduce graphical models and present convex formulations for learning a single graphical model given structured

prior information. We also discuss algorithms for learning a graphical model and present numerical results on learning a graph with a few hubs. It is also of interest to know if the convex formulations we propose are theoretically learning the right model? We address this question in Appendix A of the thesis. We can guarantee successful model-selection for our approach with far fewer samples than the state of the art convex heuristic (graphical lasso) that only uses a sparse prior. In the context of gene-regulatory networks, it is of interest to simultaneously learn multiple graphical models that share information in a structured way. For example, we may wish to learn how the correlation between genes change in the two different sub-types of cancer. In Chapter 4 we propose convex formulations to learn two (or more) graphical models, corresponding to the two sub-types and impose prior information on similarity and differences that we expect to see between the two graphical models. This improves over the state of the art in being able to be learn structured similarities and differences in graphical models, thus utilizing more prior knowledge and yielding better estimates.

1.4 Speeding up algorithms

The algorithms proposed in Chapters 3 and 4 are effective in learning the right model as seen from the empirical results. However, one bottle neck of the proposed algorithms is the reliance on expensive linear algebra computations (esp. singular value decomposition) in each iteration, making these algorithms not easily scalable to larger problem sizes. To alleviate this issue, in Chapter 5 we discuss screening rules that can speed up any algorithm by breaking down the problem into smaller sub-problems that can be solved efficiently (and also in parallel). We also propose a framework of fast block-coordinate descent methods that applies to a range of problems in structure learning of graphical models. These algorithms do not require expensive linear algebra computations such as SVD, Cholesky or QR factorizations. Instead they only rely on matrix-vector multiplications in each iteration. We observe in our empirical results that these algorithms are significantly faster than algorithms that do use SVD.

1.5 Organization of the thesis

- In Chapter 2 we look at the problem of affine rank minimization and specifically matrix completion, discussing algorithmic approaches and guarantees on

recovery. We also present results of the algorithms on the movielens data set. The results in these chapters appear in [Mohan and Fazel, 2010a, 2012, Oymak et al., 2011a, Mohan and Fazel, 2010b].

- In Chapters 3, we introduce the problem of structure learning of a graphical model and discuss algorithms for learning a single graphical model with hubs. We also present consistency results for learning a graphical model with hubs in the Appendix of this thesis. The results in this chapter appear in part in [Tan et al., 2014].

- In Chapter 4, we consider the problem of learning multiple structured graphical models. We discuss convex formulations, algorithms and empirical results that demonstrate the efficacy of our formulations. The results in this chapter appear in [Mohan et al., 2014, 2012].

- In Chapter 5, we discuss screening rules to speed up algorithms by leveraging the structure of the optimal solution. We also present fast block coordinate descent algorithms to learn structured graphical models that do not require expensive linear algebra operations such as SVD or Cholesky decomposition. This enables us to more efficiently learn graphical models over larger problem sizes as compared to traditional optimization algorithms such as ADMM that rely on SVD computations. The results in this chapter, appear in part in [Mohan et al., 2014].

Chapter 2

REWEIGHTED ALGORITHMS FOR THE AFFINE RANK MINIMIZATION PROBLEM

2.1 Affine rank minimization

The Affine Rank Minimization Problem (ARMP), or the problem of finding the *minimum rank matrix* in an affine set, arises in many engineering applications such as collaborative filtering [e.g., Candes and Recht, 2009, Srebro et al., 2005], low-dimensional Euclidean embedding [e.g., Fazel et al., 2003], low-order model fitting and system identification [e.g., Liu and Vandenberghe, 2008], and quantum tomography [e.g., Gross et al., 2010]. The problem is as follows,

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \text{rank}(X) \\ \text{subject to} & \mathcal{A}(X) = b, \end{array} \tag{2.1}$$

where $X \in \mathbb{R}^{m \times n}$ is the optimization variable, $\mathcal{A} : \mathbb{R}^{m \times n} \to \mathbb{R}^q$ is a linear map, and $b \in \mathbb{R}^q$ denotes the measurements. Also $\text{rank}(X)$ denotes the matrix rank of the matrix X. When X is restricted to be a diagonal matrix, ARMP reduces to the *cardinality minimization* or sparse vector recovery problem,

$$\begin{array}{ll} \underset{x}{\text{minimize}} & \text{card}(x) \\ \text{subject to} & Ax = b, \end{array} \tag{2.2}$$

where $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$ and $\text{card}(x)$ denotes the number of non-zeros entries of x.

The cardinality minimization problem has important applications in the field of *compressive sensing and sampling* [Candes and Tao, 2004], where given compressed measurements b of signal x and a design matrix A, the goal is to recover the true signal x. Although the standard signal processing theory requires that the signal be sampled at the Nyquist rate, the theory of compressive sensing [Candes and Tao, 2004] has shown that if the signal is sparse, much fewer measurements or samples are

required to fully recover the true signal x.

Both the cardinality minimization problem and ARMP are known to be NP-hard. An important question in this area has been to give fast algorithms to solve these problems efficiently along with *guarantees of recovery* of the true signal or low-rank matrix. Another important question is the design of matrices A for which recovery of true signals can be shown. Our contributions are in coming up with fast algorithms with recovery guarantees for ARMP. Before we move onto the contributions, we review the relevant literature in ARMP.

2.2 Literature review

We review related algorithms for recovering sparse vectors and low rank matrices. A commonly used convex heuristic for the cardinality minimization problem is ℓ_1 minimization - which involves replacing the $\|.\|_0$ norm (or cardinality of a vector) in the objective of (2.2) by the $\|.\|_1$ norm. The field of compressive sensing has shown that under certain conditions on the matrix A, this heuristic solves the original cardinality minimization problem [Candes and Tao, 2004]. There are also geometric arguments in favor of ℓ_1 norm as a good convex relaxation [see, e.g., Chandrasekaran et al., 2012a, for a unifying analysis for general linear inverse problems]. Other approaches for recovery of sparse vectors from linear measurements include greedy algorithms [see, e.g., Needell and Tropp, 2008, Goldfarb and Ma, 2011, Garg and Khandekar, 2009, for CoSaMP, IHT, and GraDes respectively] and reweighted approaches. It has also been observed empirically [e.g., Candes et al., 2008, Lobo et al., 2006] that by appropriately weighting the ℓ_1 norm and iteratively updating the weights, the recovery performance of the algorithm is enhanced. It is shown theoretically that for sparse recovery from noisy measurements, this algorithm has a better recovery error than ℓ_1 minimization under suitable assumptions on the matrix A [Needell, 2009, Zhang, 2010]. The reweighted ℓ_1 algorithm has also been generalized to the recovery of low-rank positive semi-definite matrices [Fazel et al., 2003, Mohan and Fazel, 2010c]. Another heuristic involves reweighting the ℓ_2 norm instead of the ℓ_1 norm and is known as the *iterative reweighted least squares* algorithm [Rao and Kreutz-Delgado, 1999, Wipf and Nagarajan, 2010, Daubechies et al., 2010]. This algorithm, like the reweighted ℓ_1 has been shown to improve on the recovery performance of ℓ_1 minimization.

For the ARMP, analogous algorithms have been proposed including nuclear norm

minimization [Fazel et al., 2001], reweighted nuclear norm minimization [Mohan and Fazel, 2010c, Fazel et al., 2003], as well as greedy algorithms such as AdMiRA [Lee and Bresler, 2010] which generalizes CoSaMP, SVP [Meka et al., 2010], a hard-thresholding algorithm that we also refer to as IHT, and Optspace [Keshavan and Oh, 2009]. Developing efficient implementations for nuclear norm minimization is an important research area since standard semidefinite programming solvers cannot handle large problem sizes. Towards this end, algorithms including SVT [Cai et al., 2008], NNLS [Toh and Yun, 2010], FPCA [Goldfarb and Ma, 2011] have been proposed. A spectral regularization algorithm [Mazumder et al., 2010] has also been proposed for the specific problem of matrix completion.

2.3 Summary of contributions

Our contributions are in coming up with a family of reweighted algorithms that perform better than the standard nuclear norm heuristic, along with recovery guarantees. We discuss our algorithms, *reweighted nuclear norm* and IRLS, along with efficient implementations and applications in Chapter 2. We also discuss recovery guarantees for the problem of nuclear norm minimization. First, we present a seamless way to extend recovery conditions for the ℓ_1 minimization heuristic (for the compressive sensing problem [Candes and Tao, 2004]) to recovery conditions for nuclear norm minimization (for the affine rank minimization problem). By recovery conditions for an algorithm, we mean conditions that guarantee that the algorithm recovers the solution of interest. Secondly, we present recovery conditions for the reweighted trace heuristic [Mohan and Fazel, 2010b] and for the IRLS-1 algorithm [Mohan and Fazel, 2012].

We next describe our contributions in developing reweighted algorithms for ARMP with recovery guarantees. Our reweighted algorithms are the appropriate analogs of the reweighted algorithms proposed in the compressive sensing literature [Candes et al., 2008, Lobo et al., 2006, Wipf and Nagarajan, 2010, Daubechies et al., 2010], specifically, reweighted ℓ_1 and iterative reweighted least squares (IRLS).

2.3.1 Notation

Before we discuss our contributions, we set up notation to be used in the rest of this chapter. Let $X \in \mathbb{R}^{p\times p}$ denote a $p \times p$ matrix with entries taking real values. For a symmetric matrix X, let $\text{trace}(X) = \sum_{i=1}^{p} X_{ii}$ denote the trace or sum of the diagonal entries of a matrix X. We refer to a symmetric matrix, X as positive semi-definite (resp. positive definite) if X has non-negative (resp. positive) eigen values. We denote a positive semi-definite matrix by $X \succeq 0$. For a vector $x \in \mathbb{R}^p$, let $\|x\|_q = (\sum_{i=1}^{p} x_i^q)^{1/q}$ denote the ℓ_q norm of x. For a matrix $X \in \mathbb{R}^{p\times p}$, let $\text{vec}(X) \in \mathbb{R}^{p^2}$ denote the vectorization of X obtained by stacking the columns of X on top of each other. For two matrices $X, Y \in \mathbb{R}^{p\times p}$, denote the Euclidean inner-product by $\langle X, Y\rangle$. Let the Frobenius norm of X be given by $\|X\|_F = \|\text{vec}(X)\|_2$. Note that $\|X\|_F^2 = \text{trace}(X^TX) = \langle X, X\rangle$. Let $\text{rank}(X)$ denote the matrix rank of the matrix X. The nuclear norm of a matrix X is given by $\|X\|_* = \sum_{i=1}^{p} \sigma_i(X)$, i.e. the sum of singular values of the matrix X. The nuclear norm is known to be the best convex relaxation to the non-convex matrix rank function, $\text{rank}(X)$. Let $X = U\Sigma V^T$ denote the singular value decomposition of a matrix X. The matrix Σ is a diagonal matrix whose diagonals are the singular values of the matrix X. Note that the eigen-decomposition of $X^TX = V\Sigma^2V^T$. Similarly let $(X^TX)^{q/2} = V\Sigma^qV^T$. If $X = V\sigma V^T$ is symmetric and positive definite, then denote $X^q = V\Sigma^qV^T$. Let $\sigma(X)$ denote the vector of singular values of X. Then the Schatten-q norm of X is given by $(\text{trace}((X^TX)^{q/2}))^{1/q} = \|\sigma(X)\|_q$. For $q = 1$, the Schatten-q norm of X is the same as the nuclear norm of X. For $q = 2$, we get the Frobenius norm of X. For $q = \infty$, we get the spectral norm of X, denoted by $\|X\|$. The spectral norm is the maximum singular value of X and is also equal to the standard operator norm, i.e. $\|X\| = \max_z \|Xz\|_2/\|z\|_2$. For a symmetric and positive definite X, $\log\det(X)$ denotes the log of the determinant of X. For a linear operator, $\mathcal{A}$ acting on a matrix X, its adjoint $\mathcal{A}^*$ satisfies $\langle \mathcal{A}(X), Y\rangle = \langle X, \mathcal{A}^*(Y)\rangle$.

2.4 Reweighted trace heuristic

As an extension of the reweighted ℓ_1 minimization, the *reweighted trace heuristic* [Fazel et al., 2003] has been proposed to solve ARMP with a positive semi-definite

constraint on X. The heuristic is given by,

$$X^{k+1} = \underset{X:\mathcal{A}(X)=b, X\succeq 0}{\operatorname{argmin}} \operatorname{trace}(X^k + \delta I)^{-1} X. \tag{2.3}$$

When a matrix is positive semi-definite, its nuclear norm, $\|X\|_* = \operatorname{trace}(X)$. Thus, the reweighted trace heuristic can be can be seen as a variant of reweighted nuclear norm when the matrices are restricted to be positive semi-definite. The weight matrix at the $(k+1)th$ iteration is given by $(X^k + \delta I)^{-1}$. This is analogous to reweighted ℓ_1, where the weight vector w^k at $(k+1)th$ iteration is given by $w_i^k = 1/(x_i^k + \delta)$. The reweighted trace heuristic can be extended to recovery of matrices that are not positive semi-definite. The more general reweighted heuristic is given as follows:

$$\begin{array}{ll} \text{minimize} & \operatorname{trace}(Y^k + \delta I)^{-1} Y + \operatorname{trace}(Z^k + \delta I)^{-1} Z \\ \text{subject to} & \begin{bmatrix} Y & X \\ X^T & Z \end{bmatrix} \succeq 0, \mathcal{A}(X) = b. \end{array} \tag{2.4}$$

To see the motivation behind the form in (2.4), note that ARMP can be equivalently expressed as follows [Fazel et al., 2003]:

$$\begin{array}{ll} \text{minimize} & \operatorname{rank}(Y) + \operatorname{rank}(Z) \\ \text{subject to} & \begin{bmatrix} Y & X \\ X^T & Z \end{bmatrix} \succeq 0, \mathcal{A}(X) = b. \end{array} \tag{2.5}$$

A well-known non-convex surrogate to $\|x\|_0$ norm is $\sum_i \log(|x_i| + \delta)$. Similarly, $\log\det(X)$ is a suitable non-convex surrogate [Fazel et al., 2003] for $\operatorname{rank}(X)$ when $X \succeq 0$. Replacing the rank function by $\log\det$ in (2.5), we get,

$$\begin{array}{ll} \text{minimize} & \log\det(Y) + \log\det(Z) \\ \text{subject to} & \begin{bmatrix} Y & X \\ X^T & Z \end{bmatrix} \succeq 0, \mathcal{A}(X) = b. \end{array} \tag{2.6}$$

The generalized reweighted trace heuristic is nothing but a majorization-minimization (MM) algorithm applied to minimize (2.6). Thus, we have the following convergence result [Mohan and Fazel, 2010c] on the heuristic:

Theorem 1. *Every convergent subsequence of the reweighted trace heuristic (2.4) converges to a stationary point of (2.6). Further the norm of the difference between successive iterates tends to zero.*

The generalized reweighted trace heuristic can be interpreted as a *reweighted nuclear norm* [Mohan and Fazel, 2010c] as in Algorithm 1. We note that this algorithm reduces to the reweighted trace heuristic when $X \succeq 0$. The reweighted nuclear norm was able to identify a low-order linear time-invariant (LTI) system better than nuclear norm in a system identification problem. Though showing promise, Algorithm 1 involves many matrix multiplications in each iteration and is hence less efficient. We next describe a family of algorithms that are also based on non-convex surrogates to rank, but are much more efficient to implement and perform well in practice.

Algorithm 1: Reweighted nuclear norm heuristic for ARMP

Set: $k = 0, W_1^k = I, W_2^k = I, \delta = 1e-9$;
while *Not converged* **do**
$X^{k+1} \leftarrow \underset{X:\mathcal{A}(X)=b}{\operatorname{argmin}} \|W_1^k X W_2^k\|_*$;
$\hat{X}^{k+1} \leftarrow W_1^k X^{k+1} W_2^k$;
$Y^{k+1} \leftarrow (W_1^k)^{-1}(\hat{X}^{k+1}(\hat{X}^{k+1})^T)^{\frac{1}{2}}(W_1^k)^{-1}$;
$Z^{k+1} \leftarrow (W_2^k)^{-1}((\hat{X}^{k+1})^T\hat{X}^{k+1})^{\frac{1}{2}}(W_2^k)^{-1}$;
$W_1^{k+1} \leftarrow (Y^{k+1} + \delta I)^{-1}$;
$W_2^{k+1} \leftarrow (Z^{k+1} + \delta I)^{-1}$

2.5 IRLS algorithm

Our motivation behind the IRLS algorithm is to improve upon the performance of the nuclear norm minimization, while also being efficient. We now define a family of IRLS-p algorithms for $0 \leq p \leq 1$ as follows:

Each iteration of Algorithm 2 minimizes a weighted Frobenius norm of the matrix X, since $\text{trace}(W_p^{k-1}X^TX) = \|(W_p^{k-1})^{1/2}X\|_F^2$. While minimizing the Frobenius norm subject to affine constraints doesn't lead to low-rank solutions in general, through a careful reweighting of this norm we show that Algorithm 2 does indeed produce low-rank solutions under suitable assumptions. Usually a reweighted algorithm trades

Algorithm 2: IRLS-p Algorithm for Matrix Rank Minimization with $0 \leq p \leq 1$

Data: $\mathcal{A}$, b
Set $k = 1$. Initialize $W_p^0 = I, \gamma^1 > 0$;
while *not converged* **do**
$X^k = \underset{X}{\operatorname{argmin}} \{\operatorname{trace}(W_p^{k-1} X^T X) : \mathcal{A}(X) = b\}$;
$W_p^k = (X^{k^T} X^k + \gamma^k I)^{\frac{p}{2}-1}$;
Choose $0 < \gamma^{k+1} \leq \gamma^k$;
$k = k + 1$;

off computational time for improved recovery performance when compared to the unweighted convex heuristic. As an example, the reweighted ℓ_1 algorithm for sparse recovery [Candes et al., 2008] and the reweighted nuclear norm algorithm [Mohan and Fazel, 2010c] for matrix rank minimization solve the corresponding standard convex relaxations (ℓ_1 and nuclear norm minimization respectively) in their first iteration. Thus these algorithms take at least as much time as the corresponding convex algorithms. However, the iterates of the IRLS-p family of algorithms simply minimize a weighted Frobenius norm, and have run-times comparable with the nuclear norm heuristic, while (for $p < 1$) also enjoying improved recovery performance. In the $p = 1$ case, we show that the algorithm minimizes a certain smooth approximation to the nuclear norm. This allows for efficient implementations, which we discuss in later sections. IRLS-1 also comes with theoretical guarantees similar to nuclear norm minimization.

2.5.1 Derivation of IRLS-p

Recall that replacing the rank function in (2.1) by $\|X\|_\star$ yields the nuclear norm heuristic,

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \|X\|_\star \\ \text{subject to} & \mathcal{A}(X) = b. \end{array}$$

We now consider other (convex and non-convex) smooth approximations to the rank function. Define the *smooth Schatten-p function* as

$$\begin{aligned} f_p(X) &= \text{trace}(X^T X + \gamma I)^{p/2} \\ &= \textstyle\sum_{i=1}^n (\sigma_i^2(X) + \gamma)^{\frac{p}{2}}. \end{aligned}$$

Note that $f_p(X)$ is differentiable for $p > 0$ and convex for $p \geq 1$. With $\gamma = 0$, $f_1(X) = \|X\|_\star$, which is also known as the Schatten-1 norm. With $\gamma = 0$ and $p \to 0$, $f_p(X) \to \text{rank}(X)$. For $p = 1$, one can also derive the smooth Schatten-1 function as follows.

$$f_1(X) = \max_{Z:\|Z\|_2 \leq 1} \langle Z, X \rangle - \sqrt{\gamma} d(Z) + n\sqrt{\gamma},$$

where $d(Z)$ is a smoothing term given by $d(Z) = \sum_{i=1}^n (1 - \sqrt{1 - \sigma_i^2(Z)}) = n - \text{trace}(I - Z^T Z)^{1/2}$. Plugging in $d(Z) = 0$ in (2.7) gives $\|X\|_\star + n\sqrt{\gamma}$. Hence $f_1(X)$ is a smooth approximation of $\|X\|_\star$ obtained by smoothing the conjugate of $\|X\|_\star$. By virtue of the smoothing, it is easy to see that

$$\|X\|_\star \leq f_1(X) \leq \|X\|_\star + n\sqrt{\gamma}.$$

As γ approaches 0, $f_1(X)$ becomes tightly bounded around $\|X\|_\star$. Also let X^* be the optimal solution to minimizing $\|X\|_\star$ subject to $\mathcal{A}(X) = b$ and let X^s be the optimal solution to minimizing $f_1(X)$ subject to the same constraints. Then it holds that

$$(\|X\|_\star - \|X^*\|_\star) \leq (f_1(X) - f_1(X^s)) + n\sqrt{\gamma} \quad \forall X : \mathcal{A}(X) = b.$$

Thus, to minimize $\|X\|_\star$ to a precision of ϵ, we need to minimize the smooth approximation $f_1(X)$ to a precision of $\epsilon - n\sqrt{\gamma}$.

It is therefore of interest to consider the problem

$$\begin{array}{ll} \underset{X}{\text{minimize}} & f_p(X) \\ \text{subject to} & \mathcal{A}(X) = b, \end{array} \tag{2.7}$$

the optimality conditions of which motivate the IRLS-p algorithms for $0 \leq p \leq 1$. We show that IRLS-1 solves the smooth Schatten-1 norm or nuclear norm minimization

problem, that is, finds a globally optimal solution to (2.7) with $p = 1$. For $p < 1$, we show that IRLS-p finds a stationary point of (2.7).

We now give an intuitive way to derive the IRLS-p algorithm from the KKT conditions of (2.7). The Lagrangian corresponding to (2.7) is

$$L(X, \lambda) = f_p(X) + \langle \widehat{\lambda},\ \mathcal{A}(X) - b \rangle,$$

and the KKT conditions are $\nabla_X L(X, \widehat{\lambda}) = 0$, $\mathcal{A}(X) = b$. Note that $\nabla f_p(X) = pX(X^TX + \gamma I)^{p/2-1}$ [see, e.g., Lewis, 1996]. Letting $\lambda = \frac{\widehat{\lambda}}{p}$, we have that the KKT conditions for (2.7) are given by

$$\begin{aligned} 2X(X^TX + \gamma I)^{p/2-1} + \mathcal{A}^*(\lambda) &= 0 \\ \mathcal{A}(X) &= b. \end{aligned} \tag{2.8}$$

Let $W_p^k = (X^{k^T}X^k + \gamma I)^{p/2-1}$. The first condition in (2.8) can be written as

$$X = -\frac{1}{2}\mathcal{A}^*(\lambda)(X^TX + \gamma I)^{1-p/2}.$$

This is a fixed point equation, and a solution can be obtained by iteratively solving for X as $X^{k+1} = \frac{1}{2}\mathcal{A}^*(\lambda)(W_p^k)^{-1}$, along with the condition $\mathcal{A}(X^{k+1}) = b$. Note that X^{k+1} and the dual variable λ satisfy the KKT conditions for the convex optimization problem,

$$\begin{aligned} \underset{X}{\text{minimize}} \quad & \text{trace} W_p^k X^T X \\ \text{subject to} \quad & \mathcal{A}(X) = b. \end{aligned}$$

This idea leads to the IRLS-p algorithm described in Algorithm 2. Note that we also let $p = 0$ in Algorithm 2; to derive IRLS-0, we define another non-convex surrogate function by taking limits over $f_p(X)$. For any positive scalar x, it holds that $\lim_{p\to 0} \frac{1}{p}(x^p - 1) = \log x$. Therefore,

$$\begin{aligned} \lim_{p\to 0} \frac{f_p(X) - n}{p} &= \tfrac{1}{2}\log\det(X^TX + \gamma I) \\ &= \textstyle\sum_i \tfrac{1}{2}\log(\sigma_i^2(X) + \gamma). \end{aligned}$$

Thus IRLS-0 can be seen as iteratively solving (as outlined previously) the KKT

conditions for the non-convex problem,

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \log\det(X^TX+\gamma I) \\ \text{subject to} & \mathcal{A}(X)=b. \end{array} \tag{2.9}$$

2.5.2 Convergence of IRLS-p

We show that the difference between successive iterates of the IRLS-p ($0 \leq p \leq 1$) algorithm converges to zero and that every cluster point of the iterates is a stationary point of (2.7). These results generalize the convergence results given for IRLS-1 in previous literature [Mohan and Fazel, 2010d, Fornasier et al., 2011] to IRLS-p with $0 < p \leq 1$. In this section, we drop the subscript on W_p^k for ease of notation. Our convergence analysis relies on useful auxiliary functions defined as

$$\mathcal{J}^p(X,W,\gamma) := \begin{cases} \frac{p}{2}(\text{trace}(W(X^TX+\gamma I)) + \frac{2-p}{p}\text{trace}((W)^{\frac{p}{p-2}})) & \text{if } 0<p\leq 1 \\ \text{trace}(W(X^TX+\gamma I)) - \log\det W - n & \text{if } p=0. \end{cases}$$

These functions can be obtained from the alternative characterization of Smooth Schatten-p function with details in the proofs section of this chapter. We can express the iterates of IRLS-p as

$$\begin{aligned} X^{k+1} &= \underset{X:\mathcal{A}(X)=b}{\text{argmin}}\ \mathcal{J}^p(X,W^k,\gamma^k) \\ W^{k+1} &= \underset{W\succ 0}{\text{argmin}}\ \mathcal{J}^p(X^{k+1},W,\gamma^{k+1}), \end{aligned}$$

and it follows that

$$\begin{aligned} \mathcal{J}^p(X^{k+1},W^{k+1},\gamma^{k+1}) &\leq \mathcal{J}^p(X^{k+1},W^k,\gamma^{k+1}) \\ &\leq \mathcal{J}^p(X^{k+1},W^k,\gamma^k) \\ &\leq \mathcal{J}^p(X^k,W^k,\gamma^k). \end{aligned} \tag{2.10}$$

The following theorems show the convergence of the iterates of the IRLS algorithm. The proofs are given in the proofs section of this chapter.

Theorem 2. *Given any $b \in \mathbb{R}^q$, the iterates $\{X^k\}$ of IRLS-p ($0 < p \leq 1$) satisfy*

$$\sum_{k=1}^{\infty} \|X^{k+1}-X^k\|_F^2 \leq 2D^{\frac{2}{p}},$$

where $D := \mathcal{J}^p(X^1, W^0, \gamma^0)$. *In particular, we have that* $\lim_{k\to\infty} \left(X^k - X^{k+1}\right) = 0$.

Theorem 3. *Let* $\gamma_{\min} := \lim_{k\to\infty} \gamma^k > 0$. *Then the sequence of iterates* $\{X^k\}$ *of IRLS-p* $(0 \le p \le 1)$ *is bounded, and every cluster point of the sequence is a stationary point of* (2.7) *(when* $0 < p \le 1$*), or a stationary point of (2.9) (when* $p = 0$*).*

2.5.3 IRLS-GP: An efficient implementation of IRLS for matrix completion

In Algorithm 2, note that each iteration involves solving a quadratic program in a matrix variable. This can be solved exactly with $O(m^6)$ complexity where m is the row-dimension of the matrix. However, an efficient way to solve the quadratic program would be to solve it through a gradient projection algorithm. We give an efficient implementation for the matrix completion problem, which we describe next.

The Matrix Completion Problem

The matrix completion problem is a special case of ARMP with constraints that restrict some of the entries of the matrix variable X to equal given values. The problem can be written as

$$\begin{array}{ll} \text{minimize} & \operatorname{rank}(X) \\ \text{subject to} & X_{ij} = (X_0)_{ij}, \quad (i,j) \in \Omega, \end{array}$$

where X_0 is the matrix we would like to recover and Ω denotes the set of entries which are revealed. We define the operator $\mathcal{P}_\Omega : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ as

$$(\mathcal{P}_\Omega(X))_{ij} = \begin{cases} X_{ij} & \text{if } (i,j) \in \Omega \\ 0 & \text{otherwise.} \end{cases} \tag{2.11}$$

Also, Ω^c denotes the complement of Ω, that is, all index pairs (i, j) except those in Ω.

IRLS-GP

To apply Algorithm 2 to the matrix completion problem (2.11), we replace the constraint $\mathcal{A}(X) = b$ by $\mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(X_0)$. Each iteration of IRLS solves a quadratic program (QP). A gradient projection algorithm could be used to efficiently solve

the quadratic program (QP) in each iteration of IRLS. We call this implementation IRLS-GP (Algorithm 3).

Algorithm 3: IRLS-GP for Matrix Completion

Data: Ω, $\mathcal{P}_\Omega(X_0)$
Result: $\widehat{X} : \mathcal{P}_\Omega(\widehat{X}) = b$
Set $k = 0$. Initialize $X^0 = 0, \gamma^0 > 0$, $s^0 = (\gamma^0)^{(1-\frac{p}{2})}$;
while *IRLS iterates not converged* **do**
  $W^k = (X^{kT}X^k + \gamma^k I)^{\frac{p}{2}-1}$. Set $X_{\text{temp}} = X^k$;
  while *Gradient projection iterates not converged* **do**
    $X_{\text{temp}} = \mathcal{P}_{\Omega^c}(X_{\text{temp}} - s^k X_{\text{temp}} W^k) + \mathcal{P}_\Omega(X_0)$;
  Set $X^{k+1} = X_{\text{temp}}$;
  Choose $0 < \gamma^{k+1} \leq \gamma^k$, $s^{k+1} = (\gamma^{k+1})^{(1-\frac{p}{2})}$;
  $k = k + 1$;

The step size used in the gradient descent step is $s^k = 1/L^k$, where $L^k = 2\|W^k\|$ is the Lipschitz constant of the gradient of the quadratic objective $\text{trace}(W^k X^T X)$ at the k^{th} iteration. We also warm-start the gradient projection algorithm to solve for the $(k+1)^{th}$ iterate of IRLS with the solution of the k^{th} iterate and find that this speeds up the convergence of the gradient projection algorithm in subsequent iterations. At each iteration of IRLS, computing the weighting matrix involves an inversion operation which can be expensive for large n. However we work around this through efficient SVD computations, details of which can be found in [Mohan and Fazel, 2012].

Although IRLS-GP can be much more efficient than solving the quadratic program exactly, Algorithm 3 does involve an inner loop that could take many iterations to converge in the first few outer iterations. To overcome this limitation of IRLS-GP, we propose a cleaner implementation of IRLS, that we call sIRLS. We discuss this in the next section.

2.5.4 sIRLS algorithm

In this section, we present the sIRLS-p family of algorithms that are related to the IRLS-GP implementation discussed in the previous section. We first describe the

algorithm before discussing its connection to IRLS-p.

Algorithm 4: sIRLS-p for Matrix Completion Problem

Data: $\mathcal{P}_\Omega$, b
Result: $\widehat{X} : \mathcal{P}_\Omega(\widehat{X}) = b$
Set $k = 0$. Initialize $X^0 = \mathcal{P}_\Omega(X_0), \gamma^0 > 0$, $s^0 = (\gamma^0)^{(1-\frac{p}{2})}$;
while *not converged* **do**
  $W_p^k = \left(X^{k^T} X^k + \gamma^k I\right)^{\frac{p}{2}-1}$;
  $X^{k+1} = \mathcal{P}_{\Omega^c}(X^k - s^k X^k W_p^k) + \mathcal{P}_\Omega(X_0)$;
  Choose $0 < \gamma^{k+1} \leq \gamma^k$, $s^{k+1} = (\gamma^{k+1})^{(1-\frac{p}{2})}$;
  $k = k + 1$

We note that sIRLS-p (Algorithm 4) is a gradient projection algorithm applied to

$$\begin{array}{ll} \underset{X}{\text{minimize}} & f_p(X) = \text{trace}(X^T X + \gamma I)^{\frac{p}{2}} \\ \text{subject to} & \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(X_0), \end{array} \tag{2.12}$$

while sIRLS-0 is a gradient projection algorithm applied to

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \log\det(X^T X + \gamma I) \\ \text{subject to} & \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(X_0). \end{array} \tag{2.13}$$

Indeed $\nabla f_p(X^k) = X^k W_p^k$ and the gradient projection iterates,

$$\begin{aligned} X^{k+1} &= \mathcal{P}_{\{X:\mathcal{P}_\Omega(X)=\mathcal{P}_\Omega(X_0)\}}(X^k - s^k \nabla f_p(X^k)) \\ &= \mathcal{P}_{\Omega^c}(X^k - s^k \nabla f_p(X^k)) + \mathcal{P}_\Omega(X_0) \end{aligned}$$

are exactly the same as the sIRLS-p (Algorithm 4) iterates when γ^k is a constant. In other words, for large enough k (i.e., k such that $\gamma^k = \gamma_{\min}$), the iterates of sIRLS-p and sIRLS-0 are nothing but gradient projection applied to (2.12) and (2.13) respectively, with $\gamma = \gamma_{\min}$. We have the following convergence results for sIRLS.

Theorem 4. *Every cluster point of sIRLS-p ($0 < p \leq 1$) is a stationary point of (2.12) with $\gamma = \gamma_{\min}$.*

Theorem 5. *Every cluster point of sIRLS-0 is a stationary point of (2.13) with $\gamma = \gamma_{\min}$.*

Connection between sIRLS-p and IRLS-p

We now relate the sIRLS-p family to the IRLS-p family of algorithms and its implementation IRLS-GP. Each iteration of the IRLS-p algorithm is a quadratic program, and IRLS-GP uses iterative gradient projection to solve this quadratic program. We note that sIRLS-p is nothing but IRLS-GP, with each quadratic program solved approximately by terminating at the first iteration in the gradient projection inner loop. Barring this connection, the IRLS-p (through its implementation IRLS-GP) and sIRLS-p algorithms can be seen as two different approaches to solving the smooth Schatten-p minimization problem (2.12). Indeed it would seem that sIRLS-p is an approximation to IRLS-GP implementation but as we saw earlier sIRLS-p has global convergence guarantees.
sIRLS-p can also be seen as a gradient projection algorithm applied to the smooth Schatten-p minimization problem. However, in sIRLS-p, we do not tune for the step-size, as this comes out of the derivation of the IRLS algorithm. Furthermore, for $p = 1$, sIRLS-p can be seen as a Nesterov's smoothing method [Nesterov, 2005] applied to solve the nuclear norm minimization heuristic. Thus for $p = 1$, sIRLS-1 requires $O(1/\epsilon^2)$ iterations to attain an accuracy of ϵ. However, for $p = 0$, sIRLS-0 can't be cast as a Nesterov's smoothing method, since $\log\det(X^TX)$ evaluates to $-\infty$ if X is low-rank.

In the next section, we present comparisons of the sIRLS-p algorithm against other state of the art algorithms for the matrix completion problem.

2.6 Numerical experiments for ARMP

In this section we report results for sIRLS-0 and its competitors. Detailed numerical results for sIRLS-p and IRLS-p family of algorithms can be found in [Mohan and Fazel, 2012].

2.6.1 Synthetic experiments

We classify our numerical experiments into two categories based on the *degrees of freedom ratio*, given by $\text{FR} = r(2n - r)/q$. Note that for a $n \times n$ matrix of rank r, $r(2n-r)$ is the number of degrees of freedom in the matrix. Thus if FR is large (close to 1), recovering X_0 (the true low-rank matrix) becomes harder (as the number of

Problem				sIRLS-1		sIRLS-0		SVT	
n	r	$\frac{q}{n^2}$	FR	# iter	Time	# iter	Time	# iter	Time
100	10	0.57	0.34	132	1.63	59	0.84	170	5.69
200	10	0.39	0.25	140	2.41	63	1.31	109	3.74
500	10	0.2	0.2	163	8	98	4.97	95	5.9
500	10	0.12	0.33	336	13.86	280	11.03	-	-
1000	10	0.12	0.17	195	32.21	140	20.80	85	10.71
1000	50	0.39	0.25	140	102.64	60	61.32	81	49.17
1000	20	0.12	0.33	284	57.85	241	43.11	-	-
2000	20	0.12	0.17	190	166.28	130	98.55	73	42.31
2000	40	0.12	0.33	270	322.96	220	227.07	-	-

Table 2.1: Comparison of sIRLS-0,1 with SVT. Performance on Easy Problems $FR < 0.4$.

measurements is close to the degrees of freedom) and conversely if FR is close to zero, recovering X_0 becomes easier.

We conduct subsequent numerical experiments over what we refer to in this paper as *Easy problems* ($FR < 0.4$) and *Hard problems*($FR > 0.4$). We define the recovery to be successful when the relative error, $\|X - X_0\|_F / \|X_0\|_F \leq 10^{-3}$ (with X being the output of the algorithm considered) and unsuccessful recovery otherwise. For each problem (easy or hard), the results are reported over 10 random generations of the support set, Ω and X_0. We use NS to denote the number of successful recoveries for a given problem. Also, computation times are reported in seconds. For sIRLS and IRLS-GP (implementation of IRLS), we fix $\eta = 1.1$ if $FR < 0.4$ and $\eta = 1.03$ if $FR > 0.4$ based on our earlier observations.

Comparison of sIRLS-0 and nuclear norm minimization

Table 2.1 shows that sIRLS-0 and sIRLS-1 have competitive computational times as compared to SVT [implementation available at, Candes and Becker, 2010] on easy problems. Note that SVT is an efficient implementation for nuclear norm minimization. There are also certain instances where SVT fails to have successful recovery while sIRLS-1 succeeds. Thus sIRLS-1 is competitive and in some instances better than SVT on easy problems. For hard problems, SVT fails on most instances. This demonstrates that (s)IRLS-0 performs better than nuclear norm minimization on hard

Problem		sIRLS			IHT			FPCA		Optspace		
n	FR	# iter	NS	Time	# iter	NS	Time	NS	Time	# iter	NS	Time
40	0.8	1498	10	12.91	-	0	-	5	1.69	-	0	-
100	0.87	4934	5	72.36	-	0	-	0	-	-	0	-
500	0.78	4859	9	326.06	-	0	-	0	-	-	0	-
1000	0.40	406	10	115.73	280	10	72.67	10	26.54	40	10	256.92
1000	0.57	1368	10	237.22	1059	10	244.49	0	-	133	5	769.29
1000	0.66	2961	10	554.25	-	0	-	0	-	-	0	-
1000	0.59	897	10	276.08	660	10	213.95	10	62.43	89	5	1420.81
1000	0.49	270	10	263.45	203	10	186.15	10	25.21	45	10	2924.68

Table 2.2: Comparison of sIRLS-0, IHT, FPCA and Optspace on Hard Problems when no prior information is available on the rank of the matrix to be recovered.

problems.

Comparison of sIRLS-0 and other algorithms on hard problems

As seen in Table 2.2, sIRLS has a clear advantage over IHT, Optspace and FPCA in successful recovery for hard problems. sIRLS is fully successful on all problems except the second and third on which it has a success rate of 5 and 9 respectively. On the other hand IHT, Optspace and FPCA have partial or unsuccessful recovery in many problems. Thus, when the rank of X_0 is not known a priori, sIRLS has a distinct advantage over IHT, Optspace and FPCA in successfully recovering X_0 for hard problems.

2.6.2 Movielens data set

In this section, we consider the movie lens data sets [Mov] with 100,000 ratings. In particular, we consider four different splits of the 100k ratings into (training set, test set):
(`u1.base,u1.test`), (`u2.base,u2.test`), (`u3.base,u3.test`), (`u4.base,u4.test`) for our numerical experiments. Any given set of ratings (e.g., from a data split) can be represented as a matrix. This matrix has rows representing the users and columns representing the movies and an entry (i, j) of the matrix is non-zero if we know the rating of user i for movie j. Thus estimating the remaining ratings in the

matrix corresponds to a matrix completion problem. For each data split, we train sIRLS, IHT, and Optspace on the training set and compare their performance on the corresponding test set. The performance metric here is *Normalized Mean Absolute Error* or NMAE given as follows. Let M be the matrix representation corresponding to the actual test ratings and X be the ratings matrix output by an algorithm when input the training set. Then

$$\text{NMAE} = \left(\sum_{i,j \in \text{supp}(M)} \frac{|M_{ij} - X_{ij}|}{|\text{supp}(M)|} \right) / (rt_{\max} - rt_{\min}),$$

where $rt_{\min}$ and $rt_{\max}$ are the minimum and maximum movie ratings possible and $\text{supp}(M)$ denotes the support of matrix M. The choice of γ^0, η for sIRLS is the same as for the random experiments (described in previous sections). sIRLS is terminated if the maximum number of iterations exceeds 700 or if the relative error between the successive iterates is less than 10^{-3}. We set the rank of the unknown ratings matrix to be equal to 5 while running all the three algorithms. Table 2.3 shows that the NMAE for sIRLS, IHT, and Optspace are almost the same across different splits of the data. Keshavan et al. [2009] reported the NMAE results for different algorithms when tested on data split 1. These were reported to be 0.186,0.19 and 0.242 for Optspace [Keshavan and Oh, 2009], FPCA [Goldfarb and Ma, 2011], and AdMiRA [Lee and Bresler, 2010] respectively. Thus sIRLS has a NMAE that is as good as Optspace, FPCA, IHT and has a better NMAE than AdMiRA.

	sIRLS	IHT	Optspace
split 1	0.1919	0.1925	0.1887
split 2	0.1878	0.1883	0.1878
split 3	0.1870	0.1872	0.1881
split 4	0.1899	0.1896	0.1882

Table 2.3: NMAE for sIRLS-0 for different splits of the 100k movie-lens data set.

2.7 Recovery conditions for nuclear norm minimization

We note that the problem of recovering a low-rank matrix X from the linear system $\mathcal{A}(X) = b$ is well-posed only if appropriate restrictions are made on the linear map, $\mathcal{A}$. These restrictions would ensure that the lowest rank matrix X_0 that satisfies $\mathcal{A}(X) = b$ is unique. It also turns out that tightening these restrictions would be sufficient for recovery of the lowest rank matrix through an appropriate algorithm. In the context of compressive sensing, the *restricted isometry property* [Candes and Tao, 2004, Candes, 2008] was one of the first conditions imposed on the matrix A that guaranteed uniqueness of the sparse solution, x_0 to the system $Ax = b$ and also to recover x_0 through ℓ_1 minimization.

Definition 6. *A matrix A satisfies the restricted isometry property (RIP) of order r with restricted isometry constant $0 < \delta_r < 1$ if for every $x : \|x\|_0 \leq r$, it holds that,*

$$(1 - \delta_r) \leq \frac{\|Ax\|_2^2}{\|x\|_2^2} \leq (1 + \delta_r). \tag{2.14}$$

Note that under the RIP assumption of order $\delta_{2r} < 1$, there can only exist one solution x_0 to $Ax = b$ that has $\|x\|_0 \leq r$. It was shown by Candes et al [Candes, 2008] that $\delta_{2r} < \sqrt{2} - 1$ is a sufficient condition for recovery of x_0 through ℓ_1 minimization. Although RIP provides a starting point to prove recovery of sparse vectors, it is not robust to invertible transformations of A - i.e. TA has a different RIP constant than A for invertible T. Also, the RIP only provides a sufficient condition. Let $\mathcal{N}(A)$ denote the null-space of the matrix A. To improve on these drawbacks of RIP, the null-space property [Zhang, 2008] was proposed:

Definition 7. *A matrix A satisfies the null space property (NSP) of order r if for every $z \in \mathcal{N}(A)$, it holds that*

$$\sum_{i=1}^{r} |z_i| < \sum_{i=r+1}^{n} |z_i| \tag{2.15}$$

It was shown by in [Zhang, 2008] that NSP of order r is a necessary and sufficient condition on A for x_0 to be recovered through ℓ_1 minimization.

Both RIP and NSP have intuitive extensions to the context of matrix rank minimization, as follows:

Definition 8. *A linear operator, $\mathcal{A}$ satisfies the restricted isometry property (RIP) of order r with restricted isometry constant $0 < \delta_r < 1$ if for every $X : \text{rank}(X) \leq r$, it holds that,*

$$(1 - \delta_r) \leq \frac{\|\mathcal{A}(X)\|_2^2}{\|X\|_2^2} \leq (1 + \delta_r). \tag{2.16}$$

Definition 9. *A linear operator, $\mathcal{A}$ satisfies the null space property (NSP) of order r if for every $Z \in \mathcal{N}(\mathcal{A})$, it holds that*

$$\sum_{i=1}^{r} \sigma_i(Z) < \sum_{i=r+1}^{n} \sigma_i(Z) \tag{2.17}$$

We showed in [Oymak et al., 2011a] that whenever an RIP condition or NSP condition was necessary and (or) sufficient for recovery of r-sparse vectors through ℓ_1 minimization, the extension of the corresponding conditions to the linear operator, $\mathcal{A}$ was necessary and (or) sufficient for recovery of rank r-matrices through nuclear norm minimization. This result eliminates the need to provide new RIP or NSP conditions for low-rank matrix recovery using nuclear norm minimization as done in the literature (see e.g. [Mohan and Fazel, 2010b, Candes and Plan, 2009]). Instead the best RIP or NSP result for vector recovery when extended also holds for low-rank matrix recovery. Our extension result relies on a key singular value inequality that allows for "vectorization" of matrices through their singular values. Though the RIP or NSP recovery conditions for nuclear norm minimization follow from the corresponding results for ℓ_1 norm minimization - The conditions for the reweighted trace heuristic and IRLS for low-rank matrix recovery need to be derived separately. We describe this in the next couple of sections.

2.7.1 Recovery guarantee for reweighted trace heuristic

We give a recovery guarantee for the reweighted trace heuristic for ARMP with positive semi-definite constraint on the matrix. The first iteration of reweighted trace heuristic is just trace minimization. Intuitively, we expect the reweighting to improve the solution in subsequent iterations of the heuristic. The following theorem [Mohan and Fazel, 2010b] shows that this is exactly what we can expect.

Theorem 10. *Let $\mathcal{A}$ satisfy RIP of order $3r$ with RIP constant δ_{3r}, obeying $\delta_{3r} < 2\sqrt{5} - 4$. Let X^1 be the solution obtained through nuclear norm minimization. Then*

$\|X^k - X_0\|_F \leq E(k) = 2\epsilon\sqrt{1+\delta_{3r}}\frac{\sqrt{1+\frac{10}{8}C_{1,k}^2}}{1-\delta_{3r}(1+C_{1,k}\sqrt{\frac{10}{8}})}$ $\forall k \geq 2$, *where* $C_{1,k}$ *is a constant that depends on* $E(k-1), \gamma^k, \mu$. *If* $C_{1,k} < 1\ \forall k \geq 2$, *then the sequence,* $\{E(k)\}$ *converges to a limit* $E < E(1)$. *In particular, if* $\mu > 3E(1)$ *and* $\gamma^k = \frac{E(k-1)(\mu+E(k-1))}{\mu-3E(k-1)}$ $\forall k \geq 2$ *then the sequence,* $\{E(k)\}$ *converges to a limit* $E < E(1)$.

The above theorem shows that the recovery error of the reweighted trace heuristic is strictly better than trace minimization in the presence of noise. When there is no noise, the theorem reduces to the exact recovery guarantee of the trace heuristic.

2.7.2 Recovery guarantee for IRLS-1

We first note that IRLS-1 with $\gamma \to 0$ converges to the nuclear norm solution. Thus, when $\gamma \to 0$, the existing recovery guarantee for nuclear norm minimization carries through to IRLS-1. However it may be impractical to set γ to zero due to ill-conditioning. In this section, we present necessary and sufficient conditions for recovery through IRLS-1 with $\gamma > 0$. The conditions are based on a variation of the popular Null space property (NSP) [Oymak and Hassibi, 2010, Oymak et al., 2011b] and is as follows:

Definition 11. *Given* $\tau > 0$, *the linear map* $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ *satisfies the* τ-Null space Property *(*τ*-NSP) of order* r *if for every* $Z \in \mathcal{N}(\mathcal{A})\backslash\{0\}$, *we have*

$$\sum_{i=1}^{r}\sigma_i(Z) \quad < \quad \tau\sum_{i=r+1}^{n}\sigma_i(Z) \tag{2.18}$$

where $\mathcal{N}(\mathcal{A})$ *denotes the null space of* $\mathcal{A}$ *and* $\sigma_i(Z)$ *denotes the* i^{th} *largest singular value of* Z.

If $\mathcal{A}$ satisfies τ-NSP of order r, then every matrix in the null-space of $\mathcal{A}$ has rank at least $2r$. It has been shown in the literature [Oymak and Hassibi, 2010] that certain random Gaussian maps $\mathcal{A}$ satisfy this property with high probability. We first give a few definitions and lemmas.

Definition 12. *Given* $x \in \mathbb{R}^n$, *let* $x_{[i]}$ *denote the* i^{th} *largest element of* x *so that* $x_{[1]} \geq x_{[2]} \geq \ldots x_{[n-1]} \geq x_{[n]}$. *A vector* $x \in \mathbb{R}^n$ *is said to be* majorized *by* $y \in \mathbb{R}^n$ *(denoted as* $x \prec y$*) if* $\sum_{i=1}^k x_{[i]} \leq \sum_{i=1}^k y_{[i]}$ *for* $k = 1, 2, \ldots, n-1$ *and* $\sum_{i=1}^n x_{[i]} = \sum_{i=1}^n y_{[i]}$. *A*

vector $x \in \mathbb{R}^n$ *is said to be* weakly sub-majorized *by* $y \in \mathbb{R}^n$ *(denoted as* $x \prec_w y$ *) if* $\sum_{i=1}^k x_{[i]} \leq \sum_{i=1}^k y_{[i]}$ *for* $k = 1, 2, \ldots, n$.

Lemma 13. *[Horn and Johnson, 1990] For any two matrices,* $A, B \in \mathbb{R}^{m\times n}$ *it holds that* $|\sigma(A) - \sigma(B)| \prec_w \sigma(A - B)$.

Lemma 14. *[Horn and Johnson, 1991, Marshall and Olkin, 1979] Let* $g : D \to \mathbb{R}$ *be a convex and increasing function where* $D \subset \mathbb{R}$. *Let* $x, y \in D^n$. *Then if* $x \prec_w y$, *we have* $(g(x_1), g(x_2), \ldots, g(x_n)) \prec_w (g(y_1), g(y_2), \ldots, g(y_n))$.

Since $g(x) = (x^2+\gamma)^{\frac{1}{2}}$ is a convex and increasing function on $\mathbb{R}_+$, applying Lemma 14 to the majorization inequality in Lemma 13 we have

$$\sum_{i=1}^n (|\sigma_i(A) - \sigma_i(B)|^2 + \gamma)^{\frac{1}{2}} \quad \leq \quad \sum_{i=1}^n (\sigma_i^2(A - B) + \gamma)^{\frac{1}{2}}. \tag{2.19}$$

Let X_0 be a matrix (that is not necessarily low-rank) and let the measurements be given by $b = \mathcal{A}(X_0)$. In this section, we give necessary and sufficient conditions for recovering X_0 using IRLS-1. Let $X_0^\gamma := (X_0^T X_0 + \gamma I)^{\frac{1}{2}}$ be the γ-approximation of X_0, and let $X_{0,r}$, $X_{0,r}^\gamma$ be the best rank-r approximations of X_0, X_0^γ respectively.

Recall the Null space Property τ-NSP defined earlier in (2.18). This condition requires that every nonzero matrix in the null space of $\mathcal{A}$ has a rank larger than $2r$.

Before giving the result, we define some useful notation:
Let X_0 be a matrix (that is not necessarily low-rank) and let the measurements be given by $b = \mathcal{A}(X_0)$. Let $X_0^\gamma := (X_0^T X_0 + \gamma I)^{\frac{1}{2}}$ be the γ-approximation of X_0, and let $X_{0,r}$, $X_{0,r}^\gamma$ be the best rank-r approximations of X_0, X_0^γ respectively.

