Journal of Machine Learning Research 21 (2020) 1-60 Submitted 4/19; Revised 11/19; Published 1/20

A Unified Framework for Structured Graph Learning via Spectral Constraints

Sandeep Kumar EESANDEEP@UST.HK
Department of Industrial Engineering and Data Analytics
The Hong Kong University of Science and Technology
Clear Water Bay, Hong Kong

Jiaxi Ying* JX.YING@CONNECT.UST.HK
Department of Electronic and Computer Engineering
The Hong Kong University of Science and Technology
Clear Water Bay, Hong Kong

José Vinícius de M. Cardoso JVDMC@CONNECT.UST.HK
Department of Electronic and Computer Engineering
The Hong Kong University of Science and Technology
Clear Water Bay, Hong Kong

Daniel P. Palomar PALOMAR@UST.HK
Department of Electronic and Computer Engineering
Department of Industrial Engineering and Data Analytics
The Hong Kong University of Science and Technology
Clear Water Bay, Hong Kong

Editor: Koji Tsuda

Abstract

Graph learning from data is a canonical problem that has received substantial attention in the literature. Learning a structured graph is essential for interpretability and identification of the relationships among data. In general, learning a graph with a specific structure is an NP-hard combinatorial problem and thus designing a general tractable algorithm is challenging. Some useful structured graphs include connected, sparse, multi-component, bipartite, and regular graphs. In this paper, we introduce a unified framework for structured graph learning that combines Gaussian graphical model and spectral graph theory. We propose to convert combinatorial structural constraints into spectral constraints on graph matrices and develop an optimization framework based on block majorization-minimization to solve structured graph learning problem. The proposed algorithms are provably convergent and practically amenable for a number of graph based applications such as data clustering. Extensive numerical experiments with both synthetic and real data sets illustrate the effectiveness of the proposed algorithms. An open source R package containing the code for all the experiments is available at https://CRAN.R-project.org/package=spectralGraphTopology.

Keywords: Structured graph learning, spectral graph theory, Markov random field, Gaussian graphical model, Laplacian matrix, clustering, adjacency matrix, bipartite structure, spectral similarity.

*. Corresponding author.
Part of the results in this paper appeared in NeurIPS'19 (Kumar et al., 2019).

1. Introduction

Graphs are fundamental mathematical structures consisting of a set of nodes and weighted edges connecting them. The weight associated with each edge represents the similarity between the two connected nodes. Graphical models provide an effective way to characterize relationships among data variables across numerous applications (Barabási et al., 2016; Wang et al., 2018; Friedman et al., 2008; Guo et al., 2011; Segarra et al., 2017; Banerjee et al., 2008; Meinshausen and Bühlmann, 2006). Gaussian graphical modeling (GGM) encodes conditional dependence relationships among a set of variables (Dempster, 1972; Lauritzen, 1996). GGM is a tool of increasing importance in a number of fields including finance, biology, statistical learning, and computer vision (Friedman et al., 2008). In this framework, an undirected weighted graph is matched to the variables, where each vertex corresponds to a variable and an edge is present between two vertices if the corresponding random variables are conditionally dependent (Lauritzen, 1996). Graphical models encode the dependencies among the data in the form of a graph matrix such that its non-zero entries quantify the dependencies between two variables.

Let $\mathbf{x} = [x_1, x_2, \ldots, x_p]^\top$ be a p-dimensional zero mean multivariate random variable associated with an undirected graph and $S \in \mathbb{R}^{p\times p}$ be the sample covariance matrix (SCM) calculated from n number of observations, then the GGM method learns a graph via the following optimization problem:

$$\underset{\Theta\in\mathcal{S}^p_{++}}{\mathsf{maximize}}\ \log\det(\Theta) - \mathrm{tr}\big(\Theta S\big) - \alpha h(\Theta), \tag{1}$$

where $\Theta \in \mathbb{R}^{p\times p}$ denotes the graph matrix to be estimated, p is the number of nodes (vertices) in the graph, $\mathcal{S}^p_{++} \in \mathbb{R}^{p\times p}$ denotes the set of positive definite matrices, $h(\cdot)$ is a regularization term, and $\alpha > 0$ is the regularization hyperparameter. If the observed data is distributed according to a zero mean p-variate Gaussian distribution, then the optimization in (1) corresponds to the penalized maximum likelihood estimation (MLE) of the inverse covariance (precision) matrix of a Gaussian random vector also known as Gaussian Markov Random Field (GMRF). With the refrerence to graph matrix Θ, the random vector $\mathbf{x}$ follows the Markov property: $\Theta_{ij} \neq 0$ implies x_i and x_j are conditionally dependent given the rest of the variables (Lauritzen, 1996; Dempster, 1972).

Prior knowledge about the underlying graph structure is frequently available in a number of real-world applications. For instance, in gene network analysis, genes can be grouped into pathways, and connections within a pathway might be more likely than connections between pathways, forming a cluster (Marlin and Murphy, 2009). For better interpretability and identification of the structure in the data, it is desirable to enforce specific structures on the learned graph matrix Θ. Furthermore, a structured graph can be directly applied in tasks such as community detection, clustering, and causal inference.

Few of the concrete examples where the prior information about graph structure is available. In some clustering applications, the number of clusters can be interpreted, e.g., the number of sectors in clustering the financial stocks, the number of digits in the classification of the handwritten digits (LeCun, 1998). Bipartite structure is a required for constructing two channel filter banks (Narang and Ortega, 2012); $k-$component bipartite graphs are employed for co-clustering (Dhillon, 2001); a regular graph is the best-known communication efficient structure, thus for designing a communication efficient deep learning architectures

the requirement of having a regular graph structure is known in advance (Prabhu et al., 2018; Chow et al., 2016). For multi-resolution transforms and for designing sampling algorithms, the requirement of tree graph structure is already known (Gavish et al., 2010; Shen and Ortega, 2010). For many graph signal processing (GSP) tasks, the requirement of sparse and connected graph representation is known (Sundin et al., 2017).

It is known that if the goal is structured graph learning, structure inference and graph weight estimation should be done jointly (Ambroise et al., 2009; Hao et al., 2018). Performing structure inference (also known as model selection) before weight estimation (also known as parameter estimation) results in a suboptimal procedure (Ambroise et al., 2009). Although GGM has been extended to incorporate structures on the learned graph, most of the existing methods perform graph structure learning and graph weight estimation separately. Essentially, the methods are either able to infer connectivity information (Ambroise et al., 2009) or to perform graph weight estimation in case the connectivity information is known *a priori* (Lee and Liu, 2015; Wang, 2015; Cai et al., 2016; Danaher et al., 2014; Pavez et al., 2018; Egilmez et al., 2017), but not both tasks simultaneously. Furthermore, there are few recent works considering the two tasks jointly, but those methods are limited to some specific structures (e.g., multi-component in Hao et al., 2018) that are not trivially extended to other graph structures. In addition, these methods involve computationally demanding steps that make them unsuitable for big data applications.

In general, structured graph learning is an NP-hard combinatorial problem (Anandkumar et al., 2012; Bogdanov et al., 2008), thus the task of designing an optimization method to solve it is inherently challenging. In this paper, we introduce spectral graph theory tools into the GGM learning framework so as to convert the combinatorial constraints of graph structures into constraints on the eigenvalues of graph matrices. By realizing that the structural properties of important families of graphs are encoded in the eigenvalues of their graph matrices, we develop a general framework for structured graph learning by translating the combinatorial constraints into the corresponding spectral constraints. We develop practically usable and theoretically convergent algorithms that are able to learn graph structures and weights simultaneously.

1.1. Related Work

The penalized maximum likelihood approach with sparsity regularization has been widely studied for the precision matrix estimation. An ℓ_1-norm regularization, $h(\Theta) = \sum_{i,j} |\Theta_{ij}|$, which promotes element-wise sparsity on the graph matrix Θ, is a common choice of regularization function to learn a sparse structure (Yuan and Lin, 2007; Shojaie and Michailidis, 2010a,b; Ravikumar et al., 2010; Mazumder and Hastie, 2012; Fattahi and Sojoudi, 2019). In Friedman et al. (2008), the authors came up with an efficient computational method to solve (1) and proposed the well-known GLasso algorithm. In addition, non-convex penalties are proposed for sparse precision matrix estimation to reduce estimation bias (Shen et al., 2012; Lam and Fan, 2009). However, if a specific structure is required then simply a sparse graphical modeling is not sufficient (Heinävaara et al., 2016; Tarzanagh and Michailidis, 2017). In this sense, an extension of sparse GGM model is motivated so as to incorporate more specific structures.

In this direction, the work in Ambroise et al. (2009) has considered the problem of graph connectivity inference for multi-component structures and developed a two-stage framework that combines the expectation maximization (EM) algorithm and the graphical lasso framework. The works in Lee and Liu (2015); Wang (2015); Cai et al. (2016); Danaher et al. (2014); Guo et al. (2011); Sun et al. (2015); Tan et al. (2015) have considered the problem of estimating the edge weights with known connectivity information. However, prior knowledge of the connectivity information is not always available particularly for complex data sets with unknown population structures (Hao et al., 2018; Jeziorski and Segal, 2015). Furthermore, when it comes to simultaneous connectivity inference and graph weight estimation, two-stage methods based on Bayesian models (Marlin and Murphy, 2009) and expectation maximization (Hao et al., 2018) were proposed, but these methods are computationally prohibitive and limited to multi-component graph learning.

Other important graph structures have also been considered, including factor models (Meng et al., 2014), scale free (Liu and Ihler, 2011), eigenvector centrality prior (Fiori et al., 2012), degree-distribution (Huang and Jebara, 2008), overlapping structures with multiple graphical models (Tarzanagh and Michailidis, 2017; Mohan et al., 2014), tree structures (Chow and Liu, 1968; Anandkumar et al., 2012), and topology estimation from partial observations (Coutino et al., 2019). Recently, there has been a significant interest in enforcing the Laplacian structure (Lake and Tenenbaum, 2010; Slawski and Hein, 2015; Pavez and Ortega, 2016; Kalofolias, 2016; Egilmez et al., 2017; Pavez et al., 2018; Zhao et al., 2019), but all these methods are limited to learning graphs without specific structural constraints. The work by (Segarra et al., 2017) introduces a sparse estimation problem to infer the graph topology from the given eigenbasis associated with a specific shift operator. This formulation estimates a graph topology by optimizing the eigenvalues and assume the eigenvectors to be fixed and known. On the other hand, our proposed formulation estimates a graph structure via the optimization of both the eigenvalues and the eigenvectors, thus providing a higher flexibility in capturing the spectral properties of the graph matrix so as to ensure specific structural properties on the graph to be learned.

Due to the complexity posed by the graph learning problem, owing to its combinatorial nature, existing methods are tailored to specific structures that cannot be generalized to other graph structures. In addition, existing methods often require the connectivity information for graph weight estimation, while also involving multi-stage frameworks that are computationally prohibitive. Finally, to the best of our knowledge, there is no GGM framework that is capable of learning a graph with structures such as bipartite, regular, and multi-component bipartite.

1.2. Summary of Contributions

Enforcing a structure onto a graph is generally an NP-hard combinatorial problem, which is often difficult to solve via existing methods. In this paper, we propose a unified framework for structured graph learning. Our contributions are threefold:

1. We introduce new problem formulations that convert the combinatorial structural constraints into spectral constraints on Laplacian and adjacency matrices, resulting in three main formulations:

- **Structured graph learning via Laplacian spectral constraints**
  This formulation uses the Laplacian matrix spectral properties to learn the following graphs: *multi-component, regular, multi-component regular, sparse connected*, and other specific structured graphs.
- **Structured graph learning via adjacency spectral constraints**
  This formulation uses spectral properties of the adjacency matrix for *bipartite graph* learning.
- **Structured graph learning via Laplacian and adjacency spectral constraints**
  Under this formulation we simultaneously use spectral properties of Laplacian and adjacency matrices to enforce non-trivial structures including: *bipartite-regular, multi-component bipartite*, and *multi-component bipartite-regular*.

2. We develop algorithms based on the block majorization-minimization (MM) framework also known as block successive upper-bound minimization (BSUM) to solve the proposed formulations. The algorithms are theoretically convergent and with worst case complexity $O(p^3)$, which is same as that of GLasso.

3. The effectiveness of the proposed algorithms were extensively verified through a variety of experiments involving both synthetic and real data sets. The results showed that our proposed algorithm outperforms the state-of-the-art ones under relative error and F-score. Furthermore, to the best of our knowledge, this is the first work to provide a framework for structured graph learning based on Gaussian Markov random fields and spectral graph theory.

1.3. Outline and Notation

The remainder of the paper is organized as follows. The generalized problem formulation and related background are provided in Section 2. The detailed development of the algorithms and the associated convergence results are presented in Sections 3, 4, and 5. Experimental results involving both real and synthetic data sets are provided in Section 6. Finally, Section 7 draws conclusions and discusses a number of natural extensions of the proposed framework.

In terms of notation, lower case (bold) letters denote scalars (vectors) and upper case letters denote matrices. The dimension of a matrix is omitted whenever it is clear from the context. The (i,j)-th entry of a matrix X is denoted either by $[X]_{ij}$ or X_{ij}. $X^\dagger$ and $X^\top$ denote the pseudo inverse and transpose of matrix X, respectively. The all-zero and all-one vectors or matrices of appropriate sizes are denoted by $\mathbf{0}$ and $\mathbf{1}$, respectively. $\|X\|_1$, $\|X\|_F$ denote ℓ_1-norm and Frobenius norm of X, respectively. The Euclidean norm of the vector $\mathbf{x}$ is denoted as $\|\mathbf{x}\|_2$, $\mathrm{gdet}(X)$ is defined as the generalized determinant of a positive semidefinite matrix X, i.e., the product of its non-zero eigenvalues. The inner product of two matrices is defined as $\langle X, Y\rangle = \mathrm{tr}(X^\top Y)$, where $\mathrm{tr}(\cdot)$ is the trace operator. $\mathrm{Diag}(X)$ is a diagonal matrix with diagonal elements of X filling its principal diagonal and $\mathrm{diag}(X)$ is a vector with diagonal elements of X as the vector elements. Element-wise multiplication between matrices A and B is denoted as $A \odot B$.

2. Problem Formulation

A graph is denoted by a triple $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$, where $\mathcal{V} = \{1, 2, \ldots, p\}$ is the vertex set, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set which contains the possible unordered combinations of pair of nodes $\{(i,j)\}_{i,j=1}^{p}$, and W is the adjacency (weight) matrix. We consider a simple undirected graph with nonnegative weights $W_{ij} \geq 0$, without self-loops, and whose edge set contains only distinct pairs. The entries of the weight matrix W are: $W_{ij} \neq 0$, if $(i,j) \in \mathcal{E}$ and $W_{ij} = 0$, if $(i,j) \notin \mathcal{E}$ or $i = j$. Graphs are conveniently represented by some matrix (such as Laplacian and adjacency graph matrices), whose positive entries correspond to edges in the graph. The choice of a matrix usually depends on modeling assumptions, properties of the desired graph, applications, and theoretical requirements.

A matrix $\Theta \in \mathbb{R}^{p \times p}$ is called a graph Laplacian matrix when its elements satisfy

$$\mathcal{S}_\Theta = \left\{ \Theta \in \mathbb{R}^{p \times p} | \Theta_{ij} = \Theta_{ji} \leq 0 \text{ for } i \neq j; \Theta_{ii} = -\sum_{j \neq i} \Theta_{ij} \right\}. \tag{2}$$

As a consequence, a Laplacian matrix Θ is: i) diagonally dominant; ii) positive semidefinite, implied from the diagonally dominant property (den Hertog et al., 1993, Proposition 2.2.20.); iii) and an M-matrix, i.e., a positive semidefinite matrix with non-positive off-diagonal elements (Slawski and Hein, 2015). In addition, a Laplacian matrix has zero row sum and column sum, i.e., $\Theta_{ii} + \sum_{j \neq i} \Theta_{ij} = 0$, which means that the vector $\mathbf{1} = [1, 1, \ldots, 1]^\top$ satisfies $\Theta \mathbf{1} = \mathbf{0}$ (Chung, 1997).

The adjacency matrix W associated with a Laplacian matrix Θ is defined as

$$W_{ij} = \begin{cases} -\Theta_{ij}, & \text{if } i \neq j, \\ 0, & \text{if } i = j. \end{cases} \tag{3}$$

The positive entries of W encode the edge weights of a graph, whereas $W_{ij} = 0, i \neq j$, implies there is no connectivity between vertices i and j.

Definition 1. Let Θ be a $p \times p$ symmetric positive semidefinite matrix with rank $p - k > 0$. Then $\mathbf{x} = [x_1, x_2, \ldots, x_p]^\top$ is an improper GMRF (IGMRF) of rank $p - k$ with parameters $(\boldsymbol{\mu}, \Theta)$, assuming $\boldsymbol{\mu} = \mathbf{0}$ without loss of generality, whenever its probability density is

$$p(\mathbf{x}) = (2\pi)^{\frac{-(p-k)}{2}} (\text{gdet}(\Theta))^{\frac{1}{2}} \exp\left(-\frac{1}{2} \left(\mathbf{x}^\top \Theta \mathbf{x} \right) \right), \tag{4}$$

where $\text{gdet}(\cdot)$ denotes the generalized determinant (Rue and Held, 2005) defined as the product of the non-zero eigenvalues of Θ. Furthermore, $\mathbf{x}$ is called an IGMRF with respect to a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$ having its matrix representation as Θ, where

$$\Theta_{ij} \neq 0 \iff \{i,j\} \in \mathcal{E} \; \forall \; i \neq j, \tag{5}$$

$$\Theta_{ij} = 0 \iff x_i \perp x_j | \mathbf{x} / \{x_i, x_j\}. \tag{6}$$

These relations simply state that the nonzero pattern of Θ determines $\mathcal{G}$, so we can infer from Θ whether x_i and x_j are conditionally independent. If the rank of Θ is exactly p, then $\mathbf{x}$ is called a GMRF and the parameters $\boldsymbol{\mu}$ and Θ represent the mean and precision matrix

of a p-variate Gaussian distribution (Rue and Held, 2005). In addition, if the precision matrix Θ contains only non-positive off-diagonal entries (Slawski and Hein, 2015), then the random vector $\mathbf{x}$ is called an *attractive* GMRF and if the rank of Θ is less than p then it is also called an attractive IGMRF.

2.1. A General Framework for Graph Learning under Spectral Constraints

A natural formulation to learn a Laplacian matrix with a specific structure would be via introducing eigenvalue constraints that are motivated from *a priori* information. In this regard, we introduce a general optimization framework for structured graph learning via spectral constraints on the graph Laplacian matrix as follows

$$\begin{array}{ll} \underset{\Theta}{\mathsf{maximize}} & \log\ \mathrm{gdet}(\Theta) - \mathrm{tr}\big(\Theta S\big) - \alpha h(\Theta), \\ \mathsf{subject\ to} & \Theta \in \mathcal{S}_\Theta,\ \lambda(\mathcal{T}(\Theta)) \in \mathcal{S}_\mathcal{T}, \end{array} \tag{7}$$

where S denotes the observed data statistics (e.g., sample covariance matrix), $\mathcal{S}_\Theta$ is the Laplacian matrix structural constraint set as in (2), $h(\cdot)$ is a regularization term (e.g., ℓ_1-norm), and $\lambda(\mathcal{T}(\Theta))$ denotes the eigenvalues of $\mathcal{T}(\Theta)$ with an increasing order, where $\mathcal{T}(\cdot)$ is a transformation on the matrix Θ. For example, if $\mathcal{T}$ is the identity mapping, then $\mathcal{T}(\Theta) = \Theta$, which implies the constraints are imposed on the eigenvalues of the Laplacian matrix Θ; if $\mathcal{T}(\Theta) = W$ as defined in (3), then the constraints are imposed on the eigenvalues of the adjacency matrix W. Finally, $\mathcal{S}_\mathcal{T}$ is the set containing the eigenvalues constraints.

Fundamentally, formulation (7) is designed to learn a structured graph Laplacian matrix Θ given data statistics S, where $\mathcal{S}_\Theta$ enforces a Laplacian matrix structure and $\mathcal{S}_\mathcal{T}$ allows the inclusion of a desired graph structure via constraints on the eigenvalues. Observe that, in formulation (7), we have converted the intractable combinatorial structural constraints into simple spectral constraints. In this way, the structured graph learning problem becomes a matrix optimization problem under a proper choice of spectral constraints.

Remark 1. Apart from enforcing structure onto a graph, the Laplacian matrix is also desirable for a number of practical and theoretical considerations, such as: i) Laplacian matrices are widely used in spectral graph theory, machine learning, graph regularization, graph signal processing, and graph convolution networks (Smola and Kondor, 2003; Defferrard et al., 2016; Egilmez et al., 2017; Chung, 1997); ii) in the high-dimensional setting where the number of the data samples is smaller than the dimension of the data, learning Θ as an M-matrix greatly simplifies the optimization problem because it avoids the need for the explicit regularization term $h(\cdot)$ (Slawski and Hein, 2015); iii) graph Laplacian matrices play a crucial role for the GMRF framework, which requires the matrix Θ to be positive semi-definite (Rue and Held, 2005); iv) graph Laplacian allows flexibility in incorporating well-known spectral properties of graph matrices (Chung, 1997; Spielman and Teng, 2011).

Remark 2. From a probabilistic perspective, whenever the similarity matrix S is the sample covariance matrix of Gaussian data, (7) can be viewed as a penalized maximum likelihood estimation problem of structured precision matrix of an improper attractive GMRF model (see Definition 1). In a more general setting with non-Gaussian distribution, if the similarity matrix is positive definite, (7) can be related to the log-determinant Bregman divergence regularized optimization problem (Dhillon and Tropp, 2007; Duchi et al., 2008; Slawski and

Hein, 2015), where the goal is to find the parameters of a multivariate Gaussian model that best approximates the data.

In the next subsections, we specialize the optimization framework (7) under Laplacian eigenvalue constraints, adjacency eigenvalue constraints, and joint Laplacian and adjacency eigenvalue constraints.

2.2. Structured Graph Learning via Laplacian Spectral Constraints

To enforce spectral constraints on the Laplacian matrix Θ, i.e., $\mathcal{T}(\Theta) = \Theta$ in (7), we consider the following optimization problem

$$\begin{array}{ll} \underset{\Theta,\boldsymbol{\lambda},U}{\mathsf{maximize}} & \log\ \mathrm{gdet}(\Theta) - \mathrm{tr}\big(\Theta S\big) - \alpha h(\Theta), \\ \mathsf{subject\ to} & \Theta \in \mathcal{S}_\Theta,\ \Theta = U\mathrm{Diag}(\boldsymbol{\lambda})U^\top,\ \boldsymbol{\lambda} \in \mathcal{S}_\lambda,\ U^\top U = I, \end{array} \tag{8}$$

where Θ admits the decomposition $\Theta = U\mathrm{Diag}(\boldsymbol{\lambda})U^\top$, $\mathrm{Diag}(\boldsymbol{\lambda}) \in \mathbb{R}^{p\times p}$ is a diagonal matrix containing $\boldsymbol{\lambda} = \{\lambda_i\}_{i=1}^p$ on its diagonal with $\boldsymbol{\lambda} \in \mathcal{S}_\lambda$, and $U \in \mathbb{R}^{p\times p}$ is a matrix satisfying $U^\top U = I$. We enforce Θ to be a Laplacian matrix via the constraint $\Theta \in \mathcal{S}_\Theta$, while incorporating additional spectral constraints on Θ by imposing $\Theta = U\mathrm{Diag}(\boldsymbol{\lambda})U^\top$ with $\boldsymbol{\lambda} \in \mathcal{S}_\lambda$, where $\mathcal{S}_\lambda$ contains the desired spectral constraints according to the target graph structure.

In what follows, we introduce various choices of $\mathcal{S}_\lambda$ that enables (8) to learn a variety of key graph structures.

2.2.1. k-component graph

A graph is said to be k-component if its vertex set can be partitioned into k disjoint subsets $\mathcal{V} = \cup_{i=1}^k \mathcal{V}_i$, i.e., any two nodes belonging to different subsets are not connected by an edge. Any edge in edge set $\mathcal{E}_i \subset \mathcal{E}$ have end points in $\mathcal{V}_i$, and no edge connects two different components. The k-component structural property of a graph is naturally encoded in the eigenvalues of its Laplacian matrix. The multiplicity of the zero eigenvalue of a Laplacian matrix gives the number of connected components of a graph $\mathcal{G}$. In particular, for $k = 1$ the structure is a connected graph. Figure 1 depicts a k-component graph and its Laplacian eigenvalues with $k = 3$ connected components.