Theorem 15. *Assume that* $\mathcal{A}$ *satisfies* τ*-NSP of order* r *for some* $0 < \tau < 1$. *For every* X_0 *satisfying* $\mathcal{A}(X_0) = b$ *it holds that*

$$f_1(X_0 + Z) > f_1(X_0), \quad \textit{for all} \quad Z \in \mathcal{N}(\mathcal{A})\backslash\{0\} \quad \textit{satisfying} \quad \|Z\|_\star \geq C\|X_0^\gamma - X_{0,r}^\gamma\|_\star. \tag{2.20}$$

Furthermore, we have the following bounds,

$$\begin{aligned} \|\bar{X} - X_0\|_\star &\leq C\|X_0^\gamma - X_{0,r}^\gamma\|_\star \\ \|\bar{X}_r - X_0\|_\star &\leq (2C+1)\|X_0^\gamma - X_{0,r}^\gamma\|_\star, \end{aligned}$$

where $C = 2\frac{(1+\tau)}{1-\tau}$*, and* $\bar{X}$ *is the output of IRLS-1. Conversely, if* (2.20) *holds, then* $\mathcal{A}$ *satisfies* δ*-NSP of order* r*, where* $\delta > \tau$*.*

2.8 Conclusions

In summary, we proposed two families of algorithms, the IRLS-q and sIRLS-q for $q \leq 1$. We also showed that these families of algorithms are globally convergent. For $q = 1$, IRLS-1 and sIRLS-1 converge to the nuclear norm minimizer. For $q < 1$, IRLS-q and sIRLS-q perform empirically better than the nuclear norm heuristic. We also find that sIRLS is significantly faster than IRLS empirically. Furthermore, we observe on synthetic data that sIRLS-0 is better than state of the art algorithms in recovering low-rank matrices with unknown rank. We next discussed recovery conditions for the nuclear norm minimization. In particular, we discussed a seamless way to extend recovery conditions such as restricted isometry property and null space property from the context of sparse vector recovery to the context of low-rank matrix recovery. We also considered the problem of low-rank positive semi-definite matrix recovery from linear measurements. For this problem, we showed that the reweighted trace heuristic has theoretically better recovery error as compared to the trace heuristic in the presence of noise. Finally, we also showed that the IRLS-1 algorithm for affine rank minimization problem enjoys performance guarantees similar to nuclear norm minimization under suitable null space conditions. Furthermore, the IRLS-0 algorithm has a significantly better performance than nuclear norm minimization empirically.

2.9 Proofs

2.9.1 Proof of Theorem 2

We first present two useful lemmas.

Lemma 16. *For each* $k \geq 1$*, we have*

$$\text{trace}({X^k}^T X^k)^{\frac{p}{2}} \leq \mathcal{J}^p(X^1, W^0, \gamma^0) := D \tag{2.21}$$

where $W^0 = I$*,* $\gamma^0 = 1$*. Also,* $\lambda_j(W^k) \geq D^{(1-\frac{2}{p})}$*,* $j = 1, 2, \ldots, \min\{m, n\}$

Proof. First, notice that

$$\begin{aligned}\text{trace}({X^k}^T X^k)^{\frac{p}{2}} &\leq \text{trace}({X^k}^T X^k + \gamma I)^{p/2} = \mathcal{J}^p(X^k, W^k, \gamma^k) \\ &\leq \mathcal{J}^p(X^1, W^1, \gamma^1) \leq \mathcal{J}^p(X^1, W^0, \gamma^0) = D,\end{aligned}$$

where the second and third inequalities follow from (2.10). This proves (2.21). Furthermore, from the above chain of inequalities,

$$(\|{X^k}^T X^k\| + \gamma)^{\frac{p}{2}} = \|({X^k}^T X^k + \gamma I)^{\frac{p}{2}}\| \leq D.$$

Using this and the definition of W^k, we obtain

$$\|(W^k)^{-1}\| = \|({X^k}^T X^k + \gamma I)^{1-\frac{p}{2}}\| = (\|{X^k}^T X^k\| + \gamma)^{1-\frac{p}{2}} \leq D^{(\frac{2}{p}-1)}.$$

This last relation shows that $\lambda_j(W^k) = \sigma_j(W^k) \geq 1/\|(W^k)^{-1}\| \geq D^{(1-\frac{2}{p})}$ for all j. □

Lemma 17. *A matrix X^* is a minimizer of*

$$\begin{array}{ll}\underset{X}{\text{minimize}} & \text{trace} W X^T X \\ \textit{subject to} & \mathcal{A}(X) = b\end{array}$$

if and only if $\text{trace}(W {X^*}^T Z) = 0$ *for all* $Z \in \mathcal{N}(A)$.

We are now ready to prove Theorem 2.

Proof. For each $k \geq 1$, we have that

$$\begin{aligned}2[\mathcal{J}^p(X^k, W^k, \gamma^k) - \mathcal{J}^p(X^{k+1}, W^{k+1}, \gamma^{k+1})] &\geq 2[\mathcal{J}^1(X^k, W^k, \gamma^k) - \mathcal{J}^1(X^{k+1}, W^k, \gamma^k)] \\ &= \langle X^k, X^k\rangle_{W^k} - \langle X^{k+1}, X^{k+1}\rangle_{W^k} \\ &= \langle X^k + X^{k+1}, X^k - X^{k+1}\rangle_{W^k} \\ &= \langle X^k - X^{k+1}, X^k - X^{k+1}\rangle_{W^k} \\ &= \text{trace} W^k (X^k - X^{k+1})^T (X^k - X^{k+1}) \\ &\geq D^{(1-\frac{2}{p})} \|X^k - X^{k+1}\|_F^2\end{aligned}$$

where the above expressions use Lemma 17 and Lemma 16. Summing the above inequalities over all $k \geq 1$, we have that $\lim_{n\to\infty} \left(X^n - X^{n+1}\right) = 0$. □

2.9.2 Proof of Theorem 3

Let

$$g(X,\gamma) = \begin{cases} \text{trace}(X^TX+\gamma I)^{\frac{p}{2}} & \text{if } 0<p\leq 1 \\ \log\det(X^TX+\gamma I) & \text{if } p=0 \end{cases}$$

Then $g(X^k,\gamma^k) = \mathcal{J}^p(X^k,W^k,\gamma^k)$ and it follows from (2.10) that $g(X^{k+1},\gamma^{k+1}) \leq g(X^k,\gamma^k)$ for all $k\geq 1$. Hence the sequence $\{g(X^k,\gamma^k)\}$ converges. This fact together with $\gamma_{\min}>0$ implies that the sequence $\{X^k\}$ is bounded.

We now show that every cluster point of $\{X^k\}$ is a stationary point of (2.7). Suppose to the contrary and let $\widetilde{X}$ be a cluster point of $\{X^k\}$ that is not a stationary point. By the definition of cluster point, there exists a subsequence $\{X^{n_i}\}$ of $\{X^k\}$ converging to $\widetilde{X}$. By passing to a further subsequence if necessary, we can assume that $\{X^{n_i+1}\}$ is also convergent and we denote its limit by $\widehat{X}$. By definition, X^{n_i+1} is the minimizer of

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \text{trace}W^{n_i}X^TX \\ \text{subject to} & \mathcal{A}(X)=b. \end{array} \tag{2.22}$$

Thus, X^{n_i+1} satisfies the KKT conditions of (2.22), that is,

$$X^{n_i+1}W^{n_i} \in \text{Ran}(\mathcal{A}^*) \text{ and } \mathcal{A}(X^{n_i+1})=b,$$

where $\text{Ran}(\mathcal{A}^*)$ denotes the range space of $\mathcal{A}^*$. Passing to limits, we see that

$$\widehat{X}\widetilde{W} \in \text{Ran}(\mathcal{A}^*) \text{ and } \mathcal{A}(\widehat{X})=b, \tag{2.23}$$

where $\widetilde{W}=(\widetilde{X}^T\widetilde{X}+\gamma_{\min}I)^{-1}$. From (2.23), we conclude that $\widehat{X}$ is a minimizer of the following convex optimization problem,

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \text{trace}\widetilde{W}X^TX \\ \text{subject to} & \mathcal{A}(X)=b. \end{array} \tag{2.24}$$

Next, by assumption, $\widetilde{X}$ is not a stationary point of (2.7) (for $0<p\leq 1$) nor (2.9) (for $p=0$). This implies that $\widetilde{X}$ is not a minimizer of (2.24) and thus $\text{trace}\widetilde{W}\widehat{X}^T\widehat{X} <$

trace$\widetilde{W}\widetilde{X}^T\widetilde{X}$. This is equivalent to $\mathcal{J}^p(\widehat{X},\widetilde{W},\gamma_{\min}) < \mathcal{J}^p(\widetilde{X},\widetilde{W},\gamma_{\min})$. From this last relation and (2.10) it follows that,

$$\begin{array}{rcl} \mathcal{J}^p(\widehat{X},\widehat{W},\gamma_{\min}) & < & \mathcal{J}^p(\widetilde{X},\widetilde{W},\gamma_{\min}), \\ g(\widehat{X},\gamma_{\min}) & < & g(\widetilde{X},\gamma_{\min}). \end{array} \tag{2.25}$$

On the other hand, since the sequence $\{g(X^k,\gamma^k)\}$ converges, we have that

$$\lim g(X^i,\gamma^i) = \lim g(X^{n_i},\gamma^{n_i}) = g(\widetilde{X},\gamma_{\min}) = \lim g(X^{n_{i+1}},\gamma^{n_{i+1}}) = g(\widehat{X},\gamma_{\min})$$

which contradicts (2.25). Hence, every cluster point of $\{X^k\}$ is a stationary point of (2.7) (when $0 < p \leq 1$) and a stationary point of (2.9) (when $p = 0$).

2.9.3 Proof of Theorem 4

For any two matrices X, Y we denote $\langle X, Y\rangle_W = \text{trace}WX^TY$. We first note that the iterates of sIRLS-p satisfy

$$\begin{array}{rcl} \mathcal{J}^p(X^{k+1},W^{k+1},\gamma^{k+1}) & \leq & \mathcal{J}^p(X^{k+1},W^k,\gamma^{k+1}) \\ & \leq & \mathcal{J}^p(X^{k+1},W^k,\gamma^k) \\ & \leq & \mathcal{J}^p(X^k,W^k,\gamma^k). \end{array}$$

The last inequality follows from the Lipschitz continuity (with $L^k = 2(\gamma^k)^{\frac{p}{2}-1}$) of the gradient of trace(W^kX^TX), that is,

$$\text{trace}W^kX^TX \leq \text{trace}W^kX^{kT}X^k + \langle 2X^kW^k, X - X^k\rangle + \frac{L^k}{2}\|X - X^k\|_F^2 \quad \forall X, X^k$$

and the fact that

$$\begin{array}{rcl} X^{k+1} & = & \arg\min_X \|X - (X^k - X^kW^k)\|_F^2 \\ & & \text{s.t. } \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(X_0). \end{array}$$

The convergence of $\{\mathcal{J}^p(X^k,W^k,\gamma^k)\}$ follows from Monotone convergence theorem. This also implies that the sequence $\{X^k\}$ is bounded. Hence there exists a convergent subsequence, $\{X^{n_i}\} \to X^*$. Also let $\{X^{n_i+1}\} \to \widehat{X}$. If X^* is a stationary point, we are done. Conversely, if X^* is not a stationary point to (2.12) then it follows

that, $\widehat{X} \neq X^*$. But $\widehat{X} \neq X^*$ implies (using strict convexity) that trace$(W^*\widehat{X}^T\widehat{X}) <$ trace$(W^*X^{*T}X^*)$ which also implies that $\mathcal{J}^p(\widehat{X}, \widehat{W}, \gamma_{\min}) < \mathcal{J}^p(X^*, W^*, \gamma_{\min})$. However since

$$\begin{aligned} \lim \mathcal{J}^p(X^i, W^i, \gamma^i) &= \lim \mathcal{J}^p(X^{n_i}, W^{n_i}, \gamma^{n_i}) \\ &= \lim \mathcal{J}^p(X^{n_{i+1}}, W^{n_{i+1}}, \gamma^{n_{i+1}}), \end{aligned}$$

we have a contradiction. Therefore, X^* is a stationary point to (2.12) and the theorem follows.

2.9.4 Proof of Theorem 15

Let $Z \in \mathcal{N}(A)\backslash\{0\}$ and $\|Z\|_\star \geq C\|X_0^\gamma - X_{0,r}^\gamma\|_\star$. We see that

$$\begin{aligned} f_1(X_0+Z) &= \text{trace}((X_0+Z)^T(X_0+Z)+\gamma I)^{\frac{1}{2}} = \sum_{i=1}^{n}(\sigma_i^2(X_0+Z)+\gamma)^{\frac{1}{2}} \\ &\geq \sum_{i=1}^{r}((\sigma_i(X_0)-\sigma_i(Z))^2+\gamma)^{\frac{1}{2}} + \sum_{i=r+1}^{n}((\sigma_i(Z)-\sigma_i(X_0))^2+\gamma)^{\frac{1}{2}} \\ &= \sum_{i=1}^{r}((\sigma_i^2(X_0)+\gamma)+\sigma_i^2(Z)-2\sigma_i(X_0)\sigma_i(Z))^{\frac{1}{2}} \\ &\quad + \sum_{i=r+1}^{n}((\sigma_i^2(Z)+\gamma)+\sigma_i^2(X_0)-2\sigma_i(Z)\sigma_i(X_0))^{\frac{1}{2}} \\ &\geq \sum_{i=1}^{r}|(\sigma_i^2(X_0)+\gamma)^{\frac{1}{2}}-\sigma_i(Z)| + \sum_{i=r+1}^{n}|(\sigma_i^2(Z)+\gamma)^{\frac{1}{2}}-\sigma_i(X_0)| \\ &\geq \sum_{i=1}^{r}(\sigma_i^2(X_0)+\gamma)^{\frac{1}{2}} - \sum_{i=1}^{r}\sigma_i(Z) + \sum_{i=r+1}^{n}(\sigma_i^2(Z)+\gamma)^{\frac{1}{2}} - \sum_{i=r+1}^{n}\sigma_i(X_0) \\ &\geq f_1(X_0) - \sum_{i=r+1}^{n}(\sigma_i^2(X_0)+\gamma)^{\frac{1}{2}} - \sum_{i=1}^{r}\sigma_i(Z) \\ &\quad + \sum_{i=r+1}^{n}\sigma_i(Z) - \sum_{i=r+1}^{n}\sigma_i(X_0), \end{aligned} \tag{2.26}$$

where the first inequality follows from (2.19). Since τ-NSP holds, we have

$$\begin{aligned}\sum_{i=1}^{r}\sigma_i(Z) < \tau\sum_{i=r+1}^{n}\sigma_i(Z) &= \sum_{i=r+1}^{n}\sigma_i(Z) - (1-\tau)\sum_{i=r+1}^{n}\sigma_i(Z)\\ &\le \sum_{i=r+1}^{n}\sigma_i(Z) - C\frac{1-\tau}{1+\tau}\|X_0^\gamma - X_{0,r}^\gamma\|_\star\\ &= \sum_{i=r+1}^{n}\sigma_i(Z) - 2\|X_0^\gamma - X_{0,r}^\gamma\|_\star, \end{aligned} \tag{2.27}$$

where the second inequality uses $\|Z\|_\star \ge C\|X_0^\gamma - X_{0,r}^\gamma\|_\star$ and τ-NSP. Combining (2.26) and (2.27), we obtain

$$\begin{aligned} f_1(X_0+Z) &= f_1(X_0) + \textstyle\sum_{i=r+1}^{n}\sigma_i(Z) - \sum_{i=1}^{r}\sigma_i(Z) - \sum_{i=r+1}^{n}(\sigma_i^2(X_0)+\gamma)^{\frac{1}{2}}\\ &\quad - \sum_{i=r+1}^{n}\sigma_i(X_0)\\ &\ge f_1(X_0) + \textstyle\sum_{i=r+1}^{n}\sigma_i(Z) - \sum_{i=1}^{r}\sigma_i(Z) - \sum_{i=r+1}^{n}(\sigma_i^2(X_0)+\gamma)^{\frac{1}{2}} \quad (2.28)\\ &\quad - \textstyle\sum_{i=r+1}^{n}(\sigma_i^2(X_0)+\gamma)^{\frac{1}{2}}\\ &= f_1(X_0) + \textstyle\sum_{i=r+1}^{n}\sigma_i(Z) - \sum_{i=1}^{r}\sigma_i(Z) - 2\|X_0^\gamma - X_{0,r}^\gamma\|_\star\\ &> f_1(X_0). \end{aligned}$$

This proves (2.20). Next, if $\widehat{X}$ is an optimal solution of (2.7) with $p = 1$, then $Z = \widehat{X} - X_0 \in \mathcal{N}(\mathcal{A})$ and it follows immediately from (2.28) that

$$\|\widehat{X} - X_0\|_\star = \|Z\|_\star \le C\|X_0^\gamma - X_{0,r}^\gamma\|_\star.$$

Note that problem (2.7) is convex when $p = 1$, and every stationary point of (2.7) is a global minimum. Hence, by Theorem 3, IRLS-1 converges to a global minimum of the problem (2.7). It follows that $\bar{X} = \widehat{X}$ and

$$\|\bar{X} - X_0\|_\star \le C\|X_0^\gamma - X_{0,r}^\gamma\|_\star. \tag{2.29}$$

Finally, since $|\sigma(\bar{X}) - \sigma(X_0)| \prec_w \sigma(\bar{X} - X_0)$,

$$\sum_{i=r+1}^{n}\sigma_i(\bar{X}) - \sum_{i=r+1}^{n}\sigma_i(X_0) \le \sum_{i=1}^{n}|\sigma_i(\bar{X}) - \sigma_i(X_0)| \le \|\bar{X} - X_0\|_\star. \tag{2.30}$$

Thus,

$$\begin{aligned}\|\bar{X}_r - X_0\|_\star &\leq \|\bar{X} - \bar{X}_r\|_\star + \|\bar{X} - X_0\|_\star \\ &\leq \|X_0 - X_{0r}\|_\star + \|\bar{X} - X_0\|_\star + \|\bar{X} - X_0\|_\star \\ &\leq (2C+1)\|X_0^\gamma - X_{0,r}^\gamma\|_\star,\end{aligned}$$

where the second inequality follows from (2.30) and the third inequality from (2.29). Conversely, suppose that (2.20) holds, that is, $f_1(X_0 + Z) > f_1(X_0)$ for all $\|Z\|_\star \geq C\|X_0^\gamma - X_{0,r}^\gamma\|_\star$, $Z \in \mathcal{N}(A)\backslash\{0\}$. We would like to show that $\mathcal{A}$ satisfies δ-NSP of order r. Assume to the contrary that there exists $Z \in \mathcal{N}(A)$ such that

$$\sum_{i=1}^{r} \sigma_i(Z) \geq \delta \sum_{i=r+1}^{n} \sigma_i(Z). \tag{2.31}$$

Let $\alpha = \frac{1-\tau\delta}{2(1+\tau)}$ and set $X_0 = -Z_r - \alpha(Z - Z_r)$, where Z_r denotes the best rank-r approximation to Z. Note that $\alpha < (1+\delta)/C$. Assume that Z satisfies

$$\left(\frac{1+\delta}{C} - \alpha\right) \sum_{i=r+1}^{n} \sigma_i(Z) \geq (n-r)\sqrt{\gamma}. \tag{2.32}$$

If not, Z can be multiplied by a large enough positive constant so that it satisfies both (2.32) and (2.31). Note that (2.31) can be rewritten as,

$$\|Z\|_\star \geq (1+\delta) \sum_{i=r+1}^{n} \sigma_i(Z). \tag{2.33}$$

Combining (2.32) and (2.33), we obtain that

$$\begin{aligned}\|X_0^\gamma - X_{0,r}^\gamma\|_\star = \sum_{i=r+1}^{n} (\alpha^2\sigma_i^2(Z) + \gamma)^{\frac{1}{2}} &\leq (n-r)\sqrt{\gamma} + \alpha \sum_{i=r+1}^{n} \sigma_i(Z) \\ &\leq \frac{\|Z\|_\star}{C}\end{aligned} \tag{2.34}$$

Moreover, it follows from (2.31) and the choice of X_0 that

$$\begin{aligned} f_1(X_0) &= \textstyle\sum_{i=1}^{n}(\sigma_i^2(X_0)+\gamma)^{\frac{1}{2}} \\ &\geq \textstyle\sum_{i=1}^{r}\sigma_i(X_0)+\sum_{i=r+1}^{n}\sigma_i(X_0) \\ &= \textstyle\sum_{i=1}^{r}\sigma_i(Z)+\alpha\sum_{i=r+1}^{n}\sigma_i(Z) \\ &\geq (\alpha+\delta)\textstyle\sum_{i=r+1}^{n}\sigma_i(Z). \end{aligned} \tag{2.35}$$

On the other hand, notice by the definition of α that $\alpha > \frac{1-\delta}{2}$. Also assume Z satisfies

$$(2\alpha+\delta-1)\sum_{i=r+1}^{n}\sigma_i(Z) \geq n\sqrt{r}. \tag{2.36}$$

If not, Z can be multiplied by a large enough positive constant to satisfy (2.36) and also (2.31). Combining (2.36) and (2.35), we obtain further that

$$\begin{aligned} f_1(X_0+Z) = \sum_{i=1}^{n}(\sigma_i^2(X_0+Z)+\gamma)^{\frac{1}{2}} &= r\sqrt{\gamma}+\sum_{i=r+1}^{n}((1-\alpha)^2\sigma_i^2(Z)+\gamma)^{\frac{1}{2}} \\ &\leq n\sqrt{\gamma}+(1-\alpha)\sum_{i=r+1}^{n}\sigma_i(Z) \leq f_1(X_0). \end{aligned} \tag{2.37}$$

Now it is easy to see that (2.34) and (2.37) together contradicts (2.20), which completes the proof.

2.10 Acknowledgements

The results in this chapter are based on joint work with Maryam Fazel. We also acknowledge Ting Kei Pong for helpful discussions.

Chapter 3

GRAPHICAL MODEL ESTIMATION

3.1 Learning the structure of Gaussian graphical models - Introduction

Graphical model is a way of encoding the conditional dependencies of a set of random variables $\mathcal{X}$. It is given by a graph G, where vertices correspond to the random variables, and the edges represent conditional dependencies between the random variables. A graphical model forms a *Markov random field* (MRF) if the lack of an edge between any two vertices implies that the corresponding random variables are conditionally independent given the rest. A matrix $\mathbf{\Theta}$ *defines a graph* G if the adjacency matrix of G is given by the *support matrix* of $\mathbf{\Theta}$. Gaussian random variables have the property that the graph defined by the *inverse covariance matrix* forms an MRF for the random variables. The MRF corresponding to Gaussian random variables is known as a *Gaussian graphical model*. Thus to learn the structure of a Gaussian graphical model, it is sufficient to estimate the support of the inverse covariance matrix (or precision matrix) corresponding to the random variables. Learning a *sparse* MRF is quite useful for inference due to it being efficient (esp. if the max clique size is small). Sparse MRFs are also more interpretable with applications in computational biology [Danaher et al., 2013], social networks [Robins et al., 2007], etc. For instance in gene-regulatory networks, it may be of interest to identify genes that are likely to have mutated between a healthy network and a cancer network [Mohan et al., 2013, Guo et al., 2010, Hara and Washio, 2013]. Learning MRFs that promote such differences between networks makes it easier to identify such genes. As another example, it may be of interest to identify the influential people in a social network with incomplete information. In this example, it might be of interest to learn MRFs that have a few nodes with high degree corresponding to influential people.

Our contributions are in learning the *structure* of the MRF under the Gaussianity assumption and prior information on the sparsity structure in the high dimensional setting (where the number of variables, p is greater than the number of samples, n). Specifically, we learn the precision matrix given the sample covariance matrix and

under the assumption that the true precision matrix has a support that is from a family of sparse structures. We present efficient algorithms to learn the precision matrix with guarantees in the form of *model selection consistency* and consistency under the Frobenius norm.

3.2 Literature review

There is a lot of literature on learning the structure of *Bayesian networks* [Friedman et al., 1999, Jaakkola et al., 2010]. Most of these approaches are based on devising a scoring function and picking the structure with the highest score [Acid et al., 2004, Heckerman, 1995]. Given the structure of an MRF or Bayesian network, the parameters corresponding to the probability distribution are usually learned through *maximum likelihood estimation* (MLE) or *maximum aposteriori estimation* (MAP) [Koller and Friedman, 2009]. We are particularly interested in learning both the MRF corresponding to Gaussian graphical models and also its parameters. We do this by estimating the precision matrix. Assume that we need to estimate the precision matrix corresponding to p variables given n samples. Denote the sample covariance matrix by $\mathbf{S}$. Then the maximum likelihood estimate for the precision matrix is given by:

$$\hat{\mathbf{\Theta}} = \underset{\mathbf{\Theta}}{\operatorname{argmax}} \log\det(\mathbf{\Theta}) - \langle \mathbf{\Theta}, \mathbf{S} \rangle \tag{3.1}$$

Note that the maximum likelihood estimate is just $\mathbf{S}^{-1}$, where $\mathbf{S}$ is the sample covariance matrix. However in the high-dimensional setting, where $p >> n$, $\mathbf{S}$ is not invertible and the problem of estimating the true precision matrix from few samples is ill-posed. Thus some form of regularization is required to both pick a unique solution and also to incorporate prior knowledge. A typical regularization used is the ℓ_1 norm (or Laplacian prior in the MAP estimate) given by the sum of the absolute values of the entries of the matrix and denoted by $\|.\|_1$. [Friedman et al., 2007] used the ℓ_1 norm as a regularizer yielding the following estimation procedure:

$$\hat{\mathbf{\Theta}} = \underset{\mathbf{\Theta}}{\operatorname{argmax}} \log\det(\mathbf{\Theta}) - \langle \mathbf{\Theta}, \mathbf{S} \rangle - \lambda \|\mathbf{\Theta}\|_1. \tag{3.2}$$

The advantage of ℓ_1 norm is that it promotes sparsity in the estimate allowing for fewer parameters and more interpretability of the Markov structure. The estimation procedure in (3.2) has been referred to as Graphical lasso (GL) [Friedman et al., 2007, Rothman et al., 2008] in the literature. However, in many biological and social network applications [Danaher et al., 2013, Robins et al., 2007] there is often more structure than just sparsity. In the next few sections, we discuss our contributions in learning GMRFs with structured sparsity (in contrast to plain vanilla sparsity) and provide convex formulations and efficient algorithms. Specifically, we introduce the structured graphical lasso as an extension of graphical lasso for learning a graphical model with structure. We consider a special case where the structure is given by hubs and observe empirically that our approach performs better than existing approaches. We also discuss consistency results in the appendix of this thesis. We show for the case of learning GMRFs with hub nodes that our model selection consistency result requires only $O(pk^2 \log p)$ samples (where k is the number of hubs in the true GMRF) as compared to $O(p^2 \log p)$ samples required by GL [Ravikumar et al., 2011]. When $k \sim O(1)$, our result requires much fewer samples than GL.

3.3 Structured graphical lasso

Sparsity is an important and useful prior to impose on the inverse covariance matrix or GMRF, since this leads to efficient inference and more interpretable graphical models. However in real world applications such as identifying mutant genes in gene regulatory networks or influential people in a social network graph, the structure of the MRF is more than just sparse - It is structured sparse. To leverage structured sparsity, we first need to define a framework of sparse structures that can then be imposed as priors in learning the GMRFs. Obozinksi and Jacob et al [Obozinski et al., Jacob et al., 2009] introduce the notion of overlapping groups that enable one to construct structures as a *union of overlapping groups*, where the groups are user defined. For example, when the groups are singleton elements, we get the plain vanilla sparsity (note that in this case the groups don't overlap). When the groups are entire rows or columns, the structures lead to union of rows and columns and represent MRFs with hub nodes. We now formalize the notion of groups and structures.

Notation and definitions:

Define a group, $\mathbf{G}$ as a set of indices that is a subset of $[p] \times [p]$ such that for all

$(i,j) \in \mathbf{G}$, we have that $i \leq j$. Thus a matrix that is supported on a group $\mathbf{G}$ is upper triangular. Let $\mathcal{G} = \{\mathbf{G}^1, \mathbf{G}^2, \ldots, \mathbf{G}^K\}$ denote a set of groups. Let $|\mathbf{G}^i| = \mathbf{r}^i$. We will associate with every group $\mathbf{G}^i$, a linear-operator $\mathcal{A}_i : \mathbb{R}^{\mathbf{r}^i} \rightarrow \mathbb{R}^{\mathbf{p}\times\mathbf{p}}$ such that for $v \in \mathbb{R}^{\mathbf{r}^i}$, $\mathcal{A}_i(v)$ is supported on group $\mathbf{G}^i$. Also, denote by $\mathcal{A}_i^*$, the adjoint of $\mathcal{A}_i$.

Definition 18 (Symmetric structure)**.** *A symmetric matrix $\boldsymbol{M}$ is said to have a symmetric structure under $\mathcal{G}$ if it can be expressed as $\boldsymbol{M} = \boldsymbol{V} + \boldsymbol{V}^T$ where, the support of $\boldsymbol{V}$ is a subset of a union of few groups in $\mathcal{G}$.*

We would like to define a norm that promotes matrices $\boldsymbol{\Theta}$ that can be expressed as $\mathbf{Y}+\mathbf{Z}$, where $\mathbf{Y}$ has a symmetric structure under $\mathcal{G}$ and $\mathbf{Z}$ is sparse. We note that the overlap norm defined by Obozinksi et al in [Obozinski et al.] promotes matrices that are a union of few groups (not necessarily symmetric). We now define a symmetric (and more general) variant of the overlap norm that promotes matrices that can be expressed as the sum of a symmetric structured matrix and a sparse matrix.

Definition 19 (Symmetric overlap norm)**.** *The symmetric overlap norm for a set of groups, $\mathcal{G}$ is given by,*

$$\begin{aligned} \Omega^{\mathcal{G}}(\boldsymbol{\Theta}) \quad = \quad & \min_{\{\mathbf{V}^i\}:\mathbf{V}^i\in\mathbb{R}^{|\mathbf{G}^i|},i=1,\ldots,K} && \lambda_1\|\mathbf{Z} - Diag(\mathbf{Z})\|_1 + \lambda_2\sum_{i=1}^{K}\|\mathbf{V}^i\|_1 + \lambda_3\sum_{i=1}^{K}\|\mathbf{V}^i\|_2 \\ & s.t. && \boldsymbol{\Theta} = \sum_{i=1}^{K}(\mathcal{A}_i(\mathbf{V}^i) + (\mathcal{A}_i(\mathbf{V}^i))^T) + \mathbf{Z}, \end{aligned} \tag{3.3}$$

where Diag($\mathbf{Z}$) refers to a diagonal matrix whose diagonals are given by the diagonals of $\mathbf{Z}$.

We next use the symmetric overlap norm as a regularizer for maximum likelihood estimation of GGMs to yield the following formulation:

$$\underset{\boldsymbol{\Theta}}{\text{minimize}} - \log\det(\boldsymbol{\Theta}) + \langle\boldsymbol{\Theta}, \mathbf{S}\rangle + \Omega^{\mathcal{G}}(\boldsymbol{\Theta}) \tag{3.4}$$

We refer to the above formulation as the *structured graphical lasso* (SGL). In the next section, we consider a special case of structured graphical lasso - hub graphical lasso (used to learn graphs with a few hubs).

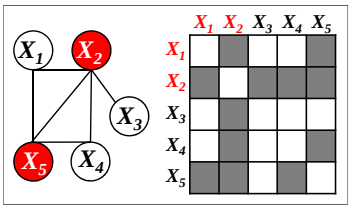


Figure 3.1: A graphical model corresponding to five random variables, $X_1, X_2, \ldots, X_5$ with two hub nodes X_2 and X_5. The adjacency matrix has a row-column structure corresponding to nodes X_2 and X_5.

3.4 Hub graphical lasso

In this section, we introduce the hub graphical lasso which is a special case of the structured graphical lasso defined earlier. Our goal is to learn a graphical model with a few hub nodes (or nodes that are densely connected). We note that a hub node in a graph corresponds to an entire row and column being non-zero in the adjacency matrix. When the random variables are Gaussian, learning a graphical model with few hubs is equivalent to learning a precision matrix whose support can be expressed as a union of a few rows and corresponding columns (corresponding to the hub nodes). In this problem, we do not know the location of the hub nodes or the number of hubs in the graph. However we use the prior that the graphical model has a few hub nodes. An example of a graph with a few hubs is shown in Figure 3.1. Graphical models with hubs find applications in biological and social network applications. For example, in a gene-regulatory network, regulatory genes regulate many other genes in the network and act as hub nodes. Similarly in a social network, influential nodes (people) are very well connected and have a much higher degree than other nodes in the network. In order to learn a precision matrix with hubs, we define the *row column overlap norm* (RCON), which is a special case of the symmetric overlap norm (where the groups are given by the upper triangular part of a row and the corresponding column):

$$\begin{aligned} \Omega^{\mathcal{H}}(\boldsymbol{\Theta}) \quad = \quad & \underset{V,Z}{\text{minimize}}\, \lambda_1 \|\boldsymbol{Z} - \text{Diag}(\boldsymbol{Z})\|_1 + \lambda_2 \|\boldsymbol{V} - \text{Diag}(\boldsymbol{V})\|_1 + \\ & \lambda_3 \sum_{i=1}^{p} \|(\boldsymbol{V} - \text{Diag}(\boldsymbol{V}))_{\cdot i}\|_q \\ & \text{s.t. } \boldsymbol{\Theta} = \boldsymbol{V} + \boldsymbol{V}^T + \boldsymbol{Z}, \end{aligned} \tag{3.5}$$

where $\boldsymbol{Y}_{\cdot i}$ denotes the ith column of matrix $\boldsymbol{Y}$ and $\|\boldsymbol{Y}_{\cdot i}\|_q$ denotes the ℓ_q norm of vector $\boldsymbol{Y}_{\cdot i}$ (where the ℓ_q norm is defined in Section 2.3.1). The RCON penalty for $q = 2$ promotes a hub structured matrix as in Figure 3.2. The hub structured matrix is a generalization of the row-column structure shown in the adjacency matrix in Figure 3.1, where we allow for hub nodes (captured by $\boldsymbol{V}$) and nodes with a sparse degree (captured by the $\boldsymbol{Z}$ term). In Figure 3.2, we see that $\boldsymbol{\Theta}$ can be decomposed into a sparse matrix $\boldsymbol{Z}$ and a matrix $\boldsymbol{V}$ that has a few non-zero columns. It is well-known in compressive sensing and statistics that the ℓ_1/ℓ_2 norm promotes a column sparse matrix (a matrix where most columns are identically zero). Because of the ℓ_1/ℓ_2 norm on $\boldsymbol{V} - \mathrm{Diag}(\boldsymbol{V})$ in (3.5), we see that $\boldsymbol{V} - \mathrm{diag}(\boldsymbol{V})$ is column-sparse in Figure 3.2. The ℓ_1 penalty on $\boldsymbol{V} - \mathrm{Diag}(\boldsymbol{V})$ in (3.5) also allows for some of the non-zero columns in $\boldsymbol{V}$ to be sparse. Also note that $\boldsymbol{Y} = \boldsymbol{V} + \boldsymbol{V}^T$ is a symmetric structured matrix under the groups described earlier.

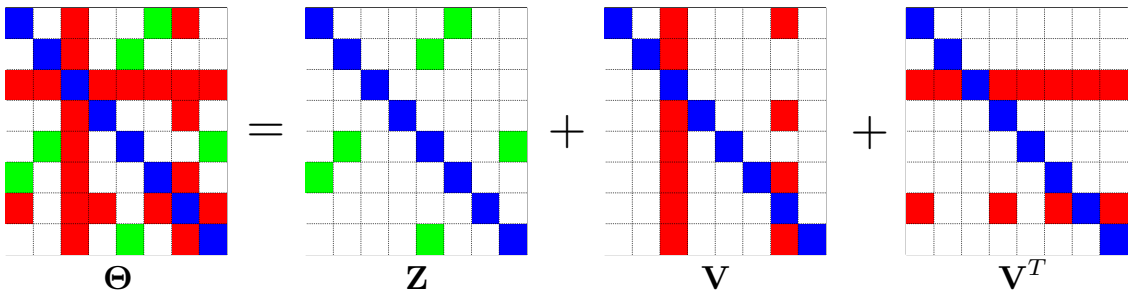


Figure 3.2: Decomposition of a symmetric matrix $\boldsymbol{\Theta}$ into $\mathbf{Z} + \mathbf{V} + \mathbf{V}^T$, where $\mathbf{Z}$ is sparse, and most columns of $\mathbf{V}$ are entirely zero. Blue, white, green, and red elements are diagonal, zero, non-zero in $\mathbf{Z}$, and non-zero due to two hubs in $\mathbf{V}$, respectively.

The hub graphical lasso uses the RCON and is given by the following formulation:

$$\underset{\boldsymbol{\Theta}}{\text{minimize}} \quad -\log\det(\boldsymbol{\Theta}) + \langle \boldsymbol{\Theta}, \mathbf{S} \rangle + \Omega^{\mathcal{H}}(\boldsymbol{\Theta}) \tag{3.6}$$

3.4.1 Properties of hub graphical lasso

We first note that HGL (3.6) can be reformulated as:

$$\begin{aligned} \underset{V,Z}{\text{minimize}} \quad & -\log\det(\boldsymbol{\Theta}) + \langle \boldsymbol{\Theta}, \mathbf{S} \rangle + \lambda_1 \|\boldsymbol{Z} - \text{Diag}(\boldsymbol{Z})\|_1 + \\ & \lambda_2 \|\boldsymbol{V} - \text{Diag}(\boldsymbol{V})\|_1 + \lambda_3 \sum_i \|(\boldsymbol{V} - \text{Diag}(\boldsymbol{V}))_i\|_q \\ & \text{s.t. } \boldsymbol{\Theta} = \boldsymbol{V} + \boldsymbol{V}^T + \boldsymbol{Z} \end{aligned} \tag{3.7}$$

We now present several properties of the HGL optimization problem (3.7), which can be used to provide guidance on the suitable range for the tuning parameters λ_1, λ_2, and λ_3. In what follows, $\hat{\boldsymbol{Z}}$ and $\hat{\boldsymbol{V}}$ denote the optimal solutions for $\mathbf{Z}$ and $\mathbf{V}$ in (3.7). Let $\frac{1}{s} + \frac{1}{q} = 1$ (recall that q appears in (3.5)).

Lemma 20. *A sufficient condition for $\hat{\boldsymbol{Z}}$ to be a diagonal matrix is that $\lambda_1 > \frac{\lambda_2+\lambda_3}{2}$.*

Lemma 21. *A sufficient condition for $\hat{\boldsymbol{V}}$ to be a diagonal matrix is that $\lambda_1 < \frac{\lambda_2}{2} + \frac{\lambda_3}{2(p-1)^{1/s}}$.*

Corollary 22. *A necessary condition for both $\hat{\boldsymbol{V}}$ and $\boldsymbol{Z}^*$ to be non-diagonal matrices is that $\frac{\lambda_2}{2} + \frac{\lambda_3}{2(p-1)^{1/s}} \leq \lambda_1 \leq \frac{\lambda_2+\lambda_3}{2}$.*

Furthermore, (3.7) reduces to the graphical lasso under a simple condition.

Lemma 23. *If $q = 1$, then (3.7) reduces to GL with tuning parameter $\min\left\{\lambda_1, \frac{\lambda_2+\lambda_3}{2}\right\}$.*

Note also that when $\lambda_2 \to \infty$ or $\lambda_3 \to \infty$, (3.7) reduces to GL with tuning parameter λ_1. However, throughout the rest of this paper, we assume that $q = 2$, and λ_2 and λ_3 are finite.

3.4.2 Tuning Parameter Selection via Bayesian information criterion

Lemmas 20, 21, and 23 make it easy to tune for the regularization parameters. In this section, we propose an alternative, *Bayesian information criterion* (BIC)-type quantity for tuning parameter selection in (3.6). The BIC is appropriate if we want a single choice of $(\lambda_1, \lambda_2, \lambda_3)$. On the other hand, the previous lemmas restrict the range of $\lambda_1, \lambda_2, \lambda_3$, making it easy to compute the solution path. Recall from Section 3.4 that the RCON (3.5) decomposes the parameter of interest into the sum of three

matrices, $\mathbf{\Theta} = \mathbf{Z} + \mathbf{V} + \mathbf{V}^T$, and places an ℓ_1 penalty on $\mathbf{Z}$, and an ℓ_1/ℓ_2 penalty on $\mathbf{V}$.

For the graphical lasso problem, many authors have proposed to select the tuning parameter λ such that $\hat{\mathbf{\Theta}}$ minimizes the following quantity:

$$-n \cdot \log\det(\hat{\mathbf{\Theta}}) + n \cdot \text{trace}(\mathbf{S}\hat{\mathbf{\Theta}}) + \log(n) \cdot |\hat{\mathbf{\Theta}}|,$$

where $|\hat{\mathbf{\Theta}}|$ is the cardinality of $\hat{\mathbf{\Theta}}$, that is, the number of unique non-zeros in $\hat{\mathbf{\Theta}}$ [see, e.g., Yuan and Lin].[1]

Using a similar idea, we propose the following BIC-type quantity for selecting the set of tuning parameters $(\lambda_1, \lambda_2, \lambda_3)$ for (3.6):

$$\text{BIC}(\hat{\mathbf{\Theta}}, \hat{\boldsymbol{V}}, \hat{\boldsymbol{Z}}) = -n\cdot\log\det(\hat{\mathbf{\Theta}}) + n\cdot\text{trace}(\mathbf{S}\hat{\mathbf{\Theta}}) + \log(n)\cdot|\hat{\boldsymbol{Z}}| + \log(n)\cdot\Big(\nu + c\cdot[|\hat{\boldsymbol{V}}| - \nu]\Big),$$

where ν is the number of estimated hub nodes, that is, $\nu = \sum_{j=1}^{p} 1_{\{\|\hat{\boldsymbol{V}}_j\|_0 > 0\}}$, c is a constant between zero and one, and $|\hat{\boldsymbol{Z}}|$ and $|\hat{\boldsymbol{V}}|$ are the cardinalities (the number of unique non-zeros) of $\hat{\boldsymbol{Z}}$ and $\hat{\boldsymbol{V}}$, respectively.[2] We select the set of tuning parameters $(\lambda_1, \lambda_2, \lambda_3)$ for which the quantity $\text{BIC}(\hat{\mathbf{\Theta}}, \hat{\boldsymbol{V}}, \hat{\boldsymbol{Z}})$ is minimized. Note that when the constant c is small, $\text{BIC}(\hat{\mathbf{\Theta}}, \hat{\boldsymbol{V}}, \hat{\boldsymbol{Z}})$ will favor more hub nodes in $\hat{\boldsymbol{V}}$. In this manuscript, we take $c = 0.2$.

3.4.3 ADMM algorithm for hub graphical lasso

In order to solve (3.7), we use an *alternating direction method of multipliers* (ADMM) algorithm [see, e.g., Eckstein, 2012, Boyd et al., 2011]. ADMM is an attractive algorithm for this problem, as it allows us to decouple some of the terms in (3.7) that are difficult to optimize jointly. In order to develop an ADMM algorithm for (3.7) with guaranteed convergence, we reformulate it as a consensus problem, as in Ma et al. [2013]. The convergence of the algorithm to the optimal solution follows from

---

[1] The term $\log(n) \cdot |\hat{\mathbf{\Theta}}|$ is motivated by the fact that the degrees of freedom for an estimate involving the ℓ_1 penalty can be approximated by the cardinality of the estimated parameter [Zou et al., 2007].

[2] The term $\log(n) \cdot |\hat{\boldsymbol{Z}}|$ is motivated by the degrees of freedom from the ℓ_1 penalty, and the term $\log(n) \cdot \Big(\nu + c \cdot [|\hat{\boldsymbol{V}}| - \nu]\Big)$ is motivated by an approximation of the degrees of freedom of the ℓ_2 penalty proposed in [Yuan and Lin].

classical results [see, e.g., the review papers Boyd et al., 2011, Eckstein, 2012].

Algorithm 5: ADMM Algorithm for Solving (3.7).

---

1. **Initialize** the parameters:
   (a) primal variables $\mathbf{\Theta}, \mathbf{V}, \mathbf{Z}, \tilde{\mathbf{\Theta}}, \tilde{\mathbf{V}}$, and $\tilde{\mathbf{Z}}$ to the $p \times p$ identity matrix.
   (b) dual variables $\mathbf{W}_1, \mathbf{W}_2$, and $\mathbf{W}_3$ to the $p \times p$ zero matrix.
   (c) constants $\rho > 0$ and $\tau > 0$.
2. **Iterate** until the stopping criterion $\frac{\|\mathbf{\Theta}_t - \mathbf{\Theta}_{t-1}\|_F^2}{\|\mathbf{\Theta}_{t-1}\|_F^2} \leq \tau$ is met, where $\mathbf{\Theta}_t$ is the value of $\mathbf{\Theta}$ obtained at the tth iteration:
   (a) Update $\mathbf{\Theta}, \mathbf{V}, \mathbf{Z}$:
      i. $\mathbf{\Theta} = \underset{\mathbf{\Theta} \in \mathcal{S}}{\arg\min} \left\{ \ell(\mathbf{X}, \mathbf{\Theta}) + \frac{\rho}{2} \|\mathbf{\Theta} - \tilde{\mathbf{\Theta}} + \mathbf{W}_1\|_F^2 \right\}$.
      ii. $\mathbf{Z} = S(\tilde{\mathbf{Z}} - \mathbf{W}_3, \frac{\lambda_1}{\rho})$, $\text{diag}(\mathbf{Z}) = \text{Diag}(\tilde{\mathbf{Z}} - \mathbf{W}_3)$. Here S denotes the soft-thresholding operator, applied element-wise to a matrix: $S(A_{ij}, b) = \text{sign}(A_{ij}) \max(|A_{ij}| - b, 0)$.
      iii. $\mathbf{C} = \tilde{\mathbf{V}} - \mathbf{W}_2 - \text{Diag}(\tilde{\mathbf{V}} - \mathbf{W}_2)$.
      iv. $\mathbf{V}_j = \max\left(1 - \frac{\lambda_3}{\rho \|S(\mathbf{C}_j, \lambda_2/\rho)\|_2}, 0\right) \cdot S(\mathbf{C}_j, \lambda_2/\rho)$ for $j = 1, \ldots, p$.
      v. $\text{diag}(\mathbf{V}) = \text{Diag}(\tilde{\mathbf{V}} - \mathbf{W}_2)$.
   (b) Update $\tilde{\mathbf{\Theta}}, \tilde{\mathbf{V}}, \tilde{\mathbf{Z}}$:
      i. $\mathbf{\Gamma} = \frac{\rho}{6} \left[(\mathbf{\Theta} + \mathbf{W}_1) - (\mathbf{V} + \mathbf{W}_2) - (\mathbf{V} + \mathbf{W}_2)^T - (\mathbf{Z} + \mathbf{W}_3) \right]$.
      ii. $\tilde{\mathbf{\Theta}} = \mathbf{\Theta} + \mathbf{W}_1 - \frac{1}{\rho}\mathbf{\Gamma}$;
      iii. $\tilde{\mathbf{V}} = \frac{1}{\rho}(\mathbf{\Gamma} + \mathbf{\Gamma}^T) + \mathbf{V} + \mathbf{W}_2$;
      iv. $\tilde{\mathbf{Z}} = \frac{1}{\rho}\mathbf{\Gamma} + \mathbf{Z} + \mathbf{W}_3$.
   (c) Update $\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3$:
      i. $\mathbf{W}_1 = \mathbf{W}_1 + \mathbf{\Theta} - \tilde{\mathbf{\Theta}}$; ii. $\mathbf{W}_2 = \mathbf{W}_2 + \mathbf{V} - \tilde{\mathbf{V}}$; iii. $\mathbf{W}_3 = \mathbf{W}_3 + \mathbf{Z} - \tilde{\mathbf{Z}}$.

Let $\ell(\mathbf{X}, \mathbf{\Theta}) = -\log\det(\mathbf{\Theta}) + \langle \mathbf{\Theta}, \mathbf{S} \rangle$. Also let $\mathbf{B} = (\mathbf{\Theta}, \mathbf{V}, \mathbf{Z})$, $\tilde{\mathbf{B}} = (\tilde{\mathbf{\Theta}}, \tilde{\mathbf{V}}, \tilde{\mathbf{Z}})$,

$$f(\mathbf{B}) = \ell(\mathbf{X}, \mathbf{\Theta}) + \lambda_1 \|\mathbf{Z} - \text{Diag}(\mathbf{Z})\|_1 + \lambda_2 \|\mathbf{V} - \text{Diag}(\mathbf{V})\|_1 + \lambda_3 \sum_{j=1}^{p} \|(\mathbf{V} - \text{Diag}(\mathbf{V}))\|_2,$$

and

$$g(\tilde{\mathbf{B}}) = \begin{cases} 0 & \text{if } \tilde{\mathbf{\Theta}} = \tilde{\mathbf{V}} + \tilde{\mathbf{V}}^T + \tilde{\mathbf{Z}} \\ \infty & \text{otherwise.} \end{cases}$$

Then, we can rewrite (3.7) as

$$\underset{\mathbf{B},\tilde{\mathbf{B}}}{\text{minimize}} \left\{ f(\mathbf{B}) + g(\tilde{\mathbf{B}}) \right\} \qquad \text{subject to } \mathbf{B} = \tilde{\mathbf{B}}. \tag{3.8}$$

The scaled augmented Lagrangian for (3.8) takes the form

$$\begin{aligned} L(\mathbf{B}, \tilde{\mathbf{B}}, \mathbf{W}) = \ell(\mathbf{X}, \mathbf{\Theta}) + \lambda_1 \|\mathbf{Z} - \text{Diag}(\mathbf{Z})\|_1 + \lambda_2 \|\mathbf{V} - \text{Diag}(\mathbf{V})\|_1 \\ + \lambda_3 \sum_{j=1}^{p} \|(\mathbf{V} - \text{Diag}(\mathbf{V}))_j\|_2 + g(\tilde{\mathbf{B}}) + \frac{\rho}{2} \|\mathbf{B} - \tilde{\mathbf{B}} + \mathbf{W}\|_F^2, \end{aligned}$$

where $\mathbf{B}$ and $\tilde{\mathbf{B}}$ are the primal variables, and $\mathbf{W} = (\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3)$ is the dual variable. Note that the scaled augmented Lagrangian can be derived from the usual Lagrangian by adding a quadratic term and completing the square [Boyd et al., 2011].

An ADMM algorithm for solving (3.7) is provided in Algorithm 5.

Remark 24. *We note that although ADMM provides an easy way to solve (3.7), it is not very efficient as it involves the computation of SVD in each iteration of the algorithm. To resolve this, we come up a faster SVD-free algorithm, which we discuss in Chapter 5 of the thesis.*

3.4.4 Experimental results

In this section, we compare HGL to two sets of proposals: proposals that learn an Erdős-Rényi Gaussian graphical model, and proposals that learn a Gaussian graphical model in which some nodes are highly-connected.

Notation and Measures of Performance

We start by defining some notation. Let $\hat{\mathbf{\Theta}}$ be the estimate of $\mathbf{\Theta} = \mathbf{\Sigma}^{-1}$ from a given proposal, and let $\hat{\mathbf{\Theta}}_j$ be its jth column. Let $\mathcal{H}$ denote the set of indices of the hub nodes in $\mathbf{\Theta}$ (that is, this is the set of true hub nodes in the graph), and let $|\mathcal{H}|$ denote the cardinality of the set. In addition, let $\hat{\mathcal{H}}_r$ be the set of *estimated hub nodes*: the

set of nodes in $\hat{\mathbf{\Theta}}$ that are among the $|\mathcal{H}|$ most highly-connected nodes, and that have at least r edges. The values chosen for $|\mathcal{H}|$ and r depend on the simulation set-up, and will be specified in each simulation study.

We now define several measures of performance that will be used to evaluate the various methods.