Theorem 1. *(Chung, 1997) The eigenvalues of any Laplacian matrix can be expressed as:*

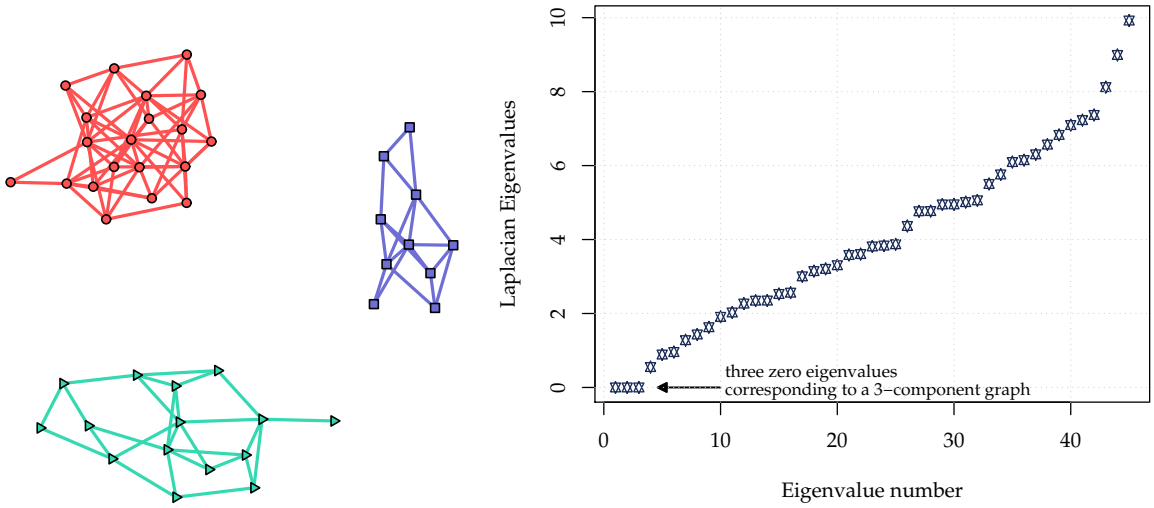
$$\mathcal{S}_\lambda = \left\{\boldsymbol{\lambda} \in \mathbb{R}^p | \{\lambda_j = 0\}_{j=1}^k,\ c_1 \le \lambda_{k+1} \le \cdots \le \lambda_p \le c_2\right\}, \tag{9}$$

where $k \ge 1$ denotes the number of connected components in the graph, and $c_1 > 0, c_2 > 0$ are constants that depend on the number of edges and their weights (Spielman and Teng, 2011).

2.2.2. d-regular graph

In a d-regular graph, all the nodes have the same weighted degree ($d_i = d,\ \forall i = 1, 2, \ldots, p$), where the weighted degree of the i-th node is defined as $d_i = \sum_j W_{ij}$, which results in

$$\Theta = dI - W,\ \ \mathrm{diag}(\Theta) = d\mathbf{1},\ \ W\mathbf{1} = d\mathbf{1}.$$



(a) A 3-component graph. (b) Eigenvalue distribution of a 3-component graph.

Figure 1: A 3-component graph and the eigenvalue distribution of its Laplacian matrix: three zero eigenvalues corresponding to three components.

Within the formulation (8), a $d-$regular structure on the matrix Θ can be enforced by including the following constraints

$$\mathcal{S}_\lambda = \{\boldsymbol{\lambda} \in \mathbb{R}^p | \lambda_1 = 0, c_1 \leq \lambda_2 \leq \cdots \leq \lambda_p \leq c_2\}, \ \mathrm{diag}(\Theta) = d\mathbf{1}. \tag{10}$$

2.2.3. k-component d-regular graph

A k-component d-regular graph, also known as clustered regular graph, is key in providing improved perceptual grouping (Kim and Choi, 2009) for clustering applications. Within the formulation (8), we can enforce this structure by including the following constraints

$$\mathcal{S}_\lambda = \left\{\boldsymbol{\lambda} \in \mathbb{R}^p | \{\lambda_j = 0\}_{j=1}^k, \ c_1 \leq \lambda_{k+1} \leq \cdots \leq \lambda_p \leq c_2\right\}, \ \mathrm{diag}(\Theta) = d\mathbf{1}. \tag{11}$$

2.2.4. Cospectral graphs

Many applications can be formulated to learn a graph with eigenvalues of the graph Laplacian Θ predefined, also called cospectral graph learning (Godsil and McKay, 1982). For instance, spectral sparsification of graphs (Spielman and Teng, 2011; Loukas and Vandergheynst, 2018) aims to learn a graph Laplacian Θ to approximate the given $\bar{\Theta}$, while keeping Θ sparse and its eigenvalues λ_i satisfying $\lambda_i = f\left(\bar{\lambda}_i\right)$, where $\left\{\bar{\lambda}_i\right\}_{i=1}^p$ are the eigenvalues of $\bar{\Theta}$ and f is some specific function. Therefore, for cospectral graph learning, we introduce the following constraint

$$\mathcal{S}_\lambda = \left\{\boldsymbol{\lambda} \in \mathbb{R}^p | \lambda_i = f\left(\bar{\lambda}_i\right), \quad \text{for} \quad i = 1, 2, \ldots, p\right\}. \tag{12}$$

2.3. Structured Graph Learning via Adjacency Spectral Constraints

To enforce spectral constraints on the adjacency matrix W, i.e., $\mathcal{T}(\Theta) = W$ in (7), we introduce the following optimization problem:

$$\begin{array}{ll} \underset{\Theta,\boldsymbol{\psi},V}{\mathsf{maximize}} & \log\ \mathrm{gdet}(\Theta) - \mathrm{tr}\big(\Theta S\big) - \alpha h(\Theta), \\ \mathsf{subject\ to} & \Theta \in \mathcal{S}_\Theta,\ \mathcal{W}(\Theta) = V\mathrm{Diag}(\boldsymbol{\psi})V^\top,\ \boldsymbol{\psi} \in \mathcal{S}_\psi,\ V^\top V = I, \end{array} \tag{13}$$

where Θ is the Laplacian matrix to be optimized, $\mathcal{W}(\Theta)$ is the linear transformation of Θ which maps the Laplacian matrix Θ to the corresponding adjacency matrix such that $[\mathcal{W}(\Theta)]_{ij} = -\Theta_{ij}$, if $i \neq j$ and $[\mathcal{W}(\Theta)]_{ij} = 0$, if $i = j$, $\mathcal{W}(\Theta)$ admits decomposition $\mathcal{W}(\Theta) = V\mathrm{Diag}(\boldsymbol{\psi})V^\top$ with $\boldsymbol{\psi} \in \mathcal{S}_\psi$ and $V^\top V = I$. We enforce Θ to be a Laplacian matrix by the constraint $\Theta \in \mathcal{S}_\Theta$, while spectral constraints on the adjacency matrix $\mathcal{W}(\Theta)$ are incorporated via $\mathcal{W}(\Theta) = V\mathrm{Diag}(\boldsymbol{\psi})V^\top$, where $\mathcal{S}_\psi$ contains the spectral constraints for the desired graph structure.

In the next subsection, we introduce a choice of $\mathcal{S}_\psi$ that enables (13) to learn bipartite graph structures.

2.3.1. General bipartite graph

A graph is said to be bipartite if its vertex set can be partitioned into two disjoint subsets $\mathcal{V} = \mathcal{V}_1 \cup \mathcal{V}_2$ such that no two nodes belonging to the same subset are connected by an edge (Zha et al., 2001), i.e., for each $(l, m) \in \mathcal{V}_i \times \mathcal{V}_i$ then $(l, m) \notin \mathcal{E}$, $i = 1, 2$. Spectral graph theory states that a graph is bipartite if and only if the spectrum of the associated adjacency matrix is symmetric about the origin (Van Mieghem, 2010, Ch.5, Thm. 22).

Theorem 2. *(Van Mieghem, 2010, Ch.5, Thm. 22) A graph is bipartite if and only if the spectrum of the associated adjacency matrix is symmetric about the origin*

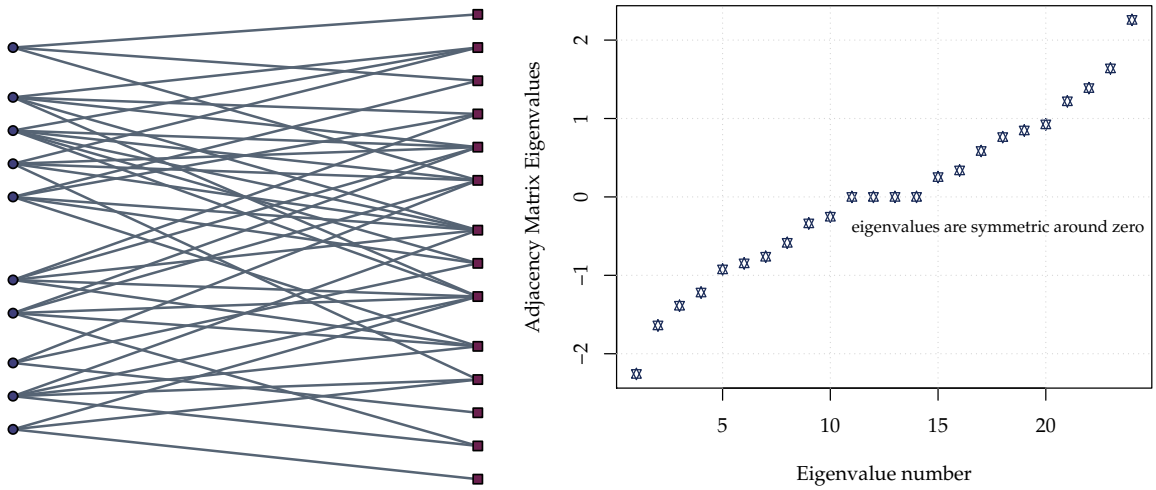
$$\mathcal{S}_\psi = \{\boldsymbol{\psi} \in \mathbb{R}^p | \psi_1 \geq \psi_2 \geq \cdots \geq \psi_p,\ \psi_i = -\psi_{p-i+1},\ i = 1, 2, \ldots, p\}. \tag{14}$$

2.3.2. Connected bipartite graph

The Perron-Frobenius theorem states that if a graph is connected, then the largest eigenvalue ψ_p of its adjacency matrix W has multiplicity 1 (Lovász, 2007, Thm. 1.2). Thus, a connected bipartite graph can be learned by including an additional constraint on the multiplicity of the largest and smallest eigenvalues, i.e., ψ_1 and ψ_p are not repeated. Figure 2 shows a connected bipartite graph and its adjacency symmetric eigenvalues.

Theorem 3. *A graph is connected bipartite if and only if the spectrum of the associated adjacency matrix is symmetric about the origin with non-repeated extreme eigenvalues*

$$\mathcal{S}_\psi = \{\boldsymbol{\psi} \in \mathbb{R}^p | \psi_1 > \psi_2 \geq \cdots \geq \psi_{p-1} > \psi_p,\ \psi_i = -\psi_{p-i+1},\ i = 1, 2, \ldots, p\}. \tag{15}$$



(a) A bipartite graph. (b) Eigenvalue distribution for a bipartite graph.

Figure 2: Bipartite graph its eigenvalue distribution: eigenvalues are symmetric around zero.

2.4. Structured Graph Learning via Joint Laplacian and Adjacency Spectral Constraints

To enforce spectral constraints on the Laplacian matrix Θ and adjacency matrix W, we introduce the following optimization problem

$$\begin{array}{ll} \underset{\Theta,\boldsymbol{\lambda},\boldsymbol{\psi},U,V}{\text{maximize}} & \log\ \text{gdet}(\Theta) - \text{tr}\big(\Theta S\big) - \alpha h(\Theta), \\ \text{subject to} & \Theta \in \mathcal{S}_\Theta,\ \Theta = U\text{Diag}(\boldsymbol{\lambda})U^\top,\ \mathcal{W}(\Theta) = V\text{Diag}(\boldsymbol{\psi})V^\top, \\ & \boldsymbol{\lambda} \in \mathcal{S}_\lambda,\ U^\top U = I,\ \boldsymbol{\psi} \in \mathcal{S}_\psi,\ \ V^\top V = I, \end{array} \tag{16}$$

where Θ admits decomposition as $\Theta = U\text{Diag}(\boldsymbol{\lambda})U^\top$, such that $\boldsymbol{\lambda} \in \mathcal{S}_\lambda$, $U^\top U = I$, and $\mathcal{W}(\Theta)$ is the corresponding adjacency matrix whose decomposition is $\mathcal{W}(\Theta) = V\text{Diag}(\boldsymbol{\psi})V^\top$, such that $\boldsymbol{\psi} \in \mathcal{S}_\psi$ and $V^\top V = I$. Observe that this formulation learns a graph Laplacian matrix Θ with a specific structure by enforcing the spectral constraints on the adjacency and Laplacian matrices simultaneously. Next, we introduce various choices of $\mathcal{S}_\lambda$ and $\mathcal{S}_\psi$ that will enable (16) to learn non-trivial structures.

2.4.1. k-component bipartite graph

A k-component bipartite graph, also known as clustered bipartite graph, is a critical graph structure to many machine learning and financial applications (Zha et al., 2001). Recall that the bipartite structure can be enforced by imposing constraints to the eigenvalues of the adjacency matrix and a k-component structure can be enforced by introducing constraints to the eigenvalues of the Laplacian matrix. These two disparate requirements can be simultaneously imposed in the current formulation (16) via the following constraints sets:

$$\mathcal{S}_\lambda = \left\{\boldsymbol{\lambda} \in \mathbb{R}^p | \{\lambda_j = 0\}_{j=1}^k,\ c_1 \le \lambda_{k+1} \le \cdots \le \lambda_p \le c_2\right\}, \tag{17}$$

$$\mathcal{S}_\psi = \left\{\boldsymbol{\psi} \in \mathbb{R}^p | \psi_i = -\psi_{p-i+1},\ \psi_1 \ge \psi_2 \ge \cdots \ge \psi_p,\ i = 1, 2, \ldots, p\right\}. \tag{18}$$

2.4.2. k-component regular bipartite graph

The eigenvalue property of a $d-$regular graph relates the eigenvalues of its adjacency matrix and Laplacian matrix, which is summarized in the following theorem.

Theorem 4. *(Mohar, 1997, Sec. 2.4) Collecting the Laplacian eigenvalues in increasing order* $\left(\{\lambda_j \uparrow\}_{j=1}^{p}\right)$ *and the adjacency eigenvalues in decreasing order* $(\{\psi_i \downarrow\}_{i=1}^{p})$*, then the eigenvalue pairs for a d-regular graph are related as follows:*

$$\lambda_i = d - \psi_i,\ i = 1, \ldots, p. \tag{19}$$

A k-component regular bipartite structure can be enforced by combining the adjacency eigenvalues property (for bipartite structure), the Laplacian eigenvalues property (for k-component structure) and the spectral properties for the regular graph structure:

$$\mathcal{S}_\lambda = \left\{\boldsymbol{\lambda} \in \mathbb{R}^p | \{\lambda_j = 0\}_{j=1}^{k},\ c_1 \leq \lambda_{k+1} \leq \cdots \leq \lambda_p \leq c_2\right\}, \tag{20}$$

$$\mathcal{S}_\psi = \left\{\boldsymbol{\psi} \in \mathbb{R}^p | \psi_i = d - \lambda_i,\ \psi_1 \geq \psi_2 \geq \cdots \geq \psi_p,\ i = 1, 2, \ldots, p\right\}. \tag{21}$$

2.5. Block Majorization-Minimization Framework

The resulting optimization programs formulated in (8), (13), and (16) are non-convex, NP-hard problems. Therefore we develop efficient optimization methods based on block MM framework (Razaviyayn et al., 2013; Sun et al., 2016). First, we present a general scheme of the block MM framework

$$\begin{array}{ll} \underset{\mathbf{x}}{\mathsf{minimize}} & f(\mathbf{x}) \\ \mathsf{subject\ to} & \mathbf{x} \in \mathcal{X}, \end{array} \tag{22}$$

where the optimization variable $\mathbf{x}$ is partitioned into m blocks as $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m)$, with $\mathbf{x}_i \in \mathcal{X}_i$, $\mathcal{X} = \prod_{i=1}^{m} \mathcal{X}_i$ is a closed convex set, and $f : \mathcal{X} \to \mathbb{R}$ is a continuous function. At the t-th iteration, each block $\mathbf{x}_i$ is updated in a cyclic order by solving the following:

$$\begin{array}{ll} \underset{\mathbf{x}_i}{\mathsf{minimize}} & g_i\left(\mathbf{x}_i | \mathbf{x}_1^{(t)}, \ldots, \mathbf{x}_{i-1}^{(t)}, \mathbf{x}_{i+1}^{(t-1)}, \ldots, \mathbf{x}_m^{(t-1)}\right), \\ \mathsf{subject\ to} & \mathbf{x}_i \in \mathcal{X}_i, \end{array} \tag{23}$$

where $g_i\left(\mathbf{x}_i | \mathbf{y}_i^{(t)}\right)$ with $\mathbf{y}_i^{(t)} \triangleq \left(\mathbf{x}_1^{(t)}, \ldots, \mathbf{x}_{i-1}^{(t)}, \mathbf{x}_i^{(t-1)}, \mathbf{x}_{i+1}^{(t-1)}, \ldots, \mathbf{x}_m^{(t-1)}\right)$ is a majorization function of $f(\mathbf{x})$ at $\mathbf{y}_i^{(t)}$ satisfying

$$g_i\left(\mathbf{x}_i | \mathbf{y}_i^{(t)}\right) \text{ is continuous in } \left(\mathbf{x}_i, \mathbf{y}_i^{(t)}\right),\ \forall\, i, \tag{24a}$$

$$g_i\left(\mathbf{x}_i^{(t)} | \mathbf{y}_i^{(t)}\right) = f\left(\mathbf{x}_1^{(t)}, \ldots, \mathbf{x}_{i-1}^{(t)}, \mathbf{x}_i^{(t)}, \mathbf{x}_{i+1}^{(t-1)}, \ldots, \mathbf{x}_m^{(t-1)}\right), \tag{24b}$$

$$g_i\left(\mathbf{x}_i | \mathbf{y}_i^{(t)}\right) \geq f\left(\mathbf{x}_1^{(t)}, \ldots, \mathbf{x}_{i-1}^{(t)}, \mathbf{x}_i, \mathbf{x}_{i+1}^{(t-1)}, \ldots, \mathbf{x}_m^{(t-1)}\right),\ \forall\, \mathbf{x}_i \in \mathcal{X}_i, \forall\, \mathbf{y}_i \in \mathcal{X}, \forall\, i, \tag{24c}$$

$$g_i'\left(\mathbf{x}_i; \mathbf{d}_i | \mathbf{y}_i^{(t)}\right) \mathbf{B}(t) i g\big|_{\mathbf{x}_i = \mathbf{x}_i^{(t)}} = f'\left(\mathbf{x}_1^{(t)}, \ldots, \mathbf{x}_{i-1}^{(t)}, \mathbf{x}_i, \mathbf{x}_{i+1}^{(t-1)}, \ldots, \mathbf{x}_m^{(t-1)}; \mathbf{d}\right),$$
$$\forall\ \mathbf{d} = (\mathbf{0}, \ldots, \mathbf{d}_i, \ldots, \mathbf{0}) \text{ such that } \mathbf{x}_i^{(t)} + \mathbf{d}_i \in \mathcal{X}_i,\ \forall\, i, \tag{24d}$$

where $f'(\mathbf{x};\mathbf{d})$ stands for the directional derivative at $\mathbf{x}$ along $\mathbf{d}$ (Razaviyayn et al., 2013). In summary, the framework is based on a sequential inexact block coordinate approach, which updates the variable in one block keeping the other blocks fixed. If the surrogate functions g_i is properly chosen, then the solution to (23) could be easier to obtain than solving (22) directly.

3. Structured Graph Learning via Laplacian Spectral Constraints (SGL)

In this section, we develop a block MM-based algorithm for **S**tructured **G**raph learning via **L**aplacian spectral constraints (SGL). In particular, we consider solving (8) under k-component Laplacian spectral constraints (9). To enforce sparsity, we introduce a sparse enforcing concave function, $h(\Theta) = \sum_{i>j} \phi(\Theta_{ij})$ with $\phi(x) \triangleq \log(\epsilon + |x|)$, which leads to the reweighted ℓ_1-norm regularization (Candès et al., 2008). The reweighted ℓ_1-norm regularization is effective in enhancing sparsity of the solution and it also reduces the bias of the estimation. Considering Θ must be symmetric, we impose the sparsity regularization only on its lower triangular part. Now, problem (8) becomes

$$\begin{array}{ll} \underset{\Theta,\boldsymbol{\lambda},U}{\mathsf{minimize}} & -\log\ \mathrm{gdet}(\Theta) + \mathrm{tr}\,(\Theta S) + \alpha \sum_{i>j} \phi(\Theta_{ij}), \\ \mathsf{subject\ to} & \Theta \in \mathcal{S}_\Theta,\ \Theta = U\mathrm{Diag}(\boldsymbol{\lambda})U^\top,\ \boldsymbol{\lambda} \in \mathcal{S}_\lambda,\ U^\top U = I. \end{array} \tag{25}$$

The optimization problem (25) is still complicated. To derive a more tractable formulation, we will introduce a linear operator $\mathcal{L}$ to handle the Laplacian structural constraints and relax the eigen-decomposition equality constraint.

3.1. Graph Laplacian Operator

We recall that a Laplacian matrix Θ as an element of $\mathcal{S}_\Theta$ satisfies i) $\Theta_{ij} = \Theta_{ji} \leq 0$, ii) $\Theta\mathbf{1} = \mathbf{0}$, implying the target matrix is symmetric with degrees of freedom of Θ equal to $p(p-1)/2$. Hence, we construct a linear operator $\mathcal{L}$ that transforms a non-negative vector $\mathbf{w} \in \mathbb{R}^{p(p-1)/2}$ into a matrix $\mathcal{L}\mathbf{w} \in \mathbb{R}^{p\times p}$ that satisfies the Laplacian constraints ($[\mathcal{L}\mathbf{w}]_{ij} = [\mathcal{L}\mathbf{w}]_{ji}, \text{for}\, i \neq j$ and $[\mathcal{L}\mathbf{w}]\mathbf{1} = \mathbf{0}$).

Definition 2. The linear operator $\mathcal{L} : \mathbb{R}^{p(p-1)/2} \to \mathbb{R}^{p\times p}$, $\mathbf{w} \mapsto \mathcal{L}\mathbf{w}$, is defined as

$$[\mathcal{L}\mathbf{w}]_{ij} = \begin{cases} -w_{i+d_j} & i > j, \\ [\mathcal{L}\mathbf{w}]_{ji} & i < j, \\ -\sum_{j\neq i}[\mathcal{L}\mathbf{w}]_{ij} & i = j, \end{cases}$$

where $d_j = -j + \frac{j-1}{2}(2p-j)$.

The adjoint operator $\mathcal{L}^*$ of $\mathcal{L}$ is defined so as to satisfy $\langle \mathcal{L}\mathbf{x}, Y\rangle = \langle \mathbf{x}, \mathcal{L}^*Y\rangle$, $\forall \mathbf{x} \in \mathbb{R}^{p(p-1)/2}$ and $Y \in \mathbb{R}^{p\times p}$.

Definition 3. The adjoint operator $\mathcal{L}^* : \mathbb{R}^{p\times p} \to \mathbb{R}^{p(p-1)/2}$, $Y \mapsto \mathcal{L}^*Y$, is defined by

$$[\mathcal{L}^*Y]_k = Y_{i,i} - Y_{i,j} - Y_{j,i} + Y_{j,j},\ \ k = i - j + \frac{j-1}{2}(2p-j),$$

where $i, j \in \mathbb{Z}^+$ satisfy $k = i - j + \frac{j-1}{2}(2p-j)$ and $i > j$.

In order to illustrate the operational meaning of $\mathcal{L}$ and $\mathcal{L}^*$ more clearly, we provide the following toy examples. Consider a weight vector $\mathbf{w} = [w_1, w_2, w_3, w_4, w_5, w_6]^\top$. The Laplacian operator $\mathcal{L}$ on $\mathbf{w}$ gives

$$\mathcal{L}\mathbf{w} = \begin{bmatrix} \sum_{i=1,2,3} w_i & -w_1 & -w_2 & -w_3 \\ -w_1 & \sum_{i=1,4,5} w_i & -w_4 & -w_5 \\ -w_2 & -w_4 & \sum_{i=2,4,6} w_i & -w_6 \\ -w_3 & -w_5 & -w_6 & \sum_{i=3,5,6} w_i \end{bmatrix}. \tag{26}$$

The operator $\mathcal{L}^*$ on a 4×4 symmetric matrix Y returns a vector

$$\mathcal{L}^* Y = \begin{bmatrix} Y_{11} - Y_{21} - Y_{12} + Y_{22} \\ Y_{11} - Y_{31} - Y_{13} + Y_{33} \\ Y_{11} - Y_{41} - Y_{14} + Y_{44} \\ Y_{22} - Y_{32} - Y_{23} + Y_{33} \\ Y_{22} - Y_{42} - Y_{24} + Y_{44} \\ Y_{33} - Y_{43} - Y_{34} + Y_{44} \end{bmatrix}. \tag{27}$$

By the definition of $\mathcal{L}$, we have Lemma 1.