- Number of correctly estimated edges: $\sum_{j<j'} \left(1_{\{|\hat{\mathbf{\Theta}}_{jj'}|>10^{-5} \text{ and } |\mathbf{\Theta}_{jj'}|\neq 0\}}\right)$.
- Proportion of correctly estimated hub edges:

$$\frac{\sum_{j\in\mathcal{H},j'\neq j} \left(1_{\{|\hat{\mathbf{\Theta}}_{jj'}|>10^{-5} \text{ and } |\mathbf{\Theta}_{jj'}|\neq 0\}}\right)}{\sum_{j\in\mathcal{H},j'\neq j} \left(1_{\{|\mathbf{\Theta}_{jj'}|\neq 0\}}\right)}.$$

- Proportion of correctly estimated hub nodes: $\frac{|\hat{\mathcal{H}}_r \cap \mathcal{H}|}{|\mathcal{H}|}$.
- Sum of squared errors: $\sum_{j<j'} \left(\hat{\mathbf{\Theta}}_{jj'} - \mathbf{\Theta}_{jj'}\right)^2$.

Data Generation

We consider three set-ups for generating a $p \times p$ adjacency matrix $\mathbf{A}$.

1. Network with hub nodes: for all $i < j$, we set $A_{ij} = 1$ with probability 0.02, and zero otherwise. We then set A_{ji} equal to A_{ij}. Next, we randomly select $|\mathcal{H}|$ hub nodes and set the elements of the corresponding rows and columns of $\mathbf{A}$ to equal one with probability 0.7 and zero otherwise.

2. Network with two connected components and hub nodes: the adjacency matrix is generated as $\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & 0 \\ 0 & \mathbf{A}_2 \end{pmatrix}$, with $\mathbf{A}_1$ and $\mathbf{A}_2$ as in Set-up I, each with $|\mathcal{H}|/2$ hub nodes.

3. Scale-free network:[3] the probability that a given node has k edges is proportional to $k^{-\alpha}$. Barabási and Albert [1999] observed that many real-world networks have $\alpha \in [2.1, 4]$; we took $\alpha = 2.5$. Note that there is no natural notion of hub

[3]Recall that our proposal is not intended for estimating a scale-free network.

nodes in a scale-free network. While some nodes in a scale-free network have more edges than one would expect in an Erdős-Rényi graph, there is no clear distinction between "hub" and "non-hub" nodes, unlike in Set-ups I and II. In our simulation settings, we consider any node that is connected to more than 5% of all other nodes to be a hub node.[4]

We then use the adjacency matrix $\mathbf{A}$ to create a matrix $\mathbf{E}$, as

$$E_{ij} \overset{\text{i.i.d.}}{\sim} \begin{cases} 0 & \text{if } A_{ij} = 0 \\ \text{Unif}([-0.75, -0.25] \cup [0.25, 0.75]) & \text{otherwise,} \end{cases}$$

and set $\bar{\mathbf{E}} = \frac{1}{2}(\mathbf{E} + \mathbf{E}^T)$. Given the matrix $\bar{\mathbf{E}}$, we set $\boldsymbol{\Sigma}^{-1}$ equal to $\bar{\mathbf{E}} + (0.1 - \Lambda_{\min}(\bar{\mathbf{E}}))\mathbf{I}$, where $\Lambda_{\min}(\bar{\mathbf{E}})$ is the smallest eigenvalue of $\bar{\mathbf{E}}$. We generate the data matrix $\mathbf{X}$ according to $\mathbf{x}_1, \ldots, \mathbf{x}_n \overset{\text{i.i.d.}}{\sim} N(\mathbf{0}, \boldsymbol{\Sigma})$. Then, variables are standardized to have standard deviation one.

Comparison to Graphical Lasso and Neighborhood Selection

In this subsection, we compare the performance of HGL to two proposals that learn a sparse Gaussian graphical model.

- The graphical lasso [Friedman et al., 2007], implemented using the `R` package `glasso`.
- The neighborhood selection approach of Meinshausen and Bülhmann [2006], implemented using the `R` package `glasso`. This approach involves performing p ℓ_1-penalized regression problems, each of which involves regressing one feature onto the others.

We consider the three simulation set-ups described in the previous section with $n = 1000$, $p = 1500$, and $|\mathcal{H}| = 30$ hub nodes in Set-ups I and II. Figure 3.3 displays the results, averaged over 100 simulated data sets. Note that the sum of squared errors

[4]The cutoff threshold of 5% is chosen in order to capture the most highly-connected nodes in the scale-free network. In our simulation study, around three nodes are connected to at least $0.05 \times p$ other nodes in the network. The precise choice of cut-off threshold has little effect on the results obtained in the figures that follow.

is not computed for Meinshausen and Bülhmann [2006], since it does not directly yield an estimate of $\boldsymbol{\Theta} = \boldsymbol{\Sigma}^{-1}$.

HGL has three tuning parameters. To obtain the curves shown in Figure 3.3, we fixed $\lambda_1 = 0.4$, considered three values of λ_3 (each shown in a different color in Figure 3.3), and used a fine grid of values of λ_2. The solid black circle in Figure 3.3 corresponds to the set of tuning parameters $(\lambda_1, \lambda_2, \lambda_3)$ for which the Bayesian information criterion is minimized. The graphical lasso and Meinshausen and Bülhmann [2006] each involves one tuning parameter; we applied them using a fine grid of the tuning parameter to obtain the curves shown in Figure 3.3.

Results for Set-up I are displayed in Figures 3.3-I(a) through 3.3-I(d), where we calculate the proportion of correctly estimated hub nodes as defined in Section 3.4.4 with $r = 300$. Since this simulation set-up exactly matches the assumptions of HGL, it is not surprising that HGL outperforms the other methods. In particular, HGL is able to identify most of the hub nodes when the number of estimated edges is approximately equal to the true number of edges. We see similar results for Set-up II in Figures 3.3-II(a) through 3.3-II(d), where the proportion of correctly estimated hub nodes is as defined in Section 3.4.4 with $r = 150$.

In Set-up III, recall that we define a node that is connected to at least 5% of all nodes to be a hub. The proportion of correctly estimated hub nodes is then as defined in Section 3.4.4 with $r = 0.05 \times p$. The results are presented in Figures 3.3-III(a) through 3.3-III(d). In this set-up, only approximately three of the nodes (on average) have more than 50 edges, and the hub nodes are not as highly-connected as in Set-up I or Set-up II. Nonetheless, HGL outperforms the graphical lasso and Meinshausen and Bülhmann [2006].

Finally, we see from Figure 3 that the set of tuning parameters $(\lambda_1, \lambda_2, \lambda_3)$ selected using BIC performs reasonably well. In particular, the graphical lasso solution always has BIC larger than HGL, and hence, is not selected.

Comparison to Additional Proposals

In this subsection, we compare the performance of HGL to three additional proposals:

- The partial correlation screening procedure of Hero and Rajaratnam [2014]. The elements of the partial correlation matrix (computed using a pseudo-inverse when $p > n$) are thresholded based on their absolute value, and a hub node

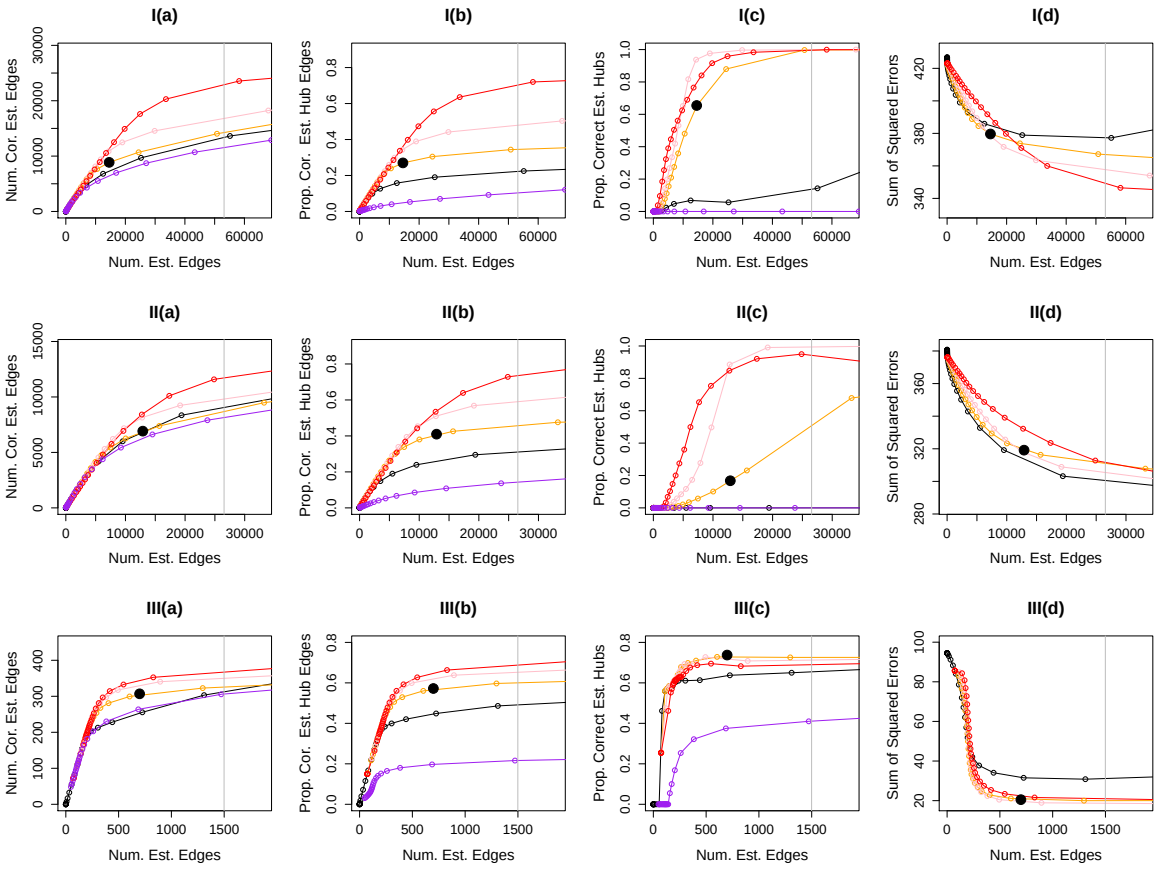


Figure 3.3: Simulation for Gaussian graphical model. Row I: Results for Set-up I. Row II: Results for Set-up II. Row III: Results for Set-up III. The results are for $n = 1000$ and $p = 1500$. In each panel, the x-axis displays the number of estimated edges, and the vertical gray line is the number of edges in the true network. The y-axes are as follows: Column (a): Number of correctly estimated edges; Column (b): Proportion of correctly estimated hub edges; Column (c): Proportion of correctly estimated hub nodes; Column (d): Sum of squared errors. The black solid circles are the results for HGL based on tuning parameters selected using the BIC-type criterion. Colored lines correspond to the graphical lasso [Friedman et al., 2007] (——); HGL with $\lambda_3 = 0.5$ (——), $\lambda_3 = 1$ (——), and $\lambda_3 = 2$ (——); neighborhood selection [Meinshausen and Bülhmann, 2006] (——).

is declared if the number of nonzero elements in the corresponding column of the thresholded partial correlation matrix is sufficiently large. Note that the purpose of Hero and Rajaratnam [2014] is to screen for hub nodes, rather than to estimate the individual edges in the network.

- The scale-free network estimation procedure of Liu and Ihler [2011]. This is the solution to the non-convex optimization problem

$$\underset{\mathbf{\Theta}\in\mathcal{S}}{\text{minimize}} \left\{ -\log\det\mathbf{\Theta} + \text{trace}(\mathbf{S\Theta}) + \alpha\sum_{j=1}^{p}\log(\|\mathbf{\Theta}_{\setminus j}\|_1+\epsilon_j) + \sum_{j=1}^{p}\beta_j|\mathbf{\Theta}_{jj}| \right\}, \tag{3.9}$$

  where $\mathbf{\Theta}_{\setminus j} = \{\mathbf{\Theta}_{jj'}|j' \neq j\}$, and ϵ_j, β_j, and α are tuning parameters. Here, $\mathcal{S} = \{\mathbf{\Theta} : \mathbf{\Theta} \succ 0 \text{ and } \mathbf{\Theta} = \mathbf{\Theta}^T\}$.

- Sparse partial correlation estimation procedure of Peng et al. [2009], implemented using the `R` package `space`. This is an extension of the neighborhood selection approach of Meinshausen and Bülhmann [2006] that combines p ℓ_1-penalized regression problems in order to obtain a symmetric estimator. The authors claimed that the proposal performs well in estimating a scale-free network.

We generated data under Set-ups I and III (described in Section 3.4.4) with $n = 250$ and $p = 500$,[5] and with $|\mathcal{H}| = 10$ for Set-up I. The results, averaged over 100 data sets, are displayed in Figures 3.4 and 3.5.

To obtain Figures 3.4 and 3.5, we applied Liu and Ihler [2011] using a fine grid of α values, and using the choices for β_j and ϵ_j specified by the authors: $\beta_j = 2\alpha/\epsilon_j$, where ϵ_j is a small constant specified in Liu and Ihler [2011]. There are two tuning parameters in Hero and Rajaratnam [2014]: (1) ρ, the value used to threshold the partial correlation matrix, and (2) d, the number of non-zero elements required for a column of the thresholded matrix to be declared a hub node. We used $d = \{10, 20\}$ in Figures 3.4 and 3.5, and used a fine grid of values for ρ. Note that the value of d has no effect on the results for Figures 3.4(a)-(b) and Figures 3.5(a)-(b), and that larger values of d tend to yield worse results in Figures 3.4(c) and 3.5(c). For Peng et al.

[5] In this subsection, a small value of p was used due to the computations required to run the R package space, as well as computational demands of the Liu and Ihler [2011] algorithm.

[2009], we used a fine grid of tuning parameter values to obtain the curves shown in Figures 3.4 and 3.5. The sum of squared errors was not reported for Peng et al. [2009] and Hero and Rajaratnam [2014] since they do not directly yield an estimate of the precision matrix. As a baseline reference, the graphical lasso is included in the comparison.

We see from Figure 3.4 that HGL outperforms the competitors when the underlying network contains hub nodes. It is not surprising that Liu and Ihler [2011] yields better results than the graphical lasso, since the former approach is implemented via an iterative procedure: in each iteration, the graphical lasso is performed with an updated tuning parameter based on the estimate obtained in the previous iteration. Hero and Rajaratnam [2014] has the worst results in Figures 3.4(a)-(b); this is not surprising, since the purpose of Hero and Rajaratnam [2014] is to screen for hub nodes, rather than to estimate the individual edges in the network.

From Figure 3.5, we see that the performance of HGL is comparable to that of Liu and Ihler [2011] and Peng et al. [2009] under the assumption of a scale-free network; note that this is the precise setting for which Liu and Ihler [2011]'s proposal is intended, and Peng et al. [2009] reported that their proposal performs well in this setting. In contrast, HGL is not intended for the scale-free network setting (as mentioned in the Introduction, it is intended for a setting with hub nodes). Again, Liu and Ihler [2011] and Peng et al. [2009] outperform the graphical lasso, and Hero and Rajaratnam [2014] has the worst results in Figures 3.5(a)-(b). Finally, we see from Figures 3.4 and 3.5 that the BIC-type criterion for HGL proposed in Section 3.4.2 yields good results.

3.4.5 Application to Gene Expression Data

We applied HGL to a publicly available cancer gene expression data set [Verhaak et al., 2010]. The data set consists of mRNA expression levels for 17,814 genes in 401 patients with glioblastoma multiforme (GBM), an extremely aggressive cancer with very poor patient prognosis. Among 7,462 genes known to be associated with cancer [Rappaport et al., 2013], we selected 500 genes with the highest variance.

We aim to reconstruct the gene regulatory network that represents the interactions among the genes, as well as to identify hub genes that tend to have many interactions with other genes. Such genes likely play an important role in regulating many

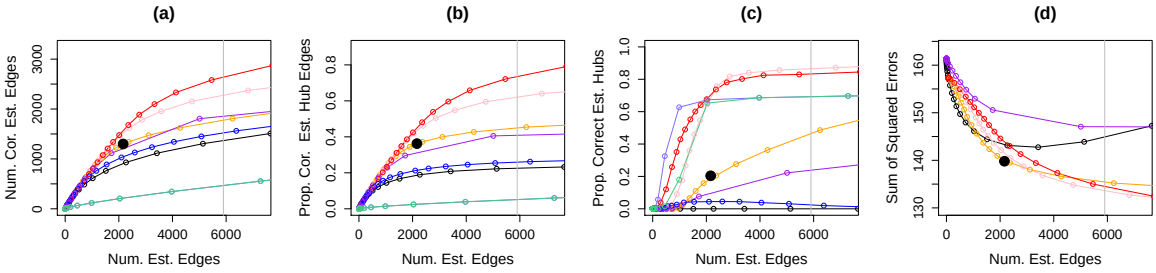


Figure 3.4: Simulation for the Gaussian graphical model. Set-up I was applied with $n = 250$ and $p = 500$. Details of the axis labels and the solid black circles are as in Figure 3.3. The colored lines correspond to the graphical lasso [Friedman et al., 2007] (——); HGL with $\lambda_3 = 1$ (——), $\lambda_3 = 2$ (——), and $\lambda_3 = 3$ (——); the hub screening procedure [Hero and Rajaratnam, 2014] with $d = 10$ (——) and $d = 20$ (——); the scale-free network approach [Liu and Ihler, 2011] (——); sparse partial correlation estimation [Peng et al., 2009] (——).

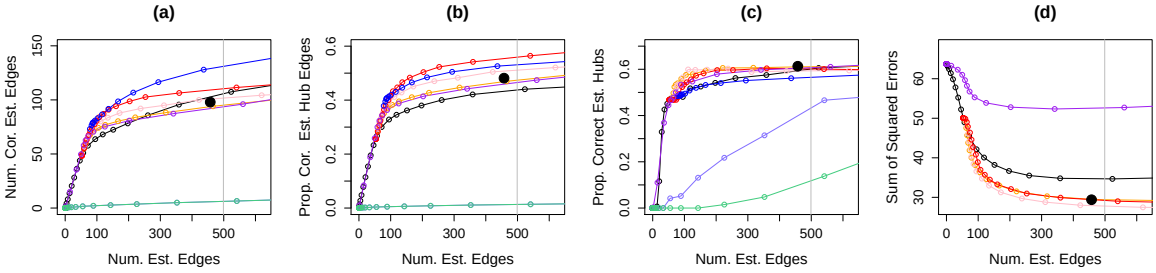


Figure 3.5: Simulation for the Gaussian graphical model. Set-up III was applied with $n = 250$ and $p = 500$. Details of the axis labels and the solid black circles are as in Figure 3.3. The colored lines correspond to the graphical lasso [Friedman et al., 2007] (——); HGL with $\lambda_3 = 1$ (——), $\lambda_3 = 2$ (——), and $\lambda_3 = 3$ (——); the hub screening procedure [Hero and Rajaratnam, 2014] with $d = 10$ (——) and $d = 20$ (——); the scale-free network approach [Liu and Ihler, 2011] (——); sparse partial correlation estimation [Peng et al., 2009] (——).

other genes in the network. Identifying such regulatory genes will lead to a better understanding of brain cancer, and eventually may lead to new therapeutic targets. Since we are interested in identifying hub genes, and not as interested in identifying edges between non-hub nodes, we fix $\lambda_1 = 0.6$ such that the matrix $\mathbf{Z}$ is sparse. We fix $\lambda_3 = 6.5$ to obtain a few hub nodes, and we select λ_2 ranging from 0.1 to 0.7 using the BIC-type criterion presented in Section 3.4.2.

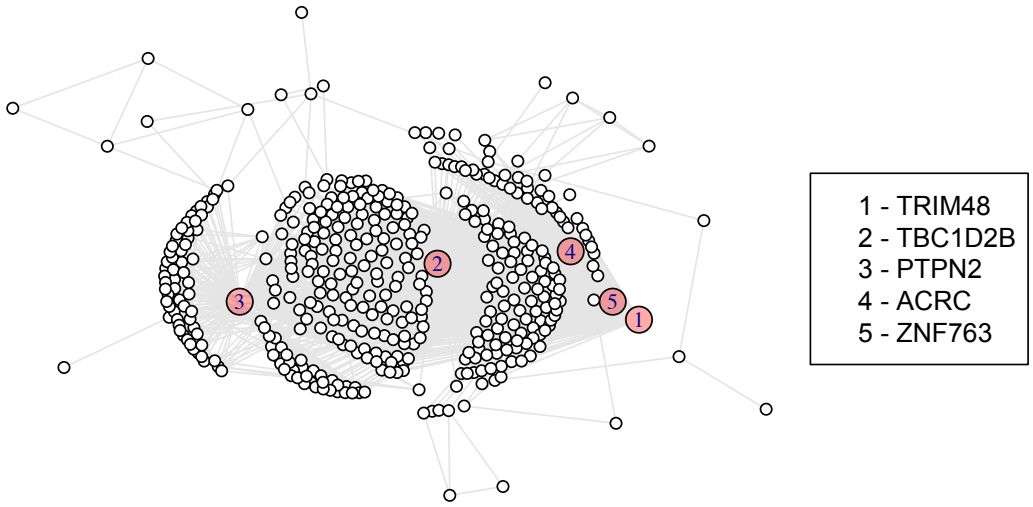


Figure 3.6: Results for HGL on the GBM data with tuning parameters selected using BIC: $\lambda_1 = 0.6$, $\lambda_2 = 0.4$, $\lambda_3 = 6.5$. Only nodes with at least two edges in the estimated network are displayed. Nodes displayed in pink were found to be hubs by the HGL algorithm.

We applied HGL with this set of tuning parameters to the empirical covariance matrix corresponding to the 401×500 data matrix, after standardizing each gene to have variance one. In Figure 3.6, we plotted the resulting network (for simplicity, only the 438 genes with at least two neighbors are displayed). We found that five genes are identified as hubs. These genes are TRIM48, TBC1D2B, PTPN2, ACRC, and ZNF763, in decreasing order of estimated edges.

Interestingly, some of these genes have known regulatory roles. PTPN2 is known to be a signaling molecule that regulates a variety of cellular processes including cell growth, differentiation, mitotic cycle, and oncogenic transformation [Maglott et al.,

2004]. ZNF763 is a DNA-binding protein that regulates the transcription of other genes [Maglott et al., 2004]. These genes do not appear to be highly-connected to many other genes in the estimate that results from applying the graphical lasso to this same data set (results not shown). These results indicate that HGL can be used to recover known regulators, as well as to suggest other potential regulators that may be targets for follow-up analysis.

3.5 Summary

In summary, we proposed the structured graphical lasso as a generalization of graphical lasso for learning a graphical model with structure. We looked at the specific structure of hubs with applications in biological networks. We proposed an ADMM algorithm for the resulting convex formulation. Simulation results show that our approach performs better than competitor algorithms in learning graphs with a few hubs. We also proposed simple screening rules to help tune for regularization parameters more easily.

3.6 Proofs

Appendix A: Derivation of Algorithm 5

Recall that the scaled augmented Lagrangian for (3.8) takes the form

$$\begin{aligned} L(\mathbf{B}, \tilde{\mathbf{B}}, \mathbf{W}) = \ell(\mathbf{X}, \boldsymbol{\Theta}) + \lambda_1 \|\mathbf{Z} - \mathrm{Diag}(\mathbf{Z})\|_1 + \lambda_2 \|\mathbf{V} - \mathrm{Diag}(\mathbf{V})\|_1 \\ + \lambda_3 \sum_{j=1}^{p} \|(\mathbf{V} - \mathrm{Diag}(\mathbf{V}))_j\|_2 + g(\tilde{\mathbf{B}}) + \frac{\rho}{2} \|\mathbf{B} - \tilde{\mathbf{B}} + \mathbf{W}\|_F^2. \end{aligned} \tag{3.10}$$

The proposed ADMM algorithm requires the following updates:

1. $\mathbf{B}^{(t+1)} \leftarrow \underset{\mathbf{B}}{\mathrm{argmin}}\ L(\mathbf{B}, \tilde{\mathbf{B}}^t, \mathbf{W}^t)$,
2. $\tilde{\mathbf{B}}^{(t+1)} \leftarrow \underset{\tilde{\mathbf{B}}}{\mathrm{argmin}}\ L(\mathbf{B}^{(t+1)}, \tilde{\mathbf{B}}, \mathbf{W}^t)$,
3. $\mathbf{W}^{(t+1)} \leftarrow \mathbf{W}^t + \mathbf{B}^{(t+1)} - \tilde{\mathbf{B}}^{(t+1)}$.

We now proceed to derive the updates for $\mathbf{B}$ and $\tilde{\mathbf{B}}$.

Updates for $\mathbf{B}$

To obtain updates for $\mathbf{B} = (\mathbf{\Theta}, \mathbf{V}, \mathbf{Z})$, we exploit the fact that (3.10) is separable in $\mathbf{\Theta}, \mathbf{V}$, and $\mathbf{Z}$. Therefore, we can simply update with respect to $\mathbf{\Theta}, \mathbf{V}$, and $\mathbf{Z}$ one-at-a-time. Update for $\mathbf{\Theta}$ depends on the form of the convex loss function, and is addressed in the main text. Updates for $\mathbf{V}$ and $\mathbf{Z}$ can be easily seen to take the form given in Algorithm 1.

Updates for $\tilde{\mathbf{B}}$

Minimizing the function in (3.10) with respect to $\tilde{\mathbf{B}}$ is equivalent to

$$\begin{aligned} \underset{\tilde{\mathbf{\Theta}}, \tilde{\mathbf{V}}, \tilde{\mathbf{Z}}}{\text{minimize}} \quad & \left\{ \frac{\rho}{2} \|\mathbf{\Theta} - \tilde{\mathbf{\Theta}} + \mathbf{W}_1\|_F^2 + \frac{\rho}{2} \|\mathbf{V} - \tilde{\mathbf{V}} + \mathbf{W}_2\|_F^2 + \frac{\rho}{2} \|\mathbf{Z} - \tilde{\mathbf{Z}} + \mathbf{W}_3\|_F^2 \right\} \\ \text{subject to} \quad & \tilde{\mathbf{\Theta}} = \tilde{\mathbf{Z}} + \tilde{\mathbf{V}} + \tilde{\mathbf{V}}^T. \end{aligned} \tag{3.11}$$

Let $\mathbf{\Gamma}$ be the $p \times p$ Lagrange multiplier matrix for the equality constraint. Then, the Lagrangian for (3.11) is

$$\frac{\rho}{2} \|\mathbf{\Theta} - \tilde{\mathbf{\Theta}} + \mathbf{W}_1\|_F^2 + \frac{\rho}{2} \|\mathbf{V} - \tilde{\mathbf{V}} + \mathbf{W}_2\|_F^2 + \frac{\rho}{2} \|\mathbf{Z} - \tilde{\mathbf{Z}} + \mathbf{W}_3\|_F^2 + \langle \mathbf{\Gamma}, \tilde{\mathbf{\Theta}} - \tilde{\mathbf{Z}} - \tilde{\mathbf{V}} - \tilde{\mathbf{V}}^T \rangle.$$

A little bit of algebra yields

$$\tilde{\mathbf{\Theta}} = \mathbf{\Theta} + \mathbf{W}_1 - \frac{1}{\rho} \mathbf{\Gamma},$$

$$\tilde{\mathbf{V}} = \frac{1}{\rho} (\mathbf{\Gamma} + \mathbf{\Gamma}^T) + \mathbf{V} + \mathbf{W}_2,$$

$$\tilde{\mathbf{Z}} = \frac{1}{\rho} \mathbf{\Gamma} + \mathbf{Z} + \mathbf{W}_3,$$

where $\mathbf{\Gamma} = \frac{\rho}{6} [(\mathbf{\Theta} + \mathbf{W}_1) - (\mathbf{V} + \mathbf{W}_2) - (\mathbf{V} + \mathbf{W}_2)^T - (\mathbf{Z} + \mathbf{W}_3)]$.

Appendix C: Some Properties of HGL

Proof of Lemma 20

Let $(\hat{\boldsymbol{\Theta}}, \hat{\boldsymbol{Z}}, \hat{\boldsymbol{V}})$ be the solution to (3.6) and suppose that $\hat{\boldsymbol{Z}}$ is not a diagonal matrix. Note that $\hat{\boldsymbol{Z}}$ is symmetric since $\boldsymbol{\Theta} \in \mathcal{S} \equiv \{\boldsymbol{\Theta} : \boldsymbol{\Theta} \succ 0 \text{ and } \boldsymbol{\Theta} = \boldsymbol{\Theta}^T\}$. Let $\hat{\boldsymbol{Z}} = \text{Diag}(\hat{\boldsymbol{Z}})$, a matrix that contains the diagonal elements of the matrix $\hat{\boldsymbol{Z}}$. Also, construct $\hat{\boldsymbol{V}}$ as follows,

$$\hat{\boldsymbol{V}}_{ij} = \begin{cases} \hat{\boldsymbol{V}}_{ij} + \frac{\hat{\boldsymbol{Z}}_{ij}}{2} & \text{if } i \neq j \\ \hat{\boldsymbol{V}}_{jj} & \text{otherwise.} \end{cases}$$

Then, we have that $\hat{\boldsymbol{\Theta}} = \hat{\boldsymbol{Z}} + \hat{\boldsymbol{V}} + \hat{\boldsymbol{V}}^T$. Thus, $(\hat{\boldsymbol{\Theta}}, \hat{\boldsymbol{Z}}, \hat{\boldsymbol{V}})$ is a feasible solution to (3.6). We now show that $(\hat{\boldsymbol{\Theta}}, \hat{\boldsymbol{Z}}, \hat{\boldsymbol{V}})$ has a smaller objective than $(\hat{\boldsymbol{\Theta}}, \hat{\boldsymbol{Z}}, \hat{\boldsymbol{V}})$ in (3.6), giving us a contradiction. Note that

$$\begin{aligned} \lambda_1\|\hat{\boldsymbol{Z}} - \text{Diag}(\hat{\boldsymbol{Z}})\|_1 + \lambda_2\|\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}})\|_1 &= \lambda_2\|\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}})\|_1 \\ &= \lambda_2 \textstyle\sum_{i\neq j} |\hat{\boldsymbol{V}}_{ij} + \frac{\hat{\boldsymbol{Z}}_{ij}}{2}| \\ &\leq \lambda_2\|\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}})\|_1 + \\ &\quad \tfrac{\lambda_2}{2}\|\hat{\boldsymbol{Z}} - \text{Diag}(\hat{\boldsymbol{Z}})\|_1, \end{aligned}$$

and

$$\begin{aligned} \lambda_3 \textstyle\sum_{j=1}^p \|(\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}}))_{\cdot j}\|_q &\leq \lambda_3 \textstyle\sum_{j=1}^p \|(\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}}))_{\cdot j}\|_q + \\ &\quad \tfrac{\lambda_3}{2} \textstyle\sum_{j=1}^p \|(\hat{\boldsymbol{Z}} - \text{Diag}(\hat{\boldsymbol{Z}}))_{\cdot j}\|_q \\ &\leq \lambda_3 \textstyle\sum_{j=1}^p \|(\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}}))_{\cdot j}\|_q + \tfrac{\lambda_3}{2}\|\hat{\boldsymbol{Z}} - \text{Diag}(\hat{\boldsymbol{Z}})\|_1, \end{aligned}$$

where the last inequality follows from the fact that for any vector $\mathbf{x} \in \mathbb{R}^p$ and $q \geq 1$, $\|\mathbf{x}\|_q$ is a nonincreasing function of q [Gentle, 2007].

Summing up the above inequalities, we get that

$$\begin{aligned} \lambda_1\|\hat{\boldsymbol{Z}} - \text{Diag}(\hat{\boldsymbol{Z}})\|_1 + \lambda_2\|\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}})\|_1 + \lambda_3 \textstyle\sum_{j=1}^p \|(\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}}))_{\cdot j}\|_q &\leq \\ \tfrac{\lambda_2+\lambda_3}{2}\|\hat{\boldsymbol{Z}} - \text{Diag}(\hat{\boldsymbol{Z}})\|_1 + \lambda_2\|\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}})\|_1 + \lambda_3 \textstyle\sum_{j=1}^p \|(\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}}))_{\cdot j}\|_q &< \\ \lambda_1\|\hat{\boldsymbol{Z}} - \text{Diag}(\hat{\boldsymbol{Z}})\|_1 + \lambda_2\|\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}})\|_1 + \lambda_3 \textstyle\sum_{j=1}^p \|(\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}}))_{\cdot j}\|_q, & \end{aligned}$$

where the last inequality uses the assumption that $\lambda_1 > \frac{\lambda_2+\lambda_3}{2}$. We arrive at a

contradiction and therefore the result holds.

Proof of Lemma 23

In this proof, we consider the case when $\lambda_1 > \frac{\lambda_2+\lambda_3}{2}$. A similar proof technique can be used to prove the case when $\lambda_1 < \frac{\lambda_2+\lambda_3}{2}$.

Let $f(\mathbf{\Theta}, \mathbf{V}, \mathbf{Z})$ denote the objective of (3.6) with $q = 1$, and $(\hat{\mathbf{\Theta}}, \hat{\boldsymbol{V}}, \hat{\boldsymbol{Z}})$ the optimal solution. By Lemma 20, the assumption that $\lambda_1 > \frac{\lambda_2+\lambda_3}{2}$ implies that $\hat{\boldsymbol{Z}}$ is a diagonal matrix. Now let $\hat{\mathbf{V}} = \frac{1}{2}\left(\hat{\boldsymbol{V}} + (\hat{\boldsymbol{V}})^T\right)$. Then

$$\begin{aligned}
f(\hat{\mathbf{\Theta}}, \hat{\boldsymbol{V}}, \hat{\boldsymbol{Z}}) &= -\log\det\hat{\mathbf{\Theta}} + \langle\hat{\mathbf{\Theta}}, \mathbf{S}\rangle + \lambda_1\|\hat{\boldsymbol{Z}} - \mathrm{Diag}(\hat{\boldsymbol{Z}})\|_1 + \\
&\quad (\lambda_2+\lambda_3)\|\hat{\boldsymbol{V}} - \mathrm{Diag}(\hat{\boldsymbol{V}})\|_1 \\
&= -\log\det\hat{\mathbf{\Theta}} + \langle\hat{\mathbf{\Theta}}, \mathbf{S}\rangle + \frac{\lambda_2+\lambda_3}{2}\|\hat{\boldsymbol{V}} + \hat{\boldsymbol{V}}^T - \mathrm{Diag}(\hat{\boldsymbol{V}} + \hat{\boldsymbol{V}}^T)\|_1 \\
&\leq -\log\det\hat{\mathbf{\Theta}} + \langle\hat{\mathbf{\Theta}}, \mathbf{S}\rangle + (\lambda_2+\lambda_3)\|\hat{\boldsymbol{V}} - \mathrm{Diag}(\hat{\boldsymbol{V}})\|_1 \\
&= f(\hat{\mathbf{\Theta}}, \hat{\boldsymbol{V}}, \hat{\boldsymbol{Z}}) \\
&\leq f(\hat{\mathbf{\Theta}}, \hat{\boldsymbol{V}}, \hat{\boldsymbol{Z}}),
\end{aligned}$$

where the last inequality follows from the assumption that $(\hat{\mathbf{\Theta}}, \hat{\boldsymbol{V}}, \hat{\boldsymbol{Z}})$ solves (3.6). By strict convexity of f, this means that $\hat{\boldsymbol{V}} = \hat{\mathbf{V}}$, i.e., $\hat{\boldsymbol{V}}$ is symmetric. This implies that

$$\begin{aligned}
f(\hat{\mathbf{\Theta}}, \hat{\boldsymbol{V}}, \hat{\boldsymbol{Z}}) &= -\log\det\hat{\mathbf{\Theta}} + \langle\hat{\mathbf{\Theta}}, \mathbf{S}\rangle + \frac{\lambda_2+\lambda_3}{2}\|\hat{\boldsymbol{V}} + \hat{\boldsymbol{V}}^T - \mathrm{Diag}(\hat{\boldsymbol{V}} + \hat{\boldsymbol{V}}^T)\|_1 \\
&= -\log\det\hat{\mathbf{\Theta}} + \langle\hat{\mathbf{\Theta}}, \mathbf{S}\rangle + \frac{\lambda_2+\lambda_3}{2}\|\hat{\mathbf{\Theta}} - \mathrm{Diag}(\hat{\mathbf{\Theta}})\|_1 \qquad (3.12) \\
&= g(\mathbf{\Theta}^*),
\end{aligned}$$

where $g(\mathbf{\Theta})$ is the objective of the graphical lasso optimization problem, evaluated at $\mathbf{\Theta}$, with tuning parameter $\frac{\lambda_2+\lambda_3}{2}$. Suppose that $\tilde{\mathbf{\Theta}}$ minimizes $g(\mathbf{\Theta})$, and $\mathbf{\Theta}^* \neq \tilde{\mathbf{\Theta}}$. Then, by (3.12) and strict convexity of g, $g(\hat{\mathbf{\Theta}}) = f(\hat{\mathbf{\Theta}}, \hat{\boldsymbol{V}}, \hat{\boldsymbol{Z}}) \leq f(\tilde{\mathbf{\Theta}}, \tilde{\mathbf{\Theta}}/2, \mathbf{0}) = g(\tilde{\mathbf{\Theta}}) < g(\hat{\mathbf{\Theta}})$, giving us a contradiction. Thus it must be that $\tilde{\mathbf{\Theta}} = \hat{\mathbf{\Theta}}$.

Proof of Lemma 21

Let $(\hat{\boldsymbol{\Theta}}, \hat{\boldsymbol{Z}}, \hat{\boldsymbol{V}})$ be the solution to (3.6) and suppose $\hat{\boldsymbol{V}}$ is not a diagonal matrix. Let $\hat{\boldsymbol{V}} = \text{Diag}(\hat{\boldsymbol{V}})$, a diagonal matrix that contains the diagonal elements of $\hat{\boldsymbol{V}}$. Also construct $\hat{\boldsymbol{Z}}$ as follows,

$$\hat{\boldsymbol{Z}}_{ij} = \begin{cases} \hat{\boldsymbol{Z}}_{ij} + \hat{\boldsymbol{V}}_{ij} + \hat{\boldsymbol{V}}_{ji} & \text{if } i \neq j \\ \hat{\boldsymbol{Z}}_{ij} & \text{otherwise.} \end{cases}$$

Then, we have that $\hat{\boldsymbol{\Theta}} = \hat{\boldsymbol{V}} + \hat{\boldsymbol{V}}^T + \hat{\boldsymbol{Z}}$. We now show that $(\hat{\boldsymbol{\Theta}}, \hat{\boldsymbol{Z}}, \hat{\boldsymbol{V}})$ has a smaller objective value than $(\hat{\boldsymbol{\Theta}}, \hat{\boldsymbol{Z}}, \hat{\boldsymbol{V}})$ in (3.6), giving us a contradiction. We start by noting that

$$\begin{aligned} \lambda_1 \|\hat{\boldsymbol{Z}} - \text{Diag}(\hat{\boldsymbol{Z}})\|_1 + \lambda_2 \|\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}})\|_1 &= \lambda_1 \|\hat{\boldsymbol{Z}} - \text{Diag}(\hat{\boldsymbol{Z}})\|_1 \\ &\leq \lambda_1 \|\hat{\boldsymbol{Z}} - \text{Diag}(\hat{\boldsymbol{Z}})\|_1 + \\ &\quad 2\lambda_1 \|\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}})\|_1. \end{aligned}$$

By Holder's Inequality, we know that $\mathbf{x}^T\mathbf{y} \leq \|\mathbf{x}\|_q \|\mathbf{y}\|_s$ where $\frac{1}{s} + \frac{1}{q} = 1$ and $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{p-1}$. Setting $\mathbf{y} = \text{sign}(\mathbf{x})$, we have that $\|\mathbf{x}\|_1 \leq (p-1)^{\frac{1}{s}} \|\mathbf{x}\|_q$. Consequently,

$$\frac{\lambda_3}{(p-1)^{\frac{1}{s}}} \|\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}})\|_1 \leq \lambda_3 \sum_{j=1}^{p} \|(\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}}))_{\cdot j}\|_q.$$

Combining these results, we have that

$$\begin{aligned} &\lambda_1 \|\hat{\boldsymbol{Z}} - \text{Diag}(\hat{\boldsymbol{Z}})\|_1 + \lambda_2 \|\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}})\|_1 + \lambda_3 \sum_{j=1}^{p} \|(\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}}))_{\cdot j}\|_q \\ &\leq \lambda_1 \|\hat{\boldsymbol{Z}} - \text{Diag}(\hat{\boldsymbol{Z}})\|_1 + 2\lambda_1 \|\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}})\|_1 \\ &< \lambda_1 \|\hat{\boldsymbol{Z}} - \text{Diag}(\hat{\boldsymbol{Z}})\|_1 + \left(\lambda_2 + \frac{\lambda_3}{(p-1)^{\frac{1}{s}}}\right) \|\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}})\|_1 \\ &\leq \lambda_1 \|\hat{\boldsymbol{Z}} - \text{Diag}(\hat{\boldsymbol{Z}})\|_1 + \lambda_2 \|\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}})\|_1 + \lambda_3 \sum_{j=1}^{p} \|(\hat{\boldsymbol{V}} - \text{Diag}(\hat{\boldsymbol{V}}))_{\cdot j}\|_q, \end{aligned}$$

where we use the assumption that $\lambda_1 < \frac{\lambda_2}{2} + \frac{\lambda_3}{2(p-1)^{\frac{1}{s}}}$. This leads to a contradiction.

3.7 Acknowledgements

The contributions in this chapter are based on joint work with Maryam Fazel, Daniela Witten, Su-In Lee, Kean-Ming Tan and Palma London.

Chapter 4

LEARNING THE STRUCTURE OF MULTIPLE GGMS

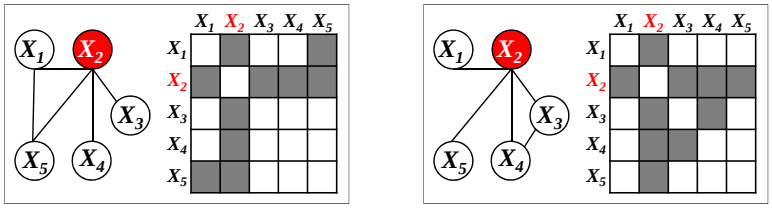
4.1 Introduction and motivation

Graphical models are often used to model biological networks such as gene regulatory networks. Note that in the graphical model, the nodes represent the genes and the edges represent the conditional dependencies between the genes. In the problem of detecting changes in gene-regulatory networks [Mohan et al., 2013, Danaher et al., 2013], it is often of interest to capture differences or similarities between two or more graphical models, that represent different conditions of the *gene-regulatory networks*. For example, the first graphical model could capture the correlations between genes in one cancer sub-type while the second could capture the correlations between genes in another cancer sub-type. Given that the two two networks correspond to a cancer condition, we expect the correlations to be similar. Yet, since the networks correspond to different sub-types, there would be distinct differences in correlations between the two networks. We note that although the two networks can be estimated separately, in the high dimensional setting (with more variables than samples), any approach that estimates the two graphical models must share information in the estimation process through the *similarity* and *differences* between the networks.

Quite a few papers in the recent literature have looked at the problem of estimating multiple graphical models that share information. [Guo et al., 2010] presented a non-convex formulation to estimate multiple graphical models under the Gaussianity assumption. [Varoquaux et al., 2010, Danaher et al., 2013, Honorio and Samaras, 2010] presented group-lasso based convex formulations to estimating multiple graphical models. However, the formulations presented only promote joint sparsity in the networks and do not account for any prior knowledge that could be captured through *structured sparsity*. [Hara and Washio, 2013] present an *additive sparsity* formulation that captures both a common substructure (similarity) and difference between networks. However, even here, the use of prior knowledge is limited to sparsity. [Zhang and Wang, 2010] follow the approach of [Meinshausen and Bülhmann, 2006] of solv-

ing multiple lassos to estimate the graphical models. They use a fused lasso penalty. A similar approach but with log-likelihood loss appeared as the FGL formulation in [Danaher et al., 2013].

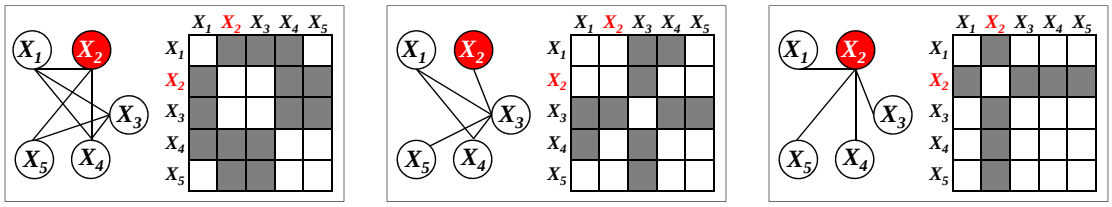
We go beyond the plain vanilla sparsity and present formulations that incorporate prior knowledge through *structured sparsity*. One example of the use of structured sparsity is in estimating nodes that have their correlations changed completely between two networks. This could happen in a gene-regulatory network when genes mutate between healthy and diseased conditions. Such a node would show up as a hub node (or a node with high degree) in the difference (of the adjacency matrices) of the two networks. We call such nodes to be *perturbed* between the two networks (Figure 4.2). We note that just the sparsity regularizer (ℓ_1 norm) is not designed to detect such structured perturbations. We instead present a suitable extension of the overlap norm proposed by Obozinksi et al [Obozinski et al.], that we refer to as the *row column overlap norm penalty* (RCON). The RCON penalty can be used to detect structured perturbations between two networks, as we discuss in the next section. We are also interested in detecting similarities between networks. In the example of estimating two gene-regulatory networks corresponding to two different sub-types of cancer, many regulatory genes are shared between the two networks, thus having similar correlations. Such nodes are referred to as *common hub nodes* or *cohub nodes* (Figure 4.1).



(a) (b)

Figure 4.1: Two networks share a *common hub* (co-hub) node. X_2 serves as a hub node in both networks. *(a):* Network 1 and its adjacency matrix. *(b):* Network 2 and its adjacency matrix.

We present convex formulations in the next section to estimate networks that share information through the medium of structured sparsity.



(a) (b) (c)

Figure 4.2: Two networks that differ due to *node perturbation* of X_2. *(a):* Network 1 and its adjacency matrix. *(b):* Network 2 and its adjacency matrix. *(c): Left:* Edges that differ between the two networks. *Right:* Shaded cells indicate edges that differ between Networks 1 and 2.

4.2 Learning multiple GGMs - PNJGL and CNJGL

In this section we discuss convex formulations to learn hub structures in Gaussian graphical models. To do this, we first introduce a RCON penalty for multiple networks as follows.

Definition 25. *The row-column overlap norm (RCON) induced by a matrix norm* $\|.\|$ *is defined as*

$$\begin{aligned} \Omega(\boldsymbol{\Theta}^1, \boldsymbol{\Theta}^2, \ldots, \boldsymbol{\Theta}^K) \quad = \quad & \min_{\boldsymbol{V}^1, \boldsymbol{V}^2, \ldots, \boldsymbol{V}^K} \left\| \begin{bmatrix} \boldsymbol{V}^1 \\ \boldsymbol{V}^2 \\ \vdots \\ \boldsymbol{V}^K \end{bmatrix} \right\| \\ & \text{subject to} \quad \boldsymbol{\Theta}^k = \boldsymbol{V}^k + (\boldsymbol{V}^k)^T \ \text{ for } k = 1, \ldots, K. \end{aligned} \tag{4.1}$$

We note that the RCON penalty in (4.1) is related to the RCON penalty introduced in the context of learning hub graphical models in the previous chapter. While the previous RCON penalty (3.5) is only applicable to learning a single graphical model, the RCON penalty in (4.1) applies to learning multiple graphical models and in that sense, can be considered a generalization of (3.5). However, there is a small difference in the two penalties, as (3.5) also allows for sparsity penalization through the ℓ_1 norm.

It is easy to check that Ω is indeed a norm for all matrix norms $\|.\|$. Also, when

$\|.\|$ is symmetric in its argument, i.e., $\|V\| = \|V^T\|$, then

$$\Omega(\mathbf{\Theta}^1, \mathbf{\Theta}^2, \ldots, \mathbf{\Theta}^K) = \frac{1}{2}\left\|\begin{bmatrix} \mathbf{\Theta}^1 \\ \mathbf{\Theta}^2 \\ \vdots \\ \mathbf{\Theta}^K \end{bmatrix}\right\|. \tag{4.2}$$

Denote by Ω_q the RCON penalty where the matrix norm, $\|\cdot\|$ is given by $\|Z\| = \sum_{i=1}^{p} \|Z_i\|_q$ or the ℓ_1/ℓ_q norm. Thus if $\|\cdot\|$ is an ℓ_1/ℓ_1 norm, then $\Omega_1(\mathbf{\Theta}^1, \mathbf{\Theta}^2, \ldots, \mathbf{\Theta}^K) = \frac{1}{2}\sum_{k=1}^{K}\sum_{i,j}|\Theta_{ij}^k|$.

Consider the task of jointly estimating K inverse covariance matrices by solving

$$\underset{\mathbf{\Theta}^1, \mathbf{\Theta}^2, \ldots, \mathbf{\Theta}^K \in \mathbb{S}_{++}^p}{\text{maximize}} \left\{ L(\mathbf{\Theta}^1, \mathbf{\Theta}^2, \ldots, \mathbf{\Theta}^K) - \lambda_1 \sum_{k=1}^{K} \|\mathbf{\Theta}^k\|_1 - \lambda_2 \sum_{k<k'} \Omega_q(\mathbf{\Theta}^k - \mathbf{\Theta}^{k'}) \right\}. \tag{4.3}$$

We refer to the convex optimization problem (4.3) as the *perturbed-node joint graphical lasso* (PNJGL). Let $\hat{\mathbf{\Theta}}^1, \hat{\mathbf{\Theta}}^2, \ldots, \hat{\mathbf{\Theta}}^K$ denote the solution to (4.3); these serve as estimates for $(\mathbf{\Sigma}^1)^{-1}, (\mathbf{\Sigma}^2)^{-1}, \ldots, (\mathbf{\Sigma}^K)^{-1}$. In (4.3), λ_1 and λ_2 are nonnegative tuning parameters, and $q \geq 1$. When $\lambda_2 = 0$, (4.3) amounts simply to applying GL to each condition separately in order to separately estimate K networks. When $\lambda_2 > 0$, there is a sharing of information between the different networks. Specifically, (4.3) estimates networks that differ pairwise by a few hub nodes.

We now consider jointly estimating K precision matrices by solving the convex optimization problem

$$\begin{aligned} \underset{\mathbf{\Theta}^1, \mathbf{\Theta}^2, \ldots, \mathbf{\Theta}^K \in \mathbb{S}_{++}^p}{\text{maximize}} \quad & L(\mathbf{\Theta}^1, \mathbf{\Theta}^2, \ldots, \mathbf{\Theta}^K) - \lambda_1 \sum_{k=1}^{K} \|\mathbf{\Theta}^k\|_1 - \\ & \lambda_2 \Omega_q(\mathbf{\Theta}^1 - \text{diag}(\mathbf{\Theta}^1), \ldots, \mathbf{\Theta}^K - \text{diag}(\mathbf{\Theta}^K)). \end{aligned} \tag{4.4}$$

We refer to (4.4) as the *co-hub node joint graphical lasso* (CNJGL) formulation. In (4.4), λ_1 and λ_2 are nonnegative tuning parameters, and $q \geq 1$. When $\lambda_2 = 0$ then this amounts to a graphical lasso optimization problem applied to each network separately; however, when $\lambda_2 > 0$, a shared structure is encouraged among the K networks. In particular, (4.4) encourages network estimates that have a common set

of hub nodes — that is, it encourages the supports of $\Theta^1, \Theta^2, \ldots, \Theta^K$ to be the same, and the union of a set of rows and columns.

4.3 *ADMM algorithms for PNJGL and CNJGL*

We now present ADMM algorithms for PNJGL and CNJGL.

Here we consider solving PNJGL with $K = 2$; the extension for $K > 2$ is slightly more complicated. To begin, we note that (4.3) can be rewritten as

$$\begin{array}{ll} \underset{\Theta^1, \Theta^2 \in \mathbb{S}^p_{++}, V \in \mathbb{R}^{p \times p}}{\text{maximize}} & \left\{ L(\Theta^1, \Theta^2) - \lambda_1 \|\Theta^1\|_1 - \lambda_1 \|\Theta^2\|_1 - \lambda_2 \sum_{j=1}^{p} \|V_j\|_q \right\} \\ \text{subject to} & \Theta^1 - \Theta^2 = V + V^T. \end{array} \tag{4.5}$$

We now reformulate (4.5) by introducing new variables, so as to decouple some of the terms in the objective function that are difficult to optimize jointly:

$$\begin{array}{ll} \underset{\Theta^1 \in \mathcal{S}^p_{++}, \Theta^2 \in \mathcal{S}^p_{++}, Z^1, Z^2, V, W}{\text{minimize}} & \left\{ -L(\Theta^1, \Theta^2) + \lambda_1 \|Z^1\|_1 + \lambda_1 \|Z^2\|_1 + \lambda_2 \sum_{j=1}^{p} \|V_j\|_q \right\} \\ \text{subject to} & \Theta^1 - \Theta^2 = V + W, V = W^T, \Theta^1 = Z^1, \Theta^2 = Z^2. \end{array} \tag{4.6}$$

The augmented Lagrangian to (4.6) is given by

$$\begin{array}{rl} - & L(\Theta^1, \Theta^2) + \lambda_1 \|Z^1\|_1 + \lambda_1 \|Z^2\|_1 + \lambda_2 \sum_{j=1}^{p} \|V_j\|_q \\ + & \langle F, \Theta^1 - \Theta^2 - (V + W) \rangle + \langle G, V - W^T \rangle + \langle Q^1, \Theta^1 - Z^1 \rangle \\ + & \langle Q^2, \Theta^2 - Z^2 \rangle + \frac{\rho}{2} \|\Theta^1 - \Theta^2 - (V + W)\|_F^2 \\ + & \frac{\rho}{2} \|V - W^T\|_F^2 + \frac{\rho}{2} \|\Theta^1 - Z^1\|_F^2 + \frac{\rho}{2} \|\Theta^2 - Z^2\|_F^2. \end{array} \tag{4.7}$$

In (4.7) there are six primal variables and four dual variables. Based on this augmented Lagrangian, the complete ADMM algorithm for (4.3) is given in Algorithm 6, in which the operator Expand is given by

$$\begin{array}{rcl} \text{Expand}(A, \rho, n_k) & = & \underset{\Theta \in \mathbb{S}^p_{++}}{\text{argmin}} \left\{ -n_k \log\det(\Theta) + \rho \|\Theta - A\|_F^2 \right\} \\ & = & \frac{1}{2} U \left(D + \sqrt{D^2 + \frac{2n_k}{\rho} I} \right) U^T, \end{array}$$

where $\boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T$ is the eigenvalue decomposition of a symmetric matrix $\boldsymbol{A}$, and as mentioned earlier, n_k is the number of observations in the kth class. The operator $\mathcal{T}_q$ is given by

$$\mathcal{T}_q(\boldsymbol{A}, \lambda) = \underset{\boldsymbol{X}}{\operatorname{argmin}} \left\{ \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{A}\|_F^2 + \lambda \sum_{j=1}^{p} \|\boldsymbol{X}_j\|_q \right\},$$

and is also known as the proximal operator corresponding to the ℓ_1/ℓ_q norm.

Algorithm 6: ADMM algorithm for the PNJGL optimization problem (4.3)

input: $\rho > 0, \mu > 1, t_{\max} > 0$;

Initialize: Primal variables to the identity matrix and dual variables to the zero matrix;

for t = *1:*$t_{\max}$ **do**

  $\rho \leftarrow \mu\rho$;

  while *Not converged* **do**

    $\boldsymbol{\Theta}^1 \leftarrow \text{Expand}\left(\frac{1}{2}(\boldsymbol{\Theta}^2 + \boldsymbol{V} + \boldsymbol{W} + \boldsymbol{Z}^1) - \frac{1}{2\rho}(\boldsymbol{Q}^1 + n_1 S^1 + \boldsymbol{F}), \rho, n_1\right)$;

    $\boldsymbol{\Theta}^2 \leftarrow \text{Expand}\left(\frac{1}{2}(\boldsymbol{\Theta}^1 - (\boldsymbol{V} + \boldsymbol{W}) + \boldsymbol{Z}^2) - \frac{1}{2\rho}(\boldsymbol{Q}^2 + n_2 S^2 - \boldsymbol{F}), \rho, n_2\right)$;

    $\boldsymbol{Z}^i \leftarrow \mathcal{T}_1\left(\boldsymbol{\Theta}^i + \frac{\boldsymbol{Q}^i}{\rho}, \frac{\lambda_1}{\rho}\right)$ for $i = 1, 2$;

    $\boldsymbol{V} \leftarrow \mathcal{T}_q\left(\frac{1}{2}(\boldsymbol{W}^T - \boldsymbol{W} + (\boldsymbol{\Theta}^1 - \boldsymbol{\Theta}^2)) + \frac{1}{2\rho}(\boldsymbol{F} - \boldsymbol{G}), \frac{\lambda_2}{2\rho}\right)$;

    $\boldsymbol{W} \leftarrow \frac{1}{2}(\boldsymbol{V}^T - \boldsymbol{V} + (\boldsymbol{\Theta}^1 - \boldsymbol{\Theta}^2)) + \frac{1}{2\rho}(\boldsymbol{F} + \boldsymbol{G}^T)$;

    $\boldsymbol{F} \leftarrow \boldsymbol{F} + \rho(\boldsymbol{\Theta}^1 - \boldsymbol{\Theta}^2 - (\boldsymbol{V} + \boldsymbol{W}))$;

    $\boldsymbol{G} \leftarrow \boldsymbol{G} + \rho(\boldsymbol{V} - \boldsymbol{W}^T)$;

    $\boldsymbol{Q}^i \leftarrow \boldsymbol{Q}^i + \rho(\boldsymbol{\Theta}^i - \boldsymbol{Z}^i)$ for $i = 1, 2$

The CNJGL formulation in (4.4) is equivalent to

$$\begin{aligned} \underset{\boldsymbol{\Theta}^i \in \mathbb{S}^p_{++}, \mathbf{V}^i \in \mathbb{R}^{p\times p}\ \forall i}{\text{minimize}} \quad & -L(\boldsymbol{\Theta}^1, \boldsymbol{\Theta}^2, \ldots, \boldsymbol{\Theta}^K) + \lambda_1 \sum_{i=1}^{K} \|\boldsymbol{\Theta}^i\|_1 + \lambda_2 \sum_{j=1}^{p} \left\| \begin{bmatrix} V^1 \\ V^2 \\ \vdots \\ V^K \end{bmatrix}_j \right\|_q \\ \text{subject to} \quad & \boldsymbol{\Theta}^i - \text{diag}(\boldsymbol{\Theta}^i) = V^i + (V^i)^T \text{ for } i = 1, \ldots, K. \end{aligned} \tag{4.8}$$

One can easily see that the problem (4.8) is equivalent to the problem

$$\begin{array}{ll} \underset{\boldsymbol{\Theta}^i \in \mathbb{S}^p_{++}, \tilde{\mathbf{V}}^i \in \mathbb{R}^{p\times p}\ \forall i}{\text{minimize}} & -L(\boldsymbol{\Theta}^1, \boldsymbol{\Theta}^2, \ldots, \boldsymbol{\Theta}^K) + \lambda_1 \sum_{i=1}^K \|\boldsymbol{\Theta}^i\|_1 + \\ & \lambda_2 \sum_{j=1}^p \left\| \begin{bmatrix} \tilde{V}^1 - \mathrm{diag}(\tilde{V}^1) \\ \tilde{V}^2 - \mathrm{diag}(\tilde{V}^2) \\ \vdots \\ \tilde{V}^K - \mathrm{diag}(\tilde{V}^K) \end{bmatrix}_j \right\|_q \\ \text{subject to} & \boldsymbol{\Theta}^i = \tilde{V}^i + (\tilde{V}^i)^T \ \text{ for } i = 1, 2, \ldots, K, \end{array} \tag{4.9}$$

in the sense that the optimal solution $\{\boldsymbol{V}^i\}$ to (4.8) and the optimal solution $\{\tilde{V}^i\}$ to (4.9) have the following relationship: $\boldsymbol{V}^i = \tilde{V}^i - \mathrm{diag}(\tilde{V}^i)$ for $i = 1, 2, \ldots, K$. We now present an ADMM algorithm for solving (4.9). We reformulate (4.9) by introducing additional variables in order to decouple some terms of the objective that are difficult to optimize jointly:

$$\begin{array}{ll} \underset{\boldsymbol{\Theta}^i \in \mathbb{S}^p_{++}, \boldsymbol{Z}^i, \tilde{V}^i, \boldsymbol{W}^i \in \mathbb{R}^{p\times p}}{\text{minimize}} & \sum_{i=1}^K n_i(-\log\det(\boldsymbol{\Theta}^i) + \mathrm{trace}(\boldsymbol{S}^i\boldsymbol{\Theta}^i)) + \lambda_1 \sum_{i=1}^K \|\boldsymbol{Z}^i\|_1 + \\ & \lambda_2 \sum_{j=1}^p \left\| \begin{bmatrix} \tilde{V}^1 - \mathrm{diag}(\tilde{V}^1) \\ \tilde{V}^2 - \mathrm{diag}(\tilde{V}^2) \\ \vdots \\ \tilde{V}^K - \mathrm{diag}(\tilde{V}^K) \end{bmatrix}_j \right\|_q \\ \text{subject to} & \boldsymbol{\Theta}^i = \tilde{V}^i + \boldsymbol{W}^i, \tilde{V}^i = (\boldsymbol{W}^i)^T, \boldsymbol{\Theta}^i = \boldsymbol{Z}^i \text{ for } i = 1, 2, \ldots, K. \end{array} \tag{4.10}$$

The augmented Lagrangian to (4.10) is given by

$$\sum_{i=1}^K n_i(-\log\det(\boldsymbol{\Theta}^i) + \mathrm{trace}(\boldsymbol{S}^i\boldsymbol{\Theta}^i)) + \lambda_1 \sum_{i=1}^K \|\boldsymbol{Z}^i\|_1 + \lambda_2 \sum_{j=1}^p \left\| \begin{bmatrix} \tilde{V}^1 - \mathrm{diag}(\tilde{V}^1) \\ \tilde{V}^2 - \mathrm{diag}(\tilde{V}^2) \\ \vdots \\ \tilde{V}^K - \mathrm{diag}(\tilde{V}^K) \end{bmatrix}_j \right\|_q$$

$$+\sum_{i=1}^{K}\left\{\langle \boldsymbol{F}^i, \boldsymbol{\Theta}^i - (\tilde{V}^i + \boldsymbol{W}^i)\rangle + \langle \boldsymbol{G}^i, \tilde{V}^i - (\boldsymbol{W}^i)^T\rangle + \langle \boldsymbol{Q}^i, \boldsymbol{\Theta}^i - \boldsymbol{Z}^i\rangle\right\}$$

$$+\frac{\rho}{2}\sum_{i=1}^{K}\left\{\|\boldsymbol{\Theta}^i - (\tilde{V}^i + \boldsymbol{W}^i)\|_F^2 + \|\tilde{V}^i - (\boldsymbol{W}^i)^T\|_F^2 + \|\boldsymbol{\Theta}^i - \boldsymbol{Z}^i\|_F^2\right\}.$$

The corresponding ADMM algorithm is given in Algorithm 7.

Algorithm 7: ADMM algorithm for the CNJGL optimization problem (4.4)

input: $\rho > 0, \mu > 1, t_{\max} > 0$;

Initialize: Primal variables to the identity matrix and dual variables to the zero matrix;

for $t = 1{:}t_{\max}$ **do**

$\rho \leftarrow \mu\rho$;

while *Not converged* **do**

$\boldsymbol{\Theta}^i \leftarrow \text{Expand}\left(\frac{1}{2}(\tilde{V}^i + \boldsymbol{W}^i + \boldsymbol{Z}^i) - \frac{1}{2\rho}(\boldsymbol{Q}^i + n_i\boldsymbol{S}^i + \boldsymbol{F}^i), \rho, n_i\right)$ for $i = 1, \ldots, K$;

$\boldsymbol{Z}^i \leftarrow \mathcal{T}_1\left(\boldsymbol{\Theta}^i + \frac{\boldsymbol{Q}^i}{\rho}, \frac{\lambda_1}{\rho}\right)$ for $i = 1, \ldots, K$;

Let $\boldsymbol{C}^i = \frac{1}{2}((\boldsymbol{W}^i)^T - \boldsymbol{W}^i + \boldsymbol{\Theta}^i) + \frac{1}{2\rho}(\boldsymbol{F}^i - \boldsymbol{G}^i)$ for $i = 1, \ldots, K$;

$$\begin{bmatrix} \tilde{V}^1 \\ \tilde{V}^2 \\ \vdots \\ \tilde{V}^K \end{bmatrix} \leftarrow \mathcal{T}_q\left(\begin{bmatrix} \boldsymbol{C}^1 - \text{diag}(\boldsymbol{C}^1) \\ \boldsymbol{C}^2 - \text{diag}(\boldsymbol{C}^2) \\ \vdots \\ \boldsymbol{C}^K - \text{diag}(\boldsymbol{C}^K) \end{bmatrix}, \frac{\lambda_2}{2\rho}\right) + \begin{bmatrix} \text{diag}(\boldsymbol{C}^1) \\ \text{diag}(\boldsymbol{C}^2) \\ \vdots \\ \text{diag}(\boldsymbol{C}^K) \end{bmatrix};$$

$\boldsymbol{W}^i \leftarrow \frac{1}{2}((\tilde{V}^i)^T - \tilde{V}^i + \boldsymbol{\Theta}^i) + \frac{1}{2\rho}(\boldsymbol{F}^i + (\boldsymbol{G}^i)^T)$ for $i = 1, \ldots, K$;

$\boldsymbol{F}^i \leftarrow \boldsymbol{F}^i + \rho(\boldsymbol{\Theta}^i - (\tilde{V}^i + \boldsymbol{W}^i))$ for $i = 1, \ldots, K$;

$\boldsymbol{G}^i \leftarrow \boldsymbol{G}^i + \rho(\tilde{V}^i - (\boldsymbol{W}^i)^T)$ for $i = 1, \ldots, K$;

$\boldsymbol{Q}^i \leftarrow \boldsymbol{Q}^i + \rho(\boldsymbol{\Theta}^i - \boldsymbol{Z}^i)$ for $i = 1, \ldots, K$

4.4 Numerical results

We give synthetic data results in this section for learning two graphical models that share information (perturbed and cohub nodes). Real data results can be found in

[Mohan et al., 2013].

4.4.1 Synthetic data experiments

We present synthetic data results for PNJGL for learning two Erdos-Renyi networks that share cohub nodes and differ by perturbed hub nodes. More detailed synthetic data experiments can be found in [Mohan et al., 2013].

Data generation for Erdos-Renyi networks

We generated the data as follows, for $p = 100$, and $n \in \{25, 50, 100, 200\}$:

Step 1: To generate an Erdos-Renyi network, we created a $p \times p$ symmetric matrix A with elements

$$A_{ij} \sim_{\text{i.i.d.}} \begin{cases} 0 & \text{with probability } 0.98, \\ \text{Unif}([-0.6, -0.3] \cup [0.3, 0.6]) & \text{otherwise.} \end{cases}$$

Step 2: We duplicated A into two matrices, A^1 and A^2. We selected two nodes at random, and for each node, we set the elements of the corresponding row and column of either A^1 or A^2 (chosen at random) to be i.i.d. draws from a $\text{Unif}([-0.6, -0.3] \cup [0.3, 0.6])$ distribution. This results in two perturbed nodes.

Step 3: We randomly selected two nodes to serve as co-hub nodes, and set each element of the corresponding rows and columns in each network to be i.i.d. draws from a $\text{Unif}([-0.6, -0.3] \cup [0.3, 0.6])$ distribution. In other words, these co-hub nodes are *identical* across the two networks.

Step 4: In order to make the matrices positive definite, we let $c = \min(\lambda_{\min}(A^1), \lambda_{\min}(A^2))$, where $\lambda_{\min}(\cdot)$ indicates the smallest eigenvalue of the matrix. We then set $(\mathbf{\Sigma}^1)^{-1}$ equal to $A^1 + (0.1 + |c|)I$ and set $(\mathbf{\Sigma}^2)^{-1}$ equal to $A^2 + (0.1 + |c|)I$, where I is the $p \times p$ identity matrix.

Step 5: We generated n independent observations each from a $N(0, \mathbf{\Sigma}^1)$ and a $N(0, \mathbf{\Sigma}^2)$ distribution, and used them to compute the sample covariance matrices S^1 and S^2.

Results

We now define several metrics used to measure algorithm performance. We wish to quantify each algorithm's (1) recovery of the support of the true inverse covariance matrices, (2) successful detection of co-hub and perturbed nodes, and (3) error in estimation of $\mathbf{\Theta}^1 = (\mathbf{\Sigma}^1)^{-1}$ and $\mathbf{\Theta}^2 = (\mathbf{\Sigma}_2)^{-1}$. Details are given in Table 4.1.

We compared the performance of PNJGL to its edge-based counterpart FGL, as well as to graphical lasso (GL). We expect PNJGL to be able to detect perturbed nodes. (PNJGL would not be expected to detect cohub nodes, since they are identical across the networks.)

The simulation results are displayed in Figure 4.3. Each row corresponds to a sample size while each column corresponds to a performance metric. In Figure 4.3, PNJGL, FGL, and GL are compared. Within each plot, each colored line corresponds to the results obtained using a fixed value of λ_2 (for either PNJGL, FGL, or GGL), as λ_1 is varied. Recall that GL corresponds to any of these four approaches with $\lambda_2 = 0$. Note that the number of positive edges (defined in Table 4.1) decreases approximately monotonically with the regularization parameter λ_1, and so on the x-axis we plot the number of positive edges, rather than λ_1, for ease of interpretation.

In Figure 4.3, we observe that PNJGL outperforms FGL and GL for a suitable range of the regularization parameter λ_2, in the sense that for a fixed number of edges estimated, PNJGL identifies more true positives, correctly identifies a greater ratio of perturbed nodes, and yields a lower Frobenius error in the estimates of $\mathbf{\Theta}^1$ and $\mathbf{\Theta}^2$. In particular, PNJGL performs best relative to FGL and GL when the number of samples is the smallest, i.e. in the high-dimensional data setting. Unlike FGL, PNJGL fully exploits the fact that differences between $\mathbf{\Theta}^1$ and $\mathbf{\Theta}^2$ are due to node perturbation. Not surprisingly, GL performs worst among the three algorithms, since it does not borrow strength across the conditions in estimating $\mathbf{\Theta}^1$ and $\mathbf{\Theta}^2$.

4.5 Summary

In summary, we proposed two convex formulations for learning structured similarities (CNJGL) and differences (PNJGL) between graphical models. These formulations use priors that encourage sharing of information between the graphical models and have a significantly better empririical performance than learning two separate graphical

(1)	Positive edges: $\sum_{i<j}\left(\mathbf{1}\{|\hat{\mathbf{\Theta}}^1_{ij}|>t_0\}+\mathbf{1}\{|\hat{\mathbf{\Theta}}^2_{ij}|>t_0\}\right)$ True positive edges: $\sum_{i<j}\left(\mathbf{1}\{|\mathbf{\Theta}^1_{ij}|>t_0 \text{ and } |\hat{\mathbf{\Theta}}^1_{ij}|>t_0\}\right)+\left(\mathbf{1}\{|\mathbf{\Theta}^2_{ij}|>t_0 \text{ and } |\hat{\mathbf{\Theta}}^2_{ij}|>t_0\}\right)$
(2)	Positive perturbed columns (PPC): PNJGL: $\sum_{i=1}^p \mathbf{1}\left\{\|\hat{\boldsymbol{V}}^1_{-i,i}\|_2>t_s\right\}$; FGL/GL: $\sum_{i=1}^p \mathbf{1}\{\|(\hat{\mathbf{\Theta}}^1-\hat{\mathbf{\Theta}}^2)_{-i,i}\|_2>t_s\}$ True positive perturbed columns (TPPC): PNJGL: $\sum_{i\in I_p} \mathbf{1}\{\|\hat{\boldsymbol{V}}^1_{-i,i}\|_2>t_s\}$; FGL/GL: $\sum_{i\in I_p} \mathbf{1}\{\|(\hat{\mathbf{\Theta}}^1-\hat{\mathbf{\Theta}}^2)_{-i,i}\|_2>t_s\}$, where I_P is the set of perturbed column indices. Positive co-hub columns (PCC): CNJGL: $\sum_{i=1}^p \mathbf{1}\left\{\|\hat{\boldsymbol{V}}^1_{-i,i}\|_2>t_s \text{and} \|\hat{\boldsymbol{V}}^2_{-i,i}\|_2>t_s\right\}$; GGL/GL: $\sum_{i=1}^p \mathbf{1}\left\{\|\hat{\mathbf{\Theta}}^1_{-i,i}\|_2>t_s \text{and} \|\hat{\mathbf{\Theta}}^2_{-i,i}\|_2>t_s\right\}$ True positive co-hub columns (TPCC): CNJGL: $\sum_{i\in I_c} \mathbf{1}\left\{\|\hat{\boldsymbol{V}}^1_{-i,i}\|_2>t_s \text{and} \|\hat{\boldsymbol{V}}^2_{-i,i}\|_2>t_s\right\}$; GGL/GL: $\sum_{i\in I_c} \mathbf{1}\left\{\|\hat{\mathbf{\Theta}}^1_{-i,i}\|_2>t_s \text{and} \|\hat{\mathbf{\Theta}}^2_{-i,i}\|_2>t_s\right\}$, where I_C is the set of co-hub column indices.
(3)	Error: $\sqrt{\sum_{i<j}(\mathbf{\Theta}^1_{ij}-\hat{\mathbf{\Theta}}^1_{ij})^2}+\sqrt{\sum_{i<j}(\mathbf{\Theta}^2_{ij}-\hat{\mathbf{\Theta}}^2_{ij})^2}$

Table 4.1: Metrics used to quantify algorithm performance. Here $\mathbf{\Theta}^1$ and $\mathbf{\Theta}^2$ denote the true inverse covariance matrices, and $\hat{\mathbf{\Theta}}^1$ and $\hat{\mathbf{\Theta}}^2$ denote the two estimated inverse covariance matrices. Here $\mathbf{1}\{A\}$ is an indicator variable that equals one if the event A holds, and equals zero otherwise. (1) Metrics based on recovery of the support of $\mathbf{\Theta}^1$ and $\mathbf{\Theta}^2$. Here $t_0 = 10^{-6}$. (2) Metrics based on identification of perturbed nodes and co-hub nodes. The metrics PPC and TPPC quantify node perturbation, and are applied to PNJGL, FGL, and GL. The metrics PCC and TPCC relate to co-hub detection, and are applied to CNJGL, GGL, and GL. We let $t_s = \mu + 5.5\sigma$, where μ is the mean and σ is the standard deviation of $\{\|\hat{V}_{-i,i}\|_2\}_{i=1}^p$ (PPC or TPPC for PNJGL), $\{\|(\hat{\mathbf{\Theta}}^1-\hat{\mathbf{\Theta}}^2)_{-i,i}\|_2\}_{i=1}^p$ (PPC or TPPC for FGL/GL), $\{\|\hat{V}^1_{-i,i}\|_2\}_{i=1}^p$ and $\{\|\hat{V}^2_{-i,i}\|_2\}_{i=1}^p$ (PPC or TPPC for CNJGL), or $\{\|\hat{\mathbf{\Theta}}^1_{-i,i}\|_2\}_{i=1}^p$ and $\{\|\hat{\mathbf{\Theta}}^2_{-i,i}\|_2\}_{i=1}^p$ (PPC or TPPC for GGL/GL). However, results are very insensitive to the value of t_s. (3) Frobenius error of estimation of $\mathbf{\Theta}^1$ and $\mathbf{\Theta}^2$.

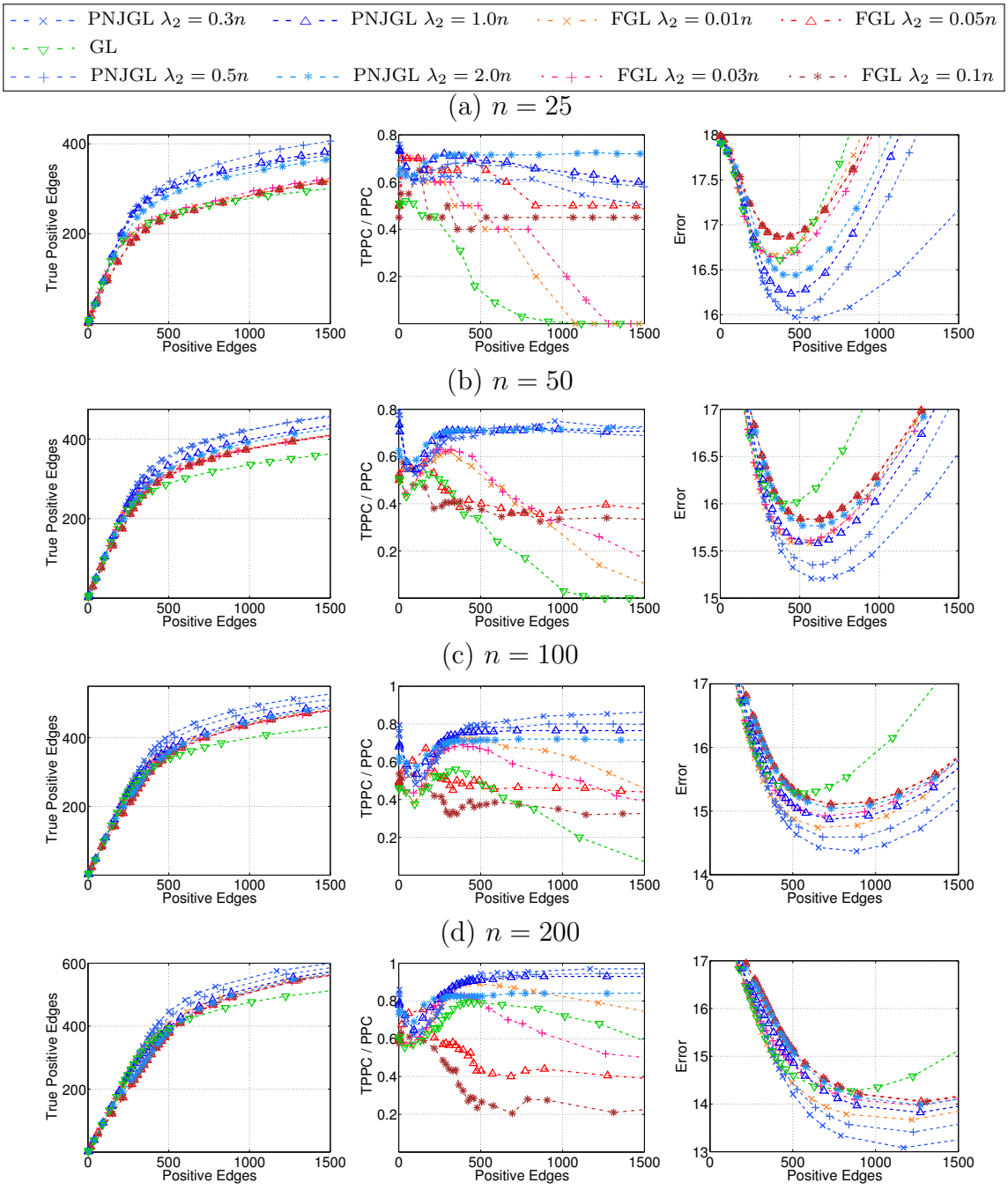


Figure 4.3: Simulation results on Erdos-Renyi network for PNJGL with $q = 2$, FGL, and GL, for *(a):* $n = 25$, *(b):* $n = 50$, *(c):* $n = 100$, *(d):* $n = 200$, when $p = 100$. Each colored line corresponds to a fixed value of λ_2, as λ_1 is varied. Axes are described in detail in Table 4.1. Results are averaged over 100 random generations of the data.

models. The priors are modeled through the RCON penalty which is related to the RCON penalty (3.5) that was used to learn a single graphical model with hub structure in the previous chapter. They also have a better performance than formulations that learn multiple graphical models using only a sparse sharing of information. Many convex formulations have been proposed for the joint estimation of graphical models – GGL, FGL [Danaher et al., 2013], CNJGL, PNJGL [Mohan et al., 2013], CSSL [Hara and Washio, 2013]. However, consistency results haven't been given for any of these formulations. One direction for future work is to show consistency results for the GGL, FGL, PNJGL and CNJGL formulations.

4.6 Acknowledgements

The results in this chapter are based on joint work with Maryam Fazel, Daniela Witten, Su-In Lee and Palma London.

Chapter 5

SCALING UP ALGORITHMS FOR LEARNING STRUCTURED GRAPHICAL MODELS

5.1 Introduction

The ADMM algorithms presented in the previous chapters work efficiently on problems of moderate size (less than 1000 variables). In order to solve the HGL, PNJGL or CNJGL (discussed in Chapters 3 and 4) optimization problems when the number of variables is large, a more scalable approach is needed. We consider two approaches to scale to larger problem sizes:

1. We present screening rules that can be used to speed up *any* algorithm for the HGL, PNJGL and CNJGL formulations. These screening rules on the regularization parameters help identify when the optimal solution is block-diagonal (up to a permutation). This helps break down the original optimization problem into potentially many sub-problems and thus speed up the algorithm. We discuss this in Sections 5.2 - 5.3.

2. We also recognize that the ADMM algorithms that we have proposed have a critical bottleneck of having to compute the eigen decomposition in each iteration. Also many algorithms for learning graphical models (e.g. QUIC [Hsieh et al., 2011]) compute a Cholesky factorization in each iteration to check for positive definiteness of the iterates. There are a couple of algorithms for graphical lasso that are SVD-free (e.g. Glasso [Friedman et al., 2007, Mazumder and Hastie, 2012, Yun et al., 2011]), which we discuss in Section 5.4.1. However these algorithms don't easily generalize from graphical lasso to learning more general structured graphical models. We therefore present fast block-coordinate descent methods that avoid expensive linear algebra computations and only rely on simple line search and matrix-vector multiplies to ensure positive definiteness of the iterates. We discuss this algorithm in Section 5.4 and observe significant speed-ups in computation over ADMM algorithm. We implement this algorithm

for two applications on learning graphs with hubs which we discuss in Sections 5.5-5.6.

We note that the first approach to speed up algorithms through screening rules is *algorithm independent* and therefore can be used to speed up any algorithm for the HGL, PNJGL and CNJGL formulations in certain range of regularization parameters. On the other hand, in the second approach we come up with specific algorithms that are SVD-free and hence are efficient in solving structured graphical model formulations such as HGL (discussed in Chapter 3), the hub covariance selection problem [Tan et al., 2014], etc.

5.2 Screening rules

In this section, we describe conditions under which any algorithm for solving the HGL, PNJGL or CNJGL problems can be sped up substantially, for an appropriate range of regularization parameter values. Our conditions depend only on the sample covariance matrices $\boldsymbol{S}^1, \ldots, \boldsymbol{S}^k$ and regularization parameters λ_1, λ_2 and are similar in flavor to the conditions given for GL,GGL and FGL [Witten et al., 2011, Mazumder and Hastie, 2012, Danaher et al., 2013] These conditions can be applied in at most $O(p^2)$ operations.

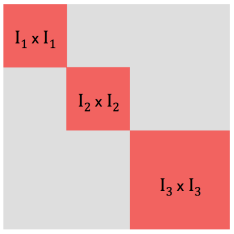


Figure 5.1: A $p \times p$ matrix is displayed, for which $\boldsymbol{I}_1, \boldsymbol{I}_2, \boldsymbol{I}_3$ denote a partition of the index set $\{1, 2, \ldots, p\}$. $\mathcal{T} = \bigcup_{i=1}^{L} \{\boldsymbol{I}_i \times \boldsymbol{I}_i\}$ is shown in red, and $\mathcal{T}^c$ is shown in gray.

We now introduce some notation that will be used throughout this section. Let $(\boldsymbol{I}_1, \boldsymbol{I}_2, \ldots, \boldsymbol{I}_L)$ be a partition of the index set $\{1, 2, \ldots, p\}$, and let $\mathcal{T} = \bigcup_{i=1}^{L} \{\boldsymbol{I}_i \times \boldsymbol{I}_i\}$. Define the *support* of a matrix $\boldsymbol{\Theta}$, denoted by $\text{supp}(\boldsymbol{\Theta})$, as the set of indices of the non-zero entries in $\boldsymbol{\Theta}$. We say $\boldsymbol{\Theta}$ is supported on $\mathcal{T}$ if $\text{supp}(\boldsymbol{\Theta}) \subseteq \mathcal{T}$. Note that any matrix supported on $\mathcal{T}$ is block-diagonal subject to some permutation of its rows

and columns. Let $|\mathcal{T}|$ denote the cardinality of the set $\mathcal{T}$, and let $\mathcal{T}^c$ denote the complement of $\mathcal{T}$. The scheme is displayed in Figure 5.1. We next discuss conditions for the optimal solution to HGL, CNJGL and PNJGL to be block-diagonal. This will then enable us to speed up algorithms for these problems.

5.2.1 Screening rule for HGL

Recall that the HGL formulation is given by,

$$\begin{array}{ll} \underset{\boldsymbol{\Theta}}{\text{minimize}} & -\log\det(\boldsymbol{\Theta}) + \langle \boldsymbol{\Theta}, \mathbf{S}\rangle + \lambda_1\|\mathbf{Z} - \text{Diag}(\mathbf{Z})\|_1 + \\ & \lambda_2\|\mathbf{V} - \text{Diag}(\mathbf{V})\|_1 + \lambda_3\|\mathbf{V} - \text{Diag}(\mathbf{V})\|_{1,2} \\ \text{subject to} & \boldsymbol{\Theta} = \mathbf{V} + \mathbf{V}^T + \mathbf{Z} \end{array} \tag{5.1}$$

We have the following conditions for the solution to (5.1) be block-diagonal.

Theorem 26 (Necessity)**.** *A necessary condition for the solution $\hat{\boldsymbol{\Theta}}$ to (5.1) to be block-diagonal with support $\mathcal{T}$ is that,*

$$\max_{(i,j)\in\mathcal{T}^c} |\mathbf{S}_{ij}| \;\leq\; \min\{\lambda_1, \tfrac{\lambda_2+\lambda_3}{2}\} \tag{5.2}$$

Theorem 27 (Sufficiency)**.** *A sufficient condition for the solution $\hat{\boldsymbol{\Theta}}$ to (5.1) to be block-diagonal with support $\mathcal{T}$ is that,*

$$\max_{(i,j)\in\mathcal{T}^c} |\mathbf{S}_{ij}| \;\leq\; \min\{\lambda_1, \tfrac{\lambda_2}{2}\} \tag{5.3}$$

Theorem 27 implies that one can screen the empirical covariance matrix $\mathbf{S}$ to check if the HGL solution is block diagonal [using standard algorithms for identifying the connected components of an undirected graph; see, e.g., Tarjan, 1972]. Suppose that the HGL solution is block diagonal with K blocks, containing $p_1, \ldots, p_K$ features, and $\sum_{k=1}^K p_k = p$. Then, one can simply solve the HGL problem on the features within each block separately. Recall that the bottleneck of the HGL algorithm is the eigen-decomposition for updating $\boldsymbol{\Theta}$. The block diagonal condition leads to massive computational speed-ups for implementing the HGL algorithm: instead of computing an eigen-decomposition for a $p \times p$ matrix in each iteration of the HGL algorithm, we

compute the eigen-decomposition of K matrices of dimensions $p_1 \times p_1, \ldots, p_K \times p_K$. The computational complexity per-iteration is reduced from $O(p^3)$ to $\sum_{k=1}^{K} O(p_k^3)$.

We illustrate the reduction in computational time due to these results in an example with $p = 500$. Without exploiting Theorem 27, the ADMM algorithm for HGL (with a particular value of λ) takes 159 seconds; in contrast, it takes only 22 seconds when Theorem 27 is applied. The estimated precision matrix has 107 connected components, the largest of which contains 212 nodes.

5.2.2 Screening rule for PNJGL

In this section, we give necessary conditions and sufficient conditions on the regularization parameters λ_1, λ_2 in the PNJGL problem (4.3) so that the resulting precision matrix estimates $\hat{\boldsymbol{\Theta}}^1, \ldots, \hat{\boldsymbol{\Theta}}^K$ have a shared block-diagonal structure (up to a permutation of the features).

We first present a necessary condition for $\hat{\boldsymbol{\Theta}}^1$ and $\hat{\boldsymbol{\Theta}}^2$ that minimize (4.3) with $K = 2$ to be block-diagonal.

Theorem 28. *Suppose that the matrices $\hat{\boldsymbol{\Theta}}^1$ and $\hat{\boldsymbol{\Theta}}^2$ that minimize (4.3) with $K = 2$ have support $\mathcal{T}$. Then, if $q \geq 1$, it must hold that*

$$n_k |\mathbf{S}_{ij}^k| \leq \lambda_1 + \lambda_2/2 \qquad \forall (i,j) \in \mathcal{T}^c, \quad \text{for } k = 1, 2, \text{ and} \tag{5.4}$$

$$|n_1 \mathbf{S}_{ij}^1 + n_2 \mathbf{S}_{ij}^2| \leq 2\lambda_1 \qquad \forall (i,j) \in \mathcal{T}^c. \tag{5.5}$$

Furthermore, if $q > 1$, then it must additionally hold that

$$\frac{n_k}{|\mathcal{T}^c|} \sum_{(i,j) \in \mathcal{T}^c} |\mathbf{S}_{ij}^k| \leq \lambda_1 + \frac{\lambda_2}{2} \left(\frac{p}{|\mathcal{T}^c|} \right)^{1/s}, \quad \text{for } k = 1, 2. \tag{5.6}$$

Remark 29. *If $|\mathcal{T}^c| = O(p^r)$ with $r > 1$, then as $p \to \infty$, (5.6) simplifies to $\frac{n_k}{|\mathcal{T}^c|} \sum_{(i,j) \in \mathcal{T}^c} |\mathbf{S}_{ij}^k| \leq \lambda_1$.*

We now present a sufficient condition for $\hat{\boldsymbol{\Theta}}^1, \ldots, \hat{\boldsymbol{\Theta}}^K$ that minimize (4.3) to be block-diagonal.

Theorem 30. *For $q \geq 1$, a sufficient condition for the matrices $\hat{\boldsymbol{\Theta}}^1, \ldots, \hat{\boldsymbol{\Theta}}^K$ that*

minimize (4.3) to each have support $\mathcal{T}$ is that

$$n_k|\mathbf{S}^k_{ij}| \leq \lambda_1 \quad \forall (i,j) \in \mathcal{T}^c, \quad \text{for } k = 1, \ldots, K.$$

Furthermore, if $q = 1$ and $K = 2$, then the necessary conditions (5.4) and (5.5) are also sufficient.

When $q = 1$ and $K = 2$, then the necessary and sufficient conditions in Theorems 28 and 30 are identical, as was previously reported in [Danaher et al., 2013]. In contrast, there is a gap between the necessary and sufficient conditions in Theorems 28 and 30 when $q > 1$ and $\lambda_2 > 0$. When $\lambda_2 = 0$, the necessary and sufficient conditions in Theorems 28 and 30 reduce to the results laid out in [Witten et al., 2011, Mazumder and Hastie, 2012] for the graphical lasso.

5.2.3 Screening rule for CNJGL

In this section, we give necessary and sufficient conditions on the regularization parameters λ_1, λ_2 in the CNJGL optimization problem (4.4) so that the resulting precision matrix estimates $\hat{\mathbf{\Theta}}^1, \ldots, \hat{\mathbf{\Theta}}^K$ have a shared block-diagonal structure (up to a permutation of the features).

Theorem 31. *Suppose that the matrices $\hat{\mathbf{\Theta}}^1, \hat{\mathbf{\Theta}}^2, \ldots, \hat{\mathbf{\Theta}}^K$ that minimize (4.4) have support $\mathcal{T}$. Then, if $q \geq 1$, it must hold that*

$$n_k|\mathbf{S}^k_{ij}| \leq \lambda_1 + \lambda_2/2 \quad \forall (i,j) \in \mathcal{T}^c, \quad \text{for } k = 1, \ldots, K. \tag{5.7}$$

Furthermore, if $q > 1$, then it must additionally hold that

$$\frac{n_k}{|\mathcal{T}^c|} \sum_{(i,j) \in T^c} |\mathbf{S}^k_{ij}| \leq \lambda_1 + \frac{\lambda_2}{2} \left(\frac{p}{|\mathcal{T}^c|} \right)^{1/s}, \quad \text{for } k = 1, \ldots, K. \tag{5.8}$$

Remark 32. *If $|\mathcal{T}^c| = O(p^r)$ with $r > 1$, then as $p \to \infty$, (5.8) simplifies to $\frac{n_k}{|\mathcal{T}^c|} \sum_{(i,j) \in \mathcal{T}^c} |\mathbf{S}^k_{ij}| \leq \lambda_1$.*

We now present a sufficient condition for $\hat{\mathbf{\Theta}}^1, \hat{\mathbf{\Theta}}^2, \ldots, \hat{\mathbf{\Theta}}^K$ that minimize (4.4) to be block-diagonal.

Theorem 33. *A sufficient condition for* $\hat{\boldsymbol{\Theta}}^1, \hat{\boldsymbol{\Theta}}^2, \ldots, \hat{\boldsymbol{\Theta}}^K$ *that minimize (4.4) to have support* $\mathcal{T}$ *is that*

$$n_k |\mathbf{S}^k_{ij}| \leq \lambda_1 \quad \forall (i,j) \in \mathcal{T}^c, \quad \text{for } k = 1, \ldots, K.$$

As was the case for the PNJGL formulation, there is a gap between the necessary and sufficient conditions for the estimated precision matrices from the CNJGL formulation to have a common block-diagonal support.

5.2.4 General Sufficient Conditions

In this section, we give sufficient conditions for the solution to a general class of optimization problems that include FGL, PNJGL, and CNJGL as special cases to be block-diagonal. Consider the optimization problem

$$\underset{\boldsymbol{\Theta}^1, \ldots, \boldsymbol{\Theta}^K \in \mathbb{S}^p_{++}}{\text{minimize}} \left\{ \sum_{k=1}^K n_k (-\log\det(\boldsymbol{\Theta}^k) + \langle \boldsymbol{\Theta}^k, \boldsymbol{S}^k \rangle) + \sum_{k=1}^K \lambda_1 \|\boldsymbol{\Theta}^k\|_1 + \lambda_2 h(\boldsymbol{\Theta}^1, \ldots, \boldsymbol{\Theta}^K) \right\}. \tag{5.9}$$

Once again, let $\mathcal{T}$ be the support of a $p \times p$ block-diagonal matrix. Let $\boldsymbol{\Theta}_{\mathcal{T}}$ denote the restriction of any $p \times p$ matrix $\boldsymbol{\Theta}$ to $\mathcal{T}$; that is, $(\boldsymbol{\Theta}_{\mathcal{T}})_{ij} = \begin{cases} \boldsymbol{\Theta}_{ij} & \text{if } (i,j) \in T \\ 0 & \text{else} \end{cases}$.
Assume that the function h satisfies

$$h(\boldsymbol{\Theta}^1, \ldots, \boldsymbol{\Theta}^K) > h(\boldsymbol{\Theta}^1_{\boldsymbol{U}}, \ldots, \boldsymbol{\Theta}^K_{\boldsymbol{U}})$$

for any matrices $\boldsymbol{\Theta}^1, \ldots, \boldsymbol{\Theta}^K$ whose support strictly contains $\boldsymbol{U}$.

Theorem 34. *A sufficient condition for the matrices* $\hat{\boldsymbol{\Theta}}^1, \ldots, \hat{\boldsymbol{\Theta}}^K$ *that solve (5.9) to have support* $\mathcal{T}$ *is that*

$$n_k |\boldsymbol{S}^k_{ij}| \leq \lambda_1 \quad \forall (i,j) \in \mathcal{T}^c, \quad \text{for } k = 1, \ldots, K.$$

Note that this sufficient condition applies to a broad class of regularizers h; indeed, the sufficient conditions for PNJGL and CNJGL given in Theorems 30 and 33 are special cases of Theorem 34. In contrast, the necessary conditions for PNJGL and CNJGL in Theorems 28 and 31 exploit the specific structure of the RCON penalty.

5.3 Evaluation of Speed-Ups on Synthetic Data

Theorems 30 and 33 provide sufficient conditions for the precision matrix estimates from PNJGL or CNJGL to be block-diagonal with a given support. How can these be used in order to obtain computational speed-ups? We construct a $p \times p$ matrix A with elements

$$A_{ij} = \begin{cases} 1 & \text{if } i = j \text{ or if } |\mathbf{S}_{ij}| > \lambda_1 \\ 0 & \text{else} \end{cases}. \tag{5.10}$$

We can then check, in $O(p^2)$ operations, whether A is (subject to some permutation of the rows and columns) block-diagonal, and can also determine the partition of the rows and columns corresponding to the blocks [see e.g. Tarjan, 1972]. Then, by Theorems 30 and 33, we can conclude that the PNJGL or CNJGL estimates are block-diagonal, with the same partition of the features into blocks. Inspection of the PNJGL and CNJGL optimization problems reveals that we can then solve the problems on the features within each block separately, in order to obtain the global solution to the original PNJGL or CNJGL optimization problems.

We now investigate the speed-ups that result from applying this approach. We consider the problem of estimating two networks of size $p = 500$. We create two inverse covariance matrices that are block diagonal with two equally-sized blocks, and sparse within each block. We then generate $n_1 = 250$ observations from a multivariate normal distribution with the first covariance matrix, and $n_2 = 250$ observations from a multivariate normal distribution with the second covariance matrix. These observations are used to generate sample covariance matrices $\boldsymbol{S}^1$ and $\boldsymbol{S}^2$. We then performed CNJGL and PNJGL with $\lambda_2 = 1$ and a range of λ_1 values, with and without the computational speed-ups just described.

Figure 5.2 displays the performance of the CNJGL and PNJGL formulations, averaged over 20 data sets generated in this way. In each panel, the x-axis shows the number of blocks into which the optimization problems were decomposed using the sufficient conditions; note that this is a surrogate for the value of λ_1 in the CNJGL or PNJGL optimization problems. Figure 5.2(a) displays the ratio of the run-time taken by the ADMM algorithm when exploiting the sufficient conditions to the run-time when not using the sufficient conditions. Figure 5.2(b) displays the true-positive ratio – that is, the ratio of the number of true positive edges in the precision

matrix estimates to the total number of edges in the precision matrix estimates. Figure 5.2(c) displays the total number of true positives for the CNJGL and PNJGL estimates. Figure 5.2 indicates that the sufficient conditions detailed in this section lead to substantial computational improvements.

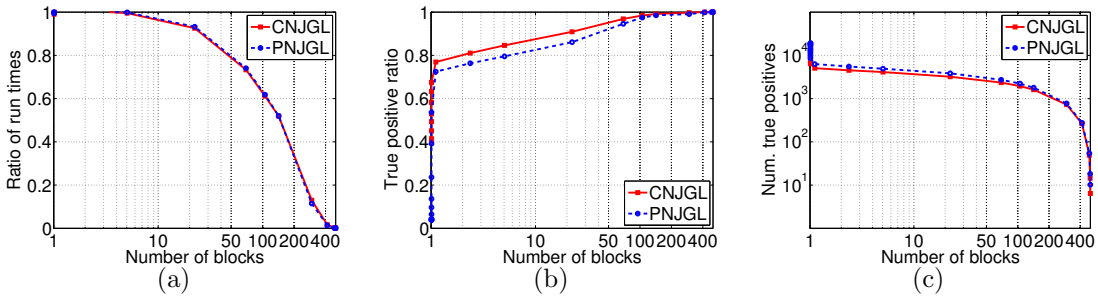


Figure 5.2: Speed-ups for CNJGL and PNJGL on a simulation set-up with $p = 500$ and $n_1 = n_2 = 250$. The true inverse covariance matrices are block-diagonal with two equally-sized sparse blocks. The x-axis in each panel displays the number of blocks into which the CNJGL or PNJGL problems are decomposed using the sufficient conditions; this is a surrogate for λ_1. The y-axes display *(a)*: the ratio of run-times with and without the sufficient conditions; *(b)*: the true positive ratio of the edges estimated; and *(c)*: the total number of true positive edges estimated.

5.4 SVD-free block coordinate descent algorithm (SBCD)

In this section, we consider the following problem:

$$\underset{\boldsymbol{V}:\boldsymbol{V}+\boldsymbol{V}^T\succ 0}{\text{minimize}} \quad f(\boldsymbol{V}) + g(\boldsymbol{V}), \tag{5.11}$$

where f is continuously differentiable and convex while g is proper, lower semicontinuous, convex and column separable. I.e. $g(\boldsymbol{V}) = \sum_i g_i(\boldsymbol{V}_{\cdot i})$, where $\boldsymbol{V}_{\cdot i}$ denotes the ith column of $\boldsymbol{V}$.

Some applications that can be modeled using (5.11) include graphical lasso [Friedman et al., 2007], joint learning of graphical models [Mohan et al., 2013, Danaher et al., 2013] and applications on learning graphs with hubs [Tan et al., 2014, Mohan et al., 2013]. In this chapter we develop a fast algorithm for (5.11) and apply it to

solve the following two problems on learning graphs with hubs:

1. Learning a Gaussian precision matrix with hub structure. Hub graphical lasso was proposed in Chapter 3 as a model for this problem. A simpler version of Hub graphical lasso is given by:

$$\begin{aligned}\underset{\boldsymbol{V}:\boldsymbol{V}+\boldsymbol{V}^T\succ 0}{\text{minimize}} \quad & -\log\det(\boldsymbol{V}+\boldsymbol{V}^T)+2\langle \boldsymbol{V},\boldsymbol{S}\rangle + \\ & \lambda\|\boldsymbol{V}-\text{Diag}(\boldsymbol{V})\|_{1,2}.\end{aligned} \tag{5.12}$$

   Here $f(\boldsymbol{V}) = -\log\det(\boldsymbol{V}+\boldsymbol{V}^T)+2\langle \boldsymbol{V},\boldsymbol{S}\rangle$ while $g(\boldsymbol{V}) = \lambda\|\boldsymbol{V}-\text{Diag}(\boldsymbol{V})\|_{1,2}$. Note that g is separable in the columns of $\boldsymbol{V}$. Thus blocks in this problem corresponds to columns of $\boldsymbol{V}$.

2. Learning a Gaussian covariance matrix with hub structure. Hub covariance selection [Tan et al., 2014] was proposed as a model for this problem. A simpler version of the hub covariance selection formulation is given by:

$$\underset{\boldsymbol{V}:\boldsymbol{V}+\boldsymbol{V}^T\succ 0}{\text{minimize}} \quad \tfrac{1}{2}\|\boldsymbol{V}+\boldsymbol{V}^T-\boldsymbol{S}\|_F^2+\lambda\|\boldsymbol{V}-\text{Diag}(\boldsymbol{V})\|_{1,2}. \tag{5.13}$$

   Here $f(\boldsymbol{V}) = \frac{1}{2}\|\boldsymbol{V}+\boldsymbol{V}^T-\boldsymbol{S}\|_F^2$ and $g(\boldsymbol{V}) = \lambda\|\boldsymbol{V}-\text{Diag}(\boldsymbol{V})\|_{1,2}$. Note that as before, g is separable in the columns of $\boldsymbol{V}$. Thus blocks in this problem corresponds to columns of $\boldsymbol{V}$.