Lemma 1. *The operator norm* $\|\mathcal{L}\|_2$ *is* $\sqrt{2p}$*, where* $\|\mathcal{L}\|_2 = \sup_{\|\mathbf{x}\|=1} \|\mathcal{L}\mathbf{x}\|_F$ *with* $\mathbf{x} \in \mathbb{R}^{p(p-1)/2}$.

Proof. Follows from the definitions of $\mathcal{L}$ and $\mathcal{L}^*$: see Appendix 8.1 for a detailed proof. □

We have introduced the operator $\mathcal{L}$ that transforms the complicated structural matrix variable Θ into a simple vector variable $\mathbf{w}$. The linear operator $\mathcal{L}$ turns out to be a crucial component of the SGL framework.

3.2. SGL Algorithm

In order to solve (25), we represent the Laplacian matrix $\Theta \in \mathcal{S}_\Theta$ as $\mathcal{L}\mathbf{w}$ and then develop an algorithm based on quadratic methods (Nikolova and Ng, 2005; Ying et al., 2018). The formulation (25) is equivalent to

$$\begin{array}{ll} \underset{\mathbf{w},\boldsymbol{\lambda},U}{\text{minimize}} & -\log \text{gdet}(\text{Diag}(\boldsymbol{\lambda})) + \text{tr}\,(S\mathcal{L}\mathbf{w}) + \alpha \sum_i \phi(w_i), \\ \text{subject to} & \mathbf{w} \geq 0,\ \mathcal{L}\mathbf{w} = U\text{Diag}(\boldsymbol{\lambda})U^\top,\ \boldsymbol{\lambda} \in \mathcal{S}_\lambda,\ U^\top U = I, \end{array} \tag{28}$$

where $\mathbf{w} \geq 0$ means each entry of $\mathbf{w}$ is non-negative. We further relax the problem by introducing the term $\frac{\beta}{2}\|\mathcal{L}\mathbf{w} - U\text{Diag}(\boldsymbol{\lambda})U^\top\|_F^2$ with $\beta > 0$, instead of exactly solving the constraint $\mathcal{L}\mathbf{w} = U\text{Diag}(\boldsymbol{\lambda})U^\top$. Note that this relaxation can be made as tight as desired by choosing β sufficiently large or iteratively increasing β. Now, the original problem can be approximated as

$$\begin{array}{ll} \underset{\mathbf{w},\boldsymbol{\lambda},U}{\text{minimize}} & -\log \text{gdet}(\text{Diag}(\boldsymbol{\lambda})) + \text{tr}\,(S\mathcal{L}\mathbf{w}) + \alpha\sum_i \phi(w_i) + \frac{\beta}{2}\|\mathcal{L}\mathbf{w} - U\text{Diag}(\boldsymbol{\lambda})U^\top\|_F^2, \\ \text{subject to} & \mathbf{w} \geq 0,\ \boldsymbol{\lambda} \in \mathcal{S}_\lambda,\ U^\top U = I. \end{array} \tag{29}$$

When solving (29) to learn a k-component graph structure with the constraints in (9), the first k zero eigenvalues as well as the corresponding eigenvectors can be dropped from the optimization formulation. Now, $\boldsymbol{\lambda}$ only contains $q = p-k$ non-zero eigenvalues in increasing order $\{\lambda_j\}_{j=k+1}^{p}$, then we can replace generalized determinant with determinant on $\mathrm{Diag}(\boldsymbol{\lambda})$ in (29). $U \in \mathbb{R}^{p\times q}$ contains the eigenvectors corresponding to the non-zero eigenvalues in the same order, and the orthogonality constraints on U becomes $U^\top U = I_q$. The non-zero eigenvalues are ordered and lie in this set

$$\mathcal{S}_\lambda = \{\ c_1 \leq \lambda_{k+1} \leq \cdots \leq \lambda_p \leq c_2\}. \tag{30}$$

Collecting the variables as a triple $\left(\mathbf{w} \in \mathbb{R}^{p(p-1)/2}, \boldsymbol{\lambda} \in \mathbb{R}^q, U \in \mathbb{R}^{p\times q}\right)$, we develop a block MM-based algorithm which updates one variable at a time while keeping the other ones fixed.

3.2.1. Update for $\mathbf{w}$

Treating $\mathbf{w}$ as a variable while fixing U and $\boldsymbol{\lambda}$, and ignoring the terms independent of $\mathbf{w}$, we have the following sub-problem

$$\underset{\mathbf{w}\geq 0}{\mathsf{minimize}} \quad \mathrm{tr}\left(S\mathcal{L}\mathbf{w}\right) + \tfrac{\beta}{2}\left\|\mathcal{L}\mathbf{w} - U\mathrm{Diag}(\boldsymbol{\lambda})U^\top\right\|_F^2 + \alpha\sum_i \phi(w_i). \tag{31}$$

Problem (31) can be written as a non-negative quadratic program

$$\underset{\mathbf{w}\geq 0}{\mathsf{minimize}} \quad f(\mathbf{w}) = f_1(\mathbf{w}) + f_2(\mathbf{w}), \tag{32}$$

where $f_1(\mathbf{w}) = \frac{1}{2}\left\|\mathcal{L}\mathbf{w}\right\|_F^2 - \mathbf{c}^\top\mathbf{w}$ and $f_2(\mathbf{w}) = \frac{\alpha}{\beta}\sum_i \log(\epsilon + w_i)$, where the absolute value in $\phi(w_i)$ is removed because each $w_i \geq 0$. Here $\mathbf{c} = \mathcal{L}^*\left(U\mathrm{Diag}(\boldsymbol{\lambda})U^\top - \beta^{-1}S\right)$.

Lemma 2. *$f_1(\mathbf{w})$ in (32) is strictly convex.*

Proof. From the definition of operator $\mathcal{L}$ and the property of its adjoint $\mathcal{L}^*$, we have

$$\left\|\mathcal{L}\mathbf{w}\right\|_F^2 = \langle\mathcal{L}\mathbf{w}, \mathcal{L}\mathbf{w}\rangle = \langle\mathbf{w}, \mathcal{L}^*\mathcal{L}\mathbf{w}\rangle = \mathbf{w}^\top\mathcal{L}^*\mathcal{L}\mathbf{w} > 0, \ \ \forall\ \mathbf{w} \neq \mathbf{0}. \tag{33}$$

The above result implies that $f_1(\mathbf{w})$ is strictly convex. □

Lemma 3. *The function $f(\mathbf{w})$ in (32) is majorized at $\mathbf{w}^{(t)}$ by the function*

$$g\left(\mathbf{w}|\mathbf{w}^{(t)}\right) = f\left(\mathbf{w}^{(t)}\right) + \left(\mathbf{w} - \mathbf{w}^{(t)}\right)^\top \nabla f\left(\mathbf{w}^{(t)}\right) + \frac{L_1}{2}\left\|\mathbf{w} - \mathbf{w}^{(t)}\right\|^2, \tag{34}$$

where $\mathbf{w}^{(t)}$ is the update from previous iteration and $L_1 = \left\|\mathcal{L}\right\|_2^2 = 2p$.

Proof. $f_1(\mathbf{w})$ in (32) is strictly convex and the majorization function can be

$$g_1\left(\mathbf{w}|\mathbf{w}^{(t)}\right) = f_1\left(\mathbf{w}^{(t)}\right) + \left(\mathbf{w} - \mathbf{w}^{(t)}\right)^\top \nabla f_1\left(\mathbf{w}^{(t)}\right) + \frac{L_1}{2}\left\|\mathbf{w} - \mathbf{w}^{(t)}\right\|^2, \tag{35}$$

According to the definition of $\|\mathcal{L}\|_2$ in Lemma 1, we get $L_1 = \|\mathcal{L}\|_2^2 = 2p$. In addition, $f_2(\mathbf{w}) = \alpha \sum_i \log(\epsilon + w_i)$ is concave and thus can be majorized at $\mathbf{w}^{(t)}$ by the first order Taylor expansion

$$g_2\left(\mathbf{w}|\mathbf{w}^{(t)}\right) = f_2\left(\mathbf{w}^{(t)}\right) + \left(\mathbf{w} - \mathbf{w}^{(t)}\right)^\top \nabla f_2\left(\mathbf{w}^{(t)}\right), \tag{36}$$

Totally, we can conclude that $g\left(\mathbf{w}|\mathbf{w}^{(t)}\right)$ in (34) is the majorization function of $f(\mathbf{w})$ in (32). More details about majorization function can be seen in (Sun et al., 2016; Song et al., 2015). □

Note that the majorization function $g\left(\mathbf{w}|\mathbf{w}^{(t)}\right)$ as in (34) is in accordance with the requirement of (24b), since $\mathbf{w}^{(t)}$, $\boldsymbol{\lambda}^{(t)}$, and $U^{(t)}$ in problem (32) are fixed. For notation brevity, we present the majorization function as $g\left(\mathbf{w}|\mathbf{w}^{(t)}\right)$ instead of $g\left(\mathbf{w}|\mathbf{w}^{(t)}, U^{(t)}, \boldsymbol{\lambda}^{(t)}\right)$.

After ignoring the constant terms in (34), the majorized problem of (32) at $\mathbf{w}^{(t)}$ is given by

$$\underset{\mathbf{w}\geq 0}{\text{minimize}} \quad g\left(\mathbf{w}|\mathbf{w}^{(t)}\right) = \tfrac{1}{2}\mathbf{w}^\top\mathbf{w} - \mathbf{w}^\top\mathbf{a}, \tag{37}$$

where $\mathbf{a} = \mathbf{w}^{(t)} - \frac{1}{L_1}\nabla f\left(\mathbf{w}^{(t)}\right)$ and $\nabla f\left(\mathbf{w}^{(t)}\right) = \mathcal{L}^*\left(\mathcal{L}\mathbf{w}^{(t)}\right) - \mathbf{c} + \mathbf{b}$ with $\mathbf{b} = \frac{\alpha}{\beta}[1/(\epsilon + w_1^{(t)}),\ \ldots,\ 1/(\epsilon + w_{p(p-1)/2}^{(t)})]^\top$.

Lemma 4. *By the KKT optimality conditions we can obtain the optimal solution to* (37) *as*

$$\mathbf{w}^{(t+1)} = \left(\mathbf{w}^{(t)} - \frac{1}{L_1}\nabla f\left(\mathbf{w}^{(t)}\right)\right)^+, \tag{38}$$

where $(x)^+ \triangleq \max(x, 0)$ *and and* f *is defined in* (32).

3.2.2. Update for U

Treating U as a variable and fixing $\mathbf{w}$ and $\boldsymbol{\lambda}$, we obtain the following sub-problem

$$\begin{array}{ll} \underset{U}{\text{minimize}} & \frac{\beta}{2}\left\|\mathcal{L}\mathbf{w} - U\text{Diag}(\boldsymbol{\lambda})U^\top\right\|_F^2, \\ \text{subject to} & U^\top U = I_q. \end{array} \tag{39}$$

An equivalent trivial reformulation is as follows

$$\begin{array}{ll} \underset{U}{\text{maximize}} & \text{tr}(U^\top\mathcal{L}\mathbf{w}U\text{Diag}(\boldsymbol{\lambda})) \\ \text{subject to} & U^\top U = I_q. \end{array} \tag{40}$$

Problem (40) is an optimization problem on the orthogonal Stiefel manifold $\text{St}(p, q) = \{U \in \mathbb{R}^{p\times q} : U^\top U = I_q\}$. From (Absil et al., 2009; Benidis et al., 2016) the solution of (40) is the eigenvectors of $\mathcal{L}\mathbf{w}$ (suitably ordered).

Lemma 5. *From the KKT optimality conditions the solution to* (40) *is given by*

$$U^{(t+1)} = \textit{eigenvectors}\left(\mathcal{L}\mathbf{w}^{(t+1)}\right)[k+1:p], \tag{41}$$

that is, the $p-k$ *principal eigenvectors of the matrix* $\mathcal{L}\mathbf{w}^{(t+1)}$ *with the corresponding eigenvalues in an increasing order.*

3.2.3. Update for λ

We obtain the following sub-problem to update $\boldsymbol{\lambda}$

$$\underset{\boldsymbol{\lambda}\in\mathcal{S}_\lambda}{\text{minimize}} \quad -\log\det\left(\text{Diag}(\boldsymbol{\lambda})\right) + \tfrac{\beta}{2}\left\|\mathcal{L}\mathbf{w} - U\text{Diag}(\boldsymbol{\lambda})U^\top\right\|_F^2. \tag{42}$$

Problem (42) can be rewritten as

$$\underset{\boldsymbol{\lambda}\in\mathcal{S}_\lambda}{\text{minimize}} \quad -\log\det\left(\text{Diag}(\boldsymbol{\lambda})\right) + \tfrac{\beta}{2}\left\|U^\top(\mathcal{L}\mathbf{w})U - \text{Diag}(\boldsymbol{\lambda})\right\|_F^2. \tag{43}$$

With a slight abuse of notation, we denote the indices for the non-zero eigenvalues λ_i in (30) from 1 to $q = p - k$ instead of $k+1$ to p. Then, problem (43) can be further simplified as

$$\underset{c_1\le\lambda_1\le\cdots\le\lambda_q\le c_2}{\text{minimize}} \quad -\sum_{i=1}^{q}\log\lambda_i + \frac{\beta}{2}\|\boldsymbol{\lambda}-\mathbf{d}\|_2^2, \tag{44}$$

where $\boldsymbol{\lambda} = [\lambda_1,\ldots,\lambda_q]^\top$ and $\mathbf{d} = [d_1,\ldots,d_q]^\top$ with d_i being the i-th diagonal element of $\text{Diag}(U^\top(\mathcal{L}\mathbf{w})U)$.

Problem (44) is a convex optimization problem, that can be solved via disciplined convex programming frameworks such as CVX (Grant and Boyd, 2014; Fu et al., 2017a). However, such frameworks usually do not scale well, hence we develop a specialized, computationally efficient algorithm to solve (44) which is derived from the KKT optimality conditions. The update rule for $\boldsymbol{\lambda}$ follows an iterative procedure summarized in Algorithm 1.

Remark 3. Problems of the form (44) are known as a regularized isotonic regression. The isotonic regression is a well-researched problem that has found applications in a number of domains (Best and Chakravarti, 1990; Lee, 1981; Barlow and Brunk, 1972; Luss and Rosset, 2014; Bartholomew, 2004). To the best of our knowledge, however, there exist no method as computationally efficient as the one presented in Algorithm 1. The proposed algorithm can obtain a globally optimal solution within a maximum of $q+1$ iterations for a q-dimensional regularized isotonic regression problem. In addition, it can be potentially adapted to solve other isotonic regression problems.

Lemma 6. *The iterative-update procedure summarized in Algorithm 1 converges to the KKT point of Problem* (44)*.*

Proof. Please refer to Appendix 8.2 for the detailed proof. □

Algorithm 1: Updating rule for $\boldsymbol{\lambda}$

Input: $d_1, d_2, \ldots, d_q$, β, c_1, and c_2;

1 $\lambda_i = \frac{1}{2}\left(d_i + \sqrt{d_i^2 + 4/\beta}\right)$, $i = 1, 2, \ldots, q$;

2 **if** $\boldsymbol{\lambda}$ *satisfies* $c_1 \leq \lambda_1 \leq \cdots \leq \lambda_q \leq c_2$ **then**

3 | **return** $\lambda_1, \ldots, \lambda_q$;

4 **while not** $c_1 \leq \lambda_1 \leq \cdots \leq \lambda_q \leq c_2$ **do**

5 | **if** $c_1 \geq \lambda_1 \geq \cdots \geq \lambda_r$ *with at least one inequality strict and* $r \geq 1$ **then**

6 | | $\lambda_1 = \cdots = \lambda_r = c_1$;

7 | **else if** $\lambda_s \geq \cdots \geq \lambda_q \geq c_2$ *with at least one inequality strict and* $s \leq q$ **then**

8 | | $\lambda_s = \cdots = \lambda_q = c_2$;

9 | **else if** $\lambda_i \geq \cdots \geq \lambda_m$ *with at least one inequality strict and* $1 \leq i \leq m \leq q$ **then**

10 | | $\bar{d}_{i\to m} = \frac{1}{m-i+1}\sum_{j=i}^{m} d_j$;

11 | | $\lambda_i = \cdots = \lambda_m = \frac{1}{2}\left(\bar{d}_{i\to m} + \sqrt{\bar{d}_{i\to m}^2 + 4/\beta}\right)$;

12 **end**

Output: $\lambda_1, \ldots, \lambda_q$.

To update $\boldsymbol{\lambda}$, Algorithm 1 iteratively check situations [cf. steps 6, 10 and 14] and updates $\boldsymbol{\lambda}$ accordingly until $c_1 \leq \lambda_1 \leq \cdots \leq \lambda_q \leq c_2$ is satisfied. If some situation is verified, then the corresponding λ_i needs to be updated accordingly. Note that the situations are independent from each other, i.e., each λ_i will not involve two situations simultaneously. Furthermore, all λ_i are updated iteratively according to the above situations until $\boldsymbol{\lambda}$ satisfies the KKT conditions.

3.2.4. SGL algorithm summary

Algorithm 2 summarizes the implementation of the structured graph learning via Laplacian spectral constraints.

Algorithm 2: SGL

Input: SCM S, k, c_1, c_2, β, α, $\mathbf{w}^{(0)}$, $\epsilon > 0$;

1 $t \leftarrow 0$;

2 **while** *stopping criteria not met* **do**

3 | update $\mathbf{w}^{(t+1)}$ as in (38);

4 | update $U^{(t+1)}$ as in (41);

5 | update $\boldsymbol{\lambda}^{(t+1)}$ by solving (44) with Algorithm 1;

6 | $t \leftarrow t + 1$;

7 **end**

Output: $\mathcal{L}\mathbf{w}^{(t+1)}$.

The most computationally demanding step in Algorithm 2 is the eigenvalue decomposition required for the update of U. Therefore, the worst-case computational complexity of Algorithm 2 is $O(p^3)$. However, this complexity can be reduced by taking advantage of the sparse structure and the properties of the Laplacian matrix while conducting its eigenvalue decomposition. As a comparison, the well-known GLasso method (Friedman et al., 2008) has

similar worst-case complexity, even though GLasso is not able to learn graphs with structural constraints. While considering specific structural requirements, the SGL algorithm has a considerable advantage over other competing structured graph learning algorithms Marlin and Murphy (2009); Hao et al. (2018); Ambroise et al. (2009).

Theorem 5. *The sequence* $\left(\mathbf{w}^{(t)}, U^{(t)}, \boldsymbol{\lambda}^{(t)}\right)$ *generated by Algorithm 2 converges to the set of KKT points of* (29).

Proof. The detailed proof is deferred to the Appendix 8.3. □

Remark 4. Note that SGL is not only limited to k-component graph learning, but can be easily adapted to learn other graph structures under aforementioned spectral constraints in (9), (10), (11), and (12). Furthermore, the SGL can also be utilized to learn popular connected graph structures (e.g., Erdos-Renyi graph, modular graph, grid graph, etc.) even without specific spectral constraints just by choosing the eigenvalue constraints corresponding to one component graph (i.e., $k = 1$) and setting c_1, c_2 to very small and large values respectively. Detailed experiments with important graph structures are carried out in the simulation section.

4. Structured Graph Learning via Adjacency Spectral Constraints (SGA)

In this section, we develop a block MM-based algorithm for **S**tructured **G**raph learning via **A**djacency spectral constraints (SGA). In particular, we consider to solve (13) for connected bipartite graph structures by introducing the spectral constraints on the adjacency eigenvalues (14). Since Θ is a connected graph, the term $\log \operatorname{gdet}(\Theta)$ can be simplified according to the following lemma.

Lemma 7. *If* Θ *is a Laplacian matrix for a connected graph, then*

$$gdet(\Theta) = det(\Theta + J), \tag{45}$$

where $J = \frac{1}{p}\mathbf{1}\mathbf{1}^\top$.

Proof. It is easy to establish (45) by the fact that $\Theta\mathbf{1} = \mathbf{0}$. □

4.1. Graph adjacency operator

To guarantee the structure of an adjacency matrix, we introduce a linear operator $\mathcal{A}$.

Definition 4. The linear operator $\mathcal{A} : \mathbb{R}^{p(p-1)/2} \to \mathbb{R}^{p\times p}$, $\mathbf{w} \mapsto \mathcal{A}\mathbf{w}$, is defined as

$$[\mathcal{A}\mathbf{w}]_{ij} = \begin{cases} w_{i+d_j} & i > j, \\ [\mathcal{A}\mathbf{w}]_{ji} & i < j, \\ 0 & i = j, \end{cases}$$

where $d_j = -j + \frac{j-1}{2}(2p - j)$.

An example of $\mathcal{A}$ acting on a weight vector of 6 elements $\mathbf{w} = [w_1, w_2, \ldots, w_6]^\top$ is given below

$$\mathcal{A}\mathbf{w} = \begin{bmatrix} 0 & w_1 & w_2 & w_3 \\ w_1 & 0 & w_4 & w_5 \\ w_2 & w_4 & 0 & w_6 \\ w_3 & w_5 & w_6 & 0 \end{bmatrix}. \tag{46}$$

The adjoint operator $\mathcal{A}^*$ of $\mathcal{A}$ is defined so as to satisfy $\langle \mathcal{A}\mathbf{x}, Y\rangle = \langle \mathbf{x}, \mathcal{A}^* Y\rangle$, $\forall \mathbf{x} \in \mathbb{R}^{p(p-1)/2}$ and $Y \in \mathbb{R}^{p\times p}$.

Definition 5. The adjoint operator $\mathcal{A}^* : \mathbb{R}^{p\times p} \to \mathbb{R}^{p(p-1)/2}$, $Y \mapsto \mathcal{A}^* Y$, is defined as

$$[\mathcal{A}^* Y]_k = Y_{ij} + Y_{ji}, \tag{47}$$

where i, $j \in \mathbb{Z}^+$ satisfy $i - j + \frac{j-1}{2}(2p - j) = k$ and $i > j$.

Lemma 8. *The operator norm* $\|\mathcal{A}\|_2$ *is* $\sqrt{2}$, $\|\mathcal{A}\|_2 = \sup_{\|\mathbf{x}\|=1} \|\mathcal{A}\mathbf{x}\|_F$ *with* $\mathbf{x} \in \mathbb{R}^{p(p-1)/2}$.

Proof. Directly from the definition of operator norm, we have

$$\|\mathcal{A}\|_2 = \sup_{\|\mathbf{x}\|=1} \|\mathcal{A}\mathbf{x}\|_F = \sup_{\|\mathbf{x}\|=1} \sqrt{2}\,\|\mathbf{x}\| = \sqrt{2}. \tag{48}$$

□

4.2. SGA Algorithm

By introducing the operators $\mathcal{L}$ and $\mathcal{A}$, using reweighted ℓ_1-norm and relaxing the constraint $\mathcal{A}\mathbf{w} = V\mathrm{Diag}(\boldsymbol{\psi})V^\top$ in (13), we obtain the following approximation

$$\begin{array}{ll} \underset{\mathbf{w},\boldsymbol{\psi},V}{\mathsf{minimize}} & -\log\det(\mathcal{L}\mathbf{w} + J) + \mathrm{tr}\,(S\mathcal{L}\mathbf{w}) + \alpha\sum_i \phi(w_i) + \frac{\gamma}{2}\|\mathcal{A}\mathbf{w} - V\mathrm{Diag}(\boldsymbol{\psi})V^\top\|_F^2, \\ \mathsf{subject\ to} & \mathbf{w} \geq 0,\ \boldsymbol{\psi} \in \mathcal{S}_\psi,\ V^\top V = I, \end{array} \tag{49}$$

where $\gamma > 0$ is the penalty parameter. Suppose there are z zero eigenvalues in the set $\mathcal{S}_\psi$ with $z \geq 0$. From the symmetry property of the eigenvalues, the zero eigenvalues are positioned in the middle, i.e., in (14) the eigenvalues $\psi_{\frac{p-z}{2}+1}$ to $\psi_{\frac{p+z}{2}}$ will be zero. Both $(p + z)$ and $(p - z)$ must be even by the symmetry property. As a consequence, the zero eigenvalues and the corresponding eigenvectors can be dropped from the formulation. Now $\boldsymbol{\psi} \in \mathbb{R}^b$ contains b non-zero eigenvalues and $V \in \mathbb{R}^{p\times b}$ contains the corresponding eigenvectors that satisfy $V^\top V = I_b$. The non-zero eigenvalues are required to lie in the following set:

$$\mathcal{S}_\psi = \left\{\psi_i = -\psi_{b+1-i},\ c_1 \geq \psi_1 \geq \psi_2 \geq \cdots \geq \psi_{b/2} \geq c_2,\ i = 1, \ldots, b/2\right\}, \tag{50}$$

where c_1 and $c_2 > 0$ are some constants which depend on the graph properties. Collecting the variables as a triple $\left(\mathbf{w} \in \mathbb{R}^{p(p-1)/2}, \boldsymbol{\psi} \in \mathbb{R}^b, V \in \mathbb{R}^{p\times b}\right)$, we develop a block MM-based method that updates one variable at a time while keeping the other ones fixed.

4.2.1. Update for $\mathbf{w}$

The optimization problem (49) with respect to $\mathbf{w}$ is

$$\underset{\mathbf{w}\geq 0}{\text{minimize}} \quad -\log\det(\mathcal{L}\mathbf{w}+J)+\operatorname{tr}(S\mathcal{L}\mathbf{w})+\alpha\sum_i \phi(w_i)+\tfrac{\gamma}{2}\|\mathcal{A}\mathbf{w}-V\mathrm{Diag}(\boldsymbol{\psi})V^\top\|_F^2. \tag{51}$$

Problem (51) is equivalent to

$$\underset{\mathbf{w}\geq 0}{\text{minimize}} \quad f(\mathbf{w})=f_1(\mathbf{w})+f_2(\mathbf{w})+f_3(\mathbf{w}), \tag{52}$$

where $f_1(\mathbf{w})=-\frac{1}{\gamma}\log\det(\mathcal{L}\mathbf{w}+J)$, $f_2(\mathbf{w})=\frac{1}{2}\|\mathcal{A}\mathbf{w}\|_F^2-\tilde{\mathbf{c}}^\top\mathbf{w}$ with $\tilde{\mathbf{c}}=\mathcal{A}^*(V\mathrm{Diag}(\boldsymbol{\psi})V^\top)-\gamma^{-1}\mathcal{L}^*S$, and $f_3(\mathbf{w})=\frac{\alpha}{\gamma}\sum_i\log(\epsilon+w_i)$.

Lemma 9. *The function $f_1(\mathbf{w})+f_2(\mathbf{w})$ in (52) is strictly convex.*

Proof. First, note that $-\log\det(\mathcal{L}\mathbf{w}+J)$ is a convex function. From the definition of $\mathcal{A}$ and the property of its adjoint $\mathcal{A}^*$, we have

$$\|\mathcal{A}\mathbf{w}\|_F^2=\langle\mathcal{A}\mathbf{w},\mathcal{A}\mathbf{w}\rangle=\langle\mathbf{w},\mathcal{A}^*\mathcal{A}\mathbf{w}\rangle=\mathbf{w}^\top\mathcal{A}^*\mathcal{A}\mathbf{w}>0,\ \forall\mathbf{w}\neq\mathbf{0}, \tag{53}$$

which implies $f_1(\mathbf{w})+f_2(\mathbf{w})$ in (52) is strictly convex. □

To get a closed-form solution, we derive a majorized function of (51).

Lemma 10. *The function $f(\mathbf{w})$ in (52) is majorized at $\mathbf{w}^{(t)}$ by the function*

$$g\left(\mathbf{w}|\mathbf{w}^{(t)}\right)=f\left(\mathbf{w}^{(t)}\right)+\left(\mathbf{w}-\mathbf{w}^{(t)}\right)^\top\nabla f\left(\mathbf{w}^{(t)}\right)+\frac{L}{2}\left\|\mathbf{w}-\mathbf{w}^{(t)}\right\|^2, \tag{54}$$

where $L=(\|\mathcal{A}\|_2^2+L_2/\gamma)$, in which $\|\mathcal{A}\|_2^2=2$, and $-\log\det(\mathcal{L}\mathbf{w}+J)$ is assumed to be L_2-Lipschitz continuous gradient [1].

Proof. Assuming $-\log\det(\mathcal{L}\mathbf{w}+J)$ is L_2-Lipschitz continuous gradient, we have $f_1(\mathbf{w})=-\frac{1}{\gamma}\log\det(\mathcal{L}\mathbf{w}+J)$ is majorized at $\mathbf{w}^{(t)}$ by the function

$$g_1\left(\mathbf{w}|\mathbf{w}^{(t)}\right)=f_1\left(\mathbf{w}^{(t)}\right)+\left(\mathbf{w}-\mathbf{w}^{(t)}\right)^\top\nabla f_1\left(\mathbf{w}^{(t)}\right)+\frac{L_2}{2\gamma}\left\|\mathbf{w}-\mathbf{w}^{(t)}\right\|^2. \tag{55}$$

By the Taylor expansion of $f_2(\mathbf{w})=\frac{1}{2}\|\mathcal{A}\mathbf{w}\|_F^2-\tilde{\mathbf{c}}^\top\mathbf{w}$ and Lemma 8, we get the majorization function of $f_2(\mathbf{w})$ can be

$$g_2(\mathbf{w})\left(\mathbf{w}|\mathbf{w}^{(t)}\right)=f_2\left(\mathbf{w}^{(t)}\right)+\left(\mathbf{w}-\mathbf{w}^{(t)}\right)^\top\nabla f_2\left(\mathbf{w}^{(t)}\right)+\frac{1}{2}\|\mathcal{A}\|_2^2\left\|\mathbf{w}-\mathbf{w}^{(t)}\right\|^2, \tag{56}$$

Since $f_3(\mathbf{w})$ is concave, the majorization function can be the first order Taylor expansion

$$g_3\left(\mathbf{w}|\mathbf{w}^{(t)}\right)=f_3\left(\mathbf{w}^{(t)}\right)+\left(\mathbf{w}-\mathbf{w}^{(t)}\right)^\top\nabla f_3\left(\mathbf{w}^{(t)}\right), \tag{57}$$

We can finish the proof by combining (55), (56) and (57). □

1. The Lipschitz constant of $-\log\det(\mathcal{L}\mathbf{w}+J)$ on the domain $\{\mathbf{w}|\lambda_{\min}(\mathcal{L}\mathbf{w}+J)\geq 1/L_2\}$ is L_2. Theoretically, the Lipschitz constant L_2 can be unbounded. However, for practical considerations we can assume it to be bounded by the algebraic connectivity of the graph. See Remark 5 for more details.

After ignoring the constant terms in (54), the majorized problem of (52) at $\mathbf{w}^{(t)}$ is given by

$$\underset{\mathbf{w}\geq 0}{\text{minimize}} \quad \tfrac{1}{2}\mathbf{w}^\top\mathbf{w} - \mathbf{w}^\top\tilde{\mathbf{a}}, \tag{58}$$

where $\tilde{\mathbf{a}} = \mathbf{w}^{(t)} - \frac{1}{L}\nabla f\left(\mathbf{w}^{(t)}\right)$ and $\nabla f\left(\mathbf{w}^{(t)}\right) = \frac{1}{\gamma}\mathcal{L}^*\left(\mathcal{L}\mathbf{w}^{(t)} + J\right)^{-1} + \mathcal{A}^*\mathcal{A}\left(\mathbf{w}^{(t)}\right) - \tilde{\mathbf{c}} + \tilde{\mathbf{b}}$ with $\tilde{\mathbf{b}} = \frac{\alpha}{\gamma}[1/(\epsilon + w_1^{(t)}),\ \ldots,\ 1/(\epsilon + w_{p(p-1)/2}^{(t)})]^\top$.

Lemma 11. *By the KKT optimality conditions we can obtain the optimal solution of* (58) *as*

$$\mathbf{w}^{(t+1)} = \left(\mathbf{w}^{(t)} - \frac{1}{L}\nabla f\left(\mathbf{w}^{(t)}\right)\right)^+, \tag{59}$$

where $(x)^+ \triangleq \max(x, 0)$ *and* f *is defined in* (52).

Remark 5. The Lipschtiz constant L_2 of the function $-\log\det(\mathcal{L}\mathbf{w} + J)$ is related to the smallest non-zero eigenvalue of $\mathcal{L}\mathbf{w}$, which is bounded away from $\frac{\epsilon_w}{(p-1)^2}$ (Lemma 1, Rajawat and Kumar, 2017), where $\epsilon_w > 0$ is the minimum non-zero graph weight. However, for practical purposes the edges with very small weights can be ignored and set to be zero, and we can assume that the non-zero weights are bounded by some constant $\epsilon_w > 0$. To strictly force the minimum weight property, one can modify the non-negativity constraint $\mathbf{w} \geq 0$ in problem (49) as $\mathbf{w} \geq \epsilon_w$. On the other hand, we do not need a tight Lipschtiz constant L_2. In fact, any $L_2' \geq L_2$ makes the function $g\left(\mathbf{w}|\mathbf{w}^{(t)}\right)$ satisfy the majorization conditions (24).

4.2.2. Update for V

To update V, we get the following sub-problem

$$\begin{array}{ll} \underset{V}{\text{maximize}} & \operatorname{tr}\left(\mathrm{V}^\top\mathcal{A}\mathbf{w}\mathrm{V}\operatorname{Diag}(\boldsymbol{\psi})\right), \\ \text{subject to} & V^\top V = I_b. \end{array} \tag{60}$$

Problem (60) is an optimization on the orthogonal Stiefel manifold $\mathrm{St}(p, b) = \{V \in \mathbb{R}^{p\times b} : V^\top V = I_b\}$.

Lemma 12. *From the KKT optimality conditions, the solution to* (60) *is given by*

$$V^{(t+1)} = \textit{eigenvectors}\left(\mathcal{A}\mathbf{w}^{(t+1)}\right)\left[1 : \frac{(p-z)}{2},\ \frac{(p+z)}{2} + 1 : p\right], \tag{61}$$

that is, the eigenvectors of the matrix $\mathcal{A}\mathbf{w}^{(t+1)}$ *in the same order of the eigenvalues, where* z *is the number of zero eigenvalues.*

The solution (61) satisfies the optimality condition of (60) on the orthogonal Stiefel manifold and more details about the derivation can be found in Absil et al. (2009).

4.2.3. Update for ψ

Solving for (49) with respect to $\boldsymbol{\psi}$, ignoring the terms independent of $\boldsymbol{\psi}$, we have the following sub-problem

$$\begin{array}{ll} \underset{\boldsymbol{\psi}}{\text{minimize}} & \|\mathcal{A}\mathbf{w} - V\text{Diag}(\boldsymbol{\psi})V^\top\|_F^2. \\ \text{subject to} & \boldsymbol{\psi} \in \mathcal{S}_\psi, \end{array} \tag{62}$$

The optimization problem (62) may also be written as

$$\underset{\boldsymbol{\psi}\in\mathcal{S}_\psi}{\text{minimize}} \quad \|\boldsymbol{\psi} - \mathbf{e}\|_2^2, \tag{63}$$

where $\boldsymbol{\psi} = [\psi_1, \psi_2, \ldots, \psi_b]^\top$ and $\mathbf{e} = [e_1, e_2, \ldots, e_b]^\top$, in which ψ_i and e_i correspond to the i-th diagonal element of $\text{Diag}(\boldsymbol{\psi})$ and $\text{Diag}(V^\top\mathcal{A}\mathbf{w}V)$, respectively. Problem (63) is a convex optimization problem with simple linear constraints. Furthermore, due to the symmetry constraint in $\mathcal{S}_\psi$ (i.e., $\psi_i = -\psi_{b+1-i}$, $i = 1, \ldots, b/2$), the variables are related such that we only need to solve for the first half of the variables $[\psi_1, \psi_2, \ldots, \psi_{b/2}]^\top$.

Lemma 13. *Consider the following isotonic regression problem for* $\tilde{\boldsymbol{\psi}} = [\psi_1, \psi_2, \ldots, \psi_{b/2}]^\top$,

$$\begin{array}{ll} \underset{\tilde{\boldsymbol{\psi}}}{\textit{minimize}} & \|\tilde{\boldsymbol{\psi}} - \tilde{\mathbf{e}}\|_2^2, \\ \textit{subject to} & c_1 \geq \psi_1 \geq \psi_2 \geq \cdots \geq \psi_{b/2} \geq c_2, \end{array} \tag{64}$$

where $c_1, c_2 > 0$ *and* $\tilde{\mathbf{e}} = [\tilde{e}_1, \tilde{e}_2, \ldots, \tilde{e}_{b/2}]^\top$, $\tilde{e}_i = \frac{e_i - e_{b+1-i}}{2}$, $i = 1, 2, \ldots, b/2$. *The first half of the solution to the problem* (63) *is same as the solution to* (64).

The detailed proof is deferred to the Appendix 8.4. The formulation (64) is also similar to problem (44) without the log determinant term. The solution to problem (64) can be obtained by following the iterative procedure discussed in Algorithm 1 by setting $\psi_i = \tilde{e}_i, i = 1, 2, \ldots, b/2$, and iteratively updating and checking all the situations until all the ψ_i satisfy $c_1 \geq \psi_1 \geq \psi_2, \ldots, \psi_{b/2} \geq c_2$. Finally, the solution for the other half of the variables $\{\psi_i\}_{i=b/2+1}^b$ in (63) are obtained by setting $\psi_{b+1-i} = -\psi_i$, for $i = 1, \ldots, b/2$.

4.2.4. SGA algorithm summary

Algorithm 3 summarizes the implementation of structured graph learning (SGA) via adjacency spectral constraints. The most computationally demanding steps of SGA are the eigenvalue decomposition and the matrix inversion of $p \times p$ matrices, which results in a worst-case complexity of $O\left(p^3\right)$. As in the SGL algorithm, the computational complexity of SGA may be reduced by taking advantage of the sparse structure and the properties of

the adjacency and Laplacian matrices while computing their eigenvalue decomposition and matrix inversion (Livne and Brandt, 2012; Koutis et al., 2011), respectively.

Algorithm 3: SGA

Input: SCM S, c_1, c_2, $\mathbf{w}^{(0)}$, γ, α, $\epsilon > 0$;
$t \leftarrow 0$;
while *stopping criteria not met* **do**
  update $\mathbf{w}^{(t+1)}$ as in (59);
  update $V^{(t+1)}$ as in (61);
  update $\boldsymbol{\psi}^{(t+1)}$ by solving (64) with Algorithm 1;
  $t \leftarrow t + 1$;
end
Output: $\mathcal{L}\mathbf{w}^{(t+1)}$.

The following theorem establishes the convergence property for SGA.

Theorem 6. *The sequence* $\left(\mathbf{w}^{(t)}, V^{(t)}, \boldsymbol{\psi}^{(t)}\right)$ *generated by Algorithm 3 converges to the set of KKT points of Problem* (49).

Proof. The proof of Theorem 6 is similar to that of Theorem 5 and thus is omitted. □

Remark 6. From a combinatorial point of view, finding a bipartite structure is equivalent to a max-cut problem between $\mathcal{V}_1$ and $\mathcal{V}_2$, which is an NP-hard problem. In a recent work, Pavez et al. (2018) considered an approximate bipartite graph estimation under structured GGM settings. The algorithm consists of a two-stage procedure: first, they learn a bipartite structure via Goemans-Williamson (GM) algorithm (Goemans and Williamson, 1995); then, their algorithm learns the Laplacian weights by the generalized graph Laplacian (GGL) learning method (Egilmez et al., 2017). The GM algorithm, dominated by a semi-definite programming and Cholesky decomposition of $p \times p$ matrix, has a worst-case complexity $O\left(p^3\right)$, while the GGL method has a computational complexity of $O\left(p^3\right)$. Therefore, the SGA algorithm enjoys a smaller worst-case computational complexity than that in Pavez et al. (2018). In addition, to the best of our knowledge, SGA is the first single stage algorithm for learning bipartite graph structure directly from the data sample covariance matrix.

5. Structured Graph Learning via Joint Laplacian and Adjacency Spectral Constraints (SGLA)

In this section we consider the problem (16) for **S**tructured **G**raph learning via joint **L**aplacian and **A**djacency spectral constraints (SGLA). Following from the discussions in previous sections: upon utilizing the Laplacian operator $\mathcal{L}$, the adjacency operator $\mathcal{A}$ and moving the constraints $\mathcal{A}\mathbf{w} = V\mathrm{Diag}(\boldsymbol{\psi})V^\top$ and $\mathcal{L}\mathbf{w} = U\mathrm{Diag}(\boldsymbol{\lambda})U^\top$ into the objective function, a tractable approximation of (16) can be written as

$$\begin{aligned} \underset{\mathbf{w},\boldsymbol{\lambda},U,\boldsymbol{\psi},V}{\text{minimize}} \quad & -\log\det\left(\mathrm{Diag}(\boldsymbol{\lambda})\right) + \mathrm{tr}\left(S\mathcal{L}\mathbf{w}\right) + \alpha\sum_i \phi(w_i) \\ & + \tfrac{\beta}{2}\|\mathcal{L}\mathbf{w} - U\mathrm{Diag}(\boldsymbol{\lambda})U^\top\|_F^2 + \tfrac{\gamma}{2}\|\mathcal{A}\mathbf{w} - V\mathrm{Diag}(\boldsymbol{\psi})V^\top\|_F^2, \\ \text{subject to} \quad & \mathbf{w} \geq 0,\ \boldsymbol{\lambda} \in \mathcal{S}_\lambda,\ U^\top U = I,\ \boldsymbol{\psi} \in \mathcal{S}_\psi,\ V^\top V = I, \end{aligned} \tag{65}$$

where β and $\gamma > 0$ are hyperparameters that control the trade-off between each term in (65). Similarly from the derivations of Algorithms 2 and 3, we collect the variables $(\mathbf{w}, \boldsymbol{\lambda}, U, \boldsymbol{\psi}, V)$ to develop a block MM-type method for solving problem (65).

Ignoring the terms independent of $\mathbf{w}$, we have the following sub-problem,

$$\underset{\mathbf{w}\geq 0}{\mathsf{minimize}} \quad \operatorname{tr}(S\mathcal{L}\mathbf{w}) + \alpha \sum_i \phi(w_i) + \tfrac{\beta}{2} \left\| \mathcal{L}\mathbf{w} - U\mathrm{Diag}(\boldsymbol{\lambda})U^\top \right\|_F^2 + \tfrac{\gamma}{2} \left\| \mathcal{A}\mathbf{w} - V\mathrm{Diag}(\boldsymbol{\psi})V^\top \right\|_F^2. \tag{66}$$

After a few algebraic manipulations, we obtain an equivalent formulation

$$\underset{\mathbf{w}\geq 0}{\mathsf{minimize}} \quad f(\mathbf{w}) = f_1(\mathbf{w}) + f_2(\mathbf{w}), \tag{67}$$

where $f_1(\mathbf{w}) = \frac{\beta}{2}\|\mathcal{L}\mathbf{w}\|_F^2 - \mathbf{c}_1^\top \mathbf{w} + \alpha \sum_i \phi(w_i)$ and $f_2(\mathbf{w}) = \frac{\gamma}{2}\|\mathcal{A}\mathbf{w}\|_F^2 - \mathbf{c}_2^\top \mathbf{w}$ in which $\mathbf{c}_1 = \beta\mathcal{L}^*(U\mathrm{Diag}(\boldsymbol{\lambda})U^\top - \beta^{-1}S)$ and $\mathbf{c}_2 = \gamma\mathcal{A}^*(V\mathrm{Diag}(\boldsymbol{\psi})V^\top)$. Now, we use the MM framework to obtain a closed form update for problem (66).

Lemma 14. *The function $f(\mathbf{w})$ in (67) is majorized at $\mathbf{w}^{(t)}$ by the function*

$$\begin{aligned} g\left(\mathbf{w}|\mathbf{w}^{(t)}\right) =& f_1\left(\mathbf{w}^{(t)}\right) + \left(\mathbf{w} - \mathbf{w}^{(t)}\right)^\top \nabla f_1\left(\mathbf{w}^{(t)}\right) + \frac{L_1}{2}\left\|\mathbf{w} - \mathbf{w}^{(t)}\right\|^2 \\ &+ f_2\left(\mathbf{w}^{(t)}\right) + \left(\mathbf{w} - \mathbf{w}^{(t)}\right)^\top \nabla f_2\left(\mathbf{w}^{(t)}\right) + \frac{L_2}{2}\left\|\mathbf{w} - \mathbf{w}^{(t)}\right\|^2, \end{aligned} \tag{68}$$

where $\mathbf{w}^{(t)}$ is the update from previous iteration, $L_1 = \beta\|\mathcal{L}\|_2^2 = 2p\beta$ and $L_2 = \gamma\|\mathcal{A}\|_2^2 = 2\gamma$. The condition for the majorization function can be easily checked (Sun et al., 2016; Song et al., 2015).

Lemma 15. *By the KKT optimality conditions the optimal solution for $\mathbf{w}^{(t+1)}$ takes the following form*

$$\mathbf{w}^{(t+1)} = \left(\mathbf{w}^{(t)} - \frac{1}{L_1 + L_2}\left(\nabla f_1\left(\mathbf{w}^{(t)}\right) + \nabla f_2\left(\mathbf{w}^{(t)}\right)\right)\right)^+, \tag{69}$$

where $(x)^+ \triangleq \max(x, 0)$, $\nabla f_1\left(\mathbf{w}^{(t)}\right) = \beta\mathcal{L}^*\left(\mathcal{L}\mathbf{w}^{(t)}\right) - \mathbf{c}_1 + \mathbf{b}_1$, $\nabla f_2\left(\mathbf{w}^{(t)}\right) = \gamma\mathcal{A}^*\left(\mathcal{A}\mathbf{w}^{(t)}\right) - \mathbf{c}_2$, and $L_1 + L_2 = 2(p\beta + \gamma)$. $\mathbf{b}_1 = \alpha[1/(\epsilon + w_1^{(t)}),\ \ldots,\ 1/(\epsilon + w_{p(p-1)/2}^{(t)})]^\top$.

The update for U and V take the same forms as discussed in (41) and (61). The update for $\boldsymbol{\lambda}$ can be solved by Algorithm 1, since the Laplacian spectral set $\mathcal{S}_\lambda$ for any graph structure can always be represented in the form (9). Next, the sub-problem for $\boldsymbol{\psi}$ takes a similar form as presented in (63) with a specific $\mathcal{S}_\psi$ depending on the graph structure. For the structures belonging to bipartite graph families, e.g., connected bipartite, multi-component bipartite, and regular bipartite graph, the sub-problem for $\boldsymbol{\psi}$ can be solved efficiently by the algorithm in 4.2.3. When we consider a more general convex set $\mathcal{S}_\psi$, the problem (63) is still a convex optimization problem and thus can be solved via disciplined convex programming frameworks like CVX. Note that the sub-problems involving $(U, \boldsymbol{\lambda})$ and $(V, \boldsymbol{\psi})$ are decoupled. As a consequence, the update steps for $U, \boldsymbol{\lambda}$ and $V, \boldsymbol{\psi}$ can be done in parallel.

5.0.1. SGLA Algorithm Summary

Algorithm 4 summarizes the implementation of the structured graph learning (SGLA) via joint Laplacian and adjacency spectral constraints. In Algorithm 4, the computationally most demanding step is the eigenvalue decomposition resulting in a worst-case complexity of $O\left(p^3\right)$. The SGLA algorithm requires two eigenvalue decomposition, while the previous two algorithms the SGL and the SGA require only one eigendecomposition per iteration. Interestingly, for the SGLA algorithm the sub-problems involving eigendecompositions are decoupled, and they can be carried out parallelly to distribute the computational load.

Algorithm 4: SGLA

Input: SCM S, $\mathbf{w}^{(0)}$, $\mathcal{S}_\lambda$, $\mathcal{S}_\psi$, β, γ, α, $\epsilon > 0$;

1 $t \leftarrow 0$;
2 **while** *stopping criteria not met* **do**
3 update $\mathbf{w}^{(t+1)}$ as in (69);
4 update $U^{(t+1)}$ as in (41);
5 update $V^{(t+1)}$ as in (61);
6 update $\boldsymbol{\lambda}^{(t+1)}$ by solving (44) with Algorithm 1;
7 update $\boldsymbol{\psi}^{(t+1)}$ by solving (64) with Algorithm 1;
8 $t \leftarrow t + 1$;
9 **end**

Output: $\mathcal{L}\mathbf{w}^{(t+1)}$.