Notation: For a vector $\boldsymbol{u}\in\mathbb{R}^p$, let $\boldsymbol{u}_i$ denote the ith entry of $\boldsymbol{u}$. For a vector $\boldsymbol{u}\in\mathbb{R}^p$, let $\boldsymbol{u}^{-i}$ be given by: $(\boldsymbol{u}^{-i})_j = \begin{cases}\boldsymbol{u}_j & \text{if } j\neq i\\ 0 & \text{otherwise}\end{cases}$. Also let $\boldsymbol{e}_i$ denote the ith coordinate vector that is all zeros except for the ith coordinate, which equals 1. Let $\mathbf{0}$ denote a vector of all zeros. For a matrix $\boldsymbol{V}$, let $\boldsymbol{V}_i$ denote the ith block of $\boldsymbol{V}$, where a block corresponds to a subset of the entries of $\boldsymbol{V}$. Let $\boldsymbol{V}_{\cdot i}$ denote the ith column of $\boldsymbol{V}$. Also let $\boldsymbol{v}^{-i}$ denote $(\boldsymbol{V}_{\cdot i})^{-i}$. Note that $(\boldsymbol{V}-\text{Diag}(\boldsymbol{V}))_{\cdot i}$ is the same as $\boldsymbol{v}^{-i}$. For a convex function f, let ∂f denote its subdifferential, i.e. the set of all subgradients of f.

5.4.1 Literature review

Learning sparse graphical models through the graphical lasso formulation [Friedman et al., 2007] has been well studied in the literature. Recently, a few efficient algorithms have been proposed for graphical lasso that can scale to large problem sizes with thousands of variables (see e.g. [Hsieh et al., 2012, Dinh et al., 2013]). On the other hand, existing algorithms (e.g. ADMM) for learning structured graphical models in the applications mentioned above mostly use a SVD based approach which can be very expensive when scaling to larger problem sizes. [Dinh et al., 2013] proposed a SVD-free proximal Newton method to solve (5.11) without any explicit constraint. Although this method applies to hub graphical lasso in Chapter 3, it doesn't apply to other problems such as hub covariance selection [Tan et al., 2014], latent variable selection [Chandrasekaran et al., 2012b], etc. Furthermore, this algorithm uses the exact Hessian to do the proximal newton update making the per iteration cost potentially very expensive. Similarly the Glasso algorithm for graphical lasso [Friedman et al., 2007, Mazumder and Hastie, 2012], which is SVD-free, can be extended to solve the hub graphical lasso (which we do so in this paper and refer to as HGlasso), but it doesn't easily extend to solve other problems. [Yun et al., 2011] considered a related problem to (5.11):

$$\underset{\boldsymbol{V}}{\text{minimize}} \quad l(\boldsymbol{V}) + h(\boldsymbol{V}), \tag{5.14}$$

where l is assumed to be convex and continuously differentiable while h is assumed to be proper, lower semicontinuous, convex and block separable. In addition, it is assumed that the domain of l given by $\{\boldsymbol{V} : l(\boldsymbol{V}) < \infty\}$ is open. We can cast the problem (5.11) into (5.14) by letting $l(\boldsymbol{V}) = f(\boldsymbol{V}) + \mathcal{I}(\boldsymbol{V} + \boldsymbol{V}^T)$, where $\mathcal{I}(\boldsymbol{X})$ is an indicator function that equals 0 if $\boldsymbol{X} \succ 0$ and equals ∞ otherwise. Note that in (5.11), l is continuously differentiable over $\boldsymbol{V} : \boldsymbol{V} + \boldsymbol{V}^T \succ 0$ and thus l has an open domain. [Yun et al., 2011] proposed a block-coordinate gradient descent method, where in each iteration one block or column of $\boldsymbol{V}$ is updated using a gradient descent update. They showed that this algorithm enjoys a global convergence to the stationary point of (5.14) if in addition to the assumptions on l, h it is assumed that the sub-level set of $l + h$ at the starting point is bounded (which is a standard assumption). A key bottle-neck in algorithms for (5.11) is checking for the positive-definiteness

of the iterates, which is usually required to maintain feasibility of iterates. Many algorithms such as ADMM use either the (eigen) singular value decomposition (SVD) or Cholesky decomposition to check for positive-definiteness. [Yun et al., 2011] instead suggested the use of Schur-complements to select an appropriate step size to ensure positive definiteness of iterates during the algorithm thus avoiding expensive SVD computations.

5.4.2 Contributions

Our contributions in this section are as follows:

1. Suppose $\boldsymbol{\Theta} \succ 0$ and the algorithm updates a row and column of $\boldsymbol{\Theta}$ to yield $\boldsymbol{\Theta} + \alpha \boldsymbol{u}\boldsymbol{e}_i^T + \alpha \boldsymbol{e}_i \boldsymbol{u}^T$. [Yun et al., 2011] obtained expressions for the range of α for which $\boldsymbol{\Theta} + \alpha \boldsymbol{u}\boldsymbol{e}_i^T + \alpha \boldsymbol{e}_i \boldsymbol{u}^T \succ 0$. We generalize this result to answer the following question: For what choice of α is $\boldsymbol{\Theta} + \alpha \boldsymbol{u}\boldsymbol{w}^T + \alpha \boldsymbol{w}\boldsymbol{u}^T \succ 0$ if $\boldsymbol{\Theta} \succ 0$? Here $\boldsymbol{u}, \boldsymbol{w} \in \mathbb{R}^p$ are arbitrary vectors.

2. We also extend the algorithm of [Yun et al., 2011] by adding an active set heuristic. We refer to our algorithm (Algorithm 8) as SVD-free block coordinate descent algorithm (SBCD).

3. Finally, our experimental results on hub graphical lasso and hub covariance selection problems [Tan et al., 2014] demonstrate that our algorithm is much faster than ADMM and another SVD-free approach, HGlasso. As an example, on synthetic data sets for hub graphical lasso, we observe that SBCD is faster than the SVD-based ADMM algorithm by a factor of 200 for $p = 1000$. Similarly, for a real data example for hub graphical lasso with $p = 500$, we are faster than ADMM by a factor of 50. We note that the use of active-set hueristic is critical for the speed ups we observe, as the algorithm can now exploit the structured sparsity in the optimal solution through the identification of active set. We discuss the active set heuristic further in Section 5.4.4 and also in the context of hub graphical lasso in Section 5.5.1. We note that a more extensive comparison of SBCD with other SVD based algorithms and SVD-free algorithms would be useful for the hub covariance selection problem and other applications, that we discuss further in future work.

5.4.3 Step-size selection for positive definiteness

Algorithm 8 (that we discuss in the subsequent section) involves a block-coordinate descent update where the block is given by a row and a column of a matrix. This translates to a rank-two update to a positive definite matrix. We therefore ask the following question:
Given $\boldsymbol{\Theta} \in \mathcal{S}^p_{++}$*, for what range of* α *is* $\boldsymbol{\Theta} + \alpha(\boldsymbol{u}\boldsymbol{w}^T + \boldsymbol{w}\boldsymbol{u}^T) \succ 0$*, where* $\boldsymbol{u}, \boldsymbol{w} \in \mathbb{R}^p$*?*
Answering this question will help us pick a step-size that can preserve the positive definiteness of the iterates (in our algorithm, our rank two update has a special structure where $\boldsymbol{w} = \boldsymbol{e}_i$). Interestingly, we can compute this range of α in closed form as we discuss next. Choosing such a step-size will ensure that in each iteration of our approach, we stay within the positive definite cone while also making progress towards the optimal solution. Note that we will give a range for α whose computation only involves matrix-vector multiplies and avoids computationally expensive SVD checks.

To arrive at a range for α, we need to examine the eigenvalues of $\boldsymbol{\Theta} + \alpha(\boldsymbol{u}\boldsymbol{w}^T + \boldsymbol{w}\boldsymbol{u}^T)$. Assume that $\boldsymbol{\Theta} \succ 0$ with eigenvalues $0 < \eta_p \leq \eta_{p-1} \cdots \eta_2 \leq \eta_1$. Before we examine the eigenvalues of $\boldsymbol{\Theta} + \alpha(\boldsymbol{u}\boldsymbol{w}^T + \boldsymbol{w}\boldsymbol{u}^T)$, let us take a look at the eigenvalues of $\mathbf{B} := \boldsymbol{u}\boldsymbol{w}^T + \boldsymbol{w}\boldsymbol{u}^T$. It is easily seen that $\text{rank}(\mathbf{B}) = 2$, hence $\mathbf{B}$ has 2 non zero eigenvalues only, say β_p, β_1. Observe that

$$\begin{aligned}\beta_1 + \beta_p &:= \text{Tr}(\mathbf{B}) = 2\boldsymbol{u}^T\boldsymbol{w} \\ \beta_1^2 + \beta_p^2 &:= \text{Tr}(\mathbf{B}^2) = 2(\boldsymbol{u}^T\boldsymbol{w})^2 + 2\|\boldsymbol{u}\|_2^2\|\boldsymbol{w}\|_2^2,\end{aligned}$$

hence

$$2\beta_1\beta_p = (\beta_1 + \beta_p)^2 - (\beta_1^2 + \beta_p^2) = 2(\boldsymbol{u}^T\boldsymbol{w})^2 - 2\|\boldsymbol{u}\|_2^2\|\boldsymbol{w}\|_2^2, \tag{5.15}$$

by Cauchy-Schwartz inequality, $\beta_1\beta_p = (\boldsymbol{u}^T\boldsymbol{w})^2 - \|\boldsymbol{u}\|_2^2\|\boldsymbol{w}\|_2^2 \leq 0$ with equality holds when $\boldsymbol{u}$ and $\boldsymbol{w}$ are linearly dependent. Assume $\boldsymbol{u}$ and $\boldsymbol{w}$ are not linearly independent, then $\beta_1\beta_p < 0$, let us say $\beta_p < 0 < \beta_1$. We summarize this in the following lemma

Lemma 35. *Let* $\mathbf{B} := \boldsymbol{u}\boldsymbol{w}^T + \boldsymbol{w}\boldsymbol{u}^T$ *where* $\boldsymbol{u}$ *and* $\boldsymbol{w}$ *are not linearly dependent, then* $\mathbf{B}$ *has* $p-2$ *zero eigenvalues and two non zero eigenvalues with opposite sign, i.e. we*

can order the eigenvalues of $\mathbf{B}$ *as*

$$\beta_p < 0 = \beta_{p-1} = \beta_{p-2} \cdots = \beta_2 < \beta_1. \tag{5.16}$$

Before we give a full characterization of the eigenvalues of $\mathbf{\Theta}+\alpha\mathbf{B}$, we need another lemma.

Lemma 36 (Weyl)**.** *Let* $\mathbf{\Theta}$ *and* $\mathbf{B}$ *be symmetric and the eigenvalues of* $\mathbf{\Theta}$, $\mathbf{B}$ *and* $\mathbf{\Theta}+\mathbf{B}$ *be denoted by* η_i, β_i *and* λ_i *respectively with orderings*

$$\begin{aligned}
\eta_p \leq \eta_{p-1} \leq \cdots \leq \eta_1, \\
\beta_p \leq \beta_{p-1} \leq \cdots \leq \beta_1, \\
\lambda_p \leq \lambda_{p-1} \leq \cdots \leq \lambda_1.
\end{aligned}$$

Then

$$\eta_p \leq \lambda_i - \beta_i \leq \eta_1 \quad i = 1, \cdots, p \tag{5.17}$$

Now we are ready to state our main theorem

Theorem 37. *Let* $\mathbf{\Theta} \succ 0$ *and* $\mathbf{B} = \boldsymbol{u}\boldsymbol{w}^T + \boldsymbol{w}\boldsymbol{u}^T$ *where* $\boldsymbol{u}$ *and* $\boldsymbol{w}$ *are not linearly dependent. Let* $\alpha > 0$*. Then* $\mathbf{\Theta} + \alpha\mathbf{B} \succ 0$ *if and only if* $\det(\mathbf{\Theta} + \alpha\mathbf{B}) > 0$*, which is also equivalent to* α *satisfying the following inequality*

$$\begin{aligned}
\delta^{-1}(\boldsymbol{w}^T\mathbf{\Theta}^{-1}\boldsymbol{u} - \sqrt{(\boldsymbol{u}^T\mathbf{\Theta}^{-1}\boldsymbol{u})(\boldsymbol{w}^T\mathbf{\Theta}^{-1}\boldsymbol{w})}) &< \alpha, \\
\delta^{-1}(\boldsymbol{w}^T\mathbf{\Theta}^{-1}\boldsymbol{u} + \sqrt{(\boldsymbol{u}^T\mathbf{\Theta}^{-1}\boldsymbol{u})(\boldsymbol{w}^T\mathbf{\Theta}^{-1}\boldsymbol{w})} &> \alpha,
\end{aligned} \tag{5.18}$$

where $\delta = (\boldsymbol{u}^T\mathbf{\Theta}^{-1}\boldsymbol{u})(\boldsymbol{w}^T\mathbf{\Theta}^{-1}\boldsymbol{w}) - (\boldsymbol{u}^T\mathbf{\Theta}^{-1}\boldsymbol{w})^2$.

Theorem 37 can be proven for the special case of $\boldsymbol{w} = \boldsymbol{e}_i$ through the use of Schur complement conditions for positive-definiteness of a matrix (see e.g. [Yun et al., 2011]). This special case is used to check for positive definiteness of iterates in our algorithms. However Theorem 37 is a more general result that provides a range for the step size α so that any symmetric rank two update of a positive definite matrix preserves positive definiteness.

5.4.4 SVD-free block-coordinate descent

Recall the optimization problem in (5.11):

$$\underset{\boldsymbol{V}:\boldsymbol{V}+\boldsymbol{V}^T\succ 0}{\text{minimize}} \quad f(\boldsymbol{V}) + g(\boldsymbol{V}), \tag{5.19}$$

where $g(\boldsymbol{V}) = \sum_{i=1}^{p} g_i(\boldsymbol{V}_{\cdot i})$. Our SVD-free approach to this problem, which we call SVD-free block-coordinate descent (or SBCD) is given in Algorithm 8 and extends the algorithm in [Yun et al., 2011] by using an active set heuristic to choose the columns of $\boldsymbol{V}$ that would get updated in each pass of the algorithm. We note that in our experimental results on hub graphical lasso and hub covariance selection that we discuss in subsequent sections, this active set heuristic provides significant speed-ups to the SBCD algorithm.

Remarks:

1. **Armijo rule for sufficient descent**: In each iteration of our Algorithm 8, we use the Armijo rule to select a step size α that ensures sufficient descent (step 2) of Algorithm 9) in the objective value of (5.11). More formally, consider the following problem: $\min_{\boldsymbol{x}} l(\boldsymbol{x}) + q(\boldsymbol{x})$ where l is smooth and convex while q is convex. Given a direction $\boldsymbol{d}$ and an initial point $\boldsymbol{x}$, the Armijo rule is used to select a step size α that ensures that $\boldsymbol{x} + \alpha\boldsymbol{d}$ has a sufficiently small objective. The Armijo rule for this problem is summarized in Algorithm 9.

2. Active set heuristic: The active set heuristic at the start of each iteration of Algorithm 8, essentially identifies the columns that need not be optimized over, since for these (active) columns the current iterate $\boldsymbol{V}$ has a low optimality error. Thus, we only update those columns that are not close to satisfying the optimality conditions through block coordinate descent. The precise check for active set is described in Step 1 of Algorithm 8.

3. Termination criterion: Note that the optimality conditions for (5.11) is $0 \in \nabla f(\boldsymbol{V}) + \partial g(\boldsymbol{V})$ (this is because the constraint $\boldsymbol{V} + \boldsymbol{V}^T \succ 0$ can be added to the domain of definition of f, which would be an open set). In Algorithm 8, we terminate when all the indices $i = 1, 2, \ldots, p$ are in the active set $\mathcal{A}$, so

Algorithm 8: SBCD algorithm with active set.

Input: f, g, starting point $\boldsymbol{V}, \boldsymbol{\Theta}$, tolerance ϵ and L, Lipschitz constant for the gradient of f.
Output: ϵ-optimal solution $\hat{\boldsymbol{V}} : \hat{\boldsymbol{V}} + (\hat{\boldsymbol{V}})^T \succ 0$.
Initialize: Active set $\mathcal{A} = \{\}$.
while $\mathcal{A}^c \neq \{\}$ **do**

1 **for** $i = 1, 2, \ldots, p$ **do**
  Set $\boldsymbol{\Gamma}_{\cdot i} = \underset{\boldsymbol{s}:\boldsymbol{s}\in\partial g_i(\boldsymbol{V}_{\cdot i})}{\operatorname{argmin}} \|(\nabla f(\boldsymbol{V}))_{\cdot i} + \boldsymbol{s}\|_2$.

2 Identify active set: $\mathcal{A} = \{i : \|(\nabla f(\boldsymbol{V}))_{\cdot i} + \boldsymbol{\Gamma}_{\cdot i}\|_\infty \leq \epsilon\}$.

3 **for** $i \in \mathcal{A}^c$ **do**

(a) $\bar{\boldsymbol{V}}_{\cdot i} = \underset{\boldsymbol{v}}{\operatorname{argmin}} \langle (\nabla f(\boldsymbol{V}))_{\cdot i}, \boldsymbol{v}\rangle + \frac{L}{2}\|\boldsymbol{v} - \boldsymbol{V}_{\cdot i}\|_2^2 + g_i(\boldsymbol{V}_{\cdot i})$

(b) Let $l(\cdot) = f(\boldsymbol{V} + \cdot \boldsymbol{e}_i^T), q(\cdot) = g(\boldsymbol{V} + \cdot \boldsymbol{e}_i^T)$ and $\boldsymbol{d} = \bar{\boldsymbol{V}}_{\cdot i} - \boldsymbol{V}_{\cdot i}$.

(c) Set α_{init} to be the upper bound in (5.18) with $\boldsymbol{u} = \boldsymbol{d}, \boldsymbol{w} = \boldsymbol{e}_i, \boldsymbol{\Theta}^{-1}$.

(d) $\boldsymbol{V}_{\cdot i} = \boldsymbol{V}_{\cdot i} + \alpha \boldsymbol{d}$ where α is the output of Algorithm 9 with input parameters $l, q, \boldsymbol{x} = \boldsymbol{0}, \boldsymbol{d}, \alpha_{\text{init}}$.

Set $\hat{\boldsymbol{V}} = \boldsymbol{V}$

Algorithm 9: Armijo rule for minimizing the sum of a smooth plus a non-smooth function

Input: Functions l, q, current iterate $\boldsymbol{x}$, direction $\boldsymbol{d}$ and initial α_{init}
Output: $\hat{\alpha}$.
Initialize: Initial step size $\alpha = \alpha_{init}$, Armijo parameters $\sigma = 0.5, \beta = 0.5, \alpha_{\min} = 10^{-6}$.

1 Set $\Delta = q(\boldsymbol{x} + \boldsymbol{d}) - q(\boldsymbol{x}) + \langle \nabla l(\boldsymbol{x}), \boldsymbol{d}\rangle$.

2 **while** $l(\boldsymbol{x} + \alpha\boldsymbol{d}) + q(\boldsymbol{x} + \alpha\boldsymbol{d}) - l(\boldsymbol{x}) - q(\boldsymbol{x}) \geq \alpha\sigma\Delta$ *and* $\alpha \geq \alpha_{\min}$ **do**
  $\alpha = \alpha\beta$.

3 If $\alpha < \alpha_{\min}$, set $\alpha = 0$.

4 Set $\hat{\alpha} = \alpha$.

that $\mathcal{A}^c = \{\}$. When this happens, by definition of the active set, we have that $\|\nabla f(\boldsymbol{V}) + \boldsymbol{\Gamma}\|_\infty \leq \epsilon$, i.e. the optimality error in the $\|.\|_\infty$ norm is less than the optimality tolerance ϵ.

4. Note that SBCD converges to the global optimal value of (5.19) as in each iteration, SBCD descends uses the Armijo rule for sufficient descent. The convergence result of the iterates shown in [Yun et al., 2011] also extends to SBCD (that uses an active set heuristic) in the following sense: In each iteration of SBCD , we skip the updates of the columns that are active since these columns have a low optimality error. Thus, we are actually passing through all the columns in each iteration of SBCD (since updating the active columns that have a low optimality error is equivalent to skipping them). We therefore satisfy the Gauss-Siedel assumption on passes through the coordinates (or columns) in [Yun et al., 2011] and therefore the convergence results in [Yun et al., 2011] extends to SBCD with an active set heuristic.

5. Updating the inverse: In each iteration of Algorithm 8, we need to compute the step-size α. To ensure that $\boldsymbol{\Theta}$ stays positive definite, we need α to satisfy 5.18. This requires the computation of the inverse of $\boldsymbol{\Theta}$. Since the update to $\boldsymbol{\Theta} = \boldsymbol{V} + \boldsymbol{V}^T$ involves a rank-2 update, the inverse of $\boldsymbol{\Theta}$ can be easily updated using the Sherman-Morrison-Woodbury identity [Woodbury, 1950]. The cost of the inverse update is $O(p^2)$.

6. Per iteration computational complexity: In each iteration, we require the computation of the gradient and the computation of step-size α. In the case where $f(\boldsymbol{V}) = -\log\det(\boldsymbol{V} + \boldsymbol{V}^T)$, the gradient computation involves the inverse of $\boldsymbol{\Theta}$ which is updated at the end of each iteration using the Sherman-Morrison-Woodbury identity (see e.g. [Woodbury, 1950]) and uses $O(p^2)$ computation. The update of α involves matrix-vector multiplies, which is again $O(p^2)$ updates. Thus in the case where $f(\boldsymbol{V}) = -\log\det(\boldsymbol{V} + \boldsymbol{V}^T)$, the per iteration cost is $O(p^2)$.

5.5 SBCD for hub graphical lasso

A simple version of the hub graphical lasso (3.6) (where we ignore the sparsity promoting regularizers for the sake of simplicity) is given by the following model:

$$\underset{\boldsymbol{V}:\boldsymbol{V}+\boldsymbol{V}^T\succ 0}{\text{minimize}} \; -\log\det(\boldsymbol{V}+\boldsymbol{V}^T)+\langle \boldsymbol{V}+\boldsymbol{V}^T,\boldsymbol{S}\rangle+\lambda\|\boldsymbol{V}-\text{Diag}(\boldsymbol{V})\|_{1,2}, \tag{5.20}$$

where we assume that the domain of $\log\det(.)$ is the set of all positive definite matrices. Thus, we have an implicit constraint that $\boldsymbol{V}+\boldsymbol{V}^T\succ 0$. Let $\boldsymbol{v}^{-i}$ denote $(\boldsymbol{V}_{\cdot i})^{-i}$.

Note that (5.20) is equivalent to:

$$\underset{\boldsymbol{V}:\boldsymbol{V}+\boldsymbol{V}^T\succ 0}{\text{minimize}} \; -\log\det(\boldsymbol{V}+\boldsymbol{V}^T)+2\langle \boldsymbol{S},\boldsymbol{V}\rangle+\lambda\sum_{i=1}^{p}\|\boldsymbol{v}^{-i}\|_2. \tag{5.21}$$

We use Algorithm 8 to solve (5.21) with $f(\boldsymbol{V})=-\log\det(\boldsymbol{V}+\boldsymbol{V}^T)+2\langle \boldsymbol{S},\boldsymbol{V}\rangle$, $g(\boldsymbol{V})=\lambda\sum_{i=1}^{p}\|\boldsymbol{v}^{-i}\|_2$. To do this, we need to compute Step 3(a) of Algorithm 8 efficiently. We first note that optimizing $\boldsymbol{V}_{\cdot i}$ while fixing other columns of $\boldsymbol{V}$ yields the following subproblem:

$$\underset{\boldsymbol{u}}{\text{minimize}} \quad f(\boldsymbol{V}+\boldsymbol{u}\boldsymbol{e}_i^T)+g(\boldsymbol{V}+\boldsymbol{u}\boldsymbol{e}_i^T), \tag{5.22}$$

which is equivalent to

$$\begin{aligned}\underset{\boldsymbol{u}}{\text{minimize}} \; & -\log\det(\boldsymbol{V}+\boldsymbol{V}^T+\boldsymbol{u}\boldsymbol{e}_i^T+\boldsymbol{e}_i\boldsymbol{u}^T)+2\langle \boldsymbol{S},\boldsymbol{V}+\boldsymbol{u}\boldsymbol{e}_i^T\rangle \\ & +\lambda\sum_{i=1}^{p}\|(V+\boldsymbol{u}\boldsymbol{e}_i^T-\text{Diag}(\boldsymbol{V}+\boldsymbol{u}\boldsymbol{e}_i^T))_{\cdot i}\|_2,\end{aligned} \tag{5.23}$$

and ignoring the constant terms gives

$$\underset{\boldsymbol{u}}{\text{minimize}} \; -\log\det(\boldsymbol{V}+\boldsymbol{V}^T+\boldsymbol{u}\boldsymbol{e}_i^T+\boldsymbol{e}_i\boldsymbol{u}^T)+2\boldsymbol{S}_{\cdot i}^T\boldsymbol{u}+\lambda\|\boldsymbol{v}^{-i}+\boldsymbol{u}^{-i}\|_2. \tag{5.24}$$

The challenge in (5.24) is that the evaluation of the objective function for Armijo rule in (step 3d) of Algorithm 8 seems to involve the log-determinant that can be expensive to compute. We therefore have the following lemma that transforms (5.24)

into a simpler optimization problem.

Lemma 38. *(5.24) is equivalent to the following optimization problem:*

$$\underset{\boldsymbol{u}}{\text{minimize}}\ h(\boldsymbol{u}) := l(\boldsymbol{u}) + q(\boldsymbol{u}), \tag{5.25}$$

where

$$l(\boldsymbol{u}) = -\log[(1 + \boldsymbol{e}_i^T\boldsymbol{\Theta}^{-1}\boldsymbol{u})^2 - (\boldsymbol{e}_i^T\boldsymbol{\Theta}^{-1}\boldsymbol{e}_i)(\boldsymbol{u}^T\boldsymbol{\Theta}^{-1}\boldsymbol{u})] + 2\boldsymbol{S}_{\cdot i}^T\boldsymbol{u}. \tag{5.26}$$

and

$$q(\boldsymbol{u}) = \lambda\|\boldsymbol{v}^{-i} + \boldsymbol{u}^{-i}\|_2. \tag{5.27}$$

Proof. Let,

$$\hat{l}(\boldsymbol{u}) := f(\boldsymbol{V} + \boldsymbol{u}\boldsymbol{e}_i^T) = -\log\det(\boldsymbol{\Theta} + \boldsymbol{u}\boldsymbol{e}_i^T + \boldsymbol{e}_i\boldsymbol{u}^T) + 2\boldsymbol{S}_{\cdot i}^T\boldsymbol{u} \tag{5.28}$$

Assume we keep track of the inverse of $\boldsymbol{\Theta} := \boldsymbol{V} + \boldsymbol{V}^T$. Then the computation of $\log\det(\boldsymbol{V} + \boldsymbol{V}^T + \boldsymbol{u}\boldsymbol{e}_i^T + \boldsymbol{e}_i\boldsymbol{u}^T)$ in $\hat{l}(\boldsymbol{u})$ can be done efficiently as follows. Note that

$$\begin{aligned}
\det(\boldsymbol{X} + \boldsymbol{u}\boldsymbol{w}^T) &= \det(\boldsymbol{X})\det(I + \boldsymbol{X}^{-1}\boldsymbol{u}\boldsymbol{w}^T) \\
&= \det(\boldsymbol{X})\prod_{i=1}^{p}(1 + \lambda_i(\boldsymbol{X}^{-1}\boldsymbol{u}\boldsymbol{w}^T)) \\
&= \det(\boldsymbol{X})(1 + \lambda_1((\boldsymbol{X}^{-1}\boldsymbol{u})\boldsymbol{w}^T) \\
&= \det(\boldsymbol{X})(1 + \boldsymbol{w}^T\boldsymbol{X}^{-1}\boldsymbol{u}),
\end{aligned} \tag{5.29}$$

where we use the fact that the non-zero eigen value of a rank one matrix $\boldsymbol{a}\boldsymbol{b}^T$ is given $\boldsymbol{b}^T\boldsymbol{a}$. Using (5.29) twice, we have that,

$$\det(\boldsymbol{\Theta} + (\boldsymbol{u}\boldsymbol{w}^T + \boldsymbol{w}\boldsymbol{u}^T)) = \det(\boldsymbol{\Theta})[(1 + \boldsymbol{w}^T\boldsymbol{\Theta}^{-1}\boldsymbol{u})^2 - (\boldsymbol{w}^T\boldsymbol{\Theta}^{-1}\boldsymbol{w})(\boldsymbol{u}^T\boldsymbol{\Theta}^{-1}\boldsymbol{u})]. \tag{5.30}$$

Using (5.30) we get that optimizing $\hat{l}(\boldsymbol{u}) + q(\boldsymbol{u})$ with respect to $\boldsymbol{u}$ is equivalent to optimizing $l(\boldsymbol{u}) + q(\boldsymbol{u})$ with respect to $\boldsymbol{u}$.

□

Note that $l(\boldsymbol{u})$ as defined in (5.26) does not involve the computation of a deter-

minant and hence is more efficient to compute. Also $\nabla l(\boldsymbol{u}) = \nabla_{\boldsymbol{u}} f(\boldsymbol{V} + \boldsymbol{u}\boldsymbol{e}_i^T)$. Now we are ready to describe the computation of step 3a) of Algorithm 8. Consider the following approximation to (5.25) at $\widetilde{\boldsymbol{u}}$

$$\underset{\boldsymbol{u}}{\text{minimize}}\ h(\boldsymbol{u}; \widetilde{\boldsymbol{u}}) := l(\widetilde{\boldsymbol{u}}) + \langle \nabla l(\widetilde{\boldsymbol{u}}), \boldsymbol{u} - \bar{\boldsymbol{u}} \rangle + \frac{1}{2}(\boldsymbol{u} - \widetilde{\boldsymbol{u}})^T \mathbf{H} (\boldsymbol{u} - \widetilde{\boldsymbol{u}}) + \lambda \|\boldsymbol{v}^{-i} + \boldsymbol{u}^{-i}\|_2, \tag{5.31}$$

where $\mathbf{H}$ is the an approximation to the Hessian of $f(\boldsymbol{u})$.

Consider a simple case $\mathbf{H} = \beta \boldsymbol{I}$, where β is an upper-bound on the Lipschitz constant of ∇f. Note that the solution to (5.31) also optimizes the subproblem in step 3a) of Algorithm 8. Define

$$\tilde{\boldsymbol{u}} = \left(1 - \frac{\lambda}{\|(\nabla l(\widetilde{\boldsymbol{u}}))^{-i} - \beta(\boldsymbol{v}^{-i} + \widetilde{\boldsymbol{u}}^{-i})\|_2}\right) + \left(\boldsymbol{v}^{-i} + \widetilde{\boldsymbol{u}}^{-i} - \frac{1}{\beta}\nabla l(\widetilde{\boldsymbol{u}})^{-i}\right) - \boldsymbol{v}^{-i},$$

where $(\boldsymbol{x})_+ = \max\{0, \boldsymbol{x}\}$,

then the solution to (5.31) is

$$\boldsymbol{u}^* = \begin{cases} \widetilde{\boldsymbol{u}}_j & \text{if } j \neq i, \\ -\left(\nabla l\left(\widetilde{\boldsymbol{u}}\right)\right)_j / \beta + \widetilde{\boldsymbol{u}}_j & \text{if } j = i. \end{cases} \tag{5.32}$$

Furthermore the gradient of l can be computed easily as:

$$\nabla l(\boldsymbol{u}) = -2\frac{(1 + \boldsymbol{e}_i^T \boldsymbol{\Theta}^{-1}\boldsymbol{u})\boldsymbol{\Theta}^{-1}\boldsymbol{e}_i - (\boldsymbol{e}_i^T \boldsymbol{\Theta}^{-1}\boldsymbol{e}_i)\boldsymbol{\Theta}^{-1}\boldsymbol{u}}{(1 + \boldsymbol{e}_i^T \boldsymbol{\Theta}^{-1}\boldsymbol{u})^2 - (\boldsymbol{e}_i^T \boldsymbol{\Theta}^{-1}\boldsymbol{e}_i)(\boldsymbol{u}^T \boldsymbol{\Theta}^{-1}\boldsymbol{u})} + 2\boldsymbol{S}_{\cdot i}, \tag{5.33}$$

where the main computation is $\boldsymbol{\Theta}^{-1}\boldsymbol{u}$.

The SBCD algorithm with active set heuristic (described in the section below) for hub graphical lasso is given in algorithm 10.

5.5.1 Active set heuristic

Under the assumption that we have only k hubs in the true precision matrix, we would expect that most of the columns of $\boldsymbol{V}$ would converge to 0 after sufficiently many iterations. Thus, it would be more efficient to skip the columns that are already optimal (or active). The active set is defined by step 2 of Algorithm 8. For the case of hub graphical lasso, the active set can easily be computed using step 2 in Algorithm

Algorithm 10: SBCD algorithm for hub graphical lasso (5.20)

Input: Sample covariance matrix $\boldsymbol{S}$, λ, optimality tolerance $\epsilon > 0$.
Output: ϵ-approximate optimal solution $\hat{\boldsymbol{V}}$.
Initialization: Set $\boldsymbol{V} = \frac{1}{2}(\text{Diag}(\boldsymbol{S}))^{-1}$, $\boldsymbol{\Theta} = \boldsymbol{V} + \boldsymbol{V}^T$ and $\boldsymbol{\Theta}^{-1} = \text{Diag}(\boldsymbol{S})$.
$\sigma \in (0, 1)$. Set active set, $\mathcal{A} = \{\}$.
while $\mathcal{A}^c \neq \{\}$ **do**

1 **for** $i = 1, 2, \ldots, p$ **do**
  if $\boldsymbol{v}^{-i} \neq \boldsymbol{0}$ **then**
    $\boldsymbol{\Gamma}_{\cdot i} = \lambda \boldsymbol{v}^{-i} / \|\boldsymbol{v}^{-i}\|_2$
  else

$$\boldsymbol{\Gamma}_{\cdot i} = \begin{cases} (2\boldsymbol{\Theta}_{\cdot i}^{-1} - 2\boldsymbol{S}_{\cdot i})^{-i} & \text{if } \|(2\boldsymbol{\Theta}_{\cdot i}^{-1} - 2\boldsymbol{S}_{\cdot i})^{-i}\|_2 \leq \lambda \\ \lambda \frac{(2\boldsymbol{\Theta}_{\cdot i}^{-1} - 2\boldsymbol{S}_{\cdot i})^{-i}}{\|(2\boldsymbol{\Theta}_{\cdot i}^{-1} - 2\boldsymbol{S}_{\cdot i})^{-i}\|_2} & \text{otherwise} \end{cases}$$

2 Identify active set: $\mathcal{A} = \left\{ i : \| - 2\boldsymbol{\Theta}_{\cdot i}^{-1} + 2\boldsymbol{S}_{\cdot i} + \boldsymbol{\Gamma}_{\cdot i} \|_\infty \leq \epsilon \right\}$.

3 **for** $i \in \mathcal{A}^c$ **do**

(a) Set $\boldsymbol{d} := \underset{\boldsymbol{u}}{\text{argmin}}\ q(\boldsymbol{u}; 0)$. The solution in closed form is given by (5.32).

(b) Set α_{init} to be the upper bound in (5.18) with $\boldsymbol{u} = \boldsymbol{d}, \boldsymbol{w} = \boldsymbol{e}_i$.

(c) Set $\boldsymbol{V}_{\cdot i} \leftarrow \boldsymbol{V}_{\cdot i} + \alpha \boldsymbol{d}$, where α is the output of Algorithm 9 with input $l, q, \boldsymbol{x} = \boldsymbol{0}, \boldsymbol{d}, \alpha_{\text{init}}$ where l, q are as in (5.26), (5.27).

(d) Update $\boldsymbol{\Theta}^{-1} = (\boldsymbol{V} + \boldsymbol{V}^T)^{-1}$ using the Sherman-Morrison-Woodbury rank two update scheme.

10. The computation of the subgradient, $\mathbf{\Gamma}$ in step 1 of Algorithm 8 is easy to do for hub graphical lasso and is given in closed form in step 1 of Algorithm 10. Let $\mathcal{A}$ be the current active set and $\mathcal{A}^c := \{1, \cdots, p\} \setminus \mathcal{A}$ be the complement of $\mathcal{A}$. In every iteration of Algorithm 10, we first identify $\mathcal{A}$ and only update the columns in $\mathcal{A}^c$ to obtain the next iterate $\boldsymbol{V}$. An initialization of $\mathcal{A}$ which works well in practice is $\{i : \|\boldsymbol{S}_{\cdot i}^{-i}\|_2 \leq \frac{\lambda}{2}\}$.
The active set heuristic allows for the efficient computation of the solution path (which refers to the set of solutions output by the algorithm as we sweep over λ) when combined with warm starts. Assume we have $\lambda \in [0, K]$, where $K = 2\|\boldsymbol{S} - \text{Diag}(\boldsymbol{S})\|_{\infty,2}$, where $\|\boldsymbol{X}\|_{\infty,2}$ denotes the maximum $\|.\|_2$ norm of the columns of the matrix $\boldsymbol{X}$. We then pick m equally spaced points in the interval $[0, K]$ and denote them by $a_i = K - K/m \times i$, $i = 0, 1, \ldots, m-1$. In our numerical experiments, we sequentially solve (5.21) with $\lambda = a_i$ by initializing it with the solution of (5.21) with $\lambda = a_{i+1}$ for $i = m-2, m-1, \ldots, 0$. We discuss this further in Section 5.5.3.

5.5.2 HGlasso algorithm for hub graphical lasso

While SBCD is a SVD-free approach, there have been other SVD-free approaches which have been proposed for learning sparse graphical models. Indeed, Glasso [Mazumder and Hastie, 2012] is a SVD-free algorithm proposed for graphical lasso. It is straight-forward to extend Glasso to the problem of Hub graphical lasso, as in Algorithm 11. We refer to this algorithm as the HGlasso algorithm. We note that we also use the active set heuristic in Algorithm 11 which speeds up the algorithm considerably.

Remark 39. *The key thing to note in HGlasso is that once $\boldsymbol{v}_{12}$ is updated, v_{22} is updated in step 3e) of Algorithm 11 as $v_{22} = \frac{1}{2w_{22}} + \frac{1}{2}\boldsymbol{\theta}_{12}^T(\boldsymbol{\Theta}_{11})^{-1}\boldsymbol{\theta}_{12}$ (note that $\boldsymbol{\theta}_{21} = \boldsymbol{\theta}_{12}$). This ensures the positive definiteness of the iterates. Indeed, since $\boldsymbol{\Theta}_{11} \succ 0$, we require that $\theta_{22} - \boldsymbol{\theta}_{12}^T\boldsymbol{\Theta}_{11}^{-1}\boldsymbol{\theta}_{12} > 0$. Note that $\theta_{22} = 2v_{22} = \frac{1}{w_{22}} + \boldsymbol{\theta}_{12}^T(\boldsymbol{\Theta}_{11})^{-1}\boldsymbol{\theta}_{12}$ and hence we have that $\theta_{22} - \boldsymbol{\theta}_{12}^T\boldsymbol{\Theta}_{11}^{-1}\boldsymbol{\theta}_{12} = \frac{1}{w_{22}} > 0$.*

Remark 40. *$\boldsymbol{\Theta}_{11}^{-1}$ can be easily computed from $\boldsymbol{W} = \boldsymbol{\Theta}^{-1}$. Indeed from the block matrix inversion lemma (see e.g. [Woodbury, 1950], [Petersen and Pedersen, 2012]), we have that $\boldsymbol{\Theta}_{11}^{-1} = \boldsymbol{W}_{11} - \boldsymbol{w}_{12}\boldsymbol{w}_{12}^T/w_{22}$.*

Algorithm 11: The HGlasso algorithm for hub graphical lasso (5.20).

Input: Sample covariance matrix $\boldsymbol{S}$, λ, optimality tolerance $\epsilon > 0$

Output: ϵ-approximate optimal solution $\boldsymbol{V}$

Initialize: $\boldsymbol{V} = (\text{Diag}(\boldsymbol{S}))^{-1}/2, \boldsymbol{\Theta} = (\text{Diag}(\boldsymbol{S}))^{-1}, \boldsymbol{W} = \boldsymbol{\Theta}^{-1} = \text{Diag}(\boldsymbol{S})$ and active set, $\mathcal{A} = \{\}$.

while $\mathcal{A}^c \neq \{\}$ **do**

1 **for** $i = 1, 2, \ldots, p$ **do**

if $\boldsymbol{v}^{-i} \neq \boldsymbol{0}$ **then**

$\boldsymbol{\Gamma}_{\cdot i} = \lambda \boldsymbol{v}^{-i} / \|\boldsymbol{v}^{-i}\|_2$

else

$$\boldsymbol{\Gamma}_{\cdot i} = \begin{cases} (2\boldsymbol{W}_{\cdot i} - 2\boldsymbol{S}_{\cdot i})^{-i} & \text{if } \|(2\boldsymbol{W}_{\cdot i} - 2\boldsymbol{S}_{\cdot i})^{-i}\|_2 \leq \lambda \\ \lambda \frac{(2\boldsymbol{W}_{\cdot i} - 2\boldsymbol{S}_{\cdot i})^{-i}}{\|(2\boldsymbol{W}_{\cdot i} - 2\boldsymbol{S}_{\cdot i})^{-i}\|_2} & \text{otherwise} \end{cases}$$

2 Identify active set: $\mathcal{A} = \{i : \| - 2\boldsymbol{W}_{\cdot i} + 2\boldsymbol{S}_{\cdot i} + \boldsymbol{\Gamma}_{\cdot i}\|_\infty \leq \epsilon\}$.

3 **for** $i \in \mathcal{A}^c$ **do**

(a) Permute rows and columns of $\boldsymbol{V}$ so that the ith column of $\boldsymbol{V}$ before permutation is now the last column of $\boldsymbol{V}$. Do the same for $\boldsymbol{\Theta}, \boldsymbol{W}$.
Let $\boldsymbol{V} = \begin{bmatrix} \boldsymbol{V}_{11} & \boldsymbol{v}_{12} \\ \boldsymbol{v}_{21}^T & v_{22} \end{bmatrix}$ where $\boldsymbol{V}_{11} \in \mathbb{R}^{p-1 \times p-1}, \boldsymbol{v}_{12} \in \mathbb{R}^{p-1}, \boldsymbol{v}_{22} \in \mathbb{R}$.
Let $\boldsymbol{\Theta}, \boldsymbol{W} = \boldsymbol{\Theta}^{-1}, \boldsymbol{S}$ have a similar block matrix decomposition.

(b) Fix $\boldsymbol{V}_{11}, \boldsymbol{v}_{21}, \boldsymbol{v}_{22}$ and optimize for $\boldsymbol{v}_{12}$:
$\hat{\boldsymbol{u}} = \underset{\boldsymbol{u}}{\text{argmin}}\, \boldsymbol{u}^T(\boldsymbol{\Theta}_{11})^{-1}\boldsymbol{u} + \boldsymbol{u}^T(2\boldsymbol{s}_{12} + 2w_{22}(\boldsymbol{\Theta}_{11})^{-1}\boldsymbol{v}_{21}) + \lambda\|\boldsymbol{u}\|_2$

(c) $\boldsymbol{v}_{12} = \hat{\boldsymbol{u}} s_{22}$.

(d) $\boldsymbol{\theta}_{12} = \boldsymbol{v}_{12} + \boldsymbol{v}_{21}$, $\boldsymbol{\theta}_{21} = \boldsymbol{v}_{12} + \boldsymbol{v}_{21}$.

(e) $v_{22} = \frac{1}{2w_{22}} + \frac{1}{2}\boldsymbol{\theta}_{21}^T(\boldsymbol{\Theta}_{11})^{-1}\boldsymbol{\theta}_{12}$.

(f) Update $\boldsymbol{W} = \boldsymbol{\Theta}^{-1}$ using the Sherman-Morrison-Woodbury identity.

(g) Unpermute rows and columns of $\boldsymbol{V}, \boldsymbol{W}, \boldsymbol{\Theta}$.

Remark 41. *In the implementation of algorithm 11, we solve the sub problem in step 2b) using the proximal gradient method.*

Remark 42. *Whenever a column of* $\boldsymbol{V}$ *is updated, a row and a column of* $\boldsymbol{\Theta} = \boldsymbol{V} + \boldsymbol{V}^T$ *is updated. This translates to a rank two update to* $\boldsymbol{\Theta}$*. Thus we can update* $\boldsymbol{W} = \boldsymbol{\Theta}^{-1}$ *in* $O(p^2)$ *by applying the Sherman-Morrison-Woodbury update scheme (see e.g. [Woodbury, 1950]).*

5.5.3 Experimental results on synthetic data

We generated an inverse covariance matrix $\boldsymbol{\Theta}_0$ with hubs similar to the Erdos-Renyi setup described in Section 3.4.4 of Chapter 3 of this thesis. For any given problem dimension, p, we let the number of hubs, k in $\boldsymbol{\Theta}_0$ equal $0.03 \times p$. We generated the sample covariance matrix $\boldsymbol{S}$, from $\boldsymbol{\Theta}_0$ using a sample size, $n = 0.1 \times p \times k$.
We use two metrics for comparison of algorithms: F-score and run-time in seconds. Let $\hat{\boldsymbol{\Theta}} = \hat{\boldsymbol{V}} + \hat{\boldsymbol{V}}^T$ be the algorithm solution. Let $\mathcal{T}_0$ be the support or set of non-zero entries of $\boldsymbol{\Theta}_0$. Let $\hat{\mathcal{T}}$ be the support of $\hat{\boldsymbol{\Theta}}$. Then the precision is given by $\frac{|\hat{\mathcal{T}} \cap \mathcal{T}_0|}{|\hat{\mathcal{T}}|}$ and the recall is given by $\frac{|\hat{\mathcal{T}} \cap \mathcal{T}_0|}{|\mathcal{T}_0|}$. The F-score equals the harmonic mean of the precision (P) and recall (R), i.e., F-score $= \frac{2 \times P \times R}{P+R}$. If the solution output by an algorithm has an F-score of 0, this implies that the algorithm has either a zero precision or zero recall. On the other hand, an F-score of 1 implies that both the precision and recall equal 1. Thus, higher the F-score, the better the performance of the algorithm on this metric. We now describe our experimental results. All the algorithms in the results of the following sections are implemented in C and use the BLAS and LAPACK linear algebra libraries. The experiments were run on MacBook Air with a 1.4 GHz Intel Core i5 processor and 4GB RAM.

1. In the first set of experiments in Figure 5.3, we plot the metrics against a range of regularization parameters λ for different choices of p. In particular, we consider $p = 200, 500, 1000$. We vary λ as follows: Let $\lambda_{\max} = 2\|S - \text{Diag}\|_{\infty,2}$, where we define the $\|\boldsymbol{X}\|_{\infty,2}$ norm as the maximum ℓ_2 norm of the columns of $\boldsymbol{X}$. We then generate a λ grid by dividing the interval $[0, \lambda_{\max}]$ into 30 equal parts. Denote these points by $\{\lambda_i\}$, $i = 1, \ldots, 30$. The solution path for a given algorithm refers to the set of solutions output by the algorithm for regularization parameters λ_1 through λ_{30}. We use warm-starts to speed up the computation of

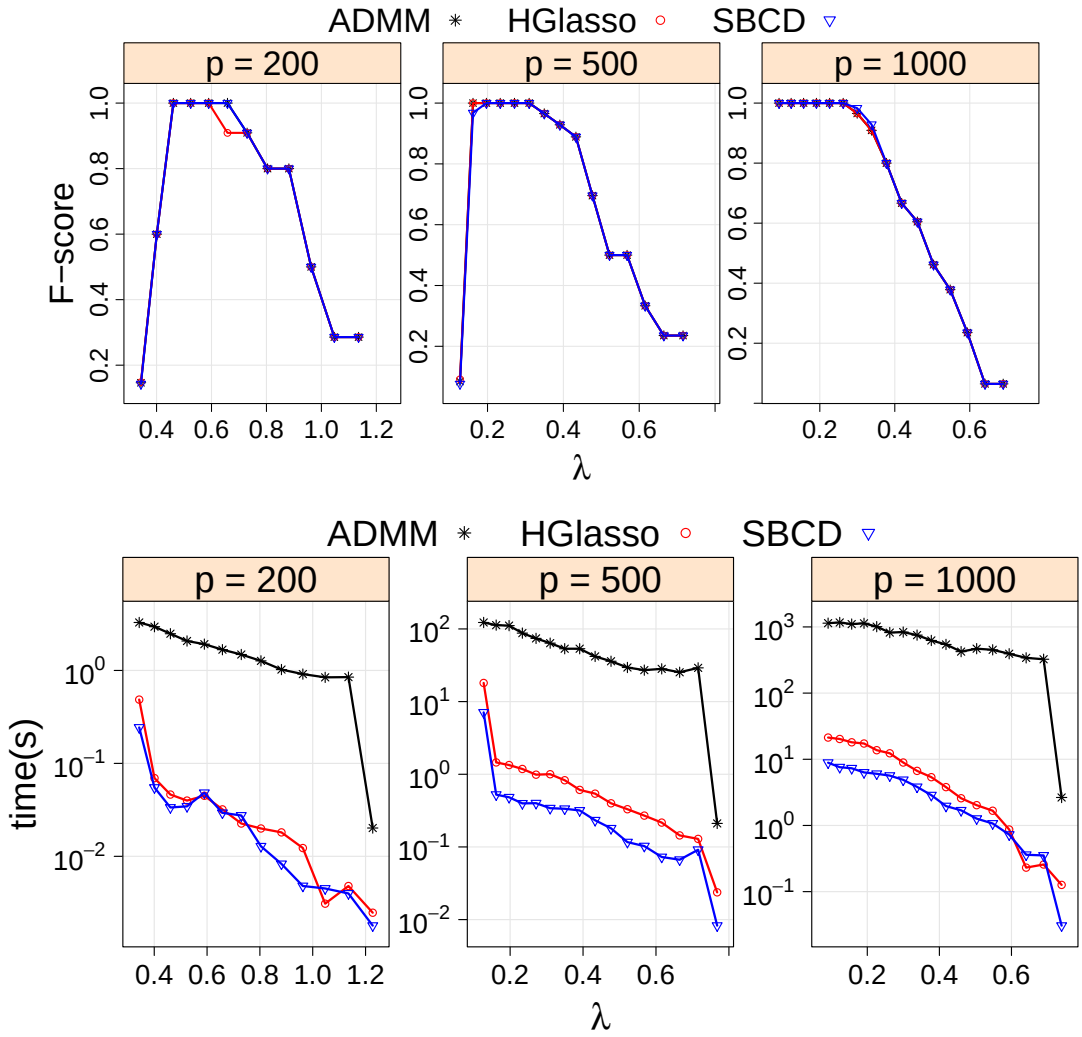


Figure 5.3: The figures compare three algorithms, ADMM, HGlasso and SBCD over two metrics, F-score and run time. The top panel shows F-score vs λ comparisons for $p = 200, 500, 1000$ which match for all of the algorithms as expected. The top panel is useful in deciding the range of λ in which the F-score is high. The bottom panel shows the run time comparison against λ for $p = 200, 500, 1000$. SBCD and HGlasso take much less time than ADMM in the range of interest where the F-score is high.