In the SGLA algorithm, imposing spectral constraints on the two graph matrices jointly is key for enforcing complicated structures. For instance, consider the case of learning a k-component bipartite graph structure, also known as bipartite graph clustering. For this structure, there exist k disjoint groups where each group is a bipartite graph. Such structural requirement makes this an extremely challenging problem. Hence, to the best of our knowledge, the SGLA algorithm is the first single-stage method capable of learning this structure directly from the sample statistics. SGLA satisfies these structural requirements by plug-in the Laplacian spectral properties for the k-component structure along with the adjacency spectral constraints for the bipartite structure (i.e., $\mathcal{S}_\lambda$ and $\mathcal{S}_\psi$ as in (17)) in Algorithm 4 [cf. step 8, 9].

The subsequence convergence result for SGLA algorithm is now established.

Theorem 7. *The sequence* $\left(\mathbf{w}^{(t)}, U^{(t)}, \boldsymbol{\lambda}^{(t)}, V^{(t)}, \boldsymbol{\psi}^{(t)}\right)$ *generated by Algorithm 4 converges to the set of KKT points of Problem* (65).

Proof. The detailed proof is deferred to the Appendix 8.5. □

6. Experiments

In this section, we present a number of comprehensive experiments for the evaluation of the performance of the proposed algorithms, i.e., SGL, SGA, and SGLA. The advantage of incorporating spectral information in the proposed framework is clearly illustrated. First, we introduce the experimental settings in Subsection 6.1 and the benchmarks for comparison in 6.2. Then the experiments are organized in the following three parts: Subsection 6.3 evalu-

ates SGL for learning the following graph structures: grid, modular, and multi-component graphs; Subsection 6.4 shows bipartite graph learning with SGA, and Subsection 6.4 shows multi-component bipartite graph learning via SGLA.

6.1. Experimental Settings

For the experiments involving synthetic data, we create several synthetic data sets based on different graph structures $\mathcal{G}$. First, we generate an IGMRF model parameterized by the true precision matrix $\Theta_{\mathcal{G}}$, which satisfies the Laplacian constraints in (2) as well as the specific graph structure. Then, a total of n samples $\{\mathbf{x}_i \in \mathbb{R}^p\}_{i=1}^n$ are drawn from the IGMRF model as $\mathbf{x}_i \sim \mathcal{N}\left(\mathbf{0}, \Theta_{\mathcal{G}}^{\dagger}\right)$. The sample covariance matrix (SCM) S is computed as

$$S = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}}_i)(\mathbf{x}_i - \bar{\mathbf{x}}_i)^{\top}, \;\; \text{with} \;\; \bar{\mathbf{x}}_i = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i. \tag{70}$$

The algorithms use S and prior information regarding the target graph families, i.e., whether the graph is k-component, bipartite, or k-component bipartite. We set $c_1 = 10^{-5}$ and $c_2 = 10^4$. We observed that the experimental performances of the algorithms are not sensitive to different values of c_1 and c_2 as long as they are reasonably small and large, respectively.

The choice for the hyperparameters β, γ, and α are discussed for each case separately. For each scenario, 20 Monte Carlo simulations are performed. We use the relative error (RE) and F-score (FS) to objectively evaluate the performance of the algorithms. Those performance measures are defined as

$$\mathsf{Relative\ Error} = \frac{\left\|\hat{\Theta} - \Theta_{\text{true}}\right\|_F}{\|\Theta_{\text{true}}\|_F}, \;\; \mathsf{F\text{-}Score} = \frac{2\mathsf{tp}}{2\mathsf{tp} + \mathsf{fp} + \mathsf{fn}}, \tag{71}$$

where $\hat{\Theta} = \mathcal{L}\hat{\mathbf{w}}$ is the final estimation result of the algorithm and Θ_{true} is the true reference graph Laplacian matrix. True positive (tp) stands for the case when there is an actual edge and the algorithm detects it; false positive (fp) stands for the case when there is no actual edge but algorithm detects one; and false negative (fn) stands for the case when the algorithm failed to detect an actual edge. The F-score takes values in $[0, 1]$ where 1 indicates perfect structure recovery (Egilmez et al., 2017). To check the performance evolution for each iteration t we evaluate the RE and FS with $\hat{\Theta}^{(t)}$. Algorithms are terminated when the relative change in $\mathbf{w}^{(t)}$ is relatively small.

6.2. Benchmarks

The CGL algorithm proposed in Egilmez et al. (2017) is the state-of-the-art method for estimating a connected combinatorial graph Laplacian matrix from the sample covariance matrix. For synthetic data experiments with connected graph structure (e.g., modular, grid, and connected bipartite), we compare the performance of the SGL algorithm to CGL. Additionally, for more insight, we also compare it to some heuristic based approaches, which are: i) the pseudo-inverse of the sample covariance matrix $S^{\dagger}$, which is denoted as Naive

and ii) the solution of following quadratic program

$$\mathbf{w}^\star = \underset{\mathbf{w}\geq 0}{\mathsf{argmin}} \left\| S^\dagger - \mathcal{L}\mathbf{w} \right\|_F^2, \tag{72}$$

which is denoted as QP.

To the best of our knowledge, there is no existing method capable of learning a graph Laplacian matrix with multiple components (e.g., k-component and k-component bipartite). Thereby, for the sake of completeness, we compare against Naive and QP, which are expected to give meaningful results for large sample size scenarios.

For experiments with real data, we compare the performance with GLasso (Friedman et al., 2008), GGL[2], constrained Laplacian rank algorithm CLR (Nie et al., 2016), spectral clustering (Ng et al., 2002), and k-means clustering. Unlike CGL, the GGL algorithm aims to estimate a generalized graph Laplacian matrix. As observed in Egilmez et al. (2017), GGL outperforms CGL in most scenarios, therefore, we omit a comparison with CGL for real data cases. Note that GGL and GLasso are incapable of estimating a standard Laplacian matrix in (2), hence it is not possible to compare those methods in synthetic experiments. For CGL, GGL, and GLasso, the sparsity hyperparameter α is chosen according to the guidelines outlined in (Egilmez et al., 2017; Zhao et al., 2012).

6.3. Performance Evaluation for the SGL Algorithm

In this Subsection, we evaluate the performance of the SGL algorithm (Algorithm 2) on grid graph, modular graph, multi-component graph, popular synthetic structures for clustering, and real data.

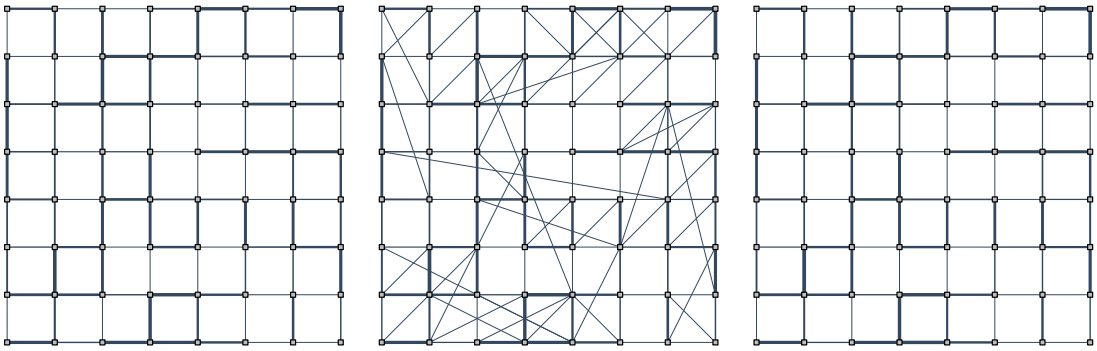
6.3.1. Grid graph

We consider a grid graph structure denoted as $\mathcal{G}_{\mathsf{grid}}(p)$, where $p = 64$ are the number of nodes, each node is connected to their four nearest neighbors (except the nodes at the boundaries), and edge weights are selected in a random fashion uniformly distributed from $[0.1, 3]$.

Figure 3 depicts the graph structures learned by SGL and CGL for $n/p = 100$. Edges whose values are smaller than 0.05 were discarded for the CGL algorithm and the SGL algorithm with the ℓ_1-norm regularization, whereas no post-processing was performed for the SGL algorithm with the reweighted ℓ_1-norm regularization. We use $\alpha = 0.0015$ for both SGL and CGL. In addition, we fix $\beta = 20$ for SGL.

Figure 4 compares the performance of the algorithms for different sample size regimes on the grid graph model. We fix the number of nodes $p = 64$ and vary the sample size n such that the algorithms are evaluated for different ratios of sample size per number of nodes. In the case of SGL, we fix $\beta = 10$ for $n/p \leq 100$, otherwise we fix $\beta = 100$. Additionally, we set $\alpha = 0$. For QP and Naive we do not need to set any parameters. It is observed in Figure 4, the SGL algorithm significantly outperforms the baseline approaches: for all the sample ratios SGL can achieve a lower average RE and higher average FS. In particular, to achieve a RE lower than 0.1, SGL requires a sample size ratio of approximately $n/p = 5$, whereas,

2. The code for the methods CGL, GGL is available at `https://github.com/STAC-USC/Graph_Learning`.



(a) True grid graph (b) CGL (c) SGL (proposed)

Figure 3: Sample results of learning $\mathcal{G}_{\mathsf{grid}}(64)$ (a) True grid graph, (b) CGL ($\mathsf{RE} = 0.0442, \mathsf{FS} = 0.7915$) and (c) SGL with reweighted ℓ_1-norm ($\mathsf{RE} = 0.0378, \mathsf{FS} = 1$).

Naive, QP, and CGL require sample size ratios of approximately $n/p = 80$, $n/p = 30$, and $n/p = 30$, respectively.

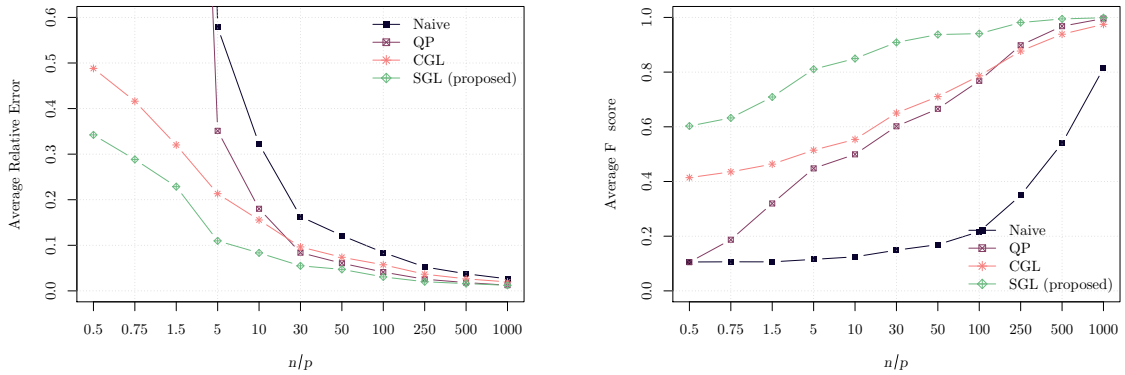


Figure 4: Average performance results for learning Laplacian matrix of a $\mathcal{G}_{\mathsf{grid}}$ graph. The SGL algorithm outperforms Naive, QP, and CGL for all the sample ratios.

6.3.2. Modular graph

We consider a random modular graph, also known as stochastic block model, $\mathcal{G}_{\mathsf{mo}}(p, k, \wp_1, \wp_2)$ with $p = 64$ nodes and $k = 4$ modules, where $\wp_1 = 0.01$ and $\wp_2 = 0.3$ are the probabilities of having an edge across modules and within modules, respectively. Edge weights are selected randomly uniformly from $[0.1, 3]$. Figure 5 illustrates the graph learning performances under different sample size ratios. From Figure 4, SGL and CGL outperforms the Naive and QP. Furthermore, for small sample size ratios (i.e., $n/p = 0.5$), SGL achieves a better performance than that of CGL, while they perform similarly for a larger sample size ratios.

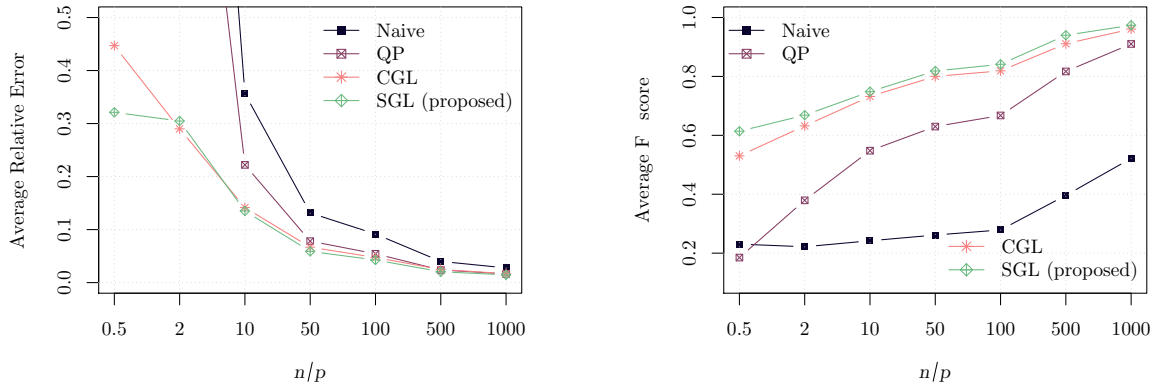


Figure 5: Average performance results for learning Laplacian matrix of a modular graph $\mathcal{G}_{\mathsf{mo}}$ with four modules. We set $\beta = 100, \alpha = 0$ for SGL. It can be observed that SGL outperforms the baseline approaches across almost the whole range of sample size ratios.

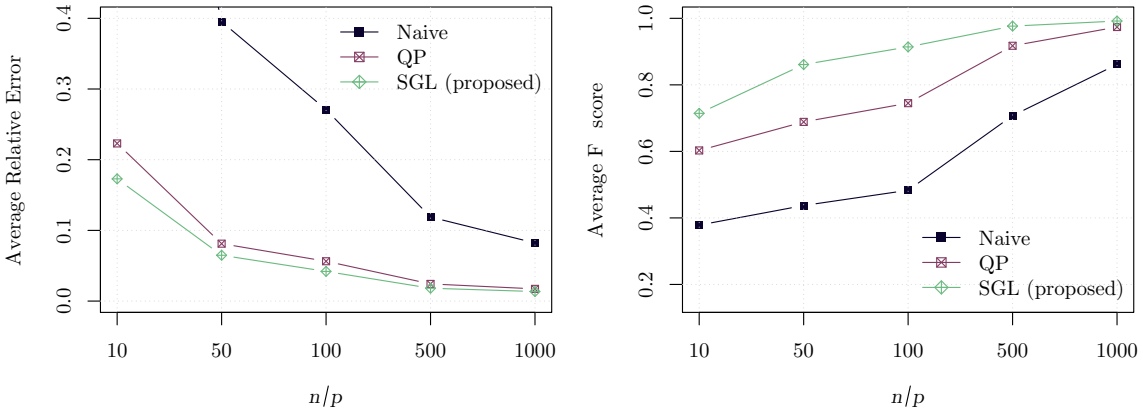


Figure 6: Average performance results as a function of the number of samples for learning the Laplacian matrix of a 4-component graph. SGL significantly outperforms the baseline approaches denoted as Naive and QP.

6.3.3. Multi-component graph

We consider learning a k-component graph also known as block-structured graph denoted as $\mathcal{G}_{\mathsf{mc}}(p, k, \wp)$, with $p = 64$, $k = 4$ and $\wp = 0.5$, where p is the number of nodes, k is the number of components, and $\wp$ is the probability of having an edge between any two nodes inside a component while the probability of having an edge between any two nodes from different components is zero. Edge weights are selected randomly uniformly from $[0.1, 3]$. Figure 6 illustrates the graph learning performances in terms of average RE and FS.

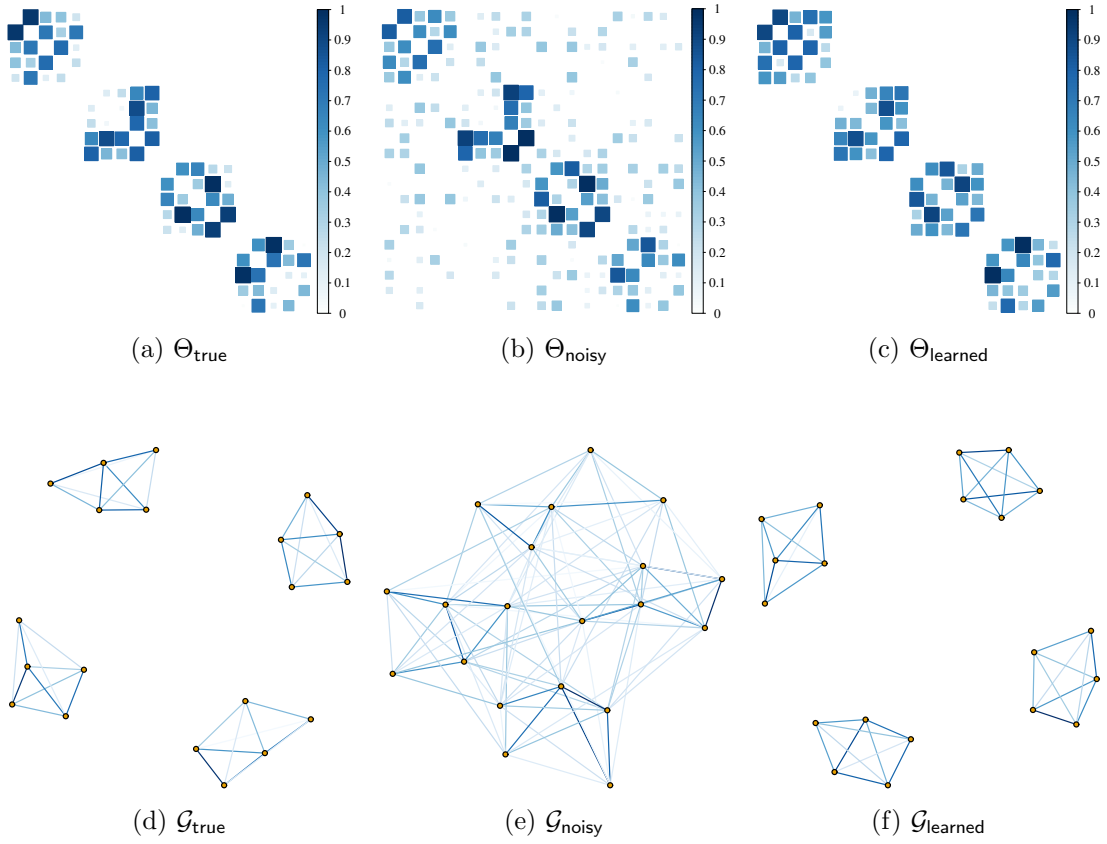
6.3.4. Multi-component graph: noisy setting

Here we consider the case of learning a multi-component graph under noise. At first, we generate a 4-component graph $\mathcal{G}_{\mathsf{mc}}(20, 4, 1)$ with equal number of nodes across different components. The nodes in each component are fully connected and the edges are drawn randomly uniformly from $[0, 1]$. Then, we add random noise to all the in-component and

out component edges. The noise is represented as an Erdos-Renyi graph $\mathcal{G}_{\mathsf{ER}}(p, \wp)$, where $p = 20$ is the number of nodes, $\wp = 0.35$ is the probability of having an edge between any two pair of nodes, and edge weights are randomly uniformly drawn from $[0, \kappa]$. Specifically, we consider a scenario where each sample $\mathbf{x}_i \sim \mathcal{N}\left(\mathbf{0}, \Theta^{\dagger}_{\mathsf{noisy}}\right)$ used for computing the SCM is drawn from the noisy precision matrix

$$\Theta_{\mathsf{noisy}} = \Theta_{\mathsf{true}} + \Theta_{\mathsf{ER}}, \tag{73}$$

where Θ_{true} is the true Laplacian matrix and Θ_{ER} represents the noise that follows an ER graph structure. Figure 7 illustrates an instance of the SGL performance for a noisy multi-component graph with fixed $n/p = 30$, $\beta = 400$, $\alpha = 0.1$, and $\kappa = 0.45$.



(a) Θ_{true} (b) Θ_{noisy} (c) $\Theta_{\mathsf{learned}}$

(d) $\mathcal{G}_{\mathsf{true}}$ (e) $\mathcal{G}_{\mathsf{noisy}}$ (f) $\mathcal{G}_{\mathsf{learned}}$

Figure 7: An example of estimating a 4-component graph. Heat maps of the graph matrices: (a) the ground truth graph Laplacian matrix Θ_{true}, (b) Θ_{noisy} after being corrupted by noise, (c) $\Theta_{\mathsf{learned}}$ the learned graph Laplacian with a performance of $(\mathsf{RE}, \mathsf{FS}) = (0.210, 1)$, which means a perfect structure recovery even in a noisy setting that heavily suppresses the ground truth weights. The panels (d), (e), and (f) correspond to the graphs represented by the Laplacian matrices in (a), (b), and (c), respectively.

6.3.5. Multi-component graph: model mismatch

SGL requires the knowledge of the number of components k for learning a multi-component graph structure. That assumption is also a requirement for similar methods such as CLR.

In case the number of components is not available *a priori*, one can infer k via methods for model selection e.g. cross validation, Bayesian information criteria (BIC), or Akaike information criteria (AIC). In addition, we also investigate the performance when inaccurate information about the true number of clusters is given.

We consider an experiment involving model mismatch: the underlying Laplacian matrix that generates the data has j components, but we actually use k, $k \neq j$, number of components to estimate it. We generate a 7-component graph $\mathcal{G}_{\mathsf{mc}}(49, 7, 1)$, where edge weights are drawn from a Uniform distribution supported on $[0, 1]$. Additionally, we consider a noisy model as in (73) i.e., $\Theta_{\mathsf{noisy}} = \Theta_{\mathsf{true}} + \Theta_{\mathsf{ER}}$, where the noise is an Erdos-Renyi graph $\mathcal{G}_{\mathsf{ER}}(49, 0.25)$ whose edges are uniformly sampled from $[0, 0.45]$.

Figure 8 shows an example where the underlying graph has seven components, but we use SGL with $k = 2$. As we can observe, even though the number of components is mismatched and the data is noisy, the SGL algorithm is still able to identify the true structure with a reasonable performance in terms of F-score and average relative error.

Therefore, even in the lack of true information regarding the number of components in a graph, the graph learned from the SGL algorithm can yield an approximate graph very close to the true graph, which can be used as an input to other algorithms for post-processing to infer a more accurate graph.

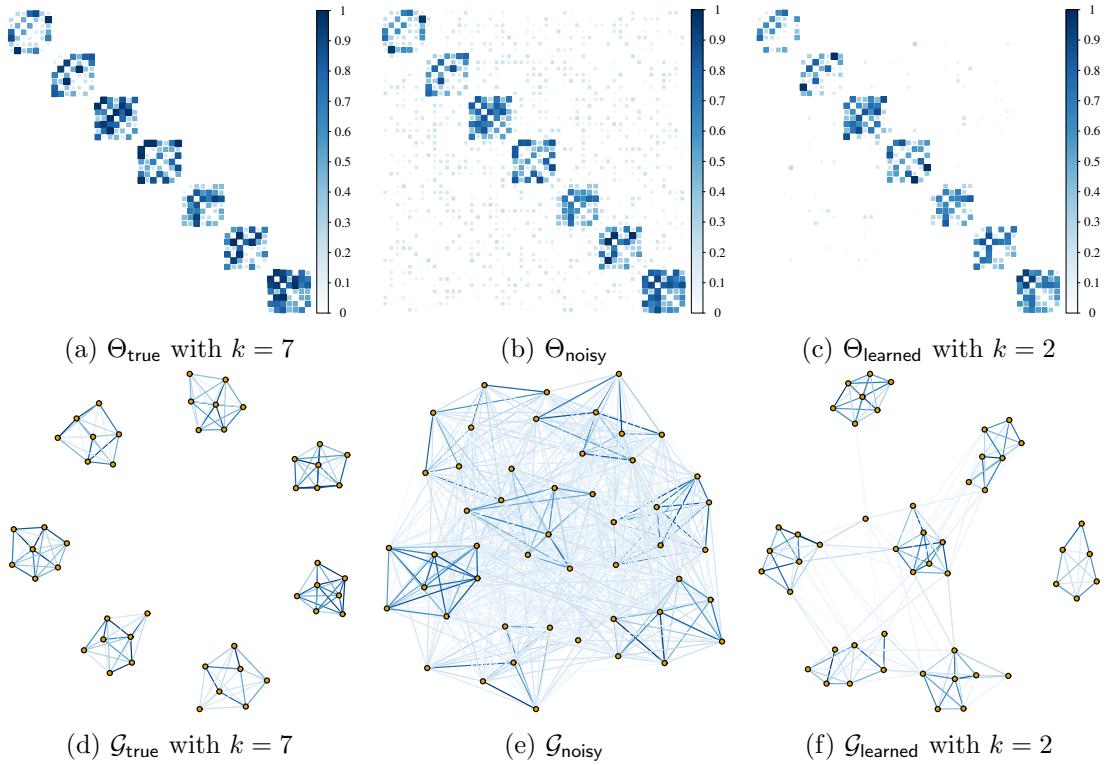
Figure 9 depicts the average performance of SGL as a function of k. The settings for the experiment are the same as in Figure 8, except now we use different number of components information for each instance. As expected, SGL has its best performance when k matches with the true number of the components in the graph. This also suggests that SGL could be seamlessly integrated with model selection techniques to dynamically determine the number of clusters to use in a single algorithm (Figueiredo and Jain, 2002; Schaeffer, 2007; Fraley and Raftery, 2007).

6.3.6. Popular multi-component structures

We now consider the classical problem of clustering using popular synthetic structures. We generate 100 nodes per cluster distributed according to structures colloquially known as *two moon*, *two circles*, *three spirals*, *three circles*, *worms* and *helix 3d*. Figure 10 depicts the results of learning the clusters structures using the proposed algorithm SGL.

6.3.7. Real data: animals data set

Herein, the animals data set (Osherson et al., 1991; Lake and Tenenbaum, 2010) is considered to learn weighted graphs. Every node denotes a unique animal and edge weights represent similarities among them. The data set consists of binary values (categorical non-Gaussian data) that represent the answers to questions such as "is warm-blooded?", "has lungs?", etc. There are a total of 102 such questions, which make up the features for 33 distinct animals. Figure 11 shows the results of estimating the graph of the animals data set using the SGL algorithm with $\beta = 1/2$ and $\alpha = 0$. For GGL and GLasso we set $\alpha = 0.05$. The input for all the algorithms is the sample covariance matrix plus an identity matrix scaled by 1/3 Egilmez et al. (2017). The evaluation of the estimated graphs is based on visual inspection. It is expected that similar animals such as (*ant*, *cockroach*), (*bee*, *butterfly*), and



(a) Θ_{true} with $k = 7$ (b) Θ_{noisy} (c) Θ_{learned} with $k = 2$

(d) $\mathcal{G}_{\text{true}}$ with $k = 7$ (e) $\mathcal{G}_{\text{noisy}}$ (f) $\mathcal{G}_{\text{learned}}$ with $k = 2$

Figure 8: Heat maps of the graph matrices: (a) the ground truth graph Laplacian of a seven-component graph Θ_{true}, (b) Θ_{noisy} after being corrupted by noise, (c) Θ_{learned} the learned graph Laplacian with a performance of $(\text{RE}, \text{FS}) = (0.18, 0.81)$. The panels (d), (e), and (f) correspond to the graphs represented by the Laplacian matrices in (a), (b), and (c), respectively. In Figure 8 (c) and (f) we are essentially getting results corresponding to a two-component graph, which is imperative from the usage of spectral constraints of $k = 2$. It is observed that the learned graph (f) consists of the true graph structure in (d) and some extra edges with very small weights which are due to the inaccurate spectral information. One can use some simple post-processing techniques (e.g., thresholding of elements in the learned matrix Θ), to recover the true component structure.

(*trout*, *salmon*) would be clustered together. Based on this premise, it can be seen that the SGL algorithm yields a clearer graph than the ones learned by GGL and GLasso.

Figure 12 compares the clustering performance of the SGL method with the state-of-the-art clustering algorithms: (a) k-means clustering, (b) spectral clustering [3], (c) CLR with $k = 10$, and (d) SGL with $k = 10$. It is remarked that all the algorithms except SGL are designed specifically for clustering and are not capable of estimating similarities (edge weights) among nodes. SGL, on the other hand, is capable of doing both clustering and the estimation of the edge weights jointly.

3. Code for spectral and k-means is available at `https://CRAN.R-project.org/package=kernlab` (Karatzoglou et al., 2004).

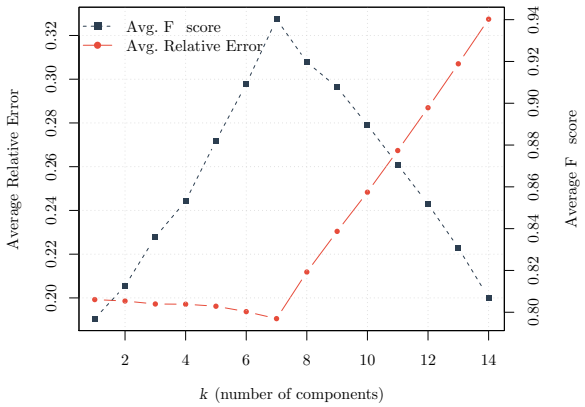
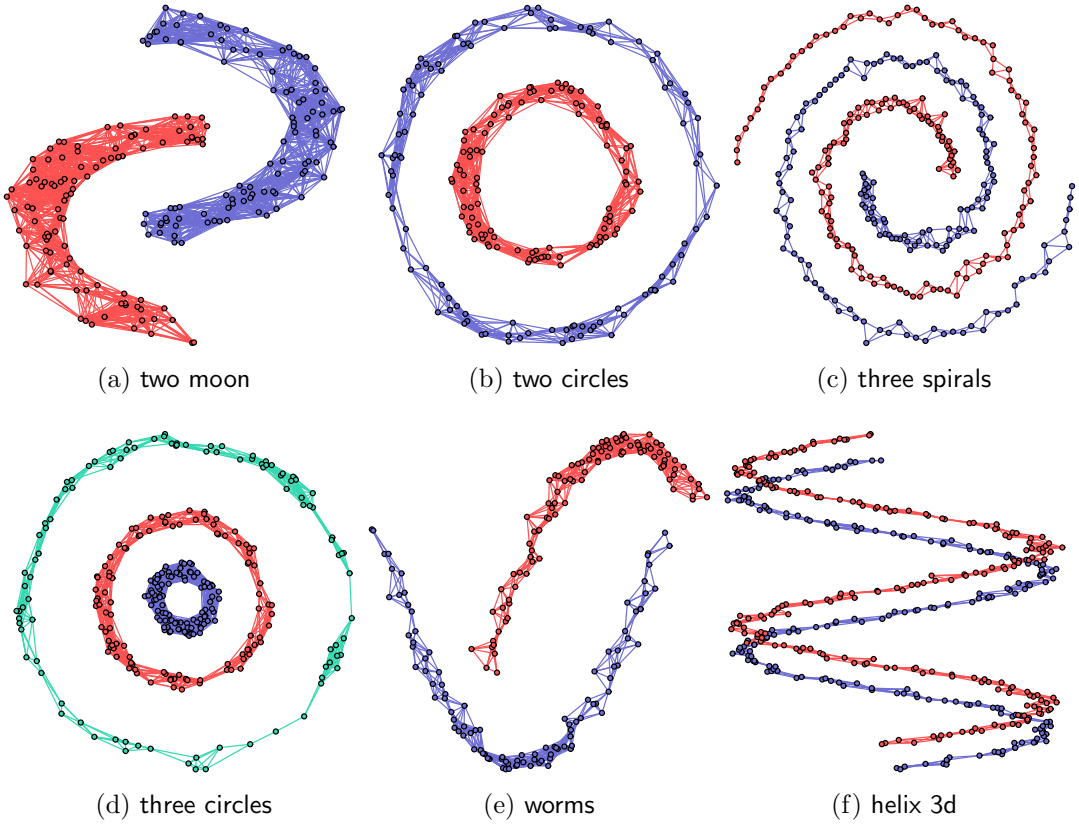


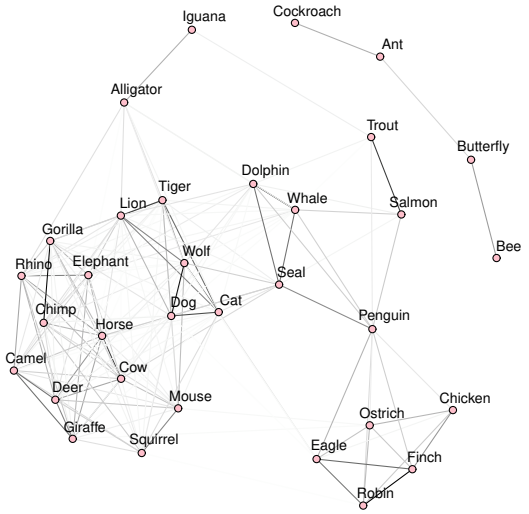
Figure 9: Average performance results as a function of the number of components k: best results are obtained for the true number of components. The performance is monotonically increasing and eventually reaches a peak in F-score when $k = 7$.



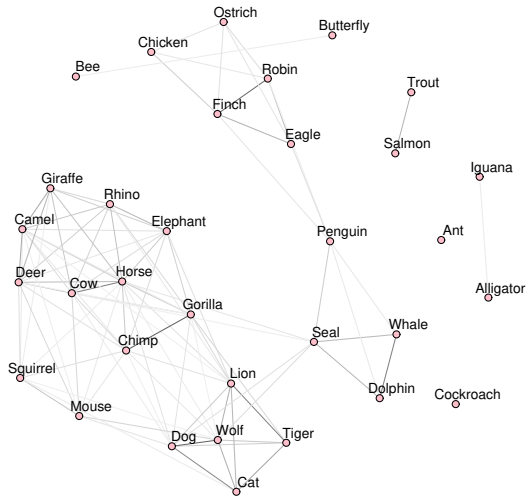
(a) two moon (b) two circles (c) three spirals

(d) three circles (e) worms (f) helix 3d

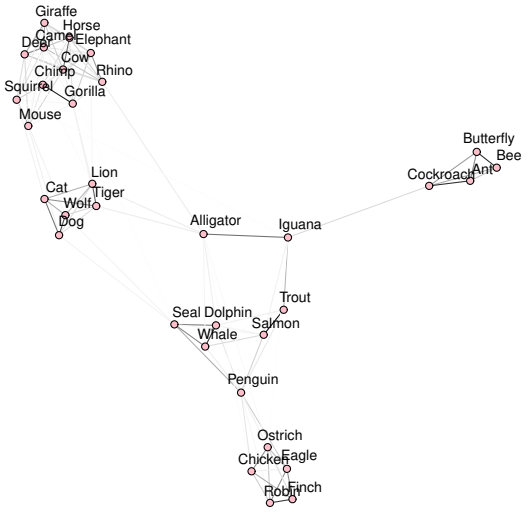
Figure 10: SGL is able to perfectly cluster the data points according to the cluster membership for all the structures.



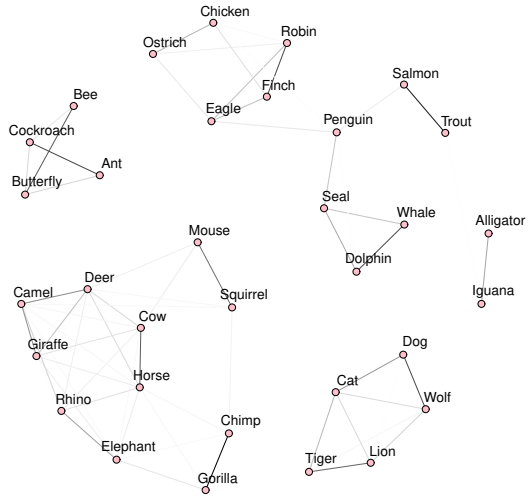
(a) GLasso (Friedman et al., 2008)



(b) GGL (Egilmez et al., 2017)

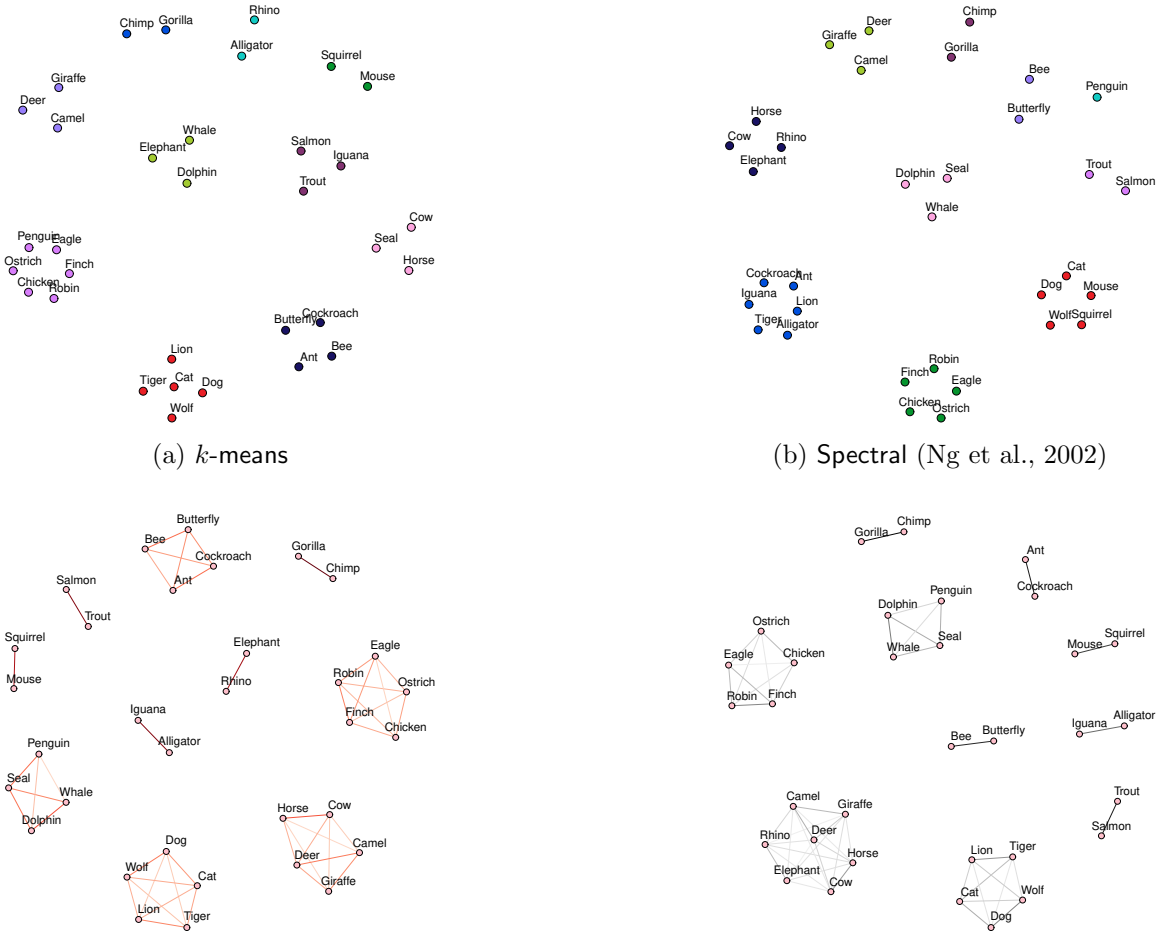


(c) SGL with $k = 1$ (proposed)



(d) SGL with $k = 5$ (proposed)

Figure 11: Learning the connectivity of the animals data set with (a) GLasso, (b) GGL, (c) SGL with $k = 1$ and (d) SGL with $k = 5$. For all graphs, darker edges denote stronger connections among animals. The methods (a) GGL, (b) GLasso, and (c) SGL $k = 1$ were expected to obtain sparse-connected graphs. But, GGL, GLasso split the graph into multiple components due to the sparsity regularization. While SGL using sparsity regularization along with spectral constraint $k = 1$ (connectedness) yields a sparse-connected graph. (d) SGL with $k = 5$ obtains a graph that depicts a more detailed representation of the network of animals by clustering similar animals within the same component. This highlights the fact that the control of the number of components may yield an improved visualization. Furthermore, the animal data is categorical (non-Gaussian) which does not follow the IGMRF assumption, hence the above result also establishes the capability of SGL under mismatch of the data model.



(a) k-means (b) Spectral (Ng et al., 2002)

(c) CLR (Nie et al., 2016) (d) SGL with $k = 10$ (proposed)

Figure 12: All the methods obtain 10 components intending to group similar animals together. Clustering with k-means and spectral methods yield components with uncommon (possibly wrong) groups. For example, in (a) *seal, cow, horse* are grouped together, and in (b) *cockroach, lion, iguana, tiger, ant, alligator* are grouped together which is also contradicts common sense. On the other hand, it is observed that both CLR (c) and SGL (d) are able to obtain groups of animals adhering to our general expectation. Although both the results vary slightly, the final results from both the methods are meaningful. For example, CLR groups all the insects (*bee, butterfly, cockroach, ant*) together in one group, while SGL splits them into two groups, one with *ant, cockroach* and another with *bee, butterfly*. On the other hand, SGL groups the herbivore mammals (*horse, elephant, giraffe, deer, camel, rhino, cow*) together in one group, while CLR splits these animals into two groups, one containing *rhino, elephant* and another group containing the rest.

6.3.8. Real data: Cancer Genome data set

We now consider the RNA-Seq Cancer Genome Atlas Research Network (Weinstein et al., 2013) data set available at UC-Irvine Machine Learning Database (Dheeru and Karra Taniskidou, 2017). This data set consists of genetic features which map 5 types of cancer namely: breast carcinoma (BRCA), kidney renal clear-cell carcinoma (KIRC), lung adenocarcinoma

(LUAD), colon adenocarcinoma (COAD), and prostate adenocarcinoma (PRAD). In Figure 13, those cancer types are labeled with colors *black, blue, red, violet*, and *green*, respectively. The data set consists of 801 labeled samples with 20531 genetic features. The goal with this data set is to cluster the nodes according to their tumor type on the basis of those genetic features. We apply SGL with $k = 5$ and $\beta = 5$.

We also compare the SGL performance against the existing state-of-the-art methods for graph learning algorithms namely GLasso with $\alpha = 1.71$ and GGL with $\alpha = 1.71$. We also include in the performance comparison the graph-based clustering algorithm CLR with $k = 5$ and $m = 10$, where m is the number of neighbors taken into the consideration when creating an initial affinity matrix. Additionally, we conducted experiments with CLR for different choices of $m = 3, 5, 7$. We observed similar performances[4] for all those values of m. In Figure 13, GLasso and GGL are not able to enforce the structural components and learn a connected graph with many wrong edges. SGL outperforms CLR in this clustering task, even though the later is a specialized clustering algorithm.

The improved performance of the SGL may be attributed to two main reasons: i) SGL is able to estimate the graph structure and weight simultaneously, which is essentially an optimal joint procedure, ii) SGL is able to capture the conditional dependencies (i.e., the precision matrix entries) among nodes, whereas CLR encodes the connectivity via the direct pairwise distances. The conditional dependencies are expected to give improved performance for clustering tasks (Hao et al., 2018). The performance with SGL shows an almost perfect clustering, which indicates that SGL may be used to perform simultaneous clustering and graph learning.

6.3.9. Effect of the parameter β

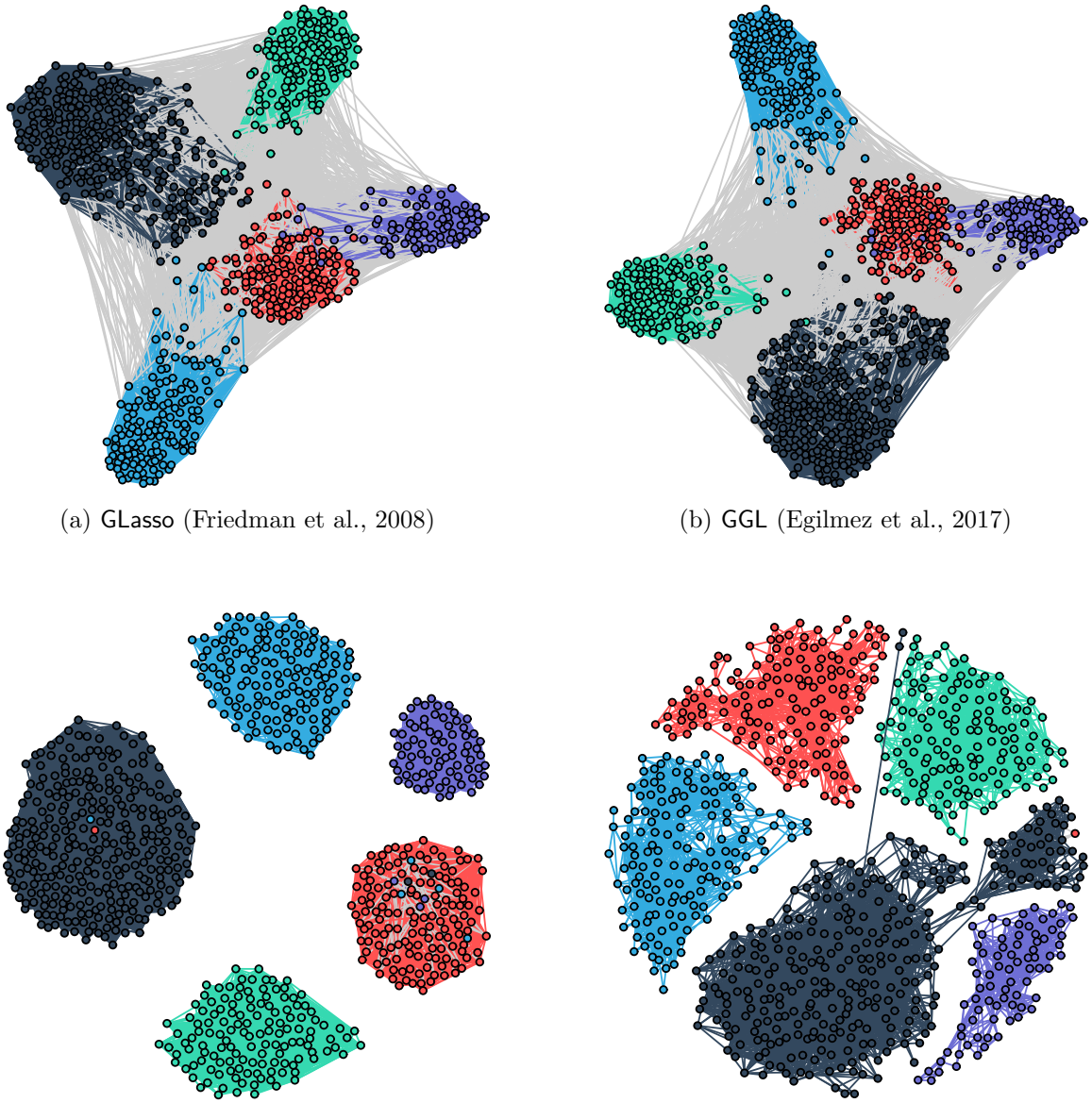
In the current subsection, we study the effect of the parameter β on the algorithmic performance in terms of RE and FS. It is observed from Figure 14 that a large value of β enables SGL to obtain a small RE and a large FS. For large values of β, the formulation puts more weight on the penalization term so as to make it closer to the spectral constraints. In addition, the increase of β tends to stabilize the values of RE and FS. Empirically, however, we observed that a huge value of β slows down the convergence speed of the algorithm. Similar observations also applies to the parameter γ for SGA and SGLA algorithms.

6.4. Performance Evaluation for the SGA Algorithm

A bipartite graph is denoted as $\mathcal{G}_{\mathsf{bi}}(p_1, p_2, \wp)$, where p_1 and p_2 are the number of nodes in each of the two disjoint groups, and $\wp$ is the probability of having an edge connecting nodes of the disjoint groups.

Figure 15 depicts the average performance of the algorithms for different sample size regimes for a bipartite graph structure $\mathcal{G}_{\mathsf{bi}}(p_1 = 40, p_2 = 24, \wp = 0.6)$ with edge weights sampled uniformly from $[1, 3]$. We set $\gamma = 10^5$. Here we consider a connected bipartite graph, thus the performance of SGA is also compared to that of CGL.

4. The authors in Nie et al. (2016) report that CLR quite robust to the choice of m.



(a) GLasso (Friedman et al., 2008)

(b) GGL (Egilmez et al., 2017)

(c) CLR (Nie et al., 2016)

(d) SGL (proposed)

Figure 13: Learning the clustered graph for the full 801 genetic samples with 20531 features of the PANCAN data set. Graph learned with (a) GLasso, (b) GGL method: grey colored edges indicate wrong connections, (c) CLR method: there are two misclassified points in the *black* group and 10 misclassified points in the *red* group, and (d) Graph learned with proposed SGL method. The label information obviously is not taken into consideration. The result for SGL is consistent with the label information: samples belonging to each set are connected together and the components are fairly separated with only one wrong connection. Furthermore, the graph for the BRCA (black) data sample highlights an inner sub-group, which could be possibly attributed to a sub type of cancer. This result shows that SGL is able to perform both clustering and edge weights learning simultaneously.