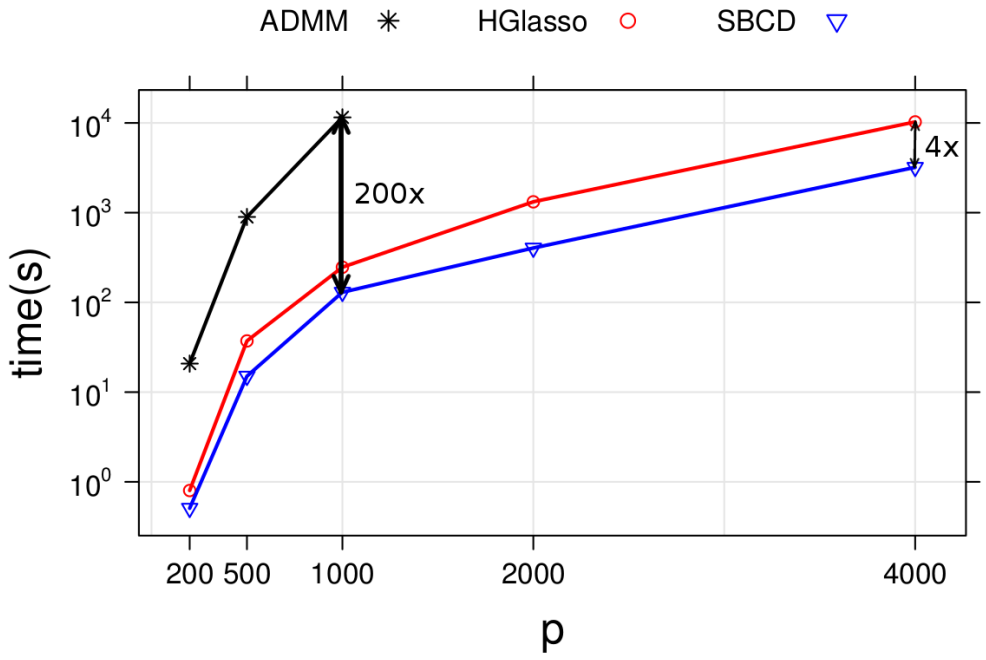


Figure 5.4: This plot compares the total solution path time of SBCD , HGlasso and ADMM across different problem dimensions p. The plot shows that SBCD is about 4 times faster than HGlasso for $p = 4000$ and about 200 times faster than ADMM for $p = 1000$. We compute the solution path time of ADMM only for $p \leq 1000$ due to ADMM taking a lot of time to run. From the plot, we see that the SBCD for $p = 4000$ is 4 times faster than ADMM for $p = 1000$.

the solution path for the algorithms. I.e. we start by running an algorithm for λ_{30} and then successively use the solution for λ_{i+1} to initialize the algorithm for the computation of solution for λ_i. Since the solution from a previous λ_i will be a good starting point than any random starting point, the computation of the solution path gets sped up when we do warm-starts. If for some λ_i, the solution output by the algorithm has more than 10% non-zero columns, then we stop the computation of the solution path at this point. This is done because, with more than 10% non-zero columns, the solution is no longer sparse. Since we are interested in structured sparse solutions, computing the remaining solution path wouldn't be of use. We set the optimality tolerance ϵ for all the algorithms to be 10^{-4}.

As can be seen from the top panel of Figure 5.3, the F-scores match up for ADMM, HGlasso and SBCD. Indeed this makes sense, as the three algorithms are solving the same HGL formulation. Another important thing to note is that for $p = 200, 500, 1000$, the range of λ captures the good regime in which the F-score is high. In the bottom panel of Figure 5.3, we see that in precisely this good regime, both HGlasso and SBCD are 150 times faster than the SVD based ADMM algorithm for $p = 500, 1000$.

2. In the second set of experiments in Figure 5.4, we compare the solution path time of ADMM, HGlasso and SBCD against the problem dimension, p. The solution path time for a given p and a given algorithm, is the total time taken by the algorithm to compute the solution path (i.e. the total run time across λ in Figure 5.4).

   As can be seen from Figure 5.4, both SBCD and HGlasso are 200 times faster than ADMM for a problem of dimension $p = 1000$. Also as the problem dimension increases, we see that SBCD has an increasingly faster solution path time as compared to HGlasso. For $p = 4000$, SBCD is 4 times faster than HGlasso. Another interesting thing to note is that SBCD for $p = 4000$ takes only 25% of the run time that ADMM takes for $p = 1000$. Although we haven't plotted the solution path time for ADMM for $p = 4000$, we expect it to be at least an order of magnitude slower than SBCD. We believe the gains in run times are due to a combination of SBCD not relying on SVD and also the ability of SBCD to exploit the sparse solution structure (for example, through the active

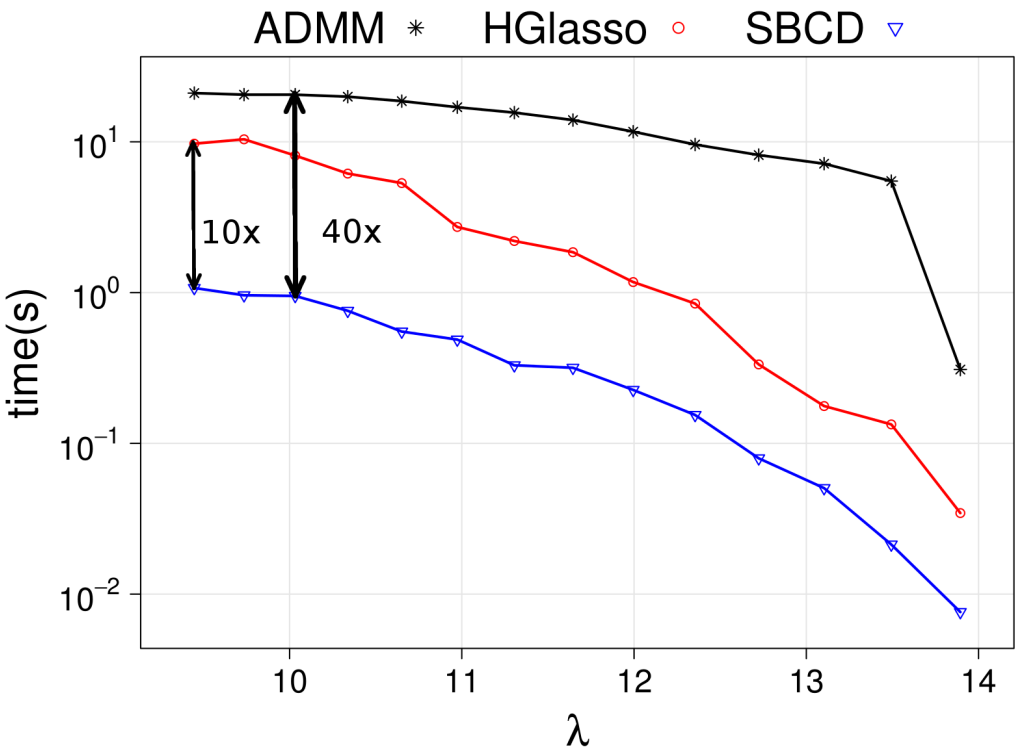


Figure 5.5: Run time (in seconds) comparison of ADMM, HGlasso and SBCD on a real data set with $p = 500, n = 401$. SBCD is around 10 times faster than HGlasso and about 40 times faster than ADMM for a wide range of λ.

set heuristic) better than the ADMM algorithm.

We note that our SBCD algorithm can be combined with screening rules from the earlier sections of this chapter to make the computation of the solution paths even faster.

5.5.4 Experimental results on real data

We next consider a real data experiment on a gliblastoma mutliforme (GBM) cancer data set over $p = 500$ genes. The sample covariance matrix S is generated from 401 samples using the procedure outlined in Section 3.4.5 of Chapter 3 of this thesis. Figure 5.5 shows the run-time results on the real data. We range λ similar to the description in synthetic data experiment in the previous section. We note that SBCD is at least 40 times faster than ADMM across a range of λ and around 10 times faster than HGlasso. We also note that we can extend SBCD to the more general HGL formulation in (3.6) in Chapter 3. We apply this extended SBCD to the real data example in Section 3.4.5 and with the same choice of regularization parameters $\lambda_1 = 0.6, \lambda_2 = 0.4, \lambda_3 = 6.5$, we observe that SBCD is significantly faster than ADMM.

5.6 SBCD for hub covariance selection

A simpler version of the hub covariance selection problem [Tan et al., 2014] (where we ignore the regularization parameters that promote sparsity) is as follows:

$$\begin{array}{ll} \underset{\boldsymbol{\Theta} \succ 0}{\text{minimize}} & \frac{1}{2}\|\boldsymbol{\Theta} - \boldsymbol{S}\|_F^2 + \lambda \|\boldsymbol{V} - \text{Diag}(\boldsymbol{V})\|_{1,2} \\ \text{s.t.} & \boldsymbol{\Theta} = \boldsymbol{V} + \boldsymbol{V}^T. \end{array} \tag{5.34}$$

[Tan et al., 2014] proposed an ADMM algorithm to solve (5.34). However this involves an eigen-value decomposition computation in each iteration. In this section, we describe the SBCD algorithm for hub covariance selection.

5.6.1 SBCD algorithm

The problem (5.34) can be reformulated as follows:

$$\underset{\boldsymbol{V}+\boldsymbol{V}^T\succ 0}{\text{minimize}} \quad \tfrac{1}{2}\|\boldsymbol{V}+\boldsymbol{V}^T-\boldsymbol{S}\|_F^2+\lambda\|\boldsymbol{V}-\text{Diag}(\boldsymbol{V})\|_{1,2} \tag{5.35}$$

Due to the simplicity of the objective in (5.35), instead of solving a first order approximation to (5.35) in each column of $\boldsymbol{V}$ while fixing all other columns (or a coordinate gradient descent step) as in Step 3a) of Algorithm 8, we actually solve (5.35) in each column exactly (keeping all other columns fixed). The update for the ith column is as follows:

$$\min_{\boldsymbol{u}} \frac{1}{2}\|\boldsymbol{V}+\boldsymbol{V}^T+\boldsymbol{u}\boldsymbol{e}_i^T+\boldsymbol{e}_i\boldsymbol{u}^T-\boldsymbol{S}\|_F^2+\lambda\|\boldsymbol{V}+\boldsymbol{u}\boldsymbol{e}_i^T-\text{Diag}(\boldsymbol{V}+\boldsymbol{u}\boldsymbol{e}_i^T)\|_{1,2} \tag{5.36}$$

Expanding (5.36), we have,

$$\begin{aligned}\underset{\boldsymbol{u}}{\text{minimize}} \quad & \|(\boldsymbol{V}+\boldsymbol{V}^T-\boldsymbol{S})_{\cdot i}+\boldsymbol{u}\|_2^2-(2\boldsymbol{V}_{ii}-\boldsymbol{S}_{ii}+\boldsymbol{u}_i)^2+\\ & \tfrac{1}{2}(2\boldsymbol{V}_{ii}+2\boldsymbol{u}_i-\boldsymbol{S}_{ii})^2+\lambda\|\boldsymbol{V}_{i}+\boldsymbol{u}-(\boldsymbol{V}_{ii}+\boldsymbol{u}_i)\boldsymbol{e}_i\|_2,\end{aligned} \tag{5.37}$$

where we have used the fact that $\|(\boldsymbol{V}+\boldsymbol{V}^T-\boldsymbol{S})_{i,\cdot}+u^T\|_2^2=\|(\boldsymbol{V}+\boldsymbol{V}^T-\boldsymbol{S})_{\cdot i}+\boldsymbol{u}\|_2^2$.

Let,

$$\begin{aligned} l(\boldsymbol{u}) &= \|(\boldsymbol{V}+\boldsymbol{V}^T-\boldsymbol{S})_{\cdot i}+\boldsymbol{u}\|_2^2-(2\boldsymbol{V}_{ii}-\boldsymbol{S}_{ii}+\boldsymbol{u}_i)^2+\tfrac{1}{2}(2\boldsymbol{V}_{ii}+2\boldsymbol{u}_i-\boldsymbol{S}_{ii})^2\\ q(\boldsymbol{u}) &= \lambda\|\boldsymbol{V}_{i}+\boldsymbol{u}-(\boldsymbol{V}_{ii}+\boldsymbol{u}_i)\boldsymbol{e}_i\|_2\end{aligned} \tag{5.38}$$

Note that (5.37) is equivalent to:

$$\underset{\boldsymbol{u}}{\text{minimize}}\, l(\boldsymbol{u})+q(\boldsymbol{u}) \tag{5.39}$$

Let $\boldsymbol{u}^{-i}$ denote a copy of the vector $\boldsymbol{u}$ with the ith coordinate set to zero. Then the optimization in (5.39) can be broken down into optimizing for $\boldsymbol{u}_i$ and $\boldsymbol{u}^{-i}$. Let $\hat{\boldsymbol{u}}$ denote the optimal solution to (5.37). Note that the optimal $\hat{\boldsymbol{u}}_i=\frac{1}{2}(\boldsymbol{S}_{ii}-2\boldsymbol{V}_{ii})$. Let $\boldsymbol{b}=(\boldsymbol{V}+\boldsymbol{V}^T-\boldsymbol{S})_{\cdot i}$, $\boldsymbol{a}=\boldsymbol{V}_{\cdot i}$ and $\boldsymbol{y}=\boldsymbol{a}^{-i}+\boldsymbol{u}^{-i}$. Then the optimization for $\boldsymbol{u}^{-i}$ is equivalent to:

$$\underset{y}{\text{minimize}}\, \|\boldsymbol{b}^{-i}+\boldsymbol{y}-\boldsymbol{a}^{-i}\|_2^2+\lambda\|\boldsymbol{y}\|_2. \tag{5.40}$$

The optimal solution to (5.40) is given by

$$\begin{aligned} \hat{\boldsymbol{y}} &= \boldsymbol{u}^{-i} + \boldsymbol{a}^{-i} \\ &= \max\left(1 - \frac{\lambda}{2\|\boldsymbol{a}^{-i} - \boldsymbol{b}^{-i}\|_2}, 0\right)(\boldsymbol{a}^{-i} - \boldsymbol{b}^{-i}). \end{aligned} \tag{5.41}$$

Hence the solution $\hat{\boldsymbol{u}}$ to (5.39) is given by:

$$\hat{\boldsymbol{u}}_j = \begin{cases} \frac{1}{2}(\boldsymbol{S}_{ii} - 2\boldsymbol{V}_{ii}) & \text{if } j = i \\ \hat{\boldsymbol{y}}_j - \boldsymbol{a}_j & \text{otherwise} \end{cases}, \tag{5.42}$$

where $\hat{\boldsymbol{y}}$ is as in (5.41).

Remark 43. *We note that instead of solving the sub-problem (5.36) through proximal gradient descent step as outlined in step 2 of Algorithm 5.11, because of the simplicity of the problem, we actually solve (5.36) exactly and the update is given by (5.41). We therefore don't need to use Armijo rule.*

Just as for hub graphical lasso, in the implementation of Algorithm 12, we maintain and update the inverse $\boldsymbol{W} = (\boldsymbol{\Theta})^{-1}$. Every time a column of $\boldsymbol{V}$ is updated, $\boldsymbol{\Theta}$ incurs a rank-two update and thus by using the Sherman-Morrison-Woodbury identity (see e.g. [Woodbury, 1950]), we can update $\boldsymbol{W}$ efficiently in $O(p^2)$.

5.6.2 Experimental results on synthetic data

We generated a covariance matrix $\boldsymbol{\Theta}_0$ with hubs similar to the Erdos-Renyi setup described in Section 4.2 of Tan et al. [2014]. For any given problem dimension, p, we let the number of hubs, k in $\boldsymbol{\Theta}_0$ equal $0.03 \times p$. We generated the sample covariance matrix S, from $\boldsymbol{\Theta}_0$ using a sample size, $n = 0.1 \times p \times k$.
We use two metrics for comparison of algorithms: F-score and run-time in seconds. The F-score equals the harmonic mean of the precision (P) and recall (R), i.e., F-score $= \frac{2 \times P \times R}{P+R}$. As mentioned earlier, higher the F-score, the better the performance of the algorithm on this metric. We now describe our experimental results.

1. In Figure 5.6, we plot the metrics against a range of regularization parameters λ for different choices of p. In particular, we consider $p = 500, 1000$. We vary

Algorithm 12: SBCD with active set heuristic for Hub covariance selection

Input: Sample covariance matrix, $\boldsymbol{S}$, λ and optimality tolerance ϵ
Output: ϵ-approximate optimal solution $\hat{\boldsymbol{V}}$
Initialize: $\boldsymbol{V} = \text{Diag}(\boldsymbol{S})/2$, $\boldsymbol{\Theta} = \boldsymbol{V} + \boldsymbol{V}^T = \text{Diag}(\boldsymbol{S})$,
$\boldsymbol{W} = (\boldsymbol{\Theta})^{-1} = (\text{Diag}(\boldsymbol{S}))^{-1}$ and active set $\mathcal{A} = \{\}$.
while $\mathcal{A} \neq \{\}$ **do**

1 **for** $i = 1, 2, \ldots, p$ **do**
  if $\boldsymbol{v}^{-i} \neq \boldsymbol{0}$ **then**
    $\boldsymbol{\Gamma}_{\cdot i} = \lambda \boldsymbol{v}^{-i} / \|\boldsymbol{v}^{-i}\|_2$
  else
    $$\boldsymbol{\Gamma}_{\cdot i} = \begin{cases} (2\boldsymbol{\Theta}_{\cdot i} - 2\boldsymbol{S}_{\cdot i})^{-i} & \text{if } \|(2\boldsymbol{\Theta}_{\cdot i} - 2\boldsymbol{S}_{\cdot i})^{-i}\|_2 \leq \lambda \\ \lambda \frac{(2\boldsymbol{\Theta}_{\cdot i} - 2\boldsymbol{S}_{\cdot i})^{-i}}{\|(2\boldsymbol{\Theta}_{\cdot i} - 2\boldsymbol{S}_{\cdot i})^{-i}\|_2} & \text{otherwise} \end{cases}$$

2 Identify active set: $\mathcal{A} = \{i : \| - 2\boldsymbol{\Theta}_{\cdot i} + 2\boldsymbol{S}_{\cdot i} + \boldsymbol{\Gamma}_{\cdot i}\|_\infty \leq \epsilon\}$.

3 **for** $i \in \mathcal{A}^c$ **do**

(a) Set $\boldsymbol{d} = \hat{\boldsymbol{u}}$, where the $\hat{\boldsymbol{u}}$ is given by (5.42).
(b) Choose a step size α that satisfies (5.18) with $\boldsymbol{u} = \boldsymbol{d}, \boldsymbol{w} = \boldsymbol{e}_i$, $\boldsymbol{\Theta}^{-1} = \boldsymbol{W}$.
(c) $\boldsymbol{V}_{\cdot i} = \boldsymbol{V}_{\cdot i} + \alpha \boldsymbol{d}$.
(d) $\boldsymbol{\Theta} = \boldsymbol{V} + \boldsymbol{V}^T$.
(e) Update $\boldsymbol{W} = (\boldsymbol{\Theta})^{-1}$ using the Sherman-Morrison-Woodbury identity.

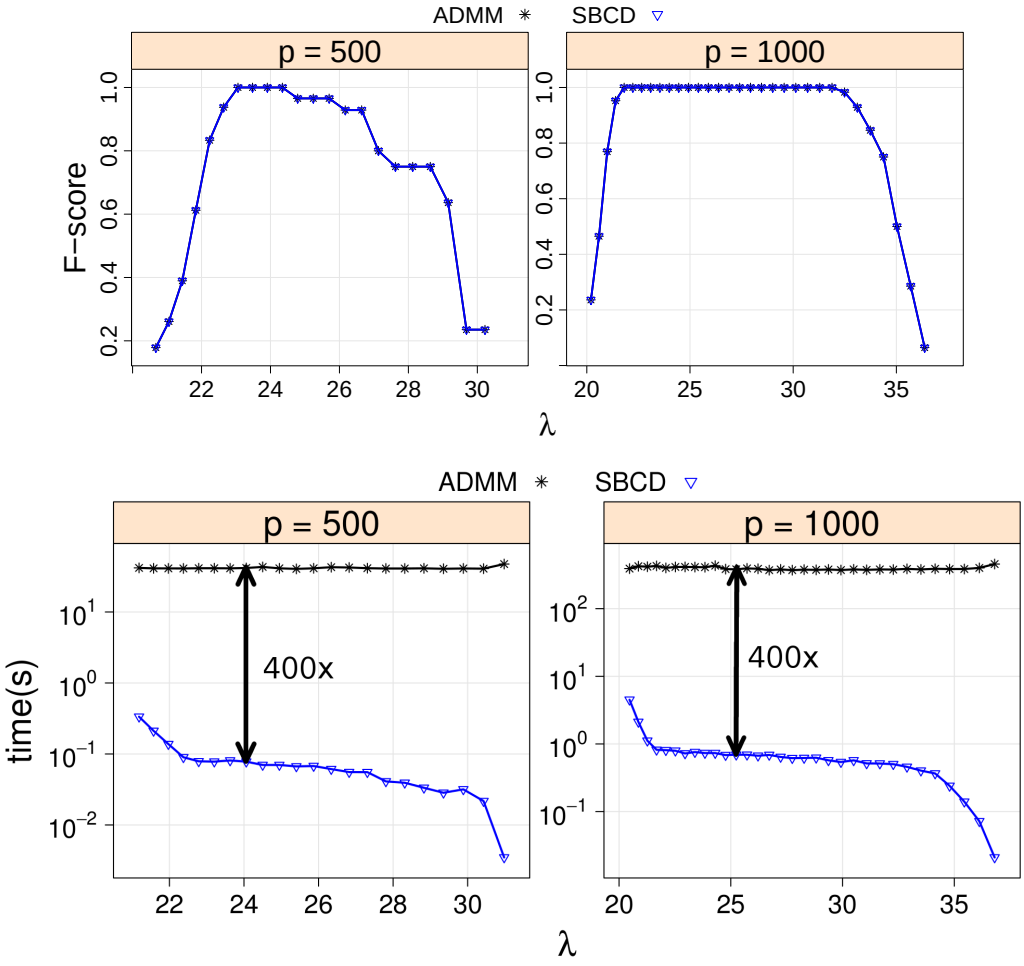


Figure 5.6: Comparison of ADMM and SBCD algorithms on the metrics of F-score and run time in seconds. The top panel shows F-score comparison of algorithms for $p = 500, 1000$ against λ. As expected, we see that the F-scores for the two algorithms match over the range of λ. The top panel is useful in deciding the range of λ in which the F-score is high. The bottom panel shows run time comparison of algorithms for $p = 500, 1000$ against λ. In the range of interest of λ, we see that SBCD is consistently faster than ADMM, sometimes by a factor of 400. There is an increasing trend of the computation time for SBCD as λ decreases, while ADMM's computation time stays almost the same for all λ.

λ as follows: Let $\lambda_{\max} = 2\|S - \text{Diag}\|_{\infty,2}$, where we define the $\|X\|_{\infty,2}$ norm as the maximum ℓ_2 norm of the columns of the X. We then generate a λ grid by dividing the interval $[0, \lambda_{\max}]$ into 30 equal parts. Denote these points by $\{\lambda_i\}$, $i = 1, \ldots, 30$. We compute the solution path starting at λ_{30} and ending at λ_1. We use warmstarts to speed up the computation of the solution path. If for some λ_i, the solution output by the algorithm has more than 10% non-zero columns, then we stop the computation of the solution path at this point. We set the optimality tolerance for all the algorithms to be 10^{-4}.
As can be seen from the top panel of Figure 5.6, the F-scores match up for ADMM and SBCD. Indeed this makes sense, as the two algorithms are solving the same HGL formulation. However, the top panel reveals the regime of λ for which the F-score is high. In the bottom panel of Figure 5.6, we see that in precisely this good regime, SBCD is up to 400 times faster than the SVD based ADMM algorithm for a significant range of λ values.

2. We next compare the solution path time of ADMM and SBCD in Figure 5.7. As can be seen from the figure, SBCD is 900 times faster than ADMM for $p = 1000$. Another interesting thing to note is that the solution path time taken by SBCD for $p = 4000$ is 8 times less than the time taken by ADMM for $p = 1000$. This illustrates that SBCD can be significantly faster than an SVD-based ADMM algorithm. We believe the gains in run times are due to a combination of SBCD not relying on SVD and also the ability of SBCD to exploit the sparse solution structure as compared to the ADMM algorithm.

Our experimental results show that SBCD is indeed much faster than an SVD-based ADMM algorithm. However, we note that there could be potentially much faster algorithms than ADMM for the hub covariance selection problem (aside from our algorithm) and we leave more extensive comparison of algorithms for future work.

5.7 Summary and conclusions

In the first part of this chapter, we discussed screening rules to detect block-diagonal structure (up to a permutation) in the optimal solution and used it to speed up the computation of our algorithms. This is done by breaking down the problem into possibly many sub-problems, one corresponding to each block identified in the optimal

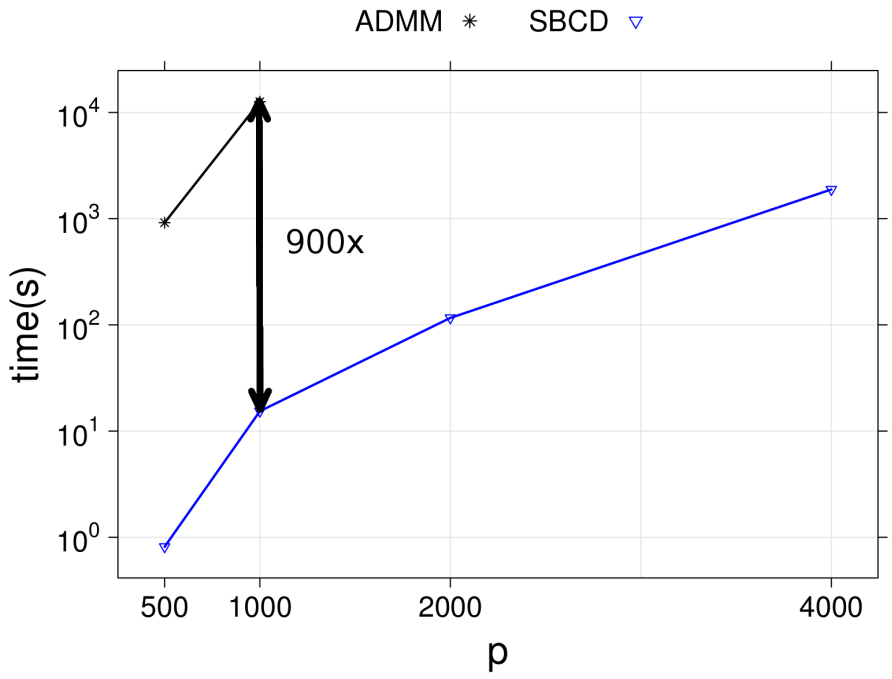


Figure 5.7: Solution path time of ADMM and SBCD vs p. SBCD is about 900 times faster than ADMM for $p = 1000$. Also note that SBCD for $p = 4000$ is 8 times for faster than ADMM for $p = 1000$.

solution structure. We observe speed-ups by a factor of 2 to 5 depending on the choice of regularization parameter.

In the second part of the chapter, we proposed a fast block coordinate descent algorithm, SBCD with an active set heuristic that does not rely on expensive linear algebra computations such as eigen-decomposition or Cholesky factorization to ensure positive definiteness. SBCD is able to better exploit structure in the problems of hub graphical lasso and hub covariance selection than a standard SVD based algorithm such as ADMM. Indeed, in our preliminary numerical experiments, we observe that SBCD is much faster than the ADMM algorithm for these problems over a range of regularization parameter λ for which the F-score is close to 1. Overall, our numerical results though preliminary, demonstrate a significant benefit to the use of an SVD-free algorithm over an SVD-based algorithm (specifically ADMM). However, there is a lot of scope for more extensive comparison of algorithms for the hub covariance selection problem. We also leave for future work the extension of our SVD-free approach to other interesting problems involving structured graphical models such as the latent variable selection problem discussed in [Chandrasekaran et al., 2012b].

5.8 Proofs

Appendix A: Conditions for HGL Solution to be Block-Diagonal

We begin by introducing some notation. For $\boldsymbol{V} \in \mathbb{R}^{p\times p}$, let $\|\boldsymbol{V}\|_{u,v}$ denote the ℓ_u/ℓ_v norm of a matrix $\boldsymbol{V}$, i.e. $\|\boldsymbol{V}\|_{u,v} = \|\boldsymbol{w}\|_u$ where $\boldsymbol{w}_i = \|\boldsymbol{V}_{\cdot i}\|_v \ \forall i$. For instance, $\|\boldsymbol{V}\|_{1,q} = \sum_{j=1}^{p} \|\boldsymbol{V}_{\cdot j}\|_q$. We define the support of a matrix $\boldsymbol{\Theta}$ as follows: $\text{supp}(\boldsymbol{\Theta}) = \{(i,j) : \boldsymbol{\Theta}_{ij} \neq 0\}$. We say that $\boldsymbol{\Theta}$ is supported on a set $\mathcal{G}$ if $\text{supp}(\boldsymbol{\Theta}) \subseteq \mathcal{G}$. Let $\{C_1, \ldots, C_K\}$ be a partition of the index set $\{1, \ldots, p\}$, and let $\mathcal{T} = \cup_{k=1}^{K}\{C_k \times C_k\}$. We let $\mathbf{A}_{\mathcal{T}}$ denote the restriction of the matrix $\mathbf{A}$ to the set $\mathcal{T}$: that is, $(\mathbf{A}_{\mathcal{T}})_{ij} = 0$ if $(i,j) \notin \mathcal{T}$ and $(\mathbf{A}_{\mathcal{T}})_{ij} = A_{ij}$ if $(i,j) \in \mathcal{T}$. Note that any matrix supported on $\mathcal{T}$ is block-diagonal with K blocks, subject to some permutation of its rows and columns. Also, let $\boldsymbol{S}_{\max} = \max_{(i,j)\in\mathcal{T}^c} |\boldsymbol{S}_{ij}|$. Define

$$\begin{aligned} \tilde{\mathbf{P}}(\boldsymbol{\Theta}) \ = \ \min_{\mathbf{V},\mathbf{Z}} \quad & \|\boldsymbol{Z} - \text{Diag}(\boldsymbol{Z})\|_1 + \hat{\lambda}_2\|\boldsymbol{V} - \text{Diag}(\boldsymbol{V})\|_1 + \\ & \hat{\lambda}_3\|\boldsymbol{V} - \text{Diag}(\boldsymbol{V})\|_{1,q} \\ \text{subject to} \quad & \boldsymbol{\Theta} = \mathbf{Z} + \mathbf{V} + \mathbf{V}^T, \end{aligned} \tag{5.43}$$

where $\hat{\lambda}_2 = \frac{\lambda_2}{\lambda_1}$ and $\hat{\lambda}_3 = \frac{\lambda_3}{\lambda_1}$. Then, optimization problem (3.7) is equivalent to

$$\underset{\boldsymbol{\Theta} \in \mathcal{S}}{\text{minimize}} \quad -\log\det(\boldsymbol{\Theta}) + \langle \boldsymbol{\Theta}, \mathbf{S} \rangle + \lambda_1 \tilde{\mathbf{P}}(\boldsymbol{\Theta}), \tag{5.44}$$

where $\mathcal{S} = \{\boldsymbol{\Theta} : \boldsymbol{\Theta} \succ 0, \boldsymbol{\Theta} = \boldsymbol{\Theta}^T\}$.

Proof of Theorem 1 (Sufficient Condition)

Proof. First, we note that if $(\boldsymbol{\Theta}, \mathbf{V}, \mathbf{Z})$ is a feasible solution to (3.7), then $(\boldsymbol{\Theta}_{\mathcal{T}}, \mathbf{V}_{\mathcal{T}}, \mathbf{Z}_{\mathcal{T}})$ is also a feasible solution to (3.7). Assume that $(\boldsymbol{\Theta}, \mathbf{V}, \mathbf{Z})$ is not supported on $\mathcal{T}$. We want to show that the objective value of (3.7) evaluated at $(\boldsymbol{\Theta}_{\mathcal{T}}, \boldsymbol{V}_{\mathcal{T}}, \boldsymbol{Z}_{\mathcal{T}})$ is smaller than the objective value of (3.7) evaluated at $(\boldsymbol{\Theta}, \mathbf{V}, \mathbf{Z})$. By Fischer's inequality [Horn and Johnson, 1990],

$$-\log\det(\boldsymbol{\Theta}) \geq -\log\det(\boldsymbol{\Theta}_{\mathcal{T}}).$$

Therefore, it remains to show that

$$\langle \boldsymbol{\Theta}, \mathbf{S} \rangle + \lambda_1 \|\boldsymbol{Z} - \mathrm{Diag}(\boldsymbol{Z})\|_1 + \lambda_2 \|\boldsymbol{V} - \mathrm{Diag}(\boldsymbol{V})\|_1 + \lambda_3 \|\boldsymbol{V} - \mathrm{Diag}(\boldsymbol{V})\|_{1,q} >$$
$$\langle \boldsymbol{\Theta}_{\mathcal{T}}, \mathbf{S} \rangle + \lambda_1 \|\boldsymbol{Z}_{\mathcal{T}} - \mathrm{Diag}(\boldsymbol{Z}_{\mathcal{T}})\|_1 + \lambda_2 \|\boldsymbol{V}_{\mathcal{T}} - \mathrm{Diag}(\boldsymbol{V}_{\mathcal{T}})\|_1 + \lambda_3 \|\boldsymbol{V}_{\mathcal{T}} - \mathrm{Diag}(\boldsymbol{V}_{\mathcal{T}})\|_{1,q},$$

or equivalently, that

$$\langle \boldsymbol{\Theta}_{\mathcal{T}^c}, \mathbf{S} \rangle + \lambda_1 \|\boldsymbol{Z}_{\mathcal{T}^c}\|_1 + \lambda_2 \|\boldsymbol{V}_{\mathcal{T}^c}\|_1 + \lambda_3 (\|\boldsymbol{V} - \mathrm{Diag}(\boldsymbol{V})\|_{1,q} - \|\boldsymbol{V}_{\mathcal{T}} - \mathrm{Diag}(\boldsymbol{V}_{\mathcal{T}})\|_{1,q}) > 0.$$

Since $\|\boldsymbol{V} - \mathrm{Diag}(\boldsymbol{V})\|_{1,q} \geq \|\boldsymbol{V}_{\mathcal{T}} - \mathrm{Diag}(\boldsymbol{V}_{\mathcal{T}})\|_{1,q}$, it suffices to show that

$$\langle \boldsymbol{\Theta}_{\mathcal{T}^c}, \mathbf{S} \rangle + \lambda_1 \|\boldsymbol{Z}_{\mathcal{T}^c}\|_1 + \lambda_2 \|\boldsymbol{V}_{\mathcal{T}^c}\|_1 > 0. \tag{5.45}$$

Note that $\langle \boldsymbol{\Theta}_{\mathcal{T}^c}, \mathbf{S} \rangle = \langle \boldsymbol{\Theta}_{\mathcal{T}^c}, \mathbf{S}_{\mathcal{T}^c} \rangle$. By the sufficient condition, $\boldsymbol{S}_{\max} < \lambda_1$ and $2\boldsymbol{S}_{\max} < \lambda_2$.

In addition, we have that

$$\begin{aligned}
|\langle \boldsymbol{\Theta}_{\mathcal{T}^c}, \mathbf{S} \rangle| &= |\langle \boldsymbol{\Theta}_{\mathcal{T}^c}, \mathbf{S}_{\mathcal{T}^c} \rangle| \\
&= |\langle \boldsymbol{V}_{\mathcal{T}^c} + \boldsymbol{V}_{\mathcal{T}^c}^T + \boldsymbol{Z}_{\mathcal{T}^c}, \mathbf{S}_{\mathcal{T}^c} \rangle| \\
&= |\langle 2\boldsymbol{V}_{\mathcal{T}^c} + \boldsymbol{Z}_{\mathcal{T}^c}, \mathbf{S}_{\mathcal{T}^c} \rangle| \\
&\leq (2\|\boldsymbol{V}_{\mathcal{T}^c}\|_1 + \|\boldsymbol{Z}_{\mathcal{T}^c}\|_1) S_{\max} \\
&< \ \lambda_2 \|\boldsymbol{V}_{\mathcal{T}^c}\|_1 + \lambda_1 \|\boldsymbol{Z}_{\mathcal{T}^c}\|_1,
\end{aligned}$$

where the last inequality follows from the sufficient condition. We have shown (5.45) as desired.

□

Proof of Theorem 2 (Necessary Condition)

We first present a simple lemma for proving Theorem 2. Throughout the proof of Theorem 2, $\|\cdot\|_\infty$ indicates the maximal absolute element of a matrix and $\|\cdot\|_{\infty,s}$ indicates the dual norm of $\|\cdot\|_{1,q}$.

Lemma 44. *The dual representation of* $\tilde{\mathbf{P}}(\boldsymbol{\Theta})$ *in (5.43) is*

$$\begin{aligned}
\tilde{\mathbf{P}}^*(\boldsymbol{\Theta}) \quad = \quad & \max_{\boldsymbol{X},\boldsymbol{Y},\boldsymbol{\Lambda}} \quad \langle \boldsymbol{\Lambda}, \boldsymbol{\Theta} \rangle \\
& \text{subject to} \quad \boldsymbol{\Lambda} + \boldsymbol{\Lambda}^T = \hat{\lambda}_2 \boldsymbol{X} + \hat{\lambda}_3 \boldsymbol{Y} \\
& \qquad\qquad \|\boldsymbol{X}\|_\infty \leq 1, \|\boldsymbol{\Lambda}\|_\infty \leq 1, \|\boldsymbol{Y}\|_{\infty,s} \leq 1 \\
& \qquad\qquad X_{ii} = 0, Y_{ii} = 0, \boldsymbol{\Lambda}_{ii} = 0 \ \textit{for} \ i = 1, \ldots, p,
\end{aligned} \tag{5.46}$$

where $\frac{1}{s} + \frac{1}{q} = 1$.

Proof. We first state the dual representations for the norms in (5.43):

$$\begin{aligned}
\|\boldsymbol{Z} - \mathrm{Diag}(\boldsymbol{Z})\|_1 \quad = \quad & \max_{\boldsymbol{\Lambda}} \quad \langle \boldsymbol{\Lambda}, \mathbf{Z} \rangle \\
& \text{subject to} \quad \|\boldsymbol{\Lambda}\|_\infty \leq 1, \boldsymbol{\Lambda}_{ii} = 0 \text{ for } i = 1, \ldots, p,
\end{aligned}$$

$$\begin{aligned}
\|\boldsymbol{V} - \mathrm{Diag}(\boldsymbol{V})\|_1 \quad = \quad & \max_{\boldsymbol{X}} \quad \langle \mathbf{X}, \mathbf{V} \rangle \\
& \text{subject to} \quad \|\boldsymbol{X}\|_\infty \leq 1, X_{ii} = 0 \text{ for } i = 1, \ldots, p,
\end{aligned}$$

$$\begin{aligned}\|\boldsymbol{V} - \mathrm{Diag}(\boldsymbol{V})\|_{1,q} \;=\; & \max_{\boldsymbol{Y}} \quad \langle \boldsymbol{Y}, \boldsymbol{V}\rangle \\ & \text{subject to} \quad \|\boldsymbol{Y}\|_{\infty,s} \le 1, \boldsymbol{Y}_{ii} = 0 \text{ for } i = 1, \ldots, p.\end{aligned}$$

Then,

$$\begin{aligned}\tilde{\mathbf{P}}(\boldsymbol{\Theta}) \;=\; & \min_{\mathbf{V},\mathbf{Z}} \quad \|\boldsymbol{Z} - \mathrm{Diag}(\boldsymbol{Z})\|_1 + \hat{\lambda}_2\|\boldsymbol{V} - \mathrm{Diag}(\boldsymbol{V})\|_1 + \hat{\lambda}_3\|\boldsymbol{V} - \mathrm{Diag}(\boldsymbol{V})\|_{1,q} \\ & \text{subject to} \quad \boldsymbol{\Theta} = \mathbf{Z} + \mathbf{V} + \mathbf{V}^T \\ = \; & \min_{\mathbf{V},\mathbf{Z}} \quad \max_{\boldsymbol{\Lambda},\mathbf{X},\mathbf{Y}} \langle \boldsymbol{\Lambda}, \mathbf{Z}\rangle + \hat{\lambda}_2\langle \mathbf{X}, \mathbf{V}\rangle + \hat{\lambda}_3\langle \mathbf{Y}, \mathbf{V}\rangle \\ & \text{subject to} \quad \|\boldsymbol{\Lambda}\|_\infty \le 1, \|\boldsymbol{X}\|_\infty \le 1, \|\boldsymbol{Y}\|_{\infty,s} \le 1 \\ & \qquad\qquad \boldsymbol{\Lambda}_{ii} = 0, \boldsymbol{X}_{ii} = 0, \boldsymbol{Y}_{ii} = 0 \text{ for } i = 1, \ldots, p \\ & \qquad\qquad \boldsymbol{\Theta} = \mathbf{Z} + \mathbf{V} + \mathbf{V}^T \\ = \; & \max_{\boldsymbol{\Lambda},\mathbf{X},\mathbf{Y}} \quad \min_{\mathbf{V},\mathbf{Z}} \langle \boldsymbol{\Lambda}, \mathbf{Z}\rangle + \hat{\lambda}_2\langle \mathbf{X}, \mathbf{V}\rangle + \hat{\lambda}_3\langle \mathbf{Y}, \mathbf{V}\rangle \\ & \text{subject to} \quad \|\boldsymbol{\Lambda}\|_\infty \le 1, \|\boldsymbol{X}\|_\infty \le 1, \|\boldsymbol{Y}\|_{\infty,s} \le 1 \\ & \qquad\qquad \boldsymbol{\Lambda}_{ii} = 0, \boldsymbol{X}_{ii} = 0, \boldsymbol{Y}_{ii} = 0 \text{ for } i = 1, \ldots, p \\ & \qquad\qquad \boldsymbol{\Theta} = \boldsymbol{Z} + \boldsymbol{V} + \boldsymbol{V}^T \\ = \; & \max_{\boldsymbol{\Lambda},\boldsymbol{X},\boldsymbol{Y}} \quad \langle \boldsymbol{\Lambda}, \boldsymbol{\Theta}\rangle \\ & \text{subject to} \quad \boldsymbol{\Lambda} + \boldsymbol{\Lambda}^T = \hat{\lambda}_2\boldsymbol{X} + \hat{\lambda}_3\boldsymbol{Y} \\ & \qquad\qquad \|\boldsymbol{X}\|_\infty \le 1, \|\boldsymbol{\Lambda}\|_\infty \le 1, \|\boldsymbol{Y}\|_{\infty,s} \le 1 \\ & \qquad\qquad \boldsymbol{X}_{ii} = 0, \boldsymbol{Y}_{ii} = 0, \boldsymbol{\Lambda}_{ii} = 0 \text{ for } i = 1, \ldots, p.\end{aligned}$$

The third equality holds since the constraints on $(\mathbf{V}, \mathbf{Z})$ and on $(\boldsymbol{\Lambda}, \mathbf{X}, \mathbf{Y})$ are both compact convex sets and so by the minimax theorem, we can swap max and min. The last equality follows from the fact that

$$\begin{aligned}& \min_{\boldsymbol{V},\boldsymbol{Z}} \quad \langle \boldsymbol{\Lambda}, \mathbf{Z}\rangle + \hat{\lambda}_2\langle \mathbf{X}, \mathbf{V}\rangle + \hat{\lambda}_3\langle \mathbf{Y}, \mathbf{V}\rangle \\ & \text{subject to} \quad \boldsymbol{\Theta} = \boldsymbol{Z} + \boldsymbol{V} + \boldsymbol{V}^T \\ = \; & \begin{cases} \langle \boldsymbol{\Lambda}, \boldsymbol{\Theta}\rangle & \text{if } \boldsymbol{\Lambda} + \boldsymbol{\Lambda}^T = \hat{\lambda}_2\boldsymbol{X} + \hat{\lambda}_3\boldsymbol{Y} \\ -\infty & \text{otherwise.} \end{cases}\end{aligned}$$

□

We now present the proof of Theorem 2.

Proof. The optimality condition for (5.44) is given by

$$\mathbf{0} = -\mathbf{\Theta}^{-1} + \mathbf{S} + \lambda_1 \mathbf{\Lambda}, \tag{5.47}$$

where $\mathbf{\Lambda}$ is a subgradient of $\tilde{\mathbf{P}}(\mathbf{\Theta})$ in (5.43) and the left-hand side of the above equation is a zero matrix of size $p \times p$.

Now suppose that $\mathbf{\Theta}^*$ that solves (5.47) is supported on $\mathcal{T}$, i.e., $\mathbf{\Theta}^*_{\mathcal{T}^c} = 0$. Then for any $(i, j) \in \mathcal{T}^c$, we have that

$$0 = \boldsymbol{S}_{ij} + \lambda_1 \mathbf{\Lambda}^*_{ij}, \tag{5.48}$$

where $\mathbf{\Lambda}^*$ is a subgradient of $\tilde{\mathbf{P}}(\mathbf{\Theta}^*)$. Note that $\mathbf{\Lambda}^*$ must be an optimal solution to the optimization problem (5.46). Therefore, it is also a feasible solution to (5.46), implying that

$$\begin{aligned} |\mathbf{\Lambda}^*_{ij} + \mathbf{\Lambda}^*_{ji}| &\leq \hat{\lambda}_2 + \hat{\lambda}_3, \\ |\mathbf{\Lambda}^*_{ij}| &\leq 1. \end{aligned}$$

From (5.48), we have that $\mathbf{\Lambda}^*_{ij} = -\frac{\boldsymbol{S}_{ij}}{\lambda_1}$ and thus,

$$\begin{aligned} \lambda_1 &\geq \lambda_1 \max_{(i,j)\in\mathcal{T}^c} |\mathbf{\Lambda}^*_{ij}| \\ &= \lambda_1 \max_{(i,j)\in\mathcal{T}^c} \frac{|\boldsymbol{S}_{ij}|}{\lambda_1} \\ &= \boldsymbol{S}_{\max}. \end{aligned}$$

Also, recall that $\hat{\lambda}_2 = \frac{\lambda_2}{\lambda_1}$ and $\hat{\lambda}_3 = \frac{\lambda_3}{\lambda_1}$. We have that

$$\begin{aligned} \lambda_2 + \lambda_3 &\geq \lambda_1 \max_{(i,j)\in\mathcal{T}^c} |\mathbf{\Lambda}^*_{ij} + \mathbf{\Lambda}^*_{ji}| \\ &= \lambda_1 \max_{(i,j)\in\mathcal{T}^c} \frac{2|\boldsymbol{S}_{ij}|}{\lambda_1} \\ &= 2\boldsymbol{S}_{\max}. \end{aligned}$$

Hence, we obtain the desired result. □

Appendix B: Proof of block-diagonal conditions for PNJGL and CNJGL

5.8.1 Dual Characterization of RCON

Lemma 45. *The dual representation of Ω is given by*

$$\begin{aligned}\Omega(\boldsymbol{\Theta}^1,\ldots,\boldsymbol{\Theta}^K) \;=\; & \max_{\boldsymbol{\Lambda}^1,\ldots,\boldsymbol{\Lambda}^K\in\mathbb{R}^{p\times p}} \; \sum_{i=1}^{K}\langle \boldsymbol{\Lambda}^i,\boldsymbol{\Theta}^i\rangle \\ & \text{subject to} \quad \left\|\left\|\begin{bmatrix} \boldsymbol{\Lambda}^1+(\boldsymbol{\Lambda}^1)^T \\ \vdots \\ \boldsymbol{\Lambda}^K+(\boldsymbol{\Lambda}^K)^T\end{bmatrix}_j\right\|\right\|_* \leq 1 \text{ for } j=1,\ldots,p,\end{aligned} \tag{5.49}$$

where $\|\cdot\|$ denotes any norm, and $\|\cdot\|_*$ its corresponding dual norm.

Proof. Recall that Ω is given by

$$\begin{aligned}\Omega(\boldsymbol{\Theta}^1,\ldots,\boldsymbol{\Theta}^K) = \min_{\boldsymbol{V}^1,\ldots,\boldsymbol{V}^K\in\mathbb{R}^{p\times p}} & \left\|\begin{bmatrix}\boldsymbol{V}^1\\ \vdots \\ \boldsymbol{V}^K\end{bmatrix}\right\| \\ \text{subject to} \quad & \boldsymbol{\Theta}^i=\boldsymbol{V}^i+(\boldsymbol{V}^i)^T,\ i=1,2,\ldots,K.\end{aligned} \tag{5.50}$$

Let $\boldsymbol{Z}=\begin{bmatrix}\boldsymbol{Z}^1\\ \vdots \\ \boldsymbol{Z}^K\end{bmatrix}$ where $\boldsymbol{Z}^k\in\mathbb{R}^{p\times p}$. Then (5.50) is equivalent to

$$\Omega(\boldsymbol{\Theta}^1,\ldots,\boldsymbol{\Theta}^K) = \min_{\boldsymbol{V}^i:\,\boldsymbol{\Theta}^i=\boldsymbol{V}^i+(\boldsymbol{V}^i)^T,\,i=1,\ldots,K} \; \max_{\boldsymbol{Z}:\|\boldsymbol{Z}\|_*\leq 1} \; \sum_{i=1}^{K}\langle \boldsymbol{Z}^i,\boldsymbol{V}^i\rangle, \tag{5.51}$$

where $\|.\|_*$ is the dual norm to $\|.\|$. Since in (5.51) the cost function is bilinear in the two sets of variables and the constraints are compact convex sets, by the minimax theorem, we can swap max and min to get

$$\Omega(\boldsymbol{\Theta}^1,\ldots,\boldsymbol{\Theta}^K) = \max_{\boldsymbol{Z}:\|\boldsymbol{Z}\|_*\leq 1} \; \min_{\boldsymbol{V}^i:\,\boldsymbol{\Theta}^i=\boldsymbol{V}^i+(\boldsymbol{V}^i)^T,\,i=1,\ldots,K} \sum_{i=1}^{K}\langle \boldsymbol{Z}^i,\boldsymbol{V}^i\rangle \ . \tag{5.52}$$

Now, note that the dual to the inner minimization problem with respect to $V^1, \ldots, V^K$ in (5.52) is given by

$$\begin{array}{ll} \underset{\boldsymbol{\Lambda}^1,\ldots,\boldsymbol{\Lambda}^K}{\text{maximize}} & \displaystyle\sum_{i=1}^{K} \langle \boldsymbol{\Lambda}^i, \boldsymbol{\Theta}^i \rangle \\ \text{subject to} & \boldsymbol{Z}^i = \boldsymbol{\Lambda}^i + (\boldsymbol{\Lambda}^i)^T, \ i = 1, 2, \ldots, K. \end{array} \tag{5.53}$$

Plugging (5.53) into (5.52), the lemma follows. □

By definition, the subdifferential of Ω is given by the set of all K-tuples $(\boldsymbol{\Lambda}^1, \ldots, \boldsymbol{\Lambda}^K)$ that are optimal solutions to problem (5.49). Note that if $(\boldsymbol{\Lambda}^1, \ldots, \boldsymbol{\Lambda}^K)$ is an optimal solution to (5.49), then any $(\boldsymbol{\Lambda}^1 + \boldsymbol{Y}^1, \ldots, \boldsymbol{\Lambda}^K + \boldsymbol{Y}^K)$ with skew-symmetric matrices $\boldsymbol{Y}^1, \ldots, \boldsymbol{Y}^K$ is also an optimal solution.

5.8.2 Proof of Theorem 28

The optimality conditions for the PNJGL optimization problem (4.3) with $K = 2$ are given by

$$-n_1(\boldsymbol{\Theta}^1)^{-1} + n_1\boldsymbol{S}^1 + \lambda_1\boldsymbol{\Gamma}^1 + \lambda_2\boldsymbol{\Lambda} = 0, \tag{5.54}$$

$$-n_2(\boldsymbol{\Theta}^2)^{-1} + n_2\boldsymbol{S}^2 + \lambda_1\boldsymbol{\Gamma}^2 - \lambda_2\boldsymbol{\Lambda} = 0, \tag{5.55}$$

where $\boldsymbol{\Gamma}^1$ and $\boldsymbol{\Gamma}^2$ are subgradients of $\|\boldsymbol{\Theta}^1\|_1$ and $\|\boldsymbol{\Theta}^2\|_1$, and $(\boldsymbol{\Lambda}, -\boldsymbol{\Lambda})$ is a subgradient of $\Omega_q(\boldsymbol{\Theta}^1 - \boldsymbol{\Theta}^2)$. (Note that $\Omega_q(\boldsymbol{\Theta}^1 - \boldsymbol{\Theta}^2)$ is a composition of Ω_q with the linear function $\boldsymbol{\Theta}^1 - \boldsymbol{\Theta}^2$, and apply the chain rule.) Also note that the right-hand side of the above equations is a zero matrix of size $p \times p$.

Now suppose that $\boldsymbol{\Theta}^1$ and $\boldsymbol{\Theta}^2$ that solve (4.3) are supported on $\mathcal{T}$. Then since $(\boldsymbol{\Theta}^1)^{-1}, (\boldsymbol{\Theta}^2)^{-1}$ are supported on $\mathcal{T}$, we have that

$$\begin{aligned} n_1\boldsymbol{S}^1_{\mathcal{T}^c} + \lambda_1\boldsymbol{\Gamma}^1_{\mathcal{T}^c} + \lambda_2\boldsymbol{\Lambda}_{\mathcal{T}^c} &= 0, \\ n_2\boldsymbol{S}^2_{\mathcal{T}^c} + \lambda_1\boldsymbol{\Gamma}^2_{\mathcal{T}^c} - \lambda_2\boldsymbol{\Lambda}_{\mathcal{T}^c} &= 0. \end{aligned} \tag{5.56}$$

Summing the two equations in (5.56) yields

$$(n_1\boldsymbol{S}^1_{\mathcal{T}^c} + n_2\boldsymbol{S}^2_{\mathcal{T}^c}) + \lambda_1(\boldsymbol{\Gamma}^1_{\mathcal{T}^c} + \boldsymbol{\Gamma}^2_{\mathcal{T}^c}) = 0. \tag{5.57}$$

It thus follows from (5.57) that

$$\|n_1 \boldsymbol{S}^1_{\mathcal{T}^c} + n_2 \boldsymbol{S}^2_{\mathcal{T}^c}\|_\infty \leq \lambda_1 \|\boldsymbol{\Gamma}^1_{\mathcal{T}^c} + \boldsymbol{\Gamma}^2_{\mathcal{T}^c}\|_\infty \leq 2\lambda_1, \tag{5.58}$$

where here $\|\cdot\|_\infty$ indicates the maximal absolute element of a matrix, and where the second inequality in (5.58) follows from the fact that the subgradient of the ℓ_1 norm is bounded in absolute value by one.

We now assume, without loss of generality, that the $\boldsymbol{\Lambda}$ that solves (5.54) and (5.55) is symmetric. (In fact, one can easily show that there exist symmetric subgradients $\boldsymbol{\Gamma}^1$, $\boldsymbol{\Gamma}^2$, and $\boldsymbol{\Lambda}$ that satisfy (5.54) and (5.55).) Moreover, recall from Lemma 45 that $\|(\boldsymbol{\Lambda} + \boldsymbol{\Lambda}^T)_j\|_s \leq 1$. Therefore, $\|\boldsymbol{\Lambda}_j\|_s \leq \frac{1}{2}$. Using Holder's inequality and noting that $\|y\|_1 = \langle y, \operatorname{sgn}(y)\rangle$ for a vector y, we obtain

$$\begin{aligned}
\|\boldsymbol{\Lambda}_{\mathcal{T}^c}\|_1 &= \langle \boldsymbol{\Lambda}_{\mathcal{T}^c}, \operatorname{sgn}(\boldsymbol{\Lambda}_{\mathcal{T}^c})\rangle \\
&\leq \|\operatorname{sgn}(\boldsymbol{\Lambda}_{\mathcal{T}^c})\|_q \|\boldsymbol{\Lambda}_{\mathcal{T}^c}\|_s \\
&\leq |\mathcal{T}^c|^{\frac{1}{q}} \|\boldsymbol{\Lambda}_{\mathcal{T}^c}\|_s \\
&\leq |\mathcal{T}^c|^{\frac{1}{q}} \|\boldsymbol{\Lambda}\|_s \\
&\leq \tfrac{1}{2} |\mathcal{T}^c|^{\frac{1}{q}} p^{\frac{1}{s}},
\end{aligned} \tag{5.59}$$

where the last inequality follows from the fact that $\|\boldsymbol{\Lambda}\|_s^s = \sum_{j=1}^p \|\boldsymbol{\Lambda}_j\|_s^s \leq p(\frac{1}{2})^s$, and where in (5.59), $\|A\|_q$ and $\|A\|_s$ indicate the ℓ_q and ℓ_s norms of $\operatorname{vec}(A)$ respectively.

From (5.56), we have for each $k \in \{1, 2\}$ that

$$\begin{aligned}
n_k \|\boldsymbol{S}^k_{\mathcal{T}^c}\|_1 &\leq \|\lambda_1 \boldsymbol{\Gamma}^k_{\mathcal{T}^c} + \lambda_2 \boldsymbol{\Lambda}_{\mathcal{T}^c}\|_1 \\
&\leq \lambda_1 \|\boldsymbol{\Gamma}^k_{\mathcal{T}^c}\|_1 + \lambda_2 \|\boldsymbol{\Lambda}_{\mathcal{T}^c}\|_1 \\
&\leq \lambda_1 |\mathcal{T}^c| + \lambda_2 \frac{|\mathcal{T}^c|^{\frac{1}{q}} p^{\frac{1}{s}}}{2},
\end{aligned}$$

where the last inequality follows from the fact that the elements of $\boldsymbol{\Gamma}^k$ are bounded in absolute value by one, and (5.59). The theorem now follows by noting from (5.56) that for each $k \in \{1, 2\}$,

$$n_k \|\boldsymbol{S}^k_{\mathcal{T}^c}\|_\infty \leq \lambda_1 \|\boldsymbol{\Gamma}^k_{\mathcal{T}^c}\|_\infty + \lambda_2 \|\boldsymbol{\Lambda}_{\mathcal{T}^c}\|_\infty \leq \lambda_1 + \frac{\lambda_2}{2}.$$

5.8.3 *Proof of Theorem 31*

Proof. The optimality conditions for the CNJGL problem (4.4) are given by

$$-n_k(\boldsymbol{\Theta}^k)^{-1} + n_k\boldsymbol{S}^k + \lambda_1\boldsymbol{\Gamma}^k + \lambda_2\boldsymbol{\Lambda}^k = 0, \quad k = 1, \ldots, K, \tag{5.60}$$

where $\boldsymbol{\Gamma}^k$ is a subgradient of $\|\boldsymbol{\Theta}^k\|_1$. Also, the K-tuple $(\boldsymbol{\Lambda}^1, \ldots, \boldsymbol{\Lambda}^K)$ is a subgradient of $\Omega_q(\boldsymbol{\Theta}^1 - \text{Diag}(\boldsymbol{\Theta}^1), \ldots, \boldsymbol{\Theta}^K - \text{Diag}(\boldsymbol{\Theta}^K))$, and the right-hand side is a $p \times p$ matrix of zeros. We can assume, without loss of generality, that the subgradients $\boldsymbol{\Gamma}^k$ and $\boldsymbol{\Lambda}^k$ that satisfy (5.60) are symmetric, since Lemma 45 indicates that if $(\boldsymbol{\Lambda}^1, \ldots, \boldsymbol{\Lambda}^K)$ is a subgradient of $\Omega_q(\boldsymbol{\Theta}^1 - \text{Diag}(\boldsymbol{\Theta}^1), \ldots, \boldsymbol{\Theta}^k - \text{Diag}(\boldsymbol{\Theta}^k))$, then $((\boldsymbol{\Lambda}^1 + (\boldsymbol{\Lambda}^1)^T)/2, \ldots, (\boldsymbol{\Lambda}^K + (\boldsymbol{\Lambda}^K)^T)/2)$ is a subgradient as well.

Now suppose that $\boldsymbol{\Theta}^1, \ldots, \boldsymbol{\Theta}^K$ that solve (4.4) are supported on T. Since $(\boldsymbol{\Theta}^k)^{-1}$ is supported on $\mathcal{T}$ for all k, we have

$$n_k\boldsymbol{S}^k_{\mathcal{T}^c} + \lambda_1\boldsymbol{\Gamma}^k_{\mathcal{T}^c} + \lambda_2\boldsymbol{\Lambda}^k_{\mathcal{T}^c} = 0. \tag{5.61}$$

We use the triangle inequality for the ℓ_1 norm (applied element wise to the matrix) to get

$$n_k\|\boldsymbol{S}^k_{\mathcal{T}^c}\|_1 \leq \lambda_1\|\boldsymbol{\Gamma}^k_{\mathcal{T}^c}\|_1 + \lambda_2\|\boldsymbol{\Lambda}^k_{\mathcal{T}^c}\|_1. \tag{5.62}$$

We have $\|\boldsymbol{\Gamma}^k\|_\infty \leq 1$ since $\boldsymbol{\Gamma}^k$ is a subgradient of the ℓ_1 norm, which gives $\|\boldsymbol{\Gamma}^k_{\mathcal{T}^c}\|_1 \leq |\mathcal{T}^c|$.

Also $\boldsymbol{\Lambda}^k$ is a part of a subgradient to Ω_q, so by Lemma 45, $\|(\boldsymbol{\Lambda}^k + (\boldsymbol{\Lambda}^k)^T)_j\|_s \leq 1$ for $j \in \{1, 2, \ldots, p\}$. Since $\boldsymbol{\Lambda}^k$ is symmetric, we have that $\|\boldsymbol{\Lambda}^k_j\|_s \leq \frac{1}{2}$. Using the same reasoning as in (5.59) of Appendix 5.8.2, we obtain

$$\|\boldsymbol{\Lambda}^k_{\mathcal{T}^c}\|_1 \leq \frac{1}{2}|\mathcal{T}^c|^{\frac{1}{q}}p^{\frac{1}{s}}. \tag{5.63}$$

Combining (5.62) and (5.63) yields

$$n_k\|\boldsymbol{S}^k_{\mathcal{T}^c}\|_1 \leq \lambda_1|\mathcal{T}^c| + \tfrac{\lambda_2}{2}|\mathcal{T}^c|^{\frac{1}{q}}p^{\frac{1}{s}}.$$

The theorem follows by noting from (5.61) that

$$n_k\|\boldsymbol{S}^k_{\mathcal{T}^c}\|_\infty \leq \lambda_1\|\boldsymbol{\Gamma}^k_{\mathcal{T}^c}\|_\infty + \lambda_2\|\boldsymbol{\Lambda}^k_{\mathcal{T}^c}\|_\infty \leq \lambda_1 + \tfrac{\lambda_2}{2}.$$

□

5.8.4 Proof of Theorem 34

Assume that the sufficient condition holds. In order to prove the theorem, we must show that

$$\begin{aligned}
&\sum_{k=1}^K n_k(-\log\det(\boldsymbol{\Theta}^k) + \langle\boldsymbol{\Theta}^k, \boldsymbol{S}^k\rangle)) + \lambda_1\sum_{k=1}^k\|\boldsymbol{\Theta}^k\|_1 + \lambda_2 h(\boldsymbol{\Theta}^1,\ldots,\boldsymbol{\Theta}^K)\\
> &\sum_{k=1}^K n_k(-\log\det(\boldsymbol{\Theta}^k_T) + \langle\boldsymbol{\Theta}^k_T, \boldsymbol{S}^k\rangle)) + \lambda_1\sum_{k=1}^k\|\boldsymbol{\Theta}^k_T\|_1 + \lambda_2 h(\boldsymbol{\Theta}^1_T,\ldots,\boldsymbol{\Theta}^K_T).
\end{aligned}$$

By assumption,

$$h(\boldsymbol{\Theta}^1,\ldots,\boldsymbol{\Theta}^K) > h(\boldsymbol{\Theta}^1_T,\ldots,\boldsymbol{\Theta}^K_T). \tag{5.64}$$

We will now show that

$$n_k\langle\boldsymbol{\Theta}^k, \boldsymbol{S}^k\rangle + \lambda_1\|\boldsymbol{\Theta}^k\|_1 \geq n_k\langle\boldsymbol{\Theta}^k_T, \boldsymbol{S}^k\rangle + \lambda_1\|\boldsymbol{\Theta}^k_T\|_1, \tag{5.65}$$

or equivalently, that

$$-n_k\langle\boldsymbol{\Theta}^k_{T^c}, \boldsymbol{S}^k\rangle \leq \lambda_1\|\boldsymbol{\Theta}^k_{T^c}\|_1. \tag{5.66}$$

Note that $\langle\boldsymbol{\Theta}^k_{T^c}, \boldsymbol{S}^k\rangle = \langle\boldsymbol{\Theta}^k_{T^c}, \boldsymbol{S}^k_{T^c}\rangle$. By the sufficient condition, $n_k\|\boldsymbol{S}^k_{T^c}\|_\infty \leq \lambda_1$. So

$$\begin{aligned}
-n_k\langle\boldsymbol{\Theta}^k_{T^c}, \boldsymbol{S}^k\rangle &= -n_k\langle\boldsymbol{\Theta}^k_{T^c}, \boldsymbol{S}^k_{T^c}\rangle\\
&\leq \|n_k\boldsymbol{S}^k_{T^c}\|_\infty\|\boldsymbol{\Theta}^k_{T^c}\|_1\\
&\leq \lambda_1\|\boldsymbol{\Theta}^k_{T^c}\|_1.
\end{aligned}$$

So (5.66) holds, and hence (5.65) holds.

Finally, we apply Fischer's inequality, which states that $\det(\boldsymbol{\Theta}^k) \leq \det(\boldsymbol{\Theta}^k_T)$, and

so

$$-\log\det(\boldsymbol{\Theta}^k) \geq -\log\det(\boldsymbol{\Theta}_T^k). \tag{5.67}$$

Combining (5.64), (5.65), and (5.67), the theorem holds.

Appendix C: Proof of step-size selection for positive definiteness

5.8.5 Proof of Theorem 37

Let the eigenvalues of $\boldsymbol{\Theta}$ be

$$0 < \eta_p \leq \eta_{p-1} \leq \cdots \leq \eta_1$$

and the eigenvalues of $\boldsymbol{\Theta} + \alpha\mathbf{B}$ be

$$\lambda_p \leq \lambda_{p-1} \leq \cdots \leq \lambda_1,$$

where $\mathbf{B} = \boldsymbol{u}\boldsymbol{w}^T + \boldsymbol{w}\boldsymbol{u}^T$. Lemma 35 shows the eigenvalues of $\mathbf{B}$ satisfies

$$\beta_p < 0 = \beta_{p-1} = \beta_{p-2} \cdots = \beta_2 < \beta_1.$$

Lemma 36 implies that

$$\eta_p \leq \lambda_i - \alpha\beta_i \leq \eta_1 \quad i = 1, \cdots, p. \tag{5.68}$$

In particular

$$\begin{aligned} \lambda_1 &\geq \eta_p + \alpha\beta_1 > 0 \\ \lambda_i &\geq \eta_p + \alpha\beta_i = \eta_p > 0 \quad i = 2, \cdots, p-1, \end{aligned}$$

This means $\boldsymbol{\Theta} + \alpha\mathbf{B}$ has at most one negative eigenvalue λ_p. Observe that

$$\det(\boldsymbol{\Theta} + \alpha\mathbf{B}) = \Pi_{i=1}^p \lambda_i,$$

hence $\det(\boldsymbol{\Theta} + \alpha\mathbf{B}) > 0$ if and only if $\lambda_p > 0$.

By (5.30), we have

$$\det(\boldsymbol{\Theta} + \alpha \mathbf{B}) = \det(\boldsymbol{\Theta})[(1 + \alpha \boldsymbol{w}^T \boldsymbol{\Theta}^{-1} \boldsymbol{u})^2 - \alpha^2 (\boldsymbol{w}^T \boldsymbol{\Theta}^{-1} \boldsymbol{w})(\boldsymbol{u}^T \boldsymbol{\Theta}^{-1} \boldsymbol{u})]$$

$\boldsymbol{\Theta} \succ 0$ implies $\det(\boldsymbol{\Theta}) > 0$, therefore $\det(\boldsymbol{\Theta} + \alpha \mathbf{B}) > 0$ is equivalent to

$$(1 + \alpha \boldsymbol{w}^T \boldsymbol{\Theta}^{-1} \boldsymbol{u})^2 - \alpha^2 (\boldsymbol{w}^T \boldsymbol{\Theta}^{-1} \boldsymbol{w})(\boldsymbol{u}^T \boldsymbol{\Theta}^{-1} \boldsymbol{u}) > 0 \tag{5.69}$$

From the statement of Theorem 37 and the Cauchy-Schwartz inequality, we have that $\delta \geq 0$. If we expand the expression on the left-hand side of 5.69 we get

$$-\delta \, \alpha^2 \; + \; 2 \, (\boldsymbol{u}^T \boldsymbol{\Theta}^{-1} \boldsymbol{w}) \, \alpha \; + \; 1 \; .$$

From this, it is straightforward to show that α satisfies 5.69 if and only if whenever $\delta = 0$, then

$$\begin{aligned} \alpha &\in \mathbb{R} \text{ when } \boldsymbol{u}^T \boldsymbol{\Theta}^{-1} \boldsymbol{w} = 0, \\ \alpha &> \frac{-1}{2 \boldsymbol{u}^T \boldsymbol{\Theta}^{-1} \boldsymbol{u}} \text{ when } \boldsymbol{u}^T \boldsymbol{\Theta}^{-1} \boldsymbol{w} > 0, \\ \alpha &< \frac{-1}{2 \boldsymbol{u}^T \boldsymbol{\Theta}^{-1} \boldsymbol{u}} \text{ when } \boldsymbol{u}^T \boldsymbol{\Theta}^{-1} \boldsymbol{w} < 0, \end{aligned}$$

and if $\delta > 0$, then

$$\begin{aligned} \delta^{-1}(\boldsymbol{w}^T \boldsymbol{\Theta}^{-1} \boldsymbol{u} - \sqrt{(\boldsymbol{u}^T \boldsymbol{\Theta}^{-1} \boldsymbol{u})(\boldsymbol{w}^T \boldsymbol{\Theta}^{-1} \boldsymbol{w})}) &< \alpha \\ \delta^{-1}(\boldsymbol{w}^T \boldsymbol{\Theta}^{-1} \boldsymbol{u} + \sqrt{(\boldsymbol{u}^T \boldsymbol{\Theta}^{-1} \boldsymbol{u})(\boldsymbol{w}^T \boldsymbol{\Theta}^{-1} \boldsymbol{w})}) &> \alpha. \end{aligned}$$

When $\delta = 0$, the equality condition of Cauchy-Schwartz inequality gives $\boldsymbol{u} = \gamma \boldsymbol{w}$. Moreover if $\delta = 0$ and $\boldsymbol{u}^T \boldsymbol{\Theta}^{-1} \boldsymbol{w} = \gamma (\boldsymbol{\Theta}^{-1})_{ii} = 0$, then we see right away that $\gamma = 0$, i.e $\boldsymbol{u} = 0$. We reach the optimal solution of (5.25).

5.9 Acknowledgements

The work on speed ups through screening rules for CNJGL, PNJGL and HGL [Mohan et al., 2013, Tan et al., 2014] is joint work with Maryam Fazel, Daniela Witten, Su-In Lee, Kean-Ming Tan and Palma London. The work on SBCD algorithm is joint work with James Burke, Maryam Fazel, and Jiashan Wang.

Chapter 6

CONCLUSIONS

6.1 Summary

In summary, we considered the topic of learning matrix structures in the high-dimensional statistical setting where the number of parameters is more than the number of samples available. We specifically focused on the two problems of matrix rank minimization (which has applications in matrix completion and system identification) and learning structured graphical models. While low-rank matrix recovery has many applications in movie and product recommendations, graphical models are useful in modeling biological and social networks. The common theme in our work is the use of prior information to learn better models in the high-dimensional setting, by which we mean models that capture specific properties of the applications through the use of prior knowledge. In the context of learning graphical models, we use structured sparse priors to learn structured graphical models that enable a better understanding in the application. For example, we are able to identify the regulatory genes better in gene-regulatory networks when we use a penalty that promotes hubs as compared to penalties that only promote sparsity. Similarly, our formulation can be used to learn graphical models with a community structure (given a candidate set of communities). We also presented efficient algorithms for both the low-rank matrix recovery problem and the problem of learning structured graphical models with performance guarantees.

6.2 Contributions

Our specific contributions are as follows:

- We present a family of iterative reweighted algorithms for the problem of matrix rank minimization. At one end of the family, we have an algorithm for a convex problem that converges to the solution of the nuclear norm heuristic, at the other end we have a fast algorithm that converges to the stationary point of a

non-convex problem. Our implementation scheme, denoted by sIRLS is fast for both nuclear norm minimization and also a non-convex log-det heuristic, which has an empirically better recovery performance (number of measurements required for recovery).
We compare the IRLS algorithms with other state of the art algorithms and observe that our algorithms have an advantage in recovery performance when the rank of the true matrix is unknown.
Though not discussed in the thesis, in [Oymak et al., 2011a], we also present a seamless way to extend results on RIP and related conditions (such as spherical section property) from the compressed sensing setup to matrix rank minimization. We do this through the use of a singular value inequality that enables us to vectorize matrices, thus getting sharp sufficient conditions for recovery of low-rank matrices.

- We next consider the problem of learning graphical models with structure. We propose a convex formulation, structured graphical lasso, to learn a graphical model with structured sparse priors. We also present algorithms for learning a single graphical model with hub structure by optimizing the convex hub graphical lasso formulation. In the context of learning multiple graphical models, we propose the CNJGL formulation for learning graphical models with shared structure and the PNJGL formulation for learning hub structure in the difference of graphical models. In Appendix A of the thesis, we present sample complexity results that show that hub graphical lasso requires fewer samples for support recovery than graphical lasso. We also consider two approaches to speed up our algorithms. The first approach involves simple screening rules to detect a block diagonal structure (upto a permutation) in the optimal solution of the formulations. This enables breaking down the optimization problem into smaller sub-problems. We also present scalable algorithms based on block-coordinate descent that do not require any expensive linear-algebra computations such as the eigen-value decomposition or Cholesky decomposition. Our empirical results show that our algorithms are efficient in learning matrices with structured sparse supports for hub graphical lasso and hub covariance selection problems.

6.3 Future work

We now mention a few future directions of our research:

- One of our approaches to speed up our algorithms was to devise screening rules to detect block-diagonal structure in the optimal solution. A natural extension of this is to devise screening rules to detect other structures in the optimal solution. For sparse structures, this can lead to significant speed-ups since the algorithm can skip the update of coordinates that are not in the structured sparse support.

- Another direction for research is to find more practical applications that fit into our structured graphical lasso framework. We pointed out two applications in our thesis, learning graphs with hubs and learning graphs with communities. Each of these applications can bring up interesting algorithmic challenges. For example, as discussed in Chapter 5, our SBCD algorithm for learning graphs with hubs relies on rank-2 updates in each iteration. When learning graphs with structures other than hubs, the rank of the update may be greater than two making it non-trivial to extend our algorithm. In the case of learning graphs with communities, this problem can be circumvented through the use of Schur-complements to check for positive definiteness. However, coming up with an extension of our SBCD algorithm to learn arbitrarily structured graphical models is open.

- It would also be an interesting challenge to extend our SBCD algorithm to learning graphical models with low-rank and sparse structure such as in the latent or hidden variable graphical model problem [Chandrasekaran et al., 2012b]. More generally, it would be interesting to extend the SBCD algorithm to solving optimization problems that have a positive semi-definite constraint. Other examples of these problems include positive semi-definite matrix completion [Bishop and Yu, 2014], low-rank distance matrix completion [Mishra et al., 2011], etc.

- Finally, another direction for future work is improving and generalizing the model-selection consistency results that we present in the appendix of this thesis. Also comparing the consistency results with other approaches including

neighborhood selection [Meinshausen and Bülhmann, 2006] would be an interesting direction to pursue.

Appendices

Appendix 1

CONSISTENCY OF THE HUB GRAPHICAL LASSO

In this section, we give sample complexity bounds for the hub graphical lasso.

A.1 Problem statement,literature review and contributions

Let $\mathcal{S}^p_{++}$(resp. $\mathcal{S}^p_{+}$) denote the set of all $p \times p$ positive definite (resp. semi-definite) matrices. Let $\Sigma_0 \in \mathcal{S}^p_{++}$ be a Gaussian covariance matrix such that $\boldsymbol{\Theta}_0 = (\Sigma_0)^{-1} = \boldsymbol{V}_0 + \boldsymbol{V}_0^T + \boldsymbol{Z}_0 + \boldsymbol{Z}_0^T$ where $\boldsymbol{V}_0$ has k non-zero columns (after excluding the positive diagonals) and $\boldsymbol{Z}_0$ has m non-zero entries. Such matrices are said to have a *row-column* structure as shown in Figure A.1. In Figure A.1, the graph induced by the non-zero entries of $\boldsymbol{V} + \boldsymbol{V}^T$ has k hub nodes (or nodes with a very high degree) while the graph induced by the non-zero entries of $\boldsymbol{Z}_0 + \boldsymbol{Z}_0^T$ is sparse. We therefore refer to the graph induced by $\boldsymbol{\Theta}$ in Figure A.1 as *hub-structured*.

We assume the following sampling model:

$$\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_n \text{ i.i.d} \sim \mathcal{N}(0, \Sigma_0), \tag{A.1}$$

$$S = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{X}_i \boldsymbol{X}_i^T, \tag{A.2}$$

where $\boldsymbol{X}_i \in \mathbb{R}^p$. Our problem statement is as follows: Given the sample covariance matrix S, how many samples, n do we need to accurately estimate the true inverse covariance matrix, $\boldsymbol{\Theta}_0 = \Sigma_0^{-1}$? I.e. we would like to estimate both the support of $\boldsymbol{\Theta}_0$ correctly (or identify the graphical model) and also bound the error between the estimate and $\boldsymbol{\Theta}_0$ as a function of the sample size. Note that we don't know a priori the number or location of the hub nodes and the sparse edges in the Gaussian graphical model. We are particularly interested in the high-dimensional setting where n scales with p and therefore, giving an estimator that has a low sample complexity is imperative. In Chapter 3, we presented convex formulations and algorithms for learning a graphical model with hub structure. In this chapter, we give sample complexities on

both support recovery and convergence rate in Frobenius norm for the hub graphical lasso formulation.

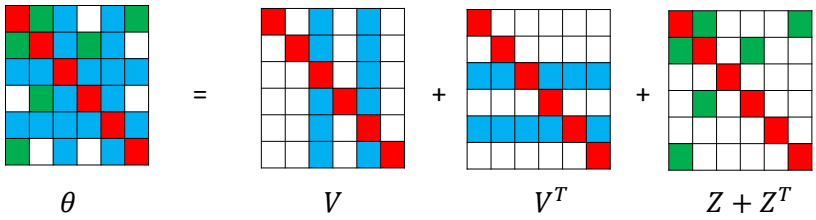


Figure A.1: A row-column structured matrix $\mathbf{\Theta}$ decomposed into its components that captures hubs in V, V^T and sparse edges in Z. In the graph associated with $\mathbf{\Theta}$, the number of hubs is $k = 2$ and number of sparse edges is $m = 6$.

A simple estimator that was proposed in the literature for learning a *sparse* inverse covariance matrix (or equivalently a sparse Gaussian graphical model) is the graphical lasso [Friedman et al., 2007]:

$$\hat{\mathbf{\Theta}} = \underset{\mathbf{\Theta} \in \mathcal{S}^p_{++}}{\operatorname{argmin}} - \log\det(\mathbf{\Theta}) + \langle \boldsymbol{S}, \mathbf{\Theta} \rangle + \lambda \|\mathbf{\Theta} - \operatorname{Diag}(\mathbf{\Theta})\|_1, \tag{A.3}$$

where $\operatorname{Diag}(\mathbf{\Theta})$ is a diagonal matrix whose entries are given by the diagonals of $\mathbf{\Theta}$. The literature has a few papers on analyzing consistency for learning graphical models penalized by the ℓ_1 norm [Ravikumar et al., 2011, 2010, Rothman et al., 2008, Lam and Fan, 2009]. We note that a sample complexity bound of $O(\sqrt{\frac{s \log p}{n}})$ under Frobenius norm was provided in [Rothman et al., 2008]. While this complexity bound states that the estimator is not far away from the true inverse covariance matrix, it does not imply that the support is recovered. To address this, it was shown in [Ravikumar et al., 2011] that if the maximum degree of nodes in the graphical model is d, then $O(d^2 \log p)$ measurements suffice to estimate the support of the inverse covariance matrix accurately. However when we have high degree nodes or *hub nodes*, the sample complexity can be as large as $O(p^2 \log p)$. Since we are in the high-dimensional setting, this sample complexity is too large to learn a graphical model with hubs. More recently, greedy methods have also been proposed for learning graphical models [Johnson et al., 2012, Jalali et al., 2011]. [Johnson et al., 2012] shows that a forward-backward greedy methods enjoys a $O(d \log p)$ sample complexity for learning a sparse

graphical model with maximum degree d. Their result, though better than that for GL, doesn't generalize to the case of graphs with *dense hubs* (d very close to p), as they require that $d < \frac{p}{10}$ (see Assumption in Section 3 of [Johnson et al., 2012]). Also, [Liu and Ihler, 2011, Defazio and Caetano, 2012] present methods for learning hub nodes in scale-free networks. However the hubs learned in these papers are far less dense than the dense hubs we consider in our model and furthermore, these papers don't present consistency results. Finally [Hero and Rajaratnam, 2014] present a screening rule for detecting hubs from partial correlation measurements. However as seen in section 3.4.4 of Chapter 3, this method doesn't perform as well as Hub graphical lasso (HGL) for learning graphs with dense hubs.

Sample complexity results for learning Ising models with hubs has been mentioned in [Tandon and Ravikumar, 2014]. However their result is for Ising model and analyzes a very different method based on neighborhood selection (see e.g. [Ravikumar et al., 2010], [Meinshausen and Bülhmann, 2006]).

Our contributions are as follows. We prove an important inequality with respect to the overlap norm that is useful for our analysis. We then consider the HGL estimator introduced in Chapter 3, that uses the row-column overlap norm as a regularizer and show that it enjoys a sample complexity of $O(\max\{k, m\}(2kp + p + m)\log p)$ for correctly learning a graphical model with k hubs and m other edges not connected to a hub node. This is strictly better than $O(p^2 \log p)$ of GL when $\max\{k^2, mk\} < p$. In particular when $k = O(1)$ and $m < p$, this estimator has a better rate than GL. Consider for instance the problem of learning a star graph that has a single hub node connected to all other nodes. In this case, there are $p - 1$ edges to be estimated. GL requires $O(p^2 \log p)$ samples [Ravikumar et al., 2011], while the HGL requires only $O(p \log p)$ samples. As also mentioned in [Ravikumar et al., 2011], the neighborhood selection method of [Meinshausen and Bülhmann, 2006] (which solves a sequence of lassos) requires $O(d \log p)$ samples for learning a graph with a maximum node degree of d. For graphs with dense hubs, this translates to a sample complexity of $O(p \log p)$. This bound matches our bound for a star graph. As also mentioned in [Ravikumar et al., 2011], the irrepresentability or incoherence condition in [Ravikumar et al., 2011] and also in this section is stricter than the incoherence conditions required for lasso. Thus, with weaker incoherence conditions, our model selection consistency result could potentially be improved further, which we leave

for future work. Indeed as seen in section 3.4.4 of Chapter 3, hub graphical lasso outperforms neighborhood selection in learning a graph with dense hubs. We also show that the Frobenius error between our estimate $\hat{\boldsymbol{\Theta}}$ and the truth $\boldsymbol{\Theta}_0$ is bounded as $\|\hat{\boldsymbol{\Theta}} - \boldsymbol{\Theta}_0\|_F \leq O\left(\sqrt{\frac{(2pk+p+m)\log p}{n}}\right)$.

A.2 Row-column overlap norm

Notation. For a matrix $\boldsymbol{V}$, let $\|\boldsymbol{V}\|_{1,q}$ denotes the sum of the ℓ_q norms of the columns of V and $\|x\|_\infty$ denotes the maximum entry in absolute value of a vector x. Also let $\|X\|$ denote the operator or spectral norm of matrix X.

The row-column overlap norm (RCON) discussed in chapter 3 is as follows:

$$\begin{aligned} \Omega_q(\boldsymbol{\Theta}) \quad = \quad & \min_{\boldsymbol{V},\boldsymbol{Z}} \quad \|\boldsymbol{V}\|_{1,q} + \lambda_1 \|\boldsymbol{Z}\|_1 \\ & \text{s.t.} \quad \boldsymbol{\Theta} = \boldsymbol{V} + \boldsymbol{V}^T + \boldsymbol{Z} + \boldsymbol{Z}^T, \end{aligned} \tag{A.4}$$

where $1 \leq q \leq \infty$. The RCON penalty was proposed in Chapter 3, section 3.4 to learn a graph with hub structure. In comparison to (A.4), the RCON penalty in Chapter 3 also has an ℓ_1 penalty on $\boldsymbol{V}$ which we don't consider in (A.4) for simplicity of exposition. It is easy to see that RCON is indeed a norm and that for $q = 1$, it reduces to a scaled version of the ℓ_1 norm. As also mentioned in [Mohan et al., 2014], the subdifferential with respect to $\boldsymbol{\Theta}$ of RCON does not have a closed form. We now present some properties of RCON that prove useful for our analysis.

Lemma 46 (Sign)**.** *Any optimal solution,* $(\boldsymbol{V}^*, \boldsymbol{Z}^*)$ *to (A.4) satisfies* $\text{sgn}(\boldsymbol{V}^*)_{ij} = \text{sgn}(\boldsymbol{V}^*)_{ji} = \text{sgn}(\boldsymbol{Z}^*_{ij}) = \text{sgn}(\boldsymbol{Z}^*_{ji})\ \forall (i,j) \in [p] \times [p]$.

Lemma 47 (RCON norm inequality)**.** *Suppose* $\boldsymbol{\Theta} \in \mathcal{S}^p$ *has a row-column structure so that it is supported on* k *rows and* k *corresponding columns along with at most* m *non-zero entries not connected to the* k *hubs (as in Figure 3.1). Then, we have that*

$$\Omega_2(\boldsymbol{\Theta} - \mathit{Diag}(\boldsymbol{\Theta})) \quad \leq \quad \sqrt{2}(\sqrt{k} + \lambda_1\sqrt{m})\|\boldsymbol{\Theta} - \mathit{Diag}(\boldsymbol{\Theta})\|_F \tag{A.5}$$

Proof. Let $\Omega_2(\boldsymbol{\Theta} - \text{Diag}(\boldsymbol{\Theta})) = \|\boldsymbol{V}^*\|_{1,2} + \lambda_1\|\boldsymbol{Z}^*\|_1$ where $\boldsymbol{\Theta} = \boldsymbol{V}^* + \boldsymbol{V}^{*T} + \boldsymbol{Z}^* + \boldsymbol{Z}^{*T}$. Let C index the k columns of $\boldsymbol{\Theta}$ and let M denote the indices of $\boldsymbol{\Theta}$ corresponding to the m non-zero entries not connected to the k hubs. Note that $|C| = k, |M| = m$.

Suppose $\boldsymbol{V}^*$ is supported on more than k columns. Construct a $(\widetilde{\boldsymbol{V}}, \widetilde{\boldsymbol{Z}})$ from $(\boldsymbol{V}^*, \boldsymbol{Z}^*)$ as follows:

$$\widetilde{\boldsymbol{V}}_{ij} = \begin{cases} \boldsymbol{V}^*_{ij} + \boldsymbol{Z}^*_{ij} & \text{if } i \in C, j \in C \\ \boldsymbol{V}^*_{ij} + \boldsymbol{V}^*_{ji} + \boldsymbol{Z}^*_{ij} + \boldsymbol{Z}^*_{ji} & \text{if } i \notin C, j \in C \\ 0 & \text{otherwise} \end{cases} \tag{A.6}$$

Similarly let $\widetilde{\boldsymbol{Z}}_{ij} = \boldsymbol{V}^*_{ij} + \boldsymbol{Z}^*_{ij}$ if $(i,j) \in M$ and $\widetilde{\boldsymbol{Z}}_{ij} = 0$ otherwise. Note that $\widetilde{\boldsymbol{V}}$ is now supported only on the k columns in C and $\widetilde{\boldsymbol{Z}}$ is supported only on M and $(\widetilde{\boldsymbol{V}}, \widetilde{\boldsymbol{Z}})$ are feasible to (A.4).

We have that,

$$\begin{aligned} \Omega_2(\boldsymbol{\Theta} - \mathrm{Diag}(\boldsymbol{\Theta})) &= \|\boldsymbol{V}^*\|_{1,2} + \lambda_1 \|\boldsymbol{Z}^*\|_1 \\ &\leq \|\widetilde{\boldsymbol{V}}\|_{1,2} + \lambda_1 \|\widetilde{\boldsymbol{Z}}\|_1 \\ &\leq \sqrt{k}\|\widetilde{\boldsymbol{V}}\|_F + \lambda_1\sqrt{m}\|\widetilde{\boldsymbol{Z}}\|_F \\ &\leq \sqrt{k}\|\widetilde{\boldsymbol{V}} + \widetilde{\boldsymbol{V}}^T\|_F + \lambda_1\sqrt{m}\|\widetilde{\boldsymbol{Z}} + \widetilde{\boldsymbol{Z}}^T\|_F \\ &\leq (\sqrt{k} + \lambda_1\sqrt{m})\sqrt{2}\sqrt{\|\widetilde{\boldsymbol{V}} + \widetilde{\boldsymbol{V}}^T\|_F^2 + \|\widetilde{\boldsymbol{Z}} + \widetilde{\boldsymbol{Z}}^T\|_F^2} \\ &= \sqrt{2}(\sqrt{k} + \lambda_1\sqrt{m})\|\boldsymbol{\Theta} - \mathrm{Diag}(\boldsymbol{\Theta})\|_F, \end{aligned} \tag{A.7}$$

where the first inequality follows from the optimality of $(\boldsymbol{V}^*, \boldsymbol{Z}^*)$ and the third inequality follows by construction and from Lemma 46. □

A.3 Maximum likelihood estimation

We now consider the Hub Graphical Lasso (HGL) approach introduced in Chapter 3, section 3.4 to obtain an estimator $\hat{\boldsymbol{\Theta}}$, which we will show is close to the true inverse covariance matrix, $\boldsymbol{\Theta}_0$ when n is sufficiently large.

$$\hat{\boldsymbol{\Theta}} = \underset{\boldsymbol{\Theta} \succ 0}{\mathrm{argmin}} \; -\log\det(\boldsymbol{\Theta}) + \langle \boldsymbol{\Theta}, \boldsymbol{S} \rangle + \lambda \Omega_2(\boldsymbol{\Theta} - \mathrm{Diag}(\boldsymbol{\Theta})). \tag{A.8}$$

Our consistency analysis for HGL is inspired in part by the analysis of GL [Ravikumar et al., 2011, Rothman et al., 2008]. We prove two results: a consistency in the Frobenius norm, and a model selection consistency result that guarantees recovery of the true graphical model. To do so, we first prove a partial consistency result which

we then use to prove model selection consistency and consequently Frobenius consistency. The partial consistency result requires mild assumptions on the true inverse covariance matrix while Frobenius and model selection consistency require stronger incoherence assumptions. In all of our results the expression *w.h.p.* would refer to "with probability at least $1 - \frac{1}{p^\gamma}$" where $\gamma > 1$. We note that although consistency results have been given for formulations using the overlap norm [Obozinski et al., Rao et al., 2013], these results usually assume a least squares loss and also have a different sampling model and hence they don't apply to our setting.

A.3.1 Partial consistency result

In this section, for simplicity we drop the subscript on Ω so that $\Omega(\boldsymbol{\Theta})$ refers to $\Omega_2(\boldsymbol{\Theta})$. Also let $\lambda\lambda_1 = \lambda_2$. We now give a partial consistency result that bounds the error between $\boldsymbol{\Theta}_0$ and any matrix $\widetilde{\boldsymbol{\Theta}}$ that has the same support as $\boldsymbol{\Theta}_0$ and has a lower objective than $\boldsymbol{\Theta}_0$. Since $\hat{\boldsymbol{\Theta}}$ may not satisfy this condition, we refer to the result below as partial consistency. We will later on prove a Frobenius consistency result for $\hat{\boldsymbol{\Theta}}$.

Lemma 48 (Partial consistency result (C))**.** *Let $\widetilde{\boldsymbol{\Theta}}, \boldsymbol{\Theta}_0 \in \mathcal{S}^p$ such that supp$(\widetilde{\boldsymbol{\Theta}}) \subset$ supp$(\boldsymbol{\Theta}_0) = T$ where $\boldsymbol{\Theta}_0$ has a row-column structure as in Figure 3.1 with k rows and columns, m sparse edges plus the diagonal. Also let $\lambda_{\max}(\boldsymbol{\Theta}_0) \leq \beta$ and assume that $\widetilde{\boldsymbol{\Theta}}$ has a smaller objective value than $\boldsymbol{\Theta}_0$ in (A.8). Then w.h.p. we have that for $C = 2\beta^2$,*

$$\|\widetilde{\boldsymbol{\Theta}} - \boldsymbol{\Theta}_0\|_F \quad \leq \quad Cr(p,k,m,n) := C(\sqrt{\tfrac{(2pk+p+m)\log p}{n}} + \sqrt{2k}\lambda + \sqrt{2m}\lambda_2) \tag{A.9}$$

Note that in the above lemma, β is a constant independent of the problem size.

- **Remark 1:** Note that we could have re-parameterized the HGL formulation, (A.8) as

$$\begin{aligned} \underset{\boldsymbol{V},\boldsymbol{Z}}{\text{argmin}} \quad & -\log\det(\boldsymbol{V} + \boldsymbol{V}^T + \boldsymbol{Z} + \boldsymbol{Z}^T) + 2\langle \boldsymbol{V}, \boldsymbol{S}\rangle + 2\langle Z, \boldsymbol{S}\rangle + \\ & \lambda\|\boldsymbol{V} - \text{Diag}(\boldsymbol{V})\|_{1,2} + \lambda_2\|\boldsymbol{Z} - \text{Diag}(\boldsymbol{Z})\|_1. \end{aligned} \tag{A.10}$$

  However the analysis in Lemma 48 using the objective (A.10) doesn't go through easily. Hence the use of the RCON inequality from Lemma 47 proves important

for Lemma 48.

- **Remark 2:** A naive application of the analysis in [Rothman et al., 2008] to the HGL formulation A.8 gives that $\|\hat{\boldsymbol{\Theta}} - \boldsymbol{\Theta}_0\|_F \leq \sqrt{\frac{p^2 \log p}{n}}$, which is a loose bound and doesn't depend on k. For GL, [Rothman et al., 2008] shows a Frobenius norm consistency result of $\sqrt{\frac{s \log p}{n}}$ where s is the sparsity of the true graphical model. For hub-structured graphical models, this translates to $\sqrt{\frac{(2pk+m) \log p}{n}}$. That is, GL has a better consistency as compared to HGL when using the analysis in [Rothman et al., 2008]. This is because the analysis uses the fact that ℓ_1 norm is decomposable over supports. That is, for any $T \subset [p] \times [p]$ and any $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^{p \times p}$, $\|\boldsymbol{X}_T + \boldsymbol{Y}_{T^c}\|_1 = \|\boldsymbol{X}_T\|_1 + \|\boldsymbol{Y}_{T^c}\|_1$ (here $\boldsymbol{X}_T$ denotes a matrix that is non-zero only over entries in T, i.e. X is restricted to support T). This property does not hold true for RCON. As mentioned earlier, we instead use the partial consistency result to subsequently show a better Frobenius consistency result for HGL.

In the next section we present our model-selection consistency result which shows exact support recovery under incoherence assumptions. Our model-selection consistency result uses Lemma 48 and can subsequently be used to show a Frobenius consistency result.

A.4 Model selection consistency analysis

In this section, we look at model selection consistency of HGL (A.8), which is is akin to asking if our estimator gives us the right 'model'. This corresponds to ensuring that there are no false edges learned in our graphical model or equivalently that we have learned the support of the true inverse covariance matrix. We take the dual-certificate construction approach also known as the primal-dual witness approach in the literature Negahban and Wainwright [2011], Ravikumar et al. [2010]. We first describe the approach for (A.8) followed by a detailed analysis of the consistency. We reformulate problem (A.8) as follows:

$$\begin{aligned} \underset{\boldsymbol{V},\boldsymbol{Z}}{\text{minimize}} \quad & -\log\det(\boldsymbol{V} + \boldsymbol{V}^T + \boldsymbol{Z} + \boldsymbol{Z}^T) + 2\langle \boldsymbol{V}, \boldsymbol{S} \rangle + 2\langle \boldsymbol{Z}, \boldsymbol{S} \rangle + \\ & \lambda \|\boldsymbol{V} - \mathrm{Diag}(\boldsymbol{V})\|_{1,2} + \lambda_2 \|\boldsymbol{Z} - \mathrm{diag}(\boldsymbol{Z})\|_1 \end{aligned} \tag{A.11}$$

Let the optimal solution to (A.11) be $(\hat{\boldsymbol{V}}, \hat{\boldsymbol{Z}})$. Note that the reformulation in (A.11) is necessary because the subgradient of the objective now has a closed form (in contrast to the subgradient of RCON) and furthermore, the subgradient of $\|\boldsymbol{V} - \text{Diag}(\boldsymbol{V})\|_{1,2}$ with respect to $\boldsymbol{V}$ is separable along the columns of $\boldsymbol{V}$. We use this property in constructing subgradients in the primal dual witness approach that we discuss next.

Let the true inverse covariance matrix $\boldsymbol{\Theta}_0 = \boldsymbol{V}_0 + \boldsymbol{V}_0^T + \boldsymbol{Z}_0 + \boldsymbol{Z}_0^T$ so that $\boldsymbol{V}_0 - \text{Diag}(\boldsymbol{V}_0)$ is column sparse with k non-zero columns and $\boldsymbol{Z}_0 - \text{Diag}(\boldsymbol{Z}_0)$ is sparse with m non-zero entries. Also let L be the support of $\boldsymbol{V}_0$ and M be the support of $\boldsymbol{Z}_0$. By assumption, $L \cap M = \emptyset$. Also let T be the support of $\boldsymbol{\Theta}_0$. The primal dual witness approach is essentially a way of constructing a solution, $(\widetilde{\boldsymbol{V}}, \widetilde{\boldsymbol{Z}})$ so that $\widetilde{\boldsymbol{V}}$ (resp. $\widetilde{\boldsymbol{Z}}$) has the same support as $\boldsymbol{V}_0$ (resp. $\boldsymbol{Z}_0$). It is then argued that $(\widetilde{\boldsymbol{V}}, \widetilde{\boldsymbol{Z}})$ is also the optimal solution to (A.11), finishing the analysis. We describe the details of the analysis below:

1. Construct $(\widetilde{\boldsymbol{V}}, \widetilde{\boldsymbol{Z}})$ as an optimal solution to:

$$\begin{aligned}(\widetilde{\boldsymbol{V}}, \widetilde{\boldsymbol{Z}}) \quad = \quad \underset{\boldsymbol{V},\boldsymbol{Z}:\boldsymbol{V}_{L^c}=0,\boldsymbol{Z}_{M^c}=0}{\text{argmin}} \quad & -\log\det(\boldsymbol{V} + \boldsymbol{V}^T + \boldsymbol{Z} + \boldsymbol{Z}^T) + \\ & 2\langle \boldsymbol{V}, \boldsymbol{S}\rangle + 2\langle \boldsymbol{Z}, \boldsymbol{S}\rangle + \\ & \lambda\|\boldsymbol{V} - \text{Diag}(\boldsymbol{V})\|_{1,2} + \lambda_2\|\boldsymbol{Z} - \text{diag}(\boldsymbol{Z})\|_1\end{aligned} \tag{A.12}$$

2. Note that $(\widetilde{\boldsymbol{V}}, \widetilde{\boldsymbol{Z}})$ satisfy: $-2(\widetilde{\boldsymbol{V}} + \widetilde{\boldsymbol{V}}^T + \widetilde{\boldsymbol{Z}} + \widetilde{\boldsymbol{Z}}^T)^{-1}_L + 2\boldsymbol{S}_L + \lambda\Gamma_L = 0$, where $\Gamma_{ii} = 0\ \forall i$. Also, $-2(\widetilde{\boldsymbol{V}} + \widetilde{\boldsymbol{V}}^T + \widetilde{\boldsymbol{Z}} + \widetilde{\boldsymbol{Z}}^T)^{-1}_M + 2\boldsymbol{S}_M + \lambda_2\Lambda_M = 0$, where $\Lambda_{ii} = 0\ \forall i$. Construct Γ_{L^c} such that $\lambda\Gamma_{L^c} = 2(\widetilde{\boldsymbol{V}} + \widetilde{\boldsymbol{V}}^T + \widetilde{\boldsymbol{Z}} + \widetilde{\boldsymbol{Z}}^T)^{-1}_{L^c} - 2\boldsymbol{S}_{L^c}$ and $\lambda_2\Lambda_{M^c} = 2(\widetilde{\boldsymbol{V}} + \widetilde{\boldsymbol{V}}^T + \widetilde{\boldsymbol{Z}} + \widetilde{\boldsymbol{Z}}^T)^{-1}_{M^c} - 2\boldsymbol{S}_{M^c}$.

3. Show that $\|\Gamma_{L^c}\|_{\infty,2} \leq 1$ and $\|\Lambda_{M^c}\|_\infty \leq 1$. This implies that (Γ, Λ) is a valid subgradient that satisfies the optimality conditions. Hence $\widetilde{\boldsymbol{V}}$ has support L and $\widetilde{\boldsymbol{Z}}$ has support M and therefore $(\widetilde{\boldsymbol{V}}, \widetilde{\boldsymbol{Z}})$ is an optimal solution to (A.11).

A.4.1 Sufficient conditions

Let $\boldsymbol{\Theta}_0 = \boldsymbol{V}_0 + \boldsymbol{V}_0{}^T + \boldsymbol{Z}_0 + \boldsymbol{Z}_0{}^T$. Let $\widetilde{\boldsymbol{V}} = \boldsymbol{V}_0 + \boldsymbol{\Delta}_V$ and $\widetilde{\boldsymbol{Z}} = \boldsymbol{Z}_0 + \boldsymbol{\Delta}_Z$. Note that $\boldsymbol{\Delta}_V$ is supported on L and $\boldsymbol{\Delta}_Z$ is supported on M. Let $\hat{\boldsymbol{\Delta}}_V = \boldsymbol{\Delta}_V + \boldsymbol{\Delta}_V^T$ and

$\hat{\boldsymbol{\Delta}}_Z = \boldsymbol{\Delta}_Z + \boldsymbol{\Delta}_Z^T$. Let $R(\hat{\boldsymbol{\Delta}}_V, \hat{\boldsymbol{\Delta}}_Z) = (\widetilde{\boldsymbol{V}} + \widetilde{\boldsymbol{V}}^T + \widetilde{\boldsymbol{Z}} + \widetilde{\boldsymbol{Z}}^T)^{-1} - ((\boldsymbol{V}_0 + \boldsymbol{V}_0{}^T + \boldsymbol{Z}_0 + \boldsymbol{Z}_0{}^T)^{-1} - (\boldsymbol{V}_0 + \boldsymbol{V}_0{}^T + \boldsymbol{Z}_0 + \boldsymbol{Z}_0{}^{-1})^{-1}(\hat{\boldsymbol{\Delta}}_V + \hat{\boldsymbol{\Delta}}_Z)(\boldsymbol{V}_0 + \boldsymbol{V}_0{}^T + \boldsymbol{Z}_0 + \boldsymbol{Z}_0{}^{-1})^{-1})$. Note that $R(\hat{\boldsymbol{\Delta}}_V, \hat{\boldsymbol{\Delta}}_Z)$ is the difference between $(\widetilde{\boldsymbol{V}} + \widetilde{\boldsymbol{V}}^T + \widetilde{\boldsymbol{Z}} + \widetilde{\boldsymbol{Z}}^T)^{-1}$ and its first order Taylor series approximation.