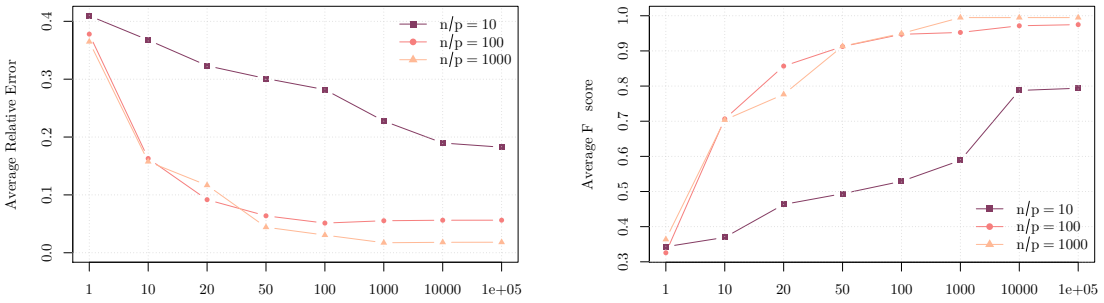


Figure 14: Effect of the parameter β on the SGL algorithm. We consider estimating a multi-component graph structure $\mathcal{G}_{\mathsf{mc}}(32, 4, 0.5)$ where edge weights were drawn randomly uniformly from $[0, 1]$. It is observed that a large value of β enables the SGL to obtain a small RE and a large FS.

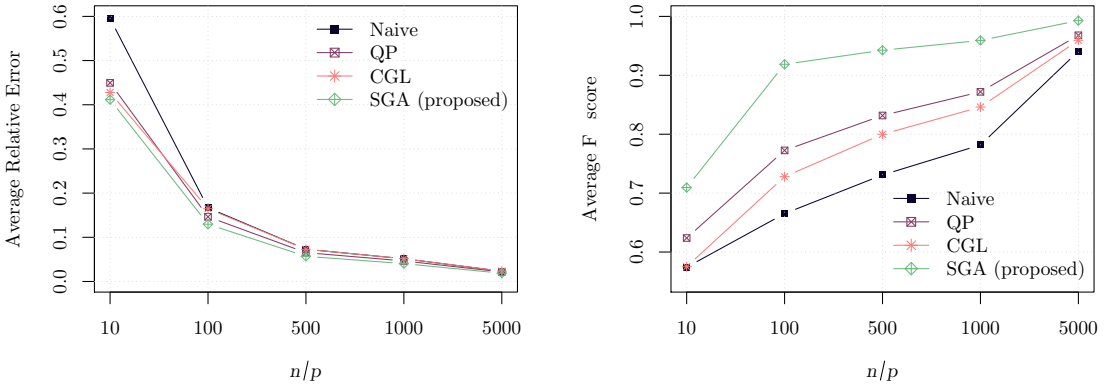


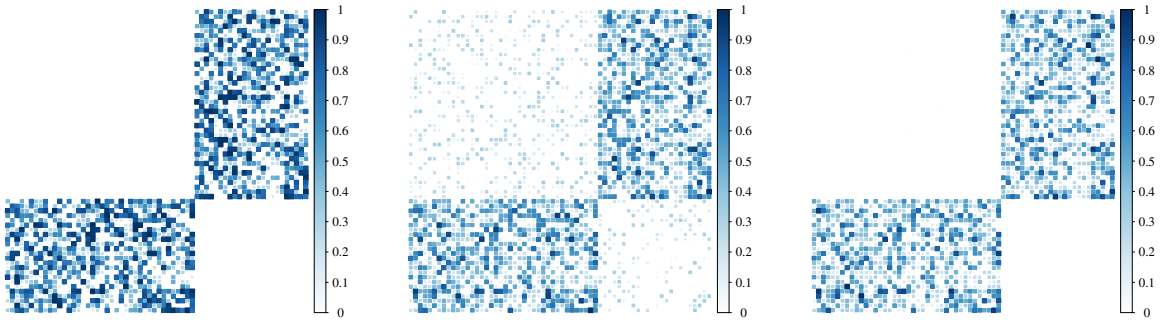
Figure 15: Average performance results for learning Laplacian matrix of a bipartite graph structure $\mathcal{G}_{\mathsf{bi}}$. The proposed SGA method significantly outperforms the base line approaches, Naive, QP and CGL.

6.4.1. Bipartite graph learning: noisy setting

We consider here learning a bipartite graph structure under noise as in equation (73), i.e., the samples used in the computation of the SCM are obtained from a noisy precision matrix, for which, the ground truth precision matrix corresponds to a bipartite graph. At first, we generate $\mathcal{G}_{\mathsf{bi}}(40, 24, 0.70)$ and the edge weights are drawn from $[0.1, 1]$. Then we add random noise to all the possible connections via an ER graph model as $\mathcal{G}_{\mathsf{ER}}(64, 0.35)$ with edges sampled from $[0, 1]$. Figure 16 illustrates an instance of the performance of SGA for learning a bipartite graph structure from noisy sample data.

6.5. Performance Evaluation for the SGLA Algorithm

Herein, we consider learning a multi-component bipartite graph structure. This structure is used in a number of applications including medicine and biology (Nie et al., 2017; Pavlopoulos et al., 2018), which makes it appealing from both practical as well as theoretical perspectives. In order to learn a multi-component bipartite graph structure from the SCM, we plug-in the Laplacian spectral constraints ($\mathcal{S}_\lambda$ as in (9)) corresponding to



(a) Θ_{true} (b) Θ_{noisy} (c) $\Theta_{\mathsf{learned}}$

Figure 16: An instance of bipartite graph learning with the SGA algorithm with data generated from a noisy graph Laplacian. We set $n/p = 500$ and $\gamma = 10^5$. (a) the ground truth Laplacian matrix (Θ_{true}), (b) Θ_{noisy} after being corrupted by noise, (c) the learned graph Laplacian with a performance of ($\mathsf{RE} = 0.219, \mathsf{FS} = 0.872$).

the multi-component structure along with the adjacency spectral constraints corresponding to the bipartite structure (i.e., $\mathcal{S}_\psi$ as in (14)) in Algorithm 4 [cf. step 8, 9]. Figure 17

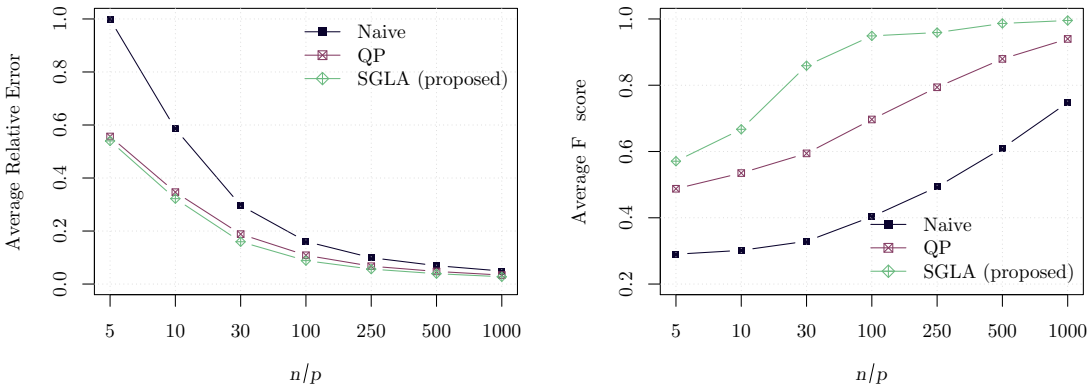


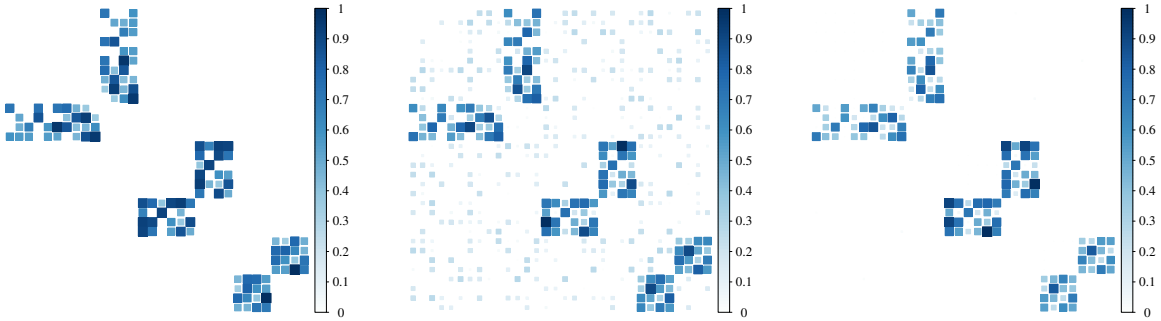
Figure 17: Average performance results for learning Laplacian matrices of a $\mathcal{G}_{\mathsf{bi}}$. The proposed SGLA method significantly outperforms the base line approaches QP and Naive.

depicts the average performance of the algorithms for different sample size regimes for a block-bipartite graph structure. The block structure contains three components $k = 3$ with unequal number of nodes, where each component represents a bipartite graph, which are denoted by $\mathcal{G}^1_{\mathsf{bi}}(20, 8, 0.5)$, $\mathcal{G}^2_{\mathsf{bi}}(12, 8, 0.6)^2$, and $\mathcal{G}^3_{\mathsf{bi}}(8, 8, 0.7)$. The edge weights are sampled uniformly from $[0.1, 3]$. We fix $\beta = 10^3$ and $\gamma = 10^3$. Since we consider a multi-component bipartite graph, we can only compare with QP and Naive. In terms of RE, the QP performance is comparable with that of SGLA, but in terms of FS, SGLA significantly outperforms the baseline methods.

6.5.1. Multi-component bipartite graph: noisy setting

We consider here learning a block-bipartite graph structure under noise as in (73), i.e., the samples used in the computation of the SCM are obtained from a noisy precision matrix,

for which the ground truth precision matrix corresponds to a multi-component bipartite graph. We first generate a graph with $p = 32$ nodes and $k = 3$ components with unequal number of nodes, where each component represents a bipartite graph structure denoted by $\mathcal{G}_{\mathsf{bi}}^1(10, 4, 0.7)$, $\mathcal{G}_{\mathsf{bi}}^2(6, 4, 0.8)^2$, and $\mathcal{G}_{\mathsf{bi}}^3(4, 4, 0.9)$. The edge weights are uniformly sampled from $[1, 3]$. We then add random noise to all the possible edges between any two nodes via an ER graph $\mathcal{G}_{\mathsf{ER}}(32, .35)$ with edges sampled from $[0, 1]$. We set $n/p = 250$, $\gamma = 10^5$ and $\beta = 10^5$. Figure 18 depicts one instance of the performance of SGLA for noisy bipartite graph structure.



(a) Θ_{true} (b) Θ_{noisy} (c) $\Theta_{\mathsf{learned}}$

Figure 18: An example of learning a 3-component bipartite graph structure with the SGLA algorithm. The samples used in the computation of the SCM are obtained from noisy Laplacian precision matrix. Heat maps of the graph matrices: (a) Θ_{true} the ground truth Laplacian matrix, (b) Θ_{noisy} after being corrupted by noise, (c) $\Theta_{\mathsf{learned}}$ the learned graph Laplacian with a performance of $(\mathsf{RE} = 0.225, \mathsf{FS} = 0.947)$, which means an almost perfect structure recovery even in a noisy model that heavily suppresses the ground truth weights.

7. Discussion and Conclusion

In this paper, we have considered spectral constraints on the eigenvalues of graph matrices in order to learn graphs with specific structures under the GGM setting. We have developed a unified optimization framework that is very general and puts forth plausible directions for future works.

7.1. Discussion

The extensions of this paper include considering more specific prior information on eigenvalues constraints, obtaining structured graph transform for graph signal processing applications, and extending the framework by considering other statistical models.

7.1.1. Other spectral prior information

In addition to the spectral properties considered in this paper, there are numerous interesting results in the literature that relate the eigenvalue properties to some specific graph structures, we refer readers to the comprehensive exposition in (Das and Bapat, 2005; Yu and Lu, 2014; Farber and Kaminer, 2011; Berman and Farber, 2011; Chung, 1997; Spielman and Teng, 2011; Chung, 1997; Mohar, 1997; Yang et al., 2003; Van Mieghem, 2010;

Schulte, 1996; Lin et al., 2014). The proposed framework has the potential to be extended to accommodate those prior spectral information.

7.1.2. Structured statistical models

Although the focus of the current paper is on the unification of spectral graph theory with the GGM framework, the proposed spectral constraints and developed algorithms are very general and can be integrated with other statistical models such as structured Ising model (Ravikumar et al., 2010), structured Gaussian covariance graphical models (Drton and Richardson, 2008), structured Gaussian graphical models with latent variables (Chandrasekaran et al., 2010), least-square formulation for graph learning (Nie et al., 2016), structured linear regression, and vector auto-regression models (Basu et al., 2015).

7.1.3. Structured graph signal processing

One of the motivations of the present work comes from the field of graph signal processing (GSP). This field provides tools for the representation, processing and analysis of complex networked-data (e.g., social, energy, finance, biology and, etc.), where the data are defined on the nodes of a graph (Shuman et al., 2013; Ortega et al., 2018). The eigenvectors of graph matrices are used to define a graph Fourier transform also known as dictionaries, used in a variety of applications including, graph-based filtering, graph-based transforms, and sampling for graph signals. More recently, it is realized that the graph eigenvectors with additional properties (e.g., sparsity) could be instrumental in performing complex tasks. For example, sparse eigenvectors of the graph matrices are important to investigate the uncertainty principle over graphs (Teke and Vaidyanathan, 2017). Few other examples include sparsifying eigenvectors (Sardellitti et al., 2019) and robust eigenvectors(Maretic et al., 2017). Within the proposed formulation, one can easily enforce desired properties on the eigenvectors by pairing the optimization step of (13), (16), and (8) with specific constraints on the eigenvectors (e.g., via regularization). Joint learning of structured graphs with specific properties on the eigenvectors have significant potential for GSP applications that have not been investigated yet, and constitute a promising research direction.

7.2. Conclusion

In summary, we have shown how to convert the combinatorial constraints of structured graph learning into algebraic constraints involving the eigenvalues of graph matrices. We have developed three algorithms SGL, SGA, and SGLA that can learn structured graphs from a large class of graph families. The algorithms are capable of learning the graph structure and their weights simultaneously by utilizing the spectral constraints of graph matrices directly into the Gaussian graphical modeling framework. The algorithms enjoy comprehensive theoretical convergence properties along with low computational complexity. Extensive numerical experiments with both synthetic and real data sets demonstrate the effectiveness of the proposed methods. We also pinpoint several extensions of future research directions.

Acknowledgments

We would like to thank the reviewers and editor for their suggestions to improve this paper. This work was supported by the Hong Kong GRF 16207019 research grant.

8. Appendix

8.1. Proof of Lemma 1

Proof. We define an index set Ω_t:

$$\Omega_t := \left\{ l \mid [\mathcal{L}\mathbf{x}]_{tt} = \sum_{l \in \Omega_t} x_l \right\}, \quad t \in [1, p]. \tag{74}$$

For any $\mathbf{x} \in \mathbb{R}^{\frac{p(p-1)}{2}}$, we have

$$\begin{aligned}
\|\mathcal{L}\mathbf{x}\|_F^2 &= 2 \sum_{k=1}^{\frac{p(p-1)}{2}} x_k^2 + \sum_{i=1}^{p} ([\mathcal{L}\mathbf{x}]_{ii})^2 && (75)\\
&= 4 \sum_{k=1}^{\frac{p(p-1)}{2}} x_k^2 + \sum_{t=1}^{p} \sum_{i,j \in \Omega_t,\ i \neq j} x_i x_j && (76)\\
&\leq 4 \sum_{k=1}^{\frac{p(p-1)}{2}} x_k^2 + \frac{1}{2} \sum_{t=1}^{p} \sum_{i,j \in \Omega_t,\ i \neq j} x_i^2 + x_j^2 && (77)\\
&= (4 + 2(|\Omega_t| - 1)) \sum_{k=1}^{\frac{p(p-1)}{2}} x_k^2 && (78)\\
&= 2p \|\mathbf{x}\|^2, && (79)
\end{aligned}$$

where the second equality is due to the fact that each x_k only appears twice on the diagonal; the first inequality achieves equality when each x_k is equal; the last equality follows the fact that $|\Omega_t| = p - 1$.

Therefore, by the definition of operator norm, we can obtain

$$\|\mathcal{L}\|_2 = \sup_{\|\mathbf{x}\|=1} \|\mathcal{L}\mathbf{x}\|_F = \sqrt{2p}, \tag{80}$$

concluding the proof.

□

8.2. Proof for Lemma 6

The Lagrangian of the optimization (44) is

$$L(\boldsymbol{\lambda}, \boldsymbol{\mu}) = -\sum_{i=1}^{q} \log \lambda_i + \frac{\beta}{2} \|\boldsymbol{\lambda} - \mathbf{d}\|_2^2 + \mu_1(c_1 - \lambda_1) + \sum_{i=2}^{q} \mu_i(\lambda_{i-1} - \lambda_i) + \mu_{q+1}(\lambda_q - c_2). \tag{81}$$

The KKT optimality conditions are derived as:

$$-\frac{1}{\lambda_i} + \beta(\lambda_i - d_i) - \mu_i + \mu_{i+1} = 0,\ i = 1, \ldots, q; \tag{82}$$

$$c_1 - \lambda_1 \leq 0; \tag{83}$$

$$\lambda_{i-1} - \lambda_i \leq 0,\ i = 2, \ldots, q; \tag{84}$$

$$\lambda_q - c_2 \leq 0; \tag{85}$$

$$\mu_i \geq 0,\ i = 1, \ldots, q+1; \tag{86}$$

$$\mu_1(c_1 - \lambda_1) = 0; \tag{87}$$

$$\mu_i(\lambda_{i-1} - \lambda_i) = 0,\ i = 2, \ldots, q; \tag{88}$$

$$\mu_{q+1}(\lambda_q - c_2) = 0; \tag{89}$$

Lemma 16. *The solution of the KKT system* (82)-(89) *is* $\lambda_i = (d_i + \sqrt{d_i^2 + 4/\beta})/2$*, for* $i = 1, \ldots, q$*, if* $c_1 \leq \lambda_1 \leq \cdots \leq c_q \leq c_2$ *hold.*

Proof. It is obvious that if the conditions $c_1 \leq \lambda_1 \leq \cdots \leq \lambda_q \leq c_2$ hold, then the solutions of the primal and dual variables satisfy all equations. □

We start from the corresponding unconstrained version of the problem (44) whose solution is

$$\lambda_i^{(0)} = \frac{1}{2}\left(d_i + \sqrt{d_i^2 + 4/\beta}\right). \tag{90}$$

If this solution satisfies all the KKT conditions (82)-(89), then it is also the optimal. Otherwise, each $\lambda_i^{(0)}$ that violates the conditions $c_1 \leq \lambda_1^{(0)} \leq \cdots \leq \lambda_q^{(0)} \leq c_2$ needs to be updated.

Situation 1: $c_1 \geq \lambda_1^{(0)} \geq \cdots \geq \lambda_r^{(0)}$, implying $c_1 - \frac{1}{c_1\beta} \geq d_1 \geq \cdots \geq d_r$, where at least one inequality is strict and $r \geq 1$. Without loss of generality, let the j-th inequality is strict with $1 \leq j \leq r$, i.e. $d_j > d_{j+1}$. The KKT optimality conditions for this pair are:

$$-\frac{1}{\lambda_j} + \beta(\lambda_j - d_j) - \mu_j + \mu_{j+1} = 0; \tag{91}$$

$$-\frac{1}{\lambda_{j+1}} + \beta(\lambda_{j+1} - d_{j+1}) - \mu_{j+1} + \mu_{j+2} = 0; \tag{92}$$

$$\lambda_j - \lambda_{j+1} \leq 0; \tag{93}$$

$$\mu_i \geq 0,\ i = j, j+1, j+2; \tag{94}$$

$$\mu_{j+1}(\lambda_j - \lambda_{j+1}) = 0; \tag{95}$$

We subtract the first two equations and obtain:

$$2\mu_{j+1} = \mu_{j+2} + \mu_j + \left(\frac{1}{\lambda_j} - \frac{1}{\lambda_{j+1}}\right) + \beta(\lambda_{j+1} - \lambda_j) + \beta(d_j - d_{j+1}) > 0, \tag{96}$$

due to the fact that $d_j > d_{j+1}$ and $\lambda_j \leq \lambda_{j+1}$. Since $\mu_{j+1} > 0$, we also have

$$2\mu_j = \mu_{j+1} + \mu_{j-1} + \left(\frac{1}{\lambda_{j-1}} - \frac{1}{\lambda_j}\right) + \beta(\lambda_j - \lambda_{j-1}) + \beta(d_{j-1} - d_j) > 0, \tag{97}$$

where $d_{j-1} \geq d_j$ and $\lambda_{j-1} \leq \lambda_j$. Similarly, we can obtain $\mu_j > 0$ with $2 \leq j \leq r$. In addition,

$$\mu_1 = -\frac{1}{\lambda_1} + \beta(\lambda_1 - d_1) + \mu_2 \tag{98}$$

$$-\frac{1}{c_1} + \beta(c_1 - d_1) + \mu_2 > 0. \tag{99}$$

Totally, we have $\mu_j > 0$ with $1 \leq j \leq r$. By (87) and (88), we obtain $\lambda_1 = \cdots = \lambda_r = c_1$. Therefore, we update

$$\lambda_1^{(1)} = \cdots = \lambda_r^{(1)} = c_1. \tag{100}$$

Situation 2: $\lambda_s^{(0)} \geq \cdots \geq \lambda_q^{(0)} \geq c_2$, implying $d_s \geq \cdots \geq d_q \geq c_2 - \frac{1}{c_2\beta}$, where at least one inequality is strict and $s \leq q$.

Similar to situation 1, we can also obtain $\mu_j > 0$ with $s+1 \leq j \leq m+1$ and thus $\lambda_s = \cdots = \lambda_q = c_2$. Therefore, we update $\lambda_s^{(0)}, \ldots, \lambda_q^{(0)}$ by $\lambda_s^{(1)} = \cdots = \lambda_q^{(1)} = c_2$.

Situation 3: $\lambda_i^{(0)} \geq \cdots \geq \lambda_m^{(0)}$, implying $d_i \geq \cdots \geq d_m$, where at least one inequality is strict and $1 \leq i \leq m \leq q$. Here we assume $\lambda_{i-1}^{(0)} < \lambda_i^{(0)}$ ($c_1 < \lambda_1^{(0)}$ if $i = 1$) and $\lambda_m^{(0)} < \lambda_{m+1}^{(0)}$ ($\lambda_q^{(0)} < c_2$ if $m = q$). Otherwise, this will be reduced to situation 1 or 3.

Similar to situation 1, we can also obtain $\mu_j > 0$ with $i+1 \leq j \leq m$ and thus $\lambda_i^{(1)} = \lambda_{i+1}^{(1)} = \cdots = \lambda_m^{(1)}$.