By construction, $(\widetilde{\boldsymbol{V}}, \widetilde{\boldsymbol{Z}})$ satisfy the optimality conditions:

$$\begin{aligned} -2(\widetilde{\boldsymbol{V}} + \widetilde{\boldsymbol{V}}^T + \widetilde{\boldsymbol{Z}} + \widetilde{\boldsymbol{Z}}^T)^{-1} + 2\boldsymbol{S} + \lambda\Gamma &= 0, \\ -2(\widetilde{\boldsymbol{V}} + \widetilde{\boldsymbol{V}}^T + \widetilde{\boldsymbol{Z}} + \widetilde{\boldsymbol{Z}}^T)^{-1} + 2\boldsymbol{S} + \lambda_2\Lambda &= 0, \end{aligned} \tag{A.13}$$

From (A.13), it is clear that $\lambda\Gamma = \lambda_2\Lambda$. Also from (A.13), we have that

$$(-2(\widetilde{\boldsymbol{V}} + \widetilde{\boldsymbol{V}}^T + \widetilde{\boldsymbol{Z}} + \widetilde{\boldsymbol{Z}}^T)^{-1} + 2(\boldsymbol{\Theta}_0)^{-1}) + (2\boldsymbol{S} - 2(\boldsymbol{\Theta}_0)^{-1}) + \lambda\Gamma = 0 \tag{A.14}$$

Let $W = S - (\boldsymbol{\Theta}_0)^{-1}$ and let $N = (\boldsymbol{\Theta}_0)^{-1}(\boldsymbol{\Delta}_V + \boldsymbol{\Delta}_V^T + \boldsymbol{\Delta}_Z + \boldsymbol{\Delta}_Z^T)(\boldsymbol{\Theta}_0)^{-1}$. Rewriting the optimality conditions in terms of $R(\hat{\boldsymbol{\Delta}}_V, \hat{\boldsymbol{\Delta}}_Z)$, we have,

$$-2R(\hat{\boldsymbol{\Delta}}_V, \hat{\boldsymbol{\Delta}}_Z) + 2W + 2N + \lambda\Gamma = 0 \tag{A.15}$$

For a matrix A, let $\bar{A}$ denote a vector obtained from A by stacking all its columns into a single column. The optimality conditions can also be written in a vectorized form as follows:

$$-2\bar{\mathbf{R}} + 2\bar{\mathbf{W}} + 2((\boldsymbol{\Theta}_0)^{-1} \otimes (\boldsymbol{\Theta}_0)^{-1})\overline{\hat{\boldsymbol{\Delta}}_V + \hat{\boldsymbol{\Delta}}_Z} + \lambda\bar{\Gamma} = 0 \tag{A.16}$$

Let $Y = (\boldsymbol{\Theta}_0)^{-1} \otimes (\boldsymbol{\Theta}_0)^{-1}$. Note that any entry of Y is indexed by a pair of indices $((i,j),(k,l))$. Let $Y_{(.,(i,j))}$ denote the column of Y indexed by (i,j). Let $\hat{Y}$ be a matrix such that $\hat{Y}_{(.,(i,j))} = Y_{(.,(i,j))} + Y_{(.,(j,i))}$ for all (i,j). Also let $Y_{L,T}$ denote Y restricted to rows in L and columns in T.

Definition 49. $\boldsymbol{\Theta}_0 \in \mathcal{S}_{++}^p$ *is said to be incoherent over the column support* L *if*

$$\sqrt{p} \max_{(i,j)\in L^c} \|\hat{Y}_{(i,j),L}(\hat{Y}_{LL})^{-1})\|_{1,2} \leq 1 - \alpha \tag{A.17}$$

for some $\alpha > 0$.

Note that our incoherence condition depends only on the support of $\boldsymbol{V}_0$. This is

because we construct a dual certificate, Λ for Z using the relation $\lambda\Gamma = \lambda_2\Lambda$. Also note that in Definition 49, the $\|x\|_{1,2}$ norm of a vector x is evaluated as the sum of the $\|.\|_2$ norms of consecutive and disjoint segments of x of length p.

Note that the incoherence condition for GL in [Ravikumar et al., 2011] is given by $\max_{(i,j)\in T^c} \|Y_{(i,j),T}(Y_{TT})^{-1}\|_1 \leq 1-\alpha$, where T is the support of sparse $\mathbf{\Theta}_0$. Our incoherence condition in definition 49 is in the worst case stronger than the corresponding condition in [Ravikumar et al., 2011] by a factor of $\sqrt{p}$. In the best case, our incoherence condition is as strong as that in [Ravikumar et al., 2011]. Under our incoherence condition, we give a sufficient condition for step 3 of the primal-dual witness approach outlined earlier to hold.

Theorem 50. *Let $\mathbf{\Theta}_0$ satisfy the incoherence condition in definition 49. Choose $\lambda \geq (\frac{8}{\alpha}-4)\max\{\|\bar{\mathbf{R}}\|_{\infty,2}, \|\bar{\mathbf{W}}\|_{\infty,2}\}$ and $\lambda_2 \geq \lambda$. Then, $\|\Gamma_{L^c}\|_{\infty,2} \leq 1$ and $\|\Lambda_{M^c}\|_\infty \leq 1$.*

Lemma 51 (Bound on $R(\hat{\mathbf{\Delta}}_V, \hat{\mathbf{\Delta}}_Z)$)**.** *Let $\hat{\mathbf{\Delta}} = \hat{\mathbf{\Delta}}_V + \hat{\mathbf{\Delta}}_Z$. Assume that $\|\hat{\mathbf{\Delta}}\| \leq \frac{1}{2\|(\mathbf{\Theta}_0)^{-1}\|}$. Then we have that,*

$$\|R(\hat{\mathbf{\Delta}}_V, \hat{\mathbf{\Delta}}_Z)\|_{\infty,2} \leq 3\|(\mathbf{\Theta}_0)^{-1}\|^3\|\hat{\mathbf{\Delta}}\|^2 \tag{A.18}$$

Proofs of the above results are given in the supplement and follows the lines of the proof given for GL in [Ravikumar et al., 2011].

A.4.2 Consistency results

The following theorem shows that the HGL estimator does not have false positives and follows from Lemma 48, Lemma 51 and Theorem 50.

Theorem 52 (No false positives)**.** *Let $\mathbf{\Theta}_0$ satisfy the incoherence condition in definition 49. Also let $n \geq 2C_1^2(8/\alpha-4)^2\max\{k,m\}(2pk+p+m)\log p$ and choose $\lambda = \lambda_2 = (8/\alpha-4)C_1\frac{(2pk+p+m)\log p}{n}$, where $C_1 = 12\beta^4/\delta^3$, where $\delta = \|(\mathbf{\Theta}_0)^{-1}\|$ and $\beta \geq \|\mathbf{\Theta}_0\|$. Then w.h.p., we have that*

$$supp(\hat{\mathbf{\Theta}}) \subseteq supp(\mathbf{\Theta}_0), \tag{A.19}$$

where $supp(\mathbf{\Theta})$ denotes the support or indices of the non-zero entries of $\mathbf{\Theta}$.

The previous result together with the partial consistency result (Lemma 48) yields a Frobenius consistency result as in Theorem 53.

Theorem 53 (Frobenius norm consistency)**.** *Let* $\mathbf{\Theta}_0$ *satisfy the incoherence condition in definition 49. Assume that* $n \geq 8C_1^2(8/\alpha - 4)^2 \max\{k, m\}(2pk + p + m)\log p$ *and choose* $\lambda = \lambda_2 = (8/\alpha - 4)C_1\frac{(2pk+p+m)\log p}{n}$*, where* $C_1 = 12\beta^4/\delta^3$*. We have w.h.p. that*

$$\|\hat{\mathbf{\Theta}} - \mathbf{\Theta}_0\|_F \leq 4\beta^2\sqrt{\frac{(2pk + p + m)\log p}{n}} \tag{A.20}$$

Finally combining, Theorem 52 and Theorem 53, we have the following model selection consistency result.

Theorem 54 (Model selection consistency)**.** *Let* $\mathbf{\Theta}_0$ *satisfy the incoherence condition in definition 49. Let* $n \geq 2C_1^2(8/\alpha - 4)^2 \max\{k, m\}(2pk + p + m)\log p$ *and choose* $\lambda = \lambda_2 = (8/\alpha - 4)C_1\frac{(2pk+p+m)\log p}{n}$*. Also let* $\zeta = \min_{(i,j)\in T} |(\mathbf{\Theta}_0)_{ij}|$ *such that* $\zeta > c\sqrt{\frac{(2pk+p+m)\log p}{n}}$ *for some constant c. Then w.h.p., we have that*

$$supp(\hat{\mathbf{\Theta}}) = supp(\mathbf{\Theta}_0) \tag{A.21}$$

Note that for a star graph with a single hub node, the above theorem gives a sample complexity of $O(p\log p)$ for HGL while GL has a sample complexity of $O(p^2\log p)$. We note that the neighborhood lasso [Meinshausen and Bülhmann, 2006] also enjoys a sample complexity of $O(p\log p)$ for star graphs. We next look at numerical results that illustrate this gap between GL and HGL for star graphs.

A.4.3 Numerical results on learning star graphs

We compare GL and HGL on learning a star graph with a single hub (i.e. $k = 1$). The plots in Figure A.2 show the Frobenius error and Probability of successful recovery (i.e. $\text{supp}(\hat{\mathbf{\Theta}}) = \text{supp}(\mathbf{\Theta}_0)$) as a function of the sample size for two different problem sizes of $p = 50$ and $p = 100$. We use a tolerance of 10^{-3} to determine non-zero elements in our estimates. The results are averaged over 150 random generations of the data. Note that HGL has a slightly better Frobenius error rate as compared to GL and this gap diminishes as the sample size increases. This corroborates with our analysis (Theorem 53) with our theoretical Frobenius error for HGL matching that of GL. Also note that HGL is able to successfully identify the true support with far fewer samples than GL. For example for $p = 100$ from Figure A.2, HGL requires only 460

samples for successful recovery with probability 1 whereas GL requires at least 2300 samples for successful recovery with probability 1. This also ties in with our model-selection consistency result, Theorem 54. More extensive numerical comparisons of HGL, GL and other algorithms on both simulated and real data can be found in Chapter 3, section 3.4 and Chapter 4, section 4.4. These numerical comparisons demonstrate that HGL can perform better than other algorithms on learning graphs with dense hubs on metrics such as number of true positives, number of hubs detected and Frobenius error between the estimate and the truth.

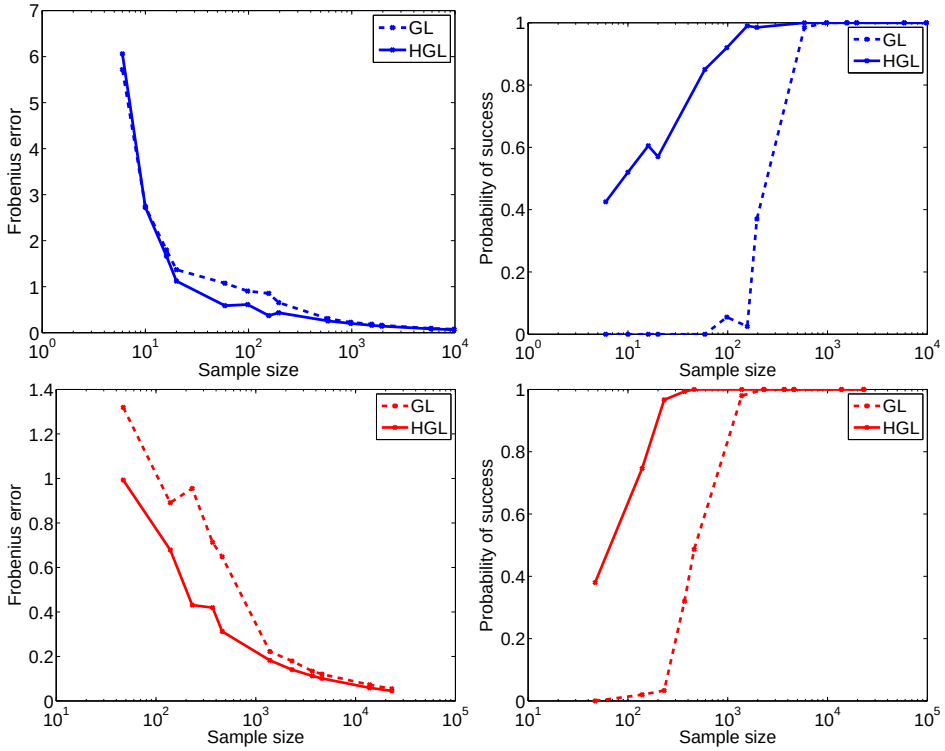


Figure A.2: Comparison of GL and HGL on Frobenius error (left panels) and successful support recovery (right panels). Top panels correspond to $p = 50$, while bottom panels correspond to $p = 100$. As seen from the figures, HGL correctly identifies the true support with much fewer samples than GL.

A.5 Discussion

We have provided model-selection consistency and Frobenius error results for learning Gaussian graphical models with hubs using HGL. Our bounds are better than those for GL when $\max\{k^2, km\} < p$ and in particular, when $k = O(1), m < p$. One question that needs to be addressed is if this bound is tight and if we can achieve a sample complexity smaller than $O(pk^2 \log p)$ for learning graphical models with k hubs using the hub graphical lasso. It would also be good to sharpen the sample complexity with respect to m, so that more edges are allowed in the graphical model that are not connected to hubs. Another direction for future work is study of other hub detection approaches such as forward-backward greedy algorithms (e.g. [Johnson et al., 2012]) or a variant of the neighborhood selection approach [Meinshausen and Bülhmann, 2006] and comparison of sample complexity for these approaches with HGL. It would also be interesting to extend our result to estimators using maximum likelihood with general overlap norms.

A.6 Proofs

Proof of Lemma 21

Suppose there exists an (i, j) such that $\text{sgn}(\boldsymbol{V}^*_{ij}) = -\text{sgn}(\boldsymbol{V}^*_{ji})$. Without loss of generality, assume that $|\boldsymbol{V}^*_{ij}| \leq |\boldsymbol{V}^*_{ji}|$. Construct a $\hat{\mathbf{V}}$ such that for all $(k, l) \neq \{(i, j) \cup (j, i)\}$, $\hat{\mathbf{V}}_{kl} = \boldsymbol{V}^*_{kl}$ and $\hat{\mathbf{V}}_{ij} = 0, \hat{\mathbf{V}}_{ji} = \boldsymbol{V}^*_{ij} + \boldsymbol{V}^*_{ji}$. Then note that, $\hat{\mathbf{V}} + \hat{\mathbf{V}}^T = \boldsymbol{V}^* + (\boldsymbol{V}^*)^T$. And therefore $(\hat{\mathbf{V}}, \boldsymbol{Z}^*)$ is a feasible solution to the RCON optimization problem. Also note by construction that $\|\hat{\mathbf{V}}\|_{1,q} < \|\boldsymbol{V}^*\|_{1,q}$. This implies $(\hat{\mathbf{V}}, \boldsymbol{Z}^*)$ has a smaller objective than $(\boldsymbol{V}^*, \boldsymbol{Z}^*)$ in the RCON optimization problem, which is a contradiction. Thus $\text{sgn}(\boldsymbol{V}^*_{ij}) = \text{sgn}(\boldsymbol{V}^*_{ji})\ \forall (i, j) \in [p] \times [p]$. Similarly, it can be argued that $\text{sgn}(\boldsymbol{Z}^*_{ij}) = \text{sgn}(\boldsymbol{Z}^*_{ji})\ \forall (i, j) \in [p] \times [p]$.

Proof of Lemma 23

Let $g(X) = -\log\det(X) + \langle X, \boldsymbol{S}\rangle + \lambda\Omega(X - \text{Diag}(X))$. Let $G(\boldsymbol{\Delta}) = g(\boldsymbol{\Theta}_0 + \boldsymbol{\Delta}) - g(\boldsymbol{\Theta}_0)$. First note that $G(\widetilde{\boldsymbol{\Theta}} - \boldsymbol{\Theta}_0) \leq 0$. The proof idea is to show that $G(\boldsymbol{\Delta}) > 0$ whenever $\|\boldsymbol{\Delta}\|_F \geq Cr(p, k, m, n)$ and $\boldsymbol{\Delta}_{T^c} = 0$. This immediately implies that $\|\widetilde{\boldsymbol{\Theta}} - \boldsymbol{\Theta}\|_F \leq Cr(p, k, m, n)$.

First, consider any $\mathbf{\Delta} : \mathbf{\Delta}_{T^c} = 0$. Let $f(\mathbf{\Theta}) = -\log\det(\mathbf{\Theta})$.Then note that $\nabla^2 f(\mathbf{\Theta}) = (\mathbf{\Theta})^{-1} \otimes (\mathbf{\Theta})^{-1}$. Also f is strongly convex at $\mathbf{\Theta}$ with strong convexity parameter μ if and only if

$$\begin{aligned} \mu &\leq \lambda_{\min}\left(\mathbf{\Theta}^{-1} \otimes \mathbf{\Theta}^{-1}\right) \\ &= \lambda^2_{\min}(\mathbf{\Theta})^{-1} \\ &= \tfrac{1}{\lambda^2_{\max}(\mathbf{\Theta})} \end{aligned}$$

Since $\lambda_{\max}(\mathbf{\Theta}_0) \leq \beta$, we have that f is strongly convex at $\mathbf{\Theta}_0$ with strong convexity parameter $\frac{1}{\beta^2}$.

By the properties of a strongly convex function, we have that

$$-\log\det(\mathbf{\Theta}_0 + \mathbf{\Delta}) \geq -\log\det(\mathbf{\Theta}_0) - \langle \mathbf{\Theta}_0^{-1}, \mathbf{\Delta}\rangle + \tfrac{1}{2\beta^2}\|\mathbf{\Delta}\|_F^2 \tag{A.22}$$

Using (A.22), we have that

$$\begin{aligned} G(\mathbf{\Delta}) &= -\log\det(\mathbf{\Theta}_0 + \mathbf{\Delta}) + \langle \mathbf{\Theta}_0 + \mathbf{\Delta}, \boldsymbol{S}\rangle + \lambda\Omega(\mathbf{\Theta}_0 - \mathrm{Diag}(\mathbf{\Theta}_0) + (\mathbf{\Delta} - \mathrm{Diag}(\mathbf{\Delta}))) \\ &\quad -(-\log\det\mathbf{\Theta}_0 + \langle \mathbf{\Theta}_0, \boldsymbol{S}\rangle + \Omega(\mathbf{\Theta}_0 - \mathrm{Diag}(\mathbf{\Theta}_0)) \\ &\geq -\log\det(\mathbf{\Theta}_0) - \langle (\mathbf{\Theta}_0)^{-1}, \mathbf{\Delta}\rangle + \tfrac{1}{2\beta^2}\|\mathbf{\Delta}\|_F^2 + \langle \mathbf{\Theta}_0 + \mathbf{\Delta}, \boldsymbol{S}\rangle + \\ &\quad \lambda\Omega(\mathbf{\Theta}_0 - \mathrm{Diag}(\mathbf{\Theta}_0) + (\mathbf{\Delta} - \mathrm{Diag}(\mathbf{\Delta}))) - \\ &\quad (-\log\det(\mathbf{\Theta}_0) + \langle \mathbf{\Theta}_0, \boldsymbol{S}\rangle + \lambda\Omega(\mathbf{\Theta}_0 - \mathrm{Diag}(\mathbf{\Theta}_0))) \\ &= -\langle (\mathbf{\Theta}_0)^{-1}, \mathbf{\Delta}\rangle + \langle \mathbf{\Delta}, \boldsymbol{S}\rangle + \lambda\Omega(\mathbf{\Theta}_0 - \mathrm{Diag}(\mathbf{\Theta}_0) + \mathbf{\Delta} - \mathrm{Diag}(\mathbf{\Delta})) - \\ &\quad \lambda\Omega(\mathbf{\Theta}_0 - \mathrm{Diag}(\mathbf{\Theta}_0))) + \tfrac{1}{2\beta^2}\|\mathbf{\Delta}\|_F^2 \\ &= \langle \mathbf{\Delta}, S - (\mathbf{\Theta}_0)^{-1}\rangle + \lambda\Omega(\mathbf{\Theta}_0 - \mathrm{Diag}(\mathbf{\Theta}_0) + \mathbf{\Delta} - \mathrm{Diag}(\mathbf{\Delta})) \\ &\quad -\lambda\Omega(\mathbf{\Theta}_0 - \mathrm{Diag}(\mathbf{\Theta}_0)) + \tfrac{1}{2\beta^2}\|\mathbf{\Delta}\|_F^2. \end{aligned} \tag{A.23}$$

Now we lower bound each of the terms in the last line of (A.23). We have w.h.p. that,

$$\begin{aligned} |\langle \mathbf{\Delta}, S - (\mathbf{\Theta}_0)^{-1}\rangle| &\leq \|\mathbf{\Delta}\|_1 \|S - (\mathbf{\Theta}_0)^{-1}\|_\infty \\ &\leq \sqrt{2pk + p + m}\|\mathbf{\Delta}\|_F \sqrt{\tfrac{\log p}{n}} \\ &= \sqrt{\tfrac{(2pk+p+m)\log p}{n}}\|\mathbf{\Delta}\|_F, \end{aligned} \tag{A.24}$$

where we have used a concentration result (see e.g. [Rothman et al., 2008]), $\|S -$

$(\mathbf{\Theta}_0)^{-1}\|_\infty \leq \sqrt{\frac{\log p}{n}}$. Denote $\widetilde{\mathbf{\Theta}}_0 = \mathbf{\Theta}_0 + \Delta$. Then we have that,

$$\begin{aligned} |\Omega(\widetilde{\mathbf{\Theta}}_0 - \text{Diag}(\widetilde{\mathbf{\Theta}}_0)) - \Omega(\mathbf{\Theta}_0 - \text{Diag}(\mathbf{\Theta}_0)))| &\leq \Omega(\mathbf{\Delta} - \text{Diag}(\mathbf{\Delta})) \\ &\leq \tfrac{D}{\lambda}\|\mathbf{\Delta} - \text{Diag}(\mathbf{\Delta})\|_F \\ &\leq \tfrac{D}{\lambda}\|\mathbf{\Delta}\|_F, \end{aligned} \tag{A.25}$$

where the first inequality follows from triangle inequality and the second inequality follows from Lemma 22 with $D = \sqrt{2k}\lambda + \sqrt{2m}\lambda_2$.

Plugging the bounds (A.24) and (A.25) into (A.23), we have that,

$$\begin{aligned} G(\mathbf{\Delta}) &\geq -\sqrt{\tfrac{(2pk+p+m)\log p}{n}}\|\mathbf{\Delta}\|_F - (\sqrt{2k}\lambda + \sqrt{2m}\lambda_2)\|\mathbf{\Delta}\|_F + \tfrac{1}{2\beta^2}\|\mathbf{\Delta}\|_F^2 \\ &\geq \|\mathbf{\Delta}\|_F^2(\tfrac{1}{2\beta^2} - \tfrac{1}{\|\mathbf{\Delta}\|_F}\sqrt{\tfrac{(2pk+p+m)\log p}{n}} - \tfrac{1}{\|\mathbf{\Delta}\|_F}(\sqrt{2k}\lambda + \sqrt{2m}\lambda_2)) \\ &> 0 \end{aligned} \tag{A.26}$$

if $\|\mathbf{\Delta}\|_F \geq 2\beta^2(\sqrt{\frac{(2pk+p+m)\log p}{n}} + \sqrt{2k}\lambda + \sqrt{2m}\lambda_2) = Cr(p,k,m,n)$. Since $G(\widetilde{\mathbf{\Theta}} - \mathbf{\Theta}_0) < 0$, it follows that $\|\widetilde{\mathbf{\Theta}} - \mathbf{\Theta}_0\|_F \leq Cr(p,k,m,n)$.

Proof of Theorem 25

Let $\mathbf{\Delta} = \mathbf{\Delta}_V + \mathbf{\Delta}_Z$ and $\hat{\mathbf{\Delta}} = \mathbf{\Delta} + \mathbf{\Delta}^T$. Note that the optimality condition in (20) (before Section 5.2.1) is equivalent to the following two conditions:

$$\begin{aligned} -2\bar{\mathbf{R}}_L + 2\bar{\mathbf{W}}_L + 2Y_{LT}\overline{\hat{\mathbf{\Delta}}}_T + \lambda\bar{\Gamma}_L &= 0 \\ -2\bar{\mathbf{R}}_{L^c} + 2\bar{\mathbf{W}}_{L^c} + 2Y_{L^cT}\overline{\hat{\mathbf{\Delta}}}_T + \lambda\bar{\Gamma}^c_L &= 0 \end{aligned} \tag{A.27}$$

Then (A.27) is equivalent to:

$$\begin{aligned} -2\bar{\mathbf{R}}_L + 2\bar{\mathbf{W}}_L + 2\hat{Y}_{LL}\overline{\mathbf{\Delta}}_L + \lambda\bar{\Gamma}_L &= 0 \\ -2\bar{\mathbf{R}}_{L^c} + 2\bar{\mathbf{W}}_{L^c} + 2\hat{Y}_{L^cL}\overline{\mathbf{\Delta}}_L + \lambda\bar{\Gamma}^c_L &= 0 \end{aligned} \tag{A.28}$$

The first equation in (A.28) yields that

$$\overline{\mathbf{\Delta}}_L = (\hat{Y}_{LL})^{-1}(\bar{\mathbf{R}}_L - \bar{\mathbf{W}}_L - \tfrac{\lambda}{2}\bar{\Gamma}_L) \tag{A.29}$$

Plugging (A.29) into the second equation in (A.27), we have that

$$\lambda\bar{\Gamma}_{L^c} = 2\bar{\mathbf{R}}_{L^c} - 2\bar{\mathbf{W}}_{L^c} - 2\hat{Y}_{L^cL}((\hat{Y}_{LL})^{-1}(\bar{\mathbf{R}}_L - \bar{\mathbf{W}}_L - \tfrac{\lambda}{2}\bar{\Gamma}_L)) \tag{A.30}$$

Note that for any vector z, $\|\hat{Y}_{L^cL}(\hat{Y}_{LL})^{-1}z\|_{\infty,2} \leq \sqrt{p}\max_{(i,j)\in L^c}\|\hat{Y}_{(i,j),L}(\hat{Y}_{LL})^{-1})\|_{1,2}\|z\|_{\infty,2}$. Hence using the fact that $\mathbf{\Theta}_0$ satisfies the incoherence condition,

$$\begin{aligned}\lambda\|\bar{\Gamma}_{L^c}\|_{\infty,2} &\leq 2\|\bar{\mathbf{R}}_{L^c}\|_{\infty,2} + 2\|\bar{\mathbf{W}}_{L^c}\|_{\infty,2} + 2(1-\alpha)\|\bar{\mathbf{R}}_L\|_{\infty,2} + \\ &\quad 2(1-\alpha)\|\bar{\mathbf{W}}_L\|_{\infty,2} + \lambda(1-\alpha)\|\bar{\Gamma}_L\|_{\infty,2} \\ &\leq (4-2\alpha)(\|\bar{\mathbf{R}}\|_{\infty,2} + \|\bar{\mathbf{W}}\|_{\infty,2}) + \lambda(1-\alpha)\end{aligned} \tag{A.31}$$

The theorem now follows.

Proof of Lemma 26

Let $\mathbf{\Theta}_0 = \boldsymbol{V}_0 + \boldsymbol{V}_0{}^T + \boldsymbol{Z}_0 + \boldsymbol{Z}_0{}^T$. Note that $R(\hat{\mathbf{\Delta}}_V, \hat{\mathbf{\Delta}}_Z) = (\mathbf{\Theta}_0 + \hat{\mathbf{\Delta}}_V + \hat{\mathbf{\Delta}}_Z)^{-1} - (\mathbf{\Theta}_0)^{-1} + (\mathbf{\Theta}_0)^{-1}(\hat{\mathbf{\Delta}}_V + \hat{\mathbf{\Delta}}_Z)^{-1}(\mathbf{\Theta}_0)^{-1}$. Let $\hat{\mathbf{\Delta}} = \hat{\mathbf{\Delta}}_V + \hat{\mathbf{\Delta}}_Z$. Now,

$$\begin{aligned}(\mathbf{\Theta}_0 + \hat{\mathbf{\Delta}})^{-1} &= (\mathbf{\Theta}_0(I + (\mathbf{\Theta}_0)^{-1}\hat{\mathbf{\Delta}}))^{-1} \\ &= (I + (\mathbf{\Theta}_0)^{-1}\hat{\mathbf{\Delta}})^{-1}(\mathbf{\Theta}_0)^{-1} \\ &= \sum_{k=0}^{\infty}(-1)^k((\mathbf{\Theta}_0)^{-1}\hat{\mathbf{\Delta}})^k(\mathbf{\Theta}_0)^{-1} \\ &= (\mathbf{\Theta}_0)^{-1} - (\mathbf{\Theta}_0)^{-1}\hat{\mathbf{\Delta}}(\mathbf{\Theta}_0)^{-1} + \sum_{k=2}^{\infty}(-1)^k((\mathbf{\Theta}_0)^{-1}\hat{\mathbf{\Delta}})^k\mathbf{\Theta}_0^{-1} \\ &= (\mathbf{\Theta}_0)^{-1} - (\mathbf{\Theta}_0)^{-1}\hat{\mathbf{\Delta}}(\mathbf{\Theta}_0)^{-1} + (\mathbf{\Theta}_0)^{-1}\hat{\mathbf{\Delta}}(\mathbf{\Theta}_0)^{-1}\hat{\mathbf{\Delta}}J\mathbf{\Theta}_0^{-1},\end{aligned} \tag{A.32}$$

where, $J = \sum_{k=0}^{\infty}(-1)^k((\mathbf{\Theta}_0)^{-1}\hat{\mathbf{\Delta}})^k$. Thus we have that,

$$\begin{aligned}\|R(\hat{\mathbf{\Delta}}_V, \hat{\mathbf{\Delta}}_Z)\|_{\infty,2} &= \|(\mathbf{\Theta}_0)^{-1}\hat{\mathbf{\Delta}}(\mathbf{\Theta}_0)^{-1}\hat{\mathbf{\Delta}}J\mathbf{\Theta}_0^{-1}\|_{\infty,2} \\ &\leq \|(\mathbf{\Theta}_0)^{-1}\hat{\mathbf{\Delta}}(\mathbf{\Theta}_0)^{-1}\hat{\mathbf{\Delta}}J\mathbf{\Theta}_0^{-1}\| \\ &\leq \|(\mathbf{\Theta}_0)^{-1}\|^3\|\hat{\mathbf{\Delta}}\|^2\|J\|\end{aligned} \tag{A.33}$$

Now note that,

$$\begin{aligned} \|J\| &= \|\sum_{k=0}^{\infty}(-1)^k((\mathbf{\Theta}_0)^{-1}\hat{\mathbf{\Delta}})^k\| \\ &\leq \sum_{k=0}^{\infty}\|((\mathbf{\Theta}_0)^{-1}\hat{\mathbf{\Delta}})^k\| \\ &\leq \sum_{k=0}^{\infty}(\|(\mathbf{\Theta}_0)^{-1}\|\|\hat{\mathbf{\Delta}}\|)^k \\ &= \frac{1}{1-\|(\mathbf{\Theta}_0)^{-1}\|\|\hat{\mathbf{\Delta}}\|} \\ &\leq 3 \end{aligned} \tag{A.34}$$

where we assumed that $\|\hat{\mathbf{\Delta}}\| \leq \frac{1}{2\|(\mathbf{\Theta}_0)^{-1}\|}$.

Hence we have that $\|R(\hat{\mathbf{\Delta}}_V, \hat{\mathbf{\Delta}}_Z)\|_{\infty,2} \leq 3\|(\mathbf{\Theta}_0)^{-1}\|^3\|\hat{\mathbf{\Delta}}\|^2$.

Proof of Theorem 27

In the final analysis, we combine the partial consistency result (Lemma 23) along with the model-selection consistency sufficient condition (Theorem 26) and bound on $R(\mathbf{\Delta}_V, \mathbf{\Delta}_Z)$ (Lemma 26) to obtain the number of samples required for successful recovery.

In the sufficient conditions, we required that $\lambda \geq (\frac{8}{\alpha} - 4)\max\{\|\bar{\mathbf{R}}\|_{\infty,2}, \|\bar{\mathbf{W}}\|_{\infty,2}\}$. Note that $\|W\|_\infty \leq \sqrt{\frac{\log p}{n}}$. Hence $\|W\|_{\infty,2} \leq \sqrt{\frac{p\log p}{n}}$. Since $\|R(\mathbf{\Delta}_V, \mathbf{\Delta}_Z)\|_{\infty,2} \leq 3\|(\mathbf{\Theta}_0)^{-1}\|^3\|\hat{\mathbf{\Delta}}\|^2$, it would be sufficient if,

$$\lambda \geq (\frac{8}{\alpha} - 4)\max\left\{\frac{3}{\delta^3}\|\hat{\mathbf{\Delta}}\|_F^2, \sqrt{\frac{p\log p}{n}}\right\}. \tag{A.35}$$

This would be true if,

$$\lambda \geq (\frac{8}{\alpha} - 4)\max\{C_1(\sqrt{\frac{(2pk+p+m)\log p}{n}} + \sqrt{2k}\lambda + \sqrt{2m}\lambda_2)^2, \sqrt{\frac{p\log p}{n}}\}, \tag{A.36}$$

where $C_1 = \frac{12\beta^4}{\delta^3}$. This would be true if,

$$\lambda \geq (\frac{8}{\alpha} - 4)\max\{2C_1k\lambda^2, C_1\frac{(2pk+p+m)\log p}{n}, 2C_1m\lambda_2^2, \sqrt{\frac{p\log p}{n}}\}. \tag{A.37}$$

This would be true if,

$$\frac{1}{2C_1k(8/\alpha-4)} \geq \lambda \geq (8/\alpha-4)\max(C_1\frac{(2pk+p+m)\log p}{n}, 2C_1m\lambda_2^2) \quad \text{(A.38)}$$

Thus when $n \geq 2C_1^2(8/\alpha-4)^2\max(k,m)(2pk+p+m)\log p$ and $\lambda = \lambda_2 = C_1(\frac{8}{\alpha}-4)\frac{(2pk+p+m)\log p}{n}$, we have model selection consistency.

Proof of Theorem 28

Combining Lemma 23 and Theorem 27, we have that

$$\begin{aligned}
\|\hat{\mathbf{\Theta}}-\mathbf{\Theta}_0\|_F &\leq 2\beta^2\left(\lambda\sqrt{2k}+\lambda_2\sqrt{2m}+\sqrt{\tfrac{(2pk+p+m)\log p}{n}}\right) \\
&\leq 2\beta^2\left((8/\alpha-4)C_1\tfrac{(2pk+p+m)\log p}{n}(\sqrt{2k}+\sqrt{2m})\right)+ \\
&\quad 2\beta^2\left(\sqrt{\tfrac{(2pk+p+m)\log p}{n}}\right) \\
&\leq 4\beta^2\sqrt{\tfrac{(2pk+p+m)\log p)}{n}},
\end{aligned} \quad \text{(A.39)}$$

where the first inequality follows from Lemma 23, whose assumptions hold because $\text{supp}(\hat{\mathbf{\Theta}}) \subset \text{supp}(\mathbf{\Theta}_0)$ (which follows from Theorem 27). The last equality follows follows from the fact that $n \geq 8C_1^2(8/\alpha-4)^2\max(k,m)(2pk+p+m)\log p$.

Proof of Theorem 29

From Theorem 28, we have that $\|\hat{\Theta}-\Theta_0\|_F \leq O\left(\sqrt{\frac{(2pk+p+m)\log p)}{n}}\right)$. Suppose $\text{supp}(\hat{\mathbf{\Theta}}) \subset \text{supp}(\mathbf{\Theta}_0)$, i.e. there exists an $(i,j) \in T$ such that $\hat{\mathbf{\Theta}}_{ij} = 0$. Then we have by the assumption in the theorem that,

$$\begin{aligned}
\zeta &\leq \|\hat{\mathbf{\Theta}}-\mathbf{\Theta}_0\|_F \\
&\leq O\left(\sqrt{\tfrac{(2pk+p+m)\log p)}{n}}\right) \\
&< \zeta,
\end{aligned} \quad \text{(A.40)}$$

giving us a contradiction. Note that the first inequality follows from the definition of ζ and the assumption that $\text{supp}(\hat{\mathbf{\Theta}}) \subset \text{supp}(\mathbf{\Theta}_0)$.

BIBLIOGRAPHY

M. Fazel, H. Hindi, and S. Boyd. A rank minimization heuristic with application to minimum order system approximation. In *Proc. American Control Conference*, Arlington, VA, 2001.

K. Mohan and M. Fazel. Reweighted nuclear norm minimization with application to system identification. *Proceedings of American controls conference*, 2010a.

P. Gopalan, C. Wang, and D. M. Blei. Modeling overlapping communities with node popularities. *Advances in neural information processing systems*, 2013.

J. Yang and J. Leskovec. Overlapping communities explain core-periphery organization of networks. *Proceedings of the IEEE*, 102(12):1892 – 1902, 2014.

M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2), 2008.

V. Filkov. Identifying gene regulatory networks from gene expression data. *Handbook of Computational Moleculary Biology*, 2005.

G. Robins, P. Pattison, Y. Kalish, and D. Lusher. An introduction to exponential random graph models for social networks. *Social Networks*, 29(2):173 – 191, 2007.

K. Mohan and M. Fazel. Iterative reweighted algorithms for matrix rank minimization. *Journal of Machine Learning Research*, 2012.

S. Oymak, K. Mohan, M. Fazel, and B. Hassibi. A simplified approach to recovery conditions for low rank matrices. *Proceedings of the International Symposium on Information Theory*, 2011a.

K. Mohan and M. Fazel. New restricted isometry results for noisy low-rank recovery. *Proceedings of the International Symposium on Information Theory*, 2010b.

K. Tan, P. London, K. Mohan, M. Fazel, S. Lee, and D. Witten. Learning graphical models with hubs. *Submitted to Journal of Machine Learning Research*, 2014.

K. Mohan, P. London, M. Fazel, D. Witten, and S. Lee. Node-based learning of multiple gaussian graphical models. *Journal of Machine Learning Research*, 2014.

K. Mohan, M. Chung, S. Han, D. Witten, S. Lee, and M. Fazel. Structured learning of gaussian graphical models. *Advances in Neural Information Processing Systems*, 2012.

E. J. Candes and B. Recht. Exact matrix completion via convex optimziation. *Foundations of Computational Mathematics*, 9:717–772, 2009.

N. Srebro, J. D. M. Rennie, and T. S. Jakkola. Maximum-margin matrix factorization. In *Advances in Neural Information Processing Systems*, 2005.

M. Fazel, H. Hindi, and S. Boyd. Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices. In *Proc. American Control Conference*, pages 2156–2162, Denver, CO, 2003.

Z. Liu and L. Vandenberghe. Interior-point method for nuclear norm approximation with application to system identification. *SIAM J. Matrix Analysis and Appl.*, 31 (3), 2008.

D. Gross, Y. K. Liu, S. T. Flammia, S. Becker, and J. Eisert. Quantum state tomography via compressed sensing. *Physical Review Letters*, 105, 2010.

E. J. Candes and T. Tao. Decoding by linear programming. *IEEE Trans Info. Theory*, 2004.

V. Chandrasekaran, B. Recht, P. A. Parrilo, and A. S. Willsky. The convex geometry of linear inverse problems. *Foundations of Computational Mathematics*, 12(6), 2012a.

D. Needell and J. A. Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301 – 321, 2008.

D. Goldfarb and S. Ma. Convergence of fixed point continuation algorithms for matrix rank minimization. *Foundations of Computational Mathematics*, 11(2), 2011.

R. Garg and R. Khandekar. Gradient descent with sparsification: An iterative algorithm for sparse recovery with restricted isometry property. In *Proc. of 26th Intl. Conf. on Machine Learning (ICML)*, 2009.

E. J. Candes, M. B. Wakin, and S. Boyd. Enhancing sparsity by reweighted l_1 minimization. *Journal of Fourier Analysis and Applications*, 14:877–905, 2008.

M. S. Lobo, M. Fazel, and S. Boyd. Portfolio optimization with linear and fixed transaction costs. *Annals of Operations Research*, 152:341–365, 2006.

D. Needell. Noisy signal recovery via iterative reweighted l1-minimization. In *Proc. Asilomar conference on Signals, Systems and Computers*, 2009.

T. Zhang. Analysis of multi-stage convex relaxation for sparse regularization. *Journal of Machine Learning Research*, 11:1081–1107, 2010.

K. Mohan and M. Fazel. Reweighted nuclear norm minimization with application to system identification. In *Proc. American Control Conference*, Baltimore, MA, 2010c.

B. D. Rao and K. Kreutz-Delgado. An affine scaling methodology for best basis selection. *IEEE Transactions on Signal Processing*, 47:187 – 200, 1999.

D. P. Wipf and S. Nagarajan. Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions. *Journal of Selected Topics in Signal Proessing (Special Issue on Compressive Sensing)*, 4(2), 2010.

I. Daubechies, R. DeVore, M. Fornasier, and C. S. Gunturk. Iteratively re-weighted least squares minimization for sparse recovery. *Commun. Pure Appl. Math*, 63(1): 1 – 38, 2010.

K. Lee and Y. Bresler. Admira: Atomic decomposition for minimum rank approximation. *IEEE Tran. Info. Theory*, 56(9), 2010.

R. Meka, P. Jain, and I. S. Dhillon. Guaranteed rank minimization via singular value projection. In *Proc. of Neural Information Processing Systems (NIPS)*, 2010.

R. H. Keshavan and S. Oh. A gradient descent algorithm on the grassman manifold for matrix completion. 2009.

J. F. Cai, E .J. Candes, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM J. on Optimization*, 20(4):1956 – 1982, 2008.

K. C. Toh and S. W. Yun. An accelerated proximal gradient algorithm for nuclear norm regularized least squares problem. *Pacific Journal of Optimization*, 6:615–640, 2010.

R. Mazumder, T. Hastie, and R. Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *Journal of Machine Learning Research*, 11:2287 – 2322, 2010.

A. S. Lewis. Derivatives of spectral functions. *Mathematics of Operations Research*, 21(3):576 – 588, 1996.

K. Mohan and M. Fazel. Iterative reweighted least squares for matrix rank minimization. In *Proc. 48th Allerton conference on Controls and communications*, Allerton, IL, 2010d.

M. Fornasier, H. Rauhut, and R. Ward. Low-rank matrix recovery via iteratively reweighted least squares minimization, 2011.

Yu. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming Series A*, 103:127 – 152, 2005.

E. J. Candes and S. Becker. Software for singular value thresholding algorithm for matrix completion. 2010. Available at http://svt.caltech.edu/code.html.

"Movie Lens data". http://www.grouplens.org/node/73.

R. H. Keshavan, A. Montanari, and S. Oh. Low-rank matrix completion with noisy observations: a quantitative comparison. In *Proc. 47th Annual Allerton Conference on Communication, Control, and Computing*, 2009.

E. J. Candes. The restricted isometry property and its implications for compressed sensing. *Compte Rendus de l'Academie des Sciences, Paris, Serie I*, 346:589–592, 2008.

Y. Zhang. Theory of compressive sensing via l_1 minimization: A non-rip analysis and extensions. 2008.

E. J. Candes and Y. Plan. Tight oracle bounds for low-rank matrix recovery from a minimal number of random measurements. *IEEE Transactions on Info. Theory*, 57(4):2342–2359, 2009.

S. Oymak and B. Hassibi. New null space results and recovery thresholds for matrix rank minimization. 2010. Available at http://arxiv.org/abs/1011.6326.

S. Oymak, K. Mohan, M. Fazel, and B. Hassibi. A simplified approach to recovery conditions for low rank matrices. In *Proc. International Symposium on Information Theory*, 2011b.

R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1991.

A. W. Marshall and I. Olkin. *Inequalities: Theory of Majorization and Its Applications*. 1979.

P. Danaher, P. Wang, and D. M. Witten. The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society: Series B*, 2013.

K. Mohan, P. London, M. Fazel, D. Witten, and S. Lee. Node-based learning of multiple gaussian graphical models. *Journal of Machine Learning Research*, 2013.

J. Guo, E. Levina, G. Michailidis, and J. Zhu. Joint estimation of multiple graphical models. *Biometrika*, 98(1):1–15, 2010.

S. Hara and T. Washio. Learning a common substructure of multiple graphical gaussian models. *Nural Networks*, 38:23–28, 2013.

N. Friedman, I. Nachman, and D. Peer. Learning bayesian network structure from massive datasets: The "sparse candidate" algorithm. *Proc. 50th Conference on Uncertainity in Artificial Intelligence*, 1999.

T. Jaakkola, D. Sontag, A. Golberson, and M. Meila. Learning Bayesian network structures using LP relaxations. *Proc. of the 13th International Conference on Artificial Intelligence and Statistics*, 2010.

S. Acid, L. M. de Campos, J. M. Fernandez-Luna, S. Rodriguez, J. M. Rodriguez, and J. L. Salcedo. A comparison of learning algorithms for bayesian networks: a case study based on data from an emergency medical service. *Artificial Intelligence in Medicine*, 30:215 – 232, 2004.

D. Heckerman. A tutorial on learning with bayesian networks. 1995. Technical report, MSR-TR-95-06.

D. Koller and N. Friedman. Probabilistic graphical models: Principles and techniques. *The MIT Press*, 2009.

J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2007.

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

P. Ravikumar, M. J. Wainwright, G. Raskutti, and B. Yu. High-dimensional covariance estimation by minimizing l1-penalized log-determinant divergence. *Electronic Journal of Statistics*, 5:935 – 980, 2011.

G. Obozinski, L. Jacob, and J.-P. Vert. Group lasso with overlaps: the latent group lasso approach. Technical report. `arxiv.org/abs/1110.0413`.

L. Jacob, G. obozinski, and J.-P. Vert. Group lasso with overlap and graph lasso. *International conference on machine learning*, 2009.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68.

H. Zou, T. Hastie, and R. Tibshirani. On the "degrees of freedom" of the lasso. *The Annals of Statistics*, 35(5):2173–2192, 2007.

J. Eckstein. Augmented lagrangian and alternating direction methods for convex optimization: A tutorial and some illustrative computational results. 2012. RUTCOR research report RRR 32-2012.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and trends in machine learning*, 3(1):1–122, 2011.

S. Ma, L. Xue, and H. Zou. Alternating direction methods for latent variable gaussian graphical model selection. *Neural Computation*, 2013.

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

Meinshausen and Bülhmann. High-dimensional graphs and variable selection with the lasso. *Annals of Statistics*, 34:1436 – 1462, 2006.

A. Hero and B. Rajaratnam. Hub discovery in partial correlation graphs. *IEEE Transactions on Information Theory*, 58(9), 2014.

Q. Liu and A. Ihler. Learning scale free networks by reweighted ℓ_1 regularization. *AISTATS*, 2011.

J. Peng, P. Wang, N. Zhou, and J. Zhu. Partial correlation estimation by joint sparse regression model. *Journal of the American Statistical Association*, 104(486):735–746, 2009.

Verhaak et al. Integrated genomic analysis identifies clinically relevant subtypes of glioblastoma characterized by abnormalities in PDGFRA, IDH1, EGFR, and NF1. *Cancer Cell*, 17(1):98–110, 2010.

N. Rappaport, N. Nativ, G. Stezler, M. Twik, Y. Guan-Golan, TI. Stein, I. Bahir, F. Belinky, CP. Morrey, M. Safran, and D. Lancet. MalaCards: an integrated compendium for diseases and their annotation. *Database (Oxford)*, 2013.

D. Maglott, J. Ostell, KD. Pruitt, and T. Tatusova. Entrez Gene: gene-centered information at NCBI. *Nucleic Acids Research*, 33(D):54–58, 2004.

J. E. Gentle. *Matrix Algebra: Theory, Computations, and Applications in Statistics*. Springer, New York, 2007.

G. Varoquaux, A. Gramfort, J.B. Poline, B. Thirion, R. Zemel, and J. Shawe-Taylor. Brain covariance selection: better individual functional connectivity models using population prior. *Advances in Neural Information Processing Systems*, 2010.

J. Honorio and D. Samaras. Multi-task learning of gaussian graphical models. *Proc. of the 27th International Conference on Machine Learning*, 2010.

B. Zhang and Y. Wang. Learning structural changes of gaussian graphical models in controlled experiments. *Proc. of the 26th Conference on Uncertainity in Artificial Intelligence*, 2010.

C.-J. Hsieh, M. Sustik, I. Dhillon, and P. Ravikumar. Sparse inverse covariance matrix estimation using quadratic approximation. *Advances in Neural Information Processing Systems*, 2011.

R. Mazumder and T. Hastie. The graphical lasso: New insights and alternatives. *Electronic Journal of Statistics*, pages 2125 – 2149, 2012.

S. Yun, P. Tseng, and K.-C. Toh. A block coordinate gradient descent method for regularized convex separable optimization and covariance selection. *Mathematical programming*, pages 331 – 355, 2011.

D. M. Witten, J. Friedman, and N. Simon. New insights and faster computations for the graphical lasso. *Journal of Computational and Graphical Statistics*, 20:892–900, 2011.

R. E. Tarjan. Depth-first search and linear graph algorithms. *SIAM Journal on Computing*, 1(2):146–160, 1972.

C.-J. Hsieh, M. Sustik, I. Dhillon, P. Ravikumar, and R. Poldrack. Big & quic: Sparse inverse covariance estimation for a million variables. *Advances in Neural Information Processing Systems*, 2012.

Q. T. Dinh, A. Kyrillidis, and V. Cevher. A proximal newton framework for composite minimization: Graph learninig without cholesky decompositions and matrix inversions. *30th International Conference on Machine Learning*, 2013.

V. Chandrasekaran, P. A. Parrilo, and A. S. Willsky. Latent variable graphical model selection via convex optimization. *Annals of Statistics*, 40(4), 2012b.

M. A. Woodbury. Inverting modified matrices. *Memorandum Report 42, Statistical research group, Princeton university*, 1950.

K. B. Petersen and M. S. Pedersen. The matrix cookbook. nov 2012. URL http://www2.imm.dtu.dk/pubdb/p.php?3274. Version 20121115.

W. E. Bishop and B. M. Yu. Deterministic symmetric positive semidefinite matrix completion. *Advances in Neural Information Processing Systems*, 2014.

B. Mishra, G. Meyer, and R. Sepulchre. Low-rank optimization for distance matrix completion. *Proceedings of the 50th IEEE Conference on Decision and Control*, 2011.

P. Ravikumar, M. J. Wainwright, and J. Lafferty. High-dimensional ising model selection using l1-regularized logistic regression. *Annals of Statistics*, 38(3):1287–1319, 2010.

C. Lam and J. Fan. Sparsistency and rates of convergence in large covariance matrix estimation. *Annals of Statistics*, 37(6B):4254 – 4278, 2009.

C. C. Johnson, A. Jalali, and P. Ravikumar. High-dimensional sparse inverse covariance estimation using greedy methods. *Proceedings of the fifteenth international conference on artificial intelligence and statistics*, 2012.

A. Jalali, C. C. Johnson, and P. Ravikumar. On learning discrete graphical models using greedy methods. *Advances in Neural Information Processing Systems*, 2011.

A. Defazio and T. Caetano. A convex formulation for learning scale-free network via submodular relaxation. *NIPS*, 2012.

R. Tandon and P. Ravikumar. Learning graphs with a few hubs. *International conference on machine learning*, 2014.

N. Rao, C. Cox, R. Nowak, and T. Rogers. Sparse overlapping sets lasso for multitask learning and its application to fmri analysis. *Advances in Neural Information Processing Systems*, 2013.

S. Negahban and M. J. Wainwright. Simultaneous support recovery in high dimensions: Benefits and perils of block ℓ_1/ℓ_∞-regularization. *IEEE Trasactions on Information Theory*, 57(6):3841–3863, 2011.

VITA

Karthik Mohan received his B.Tech and M.Tech in civil engineering from IIT Madras in 2006. He then came to UW in 2006 to pursue his Phd in civil engineering but midway decided to switch to Electrical Engineering where he worked with Prof. Maryam Fazel on optimization algorithms for machine learning problems. He received his M.S. and Phd in Electrical Engineering in 2015.