We sum up equations (91) with $i \leq j \leq m$ and obtain

$$-\frac{1}{\lambda_j} + \beta\lambda_j - \frac{1}{m-i+1}\left(\beta\sum_{j=i}^{m} d_j + \mu_i - \mu_{m+1}\right) = 0, \quad j = i, \ldots, m. \tag{101}$$

Here we need to use iterative method to find the solution that satisfies KKT conditions. It is easy to check that $\mu_i = \mu_{m+1} = 0$ when $\lambda_{i-1}^{(0)} < \lambda_i^{(0)}$ and $\lambda_m^{(0)} < \lambda_{m+1}^{(0)}$. In that case, according to (101), we have

$$\lambda_j = \frac{1}{2}\left(\bar{d}_{i\to m} + \sqrt{\bar{d}_{i\to m}^2 + 4/\beta}\right), \quad j = i, \ldots, m. \tag{102}$$

where $\bar{d}_{i\to m} = \frac{1}{m-i+1}\sum_{j=i}^{m} d_j$. Therefore, we update $\lambda_i^{(0)}, \ldots, \lambda_m^{(0)}$ by

$$\lambda_i^{(1)} = \cdots = \lambda_m^{(1)} = \frac{1}{2}\left(\bar{d}_{i\to m} + \sqrt{\bar{d}_{i\to m}^2 + 4/\beta}\right). \tag{103}$$

If there exists the case that $\lambda_{i-1}^{(1)} > \lambda_i^{(1)}$, we need to further update $\lambda_{i-1}^{(1)}, \lambda_i^{(1)}, \ldots, \lambda_m^{(1)}$ in the next iteration. It will include two cases to discuss:

1. $\lambda_{i-1}^{(1)}$ has not been updated by (103), implying that

$$\lambda_{i-1}^{(1)} = \lambda_{i-1}^{(0)} = \frac{1}{2}\left(\bar{d}_{i-1} + \sqrt{d_{i-1}^2 + 4/\beta}\right).$$

So $\lambda_{i-1}^{(1)} > \lambda_i^{(1)}$ means $d_{i-1} > \bar{d}_{i\to m}$. KKT conditions for this pair are:

$$-\frac{1}{\lambda_{i-1}} + \beta(\lambda_{i-1} - d_{i-1}) - \mu_{i-1} + \mu_i = 0; \tag{104}$$

$$-\frac{1}{\lambda_i} + \beta(\lambda_i - \bar{d}_{i\to m}) - \mu_i + \mu_{m+1} = 0; \tag{105}$$

$$\lambda_{i-1} - \lambda_i \leq 0; \tag{106}$$

$$\mu_p \geq 0,\ p = i-1, i, m+1; \tag{107}$$

$$\mu_i(\lambda_{i-1} - \lambda_i) = 0; \tag{108}$$

We subtract the first two equations and obtain

$$2\mu_i = \mu_{i-1} + \mu_{m+1} + (\frac{1}{\lambda_{i-1}} - \frac{1}{\lambda_i}) + \beta(\lambda_i - \lambda_{i-1}) + \beta(d_{i-1} - \bar{d}_{i\to m}) > 0, \tag{109}$$

and thus $\lambda_{i-1} = \lambda_i = \cdots = \lambda_m$. Then the equation (101) can be written as

$$-\frac{1}{\lambda_j} + \beta\lambda_j - \frac{1}{m-i+2}(\beta \sum_{j=i-1}^{m} d_j + \mu_{i-1} - \mu_{m+1}) = 0, \quad j = i-1, \ldots, m. \tag{110}$$

Hence, we update

$$\lambda_{i-1}^{(2)} = \cdots = \lambda_m^{(2)} = \left(\bar{d}_{(i-1)\to m} + \sqrt{\bar{d}_{(i-1)\to m}^2 + 4/\beta}\right)/2. \tag{111}$$

2. $\lambda_{i-1}^{(1)}$ has been updated by (103), implying that

$$\lambda_t^{(1)} = \cdots = \lambda_{i-1}^{(1)} = \frac{1}{2}\left(\bar{d}_{t\to(i-1)} + \sqrt{d_{t\to(i-1)}^2 + 4/\beta}\right),$$

with $t < i-1$. Then $\lambda_{i-1}^{(1)} > \lambda_i^{(1)}$ means $\bar{d}_{t\to(i-1)} > \bar{d}_{i\to m}$. Similarly, we can also obtain $\lambda_t = \lambda_{t+1} = \cdots = \lambda_m$ by deriving KKT conditions. We sum up equations (82) over $t \leq j \leq m$ and obtain

$$-\frac{1}{\lambda_j} + \beta\lambda_j - \frac{1}{m-t+1}\left(\beta\sum_{j=t}^{m} d_j + \mu_t - \mu_{m+1}\right) = 0, \quad j = t, \ldots, m. \tag{112}$$

So we update

$$\lambda_t^{(2)} = \cdots = \lambda_m^{(2)} = \frac{1}{2}\left(\bar{d}_{t\to m} + \sqrt{\bar{d}_{t\to m}^2 + 4/\beta}\right). \tag{113}$$

For the case that $\lambda_m^{(1)} > \lambda_{m+1}^{(1)}$, the update strategy is similar to the case $\lambda_{i-1}^{(1)} > \lambda_i^{(1)}$.

We iteratively check each situation and update the corresponding λ_i accordingly. We can check that the algorithm will be terminated with the maximum number of iterations of $q+1$ and $c_1 \leq \lambda_1^{(q+1)} \leq \cdots \leq \lambda_q^{(q+1)} \leq c_2$ holds for all variables. Because each updating above is derived by KKT optimality conditions for Problem (44), the iterative-update procedure summarized in Algorithm 1 converges to the KKT point of Problem (44).

8.3. Proof for Theorem 5

Proof. The proof of algorithm convergence is partly based on the proof of BSUM in Razaviyayn et al. (2013). We first show the linear independence constraint qualification on unitary constraint set $\mathcal{S}_U \triangleq \{U \in \mathbb{R}^{p\times q} | U^\top U = I_q\}$.

Lemma 17. *Linear independence constraint qualification (LICQ) holds on each $U \in \mathcal{S}_U$.*

Proof. We rewrite $\mathcal{S}_U$ as

$$\left\{ U \in \mathbb{R}^{p\times q} | g_{ij}(U) = \sum_{k=1}^{p} u_{ki}u_{kj} - I_{ij}, \forall 1 \le i \le j \le q \right\}, \tag{114}$$

where u_{ij} and I_{ij} are the elements of U and identity matrix I in i-th row and j-th column, respectively. It is observed that

$$\nabla g_{ij}(U) = \begin{cases} [0_{p\times(i-1)}; 2u_i; 0_{p\times(q-i)}], & \text{if } i = j; \\ [0_{p\times(i-1)}; u_j; 0_{p\times(j-i-1)}; u_i; 0_{p\times(q-j)}], & \text{otherwise.} \end{cases} \tag{115}$$

We can see u_i from $\nabla g_{ii}(U)$ will only appear in i-th column, but u_i from $\nabla g_{ij}(U)$ with $i \neq j$ will not appear in i-th column. Consequently, each $\nabla g_{ij}(U)$ cannot be expressed as a linear combination of the others, thus each $\nabla g_{ij}(U)$ is linear independent. □

Now we prove Theorem 5. It is easy to check that the level set $\{(\mathbf{w}, U, \boldsymbol{\lambda}) | f(\mathbf{w}, U, \boldsymbol{\lambda}) \le f(\mathbf{w}^{(0)}, U^{(0)}, \boldsymbol{\lambda}^{(0)})\}$ is compact, where $f(\mathbf{w}, U, \boldsymbol{\lambda})$ is the cost function in Problem (25). Furthermore, the sub-problems (31) and (42) have unique solutions since they are strictly convex problems and we get the global optima. According to Theorem 2 in Razaviyayn et al. (2013), we obtain that the sequence $(\mathbf{w}^{(t)}, U^{(t)}, \boldsymbol{\lambda}^{(t)})$ generated by Algorithm 2 converges to the set of stationary points. Note that U is constrained on the orthogonal Stiefel manifold that is nonconvex, while block MM framework does not cover nonconvex constraints. But the subsequence convergence can still be established (Fu et al., 2017b) due to the fact that the cost function value here is non-increasing and bounded below in each iteration.

Next we will further show that each limit of the sequence $(\mathbf{w}^{(t)}, U^{(t)}, \boldsymbol{\lambda}^{(t)})$ satisfies KKT conditions of Problem (29). Let $(\mathbf{w}^\infty, U^\infty, \boldsymbol{\lambda}^\infty)$ be a limit point of the generated sequence. The Lagrangian function of (29) is

$$\begin{aligned} L(\mathbf{w}, U, \boldsymbol{\lambda}, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, M) = &- \log \mathrm{gdet}(\mathrm{Diag}(\boldsymbol{\lambda})) + \mathrm{tr}\,(S\mathcal{L}\mathbf{w}) + \alpha\sum_i \phi(w_i) + \frac{\beta}{2}\|\mathcal{L}\mathbf{w} - U\mathrm{Diag}(\boldsymbol{\lambda})U^\top\|_F^2 \\ &- \boldsymbol{\mu}_1^\top \mathbf{w} + \boldsymbol{\mu}_2^\top h(\boldsymbol{\lambda}) + \mathrm{tr}\left(M^\top(U^\top U - I_q)\right), \end{aligned} \tag{116}$$

where $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$ and M are KKT multipliers, and $\boldsymbol{\mu}_2^\top h(\boldsymbol{\lambda}) = \mu_{2,1}(\alpha_1 - \lambda_1) + \sum_{i=2}^{q} \mu_{2,i}(\lambda_{i-1} - \lambda_i) + \mu_{2,q+1}(\lambda_q - \alpha_2)$ with $\boldsymbol{\mu}_2 = [\mu_{2,1}, \ldots, \mu_{2,q+1}]^\top$.

(1) we can see $\boldsymbol{\lambda}^\infty$ is derived from KKT conditions of sub-problem (42). Obviously, $\boldsymbol{\lambda}^\infty$ also satisfies KKT conditions of Problem (29).

(2) We will show $\mathbf{w}^\infty$ satisfies KKT conditions (29). The KKT conditions with $\mathbf{w}$ can be derived as:

$$\mathcal{L}^*\mathcal{L}\mathbf{w} - \mathcal{L}^*\left(U\text{Diag}(\boldsymbol{\lambda})U^\top - \beta^{-1}S\right) - \beta^{-1}\boldsymbol{\mu}_1 + \mathbf{b} = 0; \tag{117}$$

$$\boldsymbol{\mu}_1^\top \mathbf{w} = 0; \tag{118}$$

$$\mathbf{w} \geq 0; \tag{119}$$

$$\boldsymbol{\mu}_1 \geq 0; \tag{120}$$

where $\mathbf{b} = \frac{\alpha}{\beta}[1/(\epsilon + w_1^{(t)}),\ \ldots,\ 1/(\epsilon + w_{p(p-1)/2}^{(t)})]^\top$. On the other hand, due to the fact that $\mathbf{w}^\infty$ is derived by KKT system of (38) see Lemma 4, we obtain

$$\mathbf{w}^\infty - \left(\mathbf{w}^\infty - \frac{1}{L_1}\left(\mathcal{L}^*\mathcal{L}\mathbf{w}^\infty - \mathcal{L}^*(U^\infty\text{Diag}(\boldsymbol{\lambda}^\infty)U^{\infty\top} - \beta^{-1}S) + \mathbf{b}^\infty\right)\right) - \boldsymbol{\mu} = 0, \tag{121}$$

where $\boldsymbol{\mu}$ is the KKT multiplier satisfying $\boldsymbol{\mu} \geq \mathbf{0}$ and $\boldsymbol{\mu}\mathbf{w}^\infty = 0$, and $\boldsymbol{b}^\infty = \frac{\alpha}{\beta}[1/(\epsilon + w_1^\infty),\ \ldots,\ 1/(\epsilon + w_{p(p-1)/2}^\infty)]^\top$. Now we could get

$$\mathcal{L}^*\mathcal{L}\mathbf{w}^\infty - \mathcal{L}^*\left(U^\infty\text{Diag}(\boldsymbol{\lambda})(U^\infty)^\top - \beta^{-1}S\right) + \boldsymbol{b}^\infty - L_1\boldsymbol{\mu} = 0, \tag{122}$$

Therefore, by scaling $\boldsymbol{\mu}$ such that $L_1\boldsymbol{\mu} = \beta^{-1}\boldsymbol{\mu}_1$, we can conclude that $\mathbf{w}^\infty$ also satisfies KKT conditions (29).

(3) The KKT conditions with respect to U are as below:

$$\mathcal{L}\mathbf{w}U\text{Diag}(\boldsymbol{\lambda}) - \frac{1}{2}U\left(\text{Diag}(\boldsymbol{\lambda})^2 + \beta^{-1}(M + M^\top)\right) = 0; \tag{123}$$

$$U^\top U = I_q. \tag{124}$$

Since U^∞ admits the first order optimality condition on orthogonal Stiefel manifold, we have

$$\mathcal{L}\mathbf{w}^\infty U^\infty\text{Diag}(\boldsymbol{\lambda}^\infty) - U^\infty\left((U^\infty)^\top\mathcal{L}\mathbf{w}^\infty U^\infty\text{Diag}(\boldsymbol{\lambda}) - \frac{1}{2}[(U^\infty)^\top\mathcal{L}\mathbf{w}^\infty U^\infty, \text{Diag}(\boldsymbol{\lambda}^\infty)]\right) = 0, \tag{125}$$

where $[A, B] = AB - BA$. Note that $(U^\infty)^\top\mathcal{L}\mathbf{w}^\infty U^\infty$ is a diagonal matrix according to the update of U^∞. So there must exist a M such that U^∞ satisfies (123). Therefore, $(\mathbf{w}^\infty, U^\infty, \boldsymbol{\lambda}^\infty)$ satisfies KKT conditions of Problem (29). □

8.4. Proof for Lemma 13

Proof. Note that both $\boldsymbol{\psi}$ and $\mathbf{e}$ are diagonal, and additionally we require the values for $\boldsymbol{\psi}$ to be symmetric across zero, i.e., i.e., $\psi_i = -\psi_{b+1-i}$, $i = 1, \ldots, b/2$. We can express the

least square $\|\boldsymbol{\psi}-\mathbf{e}\|_2^2$ in (63) as

$$\begin{aligned}\|\boldsymbol{\psi}-\mathbf{e}\|_2^2 &= \sum_{i=1}^{b/2}(\psi_i-e_i)^2+(-\psi_i-e_{b-i+1})^2\\ &= 2\sum_{i=1}^{b/2}\left(\psi_i^2-2\psi_i\frac{e_i-e_{b-i+1}}{2}+\frac{e_i^2+e_{b-i+1}^2}{2}\right)\\ &= 2\sum_{i=1}^{b/2}\left(\left(\psi_i-\frac{e_i-e_{b-i+1}}{2}\right)^2+\left(\frac{e_i-e_{b-i+1}}{2}\right)^2+\frac{e_i^2+e_{b-i+1}^2}{2}\right).\end{aligned} \tag{126}$$

Thus, we can write

$$\|\boldsymbol{\psi}-\mathbf{e}\|_2^2 = \text{constant}+\|\tilde{\boldsymbol{\psi}}-\tilde{\mathbf{e}}\|_2^2 \tag{127}$$

where $\tilde{\boldsymbol{\psi}}=[\psi_1,\psi_2,\ldots,\psi_{b/2}]$ are the first half elements of $\boldsymbol{\psi}$, and $\tilde{\mathbf{e}}=[\tilde{e}_1,\tilde{e}_2,\ldots,\tilde{e}_{b/2}]$, with $\{\tilde{e}_i=\frac{e_i-e_{b-i+1}}{2}\}_{i=1}^{b/2}$. □

8.5. Proof for Theorem 7

Proof. We cannot directly apply the established convergence results of BSUM in (Razaviyayn et al., 2013) to prove our algorithm because there are two variables U and V that may not have unique solutions in iterations while the paper (Razaviyayn et al., 2013) only allows one variable to enjoy multiple optimal solutions. However, we can still establish the subsequence convergence as below by following the proof in (Razaviyayn et al., 2013) together with the fact that $\boldsymbol{\lambda}$ and U are updated independently with $\boldsymbol{\psi}$ and V.

For the convenience of description, let $\mathbf{x}=(\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3,\mathbf{x}_4,\mathbf{x}_5)=(\mathbf{w},\boldsymbol{\lambda},U,\boldsymbol{\psi},V)$ with each $\mathbf{x}_i\in\mathcal{X}_i$. Since the iterates $\mathbf{x}^k$ are in a compact set, there must be a limit point for the sequence $\{\mathbf{x}^k\}$. Now we need to show every limit point of the iterates is a stationary point of (65).

Let $\bar{\mathbf{x}}=(\bar{\mathbf{x}}_1,\bar{\mathbf{x}}_2,\bar{\mathbf{x}}_3,\bar{\mathbf{x}}_4,\bar{\mathbf{x}}_5)$ is a limit point of $\{\mathbf{x}^k\}$, and $\{\mathbf{x}^{k_j}\}$ be the subsequence converging to $\bar{\mathbf{x}}$. Without loss of generality, we can assume that

$$\mathbf{x}_1^{k_j}=\arg\min_{\mathbf{x}_1} g(\mathbf{x}_1|\mathbf{x}^{k_j-1}), \tag{128}$$

where $g(\mathbf{x}_1|\mathbf{x}^{k_j-1})$ is the majorized function defined in (68).

For the convenience of proof, we change the updating order along with $(\mathbf{w}, \boldsymbol{\lambda}, U, \boldsymbol{\psi}, V)$ and have the following updating procedure:

$$\mathbf{x}_1^{k_j} = \arg\min_{\mathbf{x}_1} g(\mathbf{x}_1|\mathbf{x}^{k_j-1}), \tag{129}$$

$$\mathbf{x}^{k_j} = [\mathbf{x}_1^{k_j}, \mathbf{x}_2^{k_j-1}, \mathbf{x}_3^{k_j-1}, \mathbf{x}_4^{k_j-1}, \mathbf{x}_5^{k_j-1}]^\top; \tag{130}$$

$$\mathbf{x}_2^{k_j+1} = \arg\min_{\mathbf{x}_2} F_2(\mathbf{x}_2|\mathbf{x}^{k_j}), \tag{131}$$

$$\mathbf{x}^{k_j+1} = [\mathbf{x}_1^{k_j}, \mathbf{x}_2^{k_j+1}, \mathbf{x}_3^{k_j}, \mathbf{x}_4^{k_j}, \mathbf{x}_5^{k_j}]^\top; \tag{132}$$

$$\mathbf{x}_3^{k_j+2} = \arg\min_{\mathbf{x}_3} F_3(\mathbf{x}_3|\mathbf{x}^{k_j+1}), \tag{133}$$

$$\mathbf{x}^{k_j+2} = [\mathbf{x}_1^{k_j+1}, \mathbf{x}_2^{k_j+1}, \mathbf{x}_3^{k_j+2}, \mathbf{x}_4^{k_j+1}, \mathbf{x}_5^{k_j+1}]^\top; \tag{134}$$

$$\mathbf{x}_4^{k_j+3} = \arg\min_{\mathbf{x}_4} F_4(\mathbf{x}_4|\mathbf{x}^{k_j+2}), \tag{135}$$

$$\mathbf{x}^{k_j+3} = [\mathbf{x}_1^{k_j+2}, \mathbf{x}_2^{k_j+2}, \mathbf{x}_3^{k_j+2}, \mathbf{x}_4^{k_j+3}, \mathbf{x}_5^{k_j+2}]^\top; \tag{136}$$

$$\mathbf{x}_5^{k_j+4} = \arg\min_{\mathbf{x}_5} F_5(\mathbf{x}_5|\mathbf{x}^{k_j+3}), \tag{137}$$

$$\mathbf{x}^{k_j+4} = [\mathbf{x}_1^{k_j+3}, \mathbf{x}_2^{k_j+3}, \mathbf{x}_3^{k_j+3}, \mathbf{x}_4^{k_j+3}, \mathbf{x}_5^{k_j+4}]^\top; \tag{138}$$

where $F_i(\mathbf{x}_i|\mathbf{x}^{k_j+i-1})$ is the cost function $F(\mathbf{x})$ in (65) with respect to $\mathbf{x}_i$ and other variables fixed, i.e., $F_i(\mathbf{x}_i|\mathbf{x}^{k_j+i-1}) = F(\mathbf{x}_1^{k_j+i-1}, \ldots, \mathbf{x}_{i-1}^{k_j+i-1}, \mathbf{x}_i, \mathbf{x}_{i+1}^{k_j+i-1}, \mathbf{x}_5^{k_j+i-1})$, $i \in \{2,3,4,5\}$. Note that here we increase the iteration number k when updating each variable, which is different with the notation t in Algorithm 4 where we increase t only after updating all the variables.

We further restrict the subsequence such that

$$\lim_{j\to\infty} \mathbf{x}^{k_j+i} = \mathbf{z}^i, \quad \forall i = -1, 0, 1, \ldots, 4, \tag{139}$$

where $\mathbf{z}^0 = \bar{\mathbf{x}}$.

Since the cost function $F(\mathbf{x})$ in (65) is continuous and nonincreasing, we have

$$F(\mathbf{z}^{-1}) = F(\mathbf{z}^0) = \cdots = F(\mathbf{z}^4). \tag{140}$$

By the majorized function property (24b), we have

$$F(\mathbf{x}^{k_j}) \le g(\mathbf{x}_1^{k_j}|\mathbf{x}^{k_j-1}) \le g(\mathbf{x}_1^{k_j-1}|\mathbf{x}^{k_j-1}) = F(\mathbf{x}^{k_j-1}). \tag{141}$$

By the continuity of $g(\cdot)$ according to (24a), we take the limit $j \to \infty$ and obtain

$$g(\mathbf{z}_1^0|\mathbf{z}^{-1}) = g(\mathbf{z}_1^{-1}|\mathbf{z}^{-1}), \tag{142}$$

Since $\mathbf{z}_1^0$ is the minimizer of $g(\mathbf{x}_1|\mathbf{z}^{-1})$ and $g(\mathbf{x}_1|\mathbf{z}^{-1})$ has the unique minimizer, we have $\mathbf{z}_1^0 = \mathbf{z}_1^{-1}$. We can see only $\mathbf{z}_1$ is updated from $\mathbf{z}^{-1}$ to $\mathbf{z}^0$, so we further obtain $\mathbf{z}^0 = \mathbf{z}^{-1}$.

Regarding $\mathbf{x}_2$, we have

$$F(\mathbf{x}^{k_j+1}) \le F_2(\mathbf{x}_2^{k_j+1}|\mathbf{x}^{k_j}) \le F_2(\mathbf{x}_2^{k_j}|\mathbf{x}^{k_j}) = F(\mathbf{x}^{k_j}). \tag{143}$$

By the continuity of $F_2(\cdot)$, we take the limit $j \to \infty$ and get

$$F_2(\mathbf{z}_2^1|\mathbf{z}^0) = F_2(\mathbf{z}_2^0|\mathbf{z}^0). \tag{144}$$

Considering $F_2(\mathbf{x}_2|\mathbf{z}^0)$ has the unique minimizer, we have $\mathbf{z}_2^1 = \mathbf{z}_2^0$. Similarly, only $\mathbf{z}_2$ is updated from $\mathbf{z}^0$ to $\mathbf{z}^1$ and thus $\mathbf{z}^1 = \mathbf{z}^0$. We can also obtain $\mathbf{z}^3 = \mathbf{z}^2$ by the uniqueness of the minimizer of $F_4(\cdot)$.

Since $\mathbf{x}_2$ and $\mathbf{x}_3$ are updated independently with $\mathbf{x}_4$ and $\mathbf{x}_5$, we have

$$\begin{aligned} F_4(\mathbf{x}_4^{k_j+3}|\mathbf{x}^{k_j+2}) &= F_4(\mathbf{x}_4^{k_j+3}|\mathbf{x}^{k_j}) - (F(\mathbf{x}^{k_j+1}) - F(\mathbf{x}^{k_j+2})) - (F(\mathbf{x}^{k_j}) - F(\mathbf{x}^{k_j+1})) && (145)\\ &\leq F_4(\mathbf{x}_4^{k_j}|\mathbf{x}^{k_j}) - (F(\mathbf{x}^{k_j+1}) - F(\mathbf{x}^{k_j+2})) - (F(\mathbf{x}^{k_j}) - F(\mathbf{x}^{k_j+1})) && (146)\\ &= F(\mathbf{x}^{k_j}) - (F(\mathbf{x}^{k_j+1}) - F(\mathbf{x}^{k_j+2})) - (F(\mathbf{x}^{k_j}) - F(\mathbf{x}^{k_j+1})). && (147) \end{aligned}$$

and

$$F_4(\mathbf{x}_4^{k_j+3}|\mathbf{x}^{k_j+2}) \geq F(\mathbf{x}^{k_j+3}). \tag{148}$$

Then we take the limit $j \to \infty$ and obtain

$$F(\mathbf{z}^3) \leq F_4(\mathbf{z}_4^3|\mathbf{z}^2) = F_4(\mathbf{z}_4^3|\mathbf{z}^0) \leq F_4(\mathbf{z}_4^0|\mathbf{z}^0) = F(\mathbf{z}^0). \tag{149}$$

Together with $F(\mathbf{z}^3) = F(\mathbf{z}^0)$, we have

$$F_4(\mathbf{z}_4^3|\mathbf{z}^0) = F_4(\mathbf{z}_4^0|\mathbf{z}^0). \tag{150}$$

Since $\mathbf{z}_4^3$ is the minimizer of $F_4(\mathbf{x}_4|\mathbf{z}^2)$ as well as $F_4(\mathbf{x}_4|\mathbf{z}^0)$, and the minimizer of $F_4(\mathbf{x}|\mathbf{z}^0)$ is unique, we can get

$$\mathbf{z}_4^3 = \mathbf{z}_4^0. \tag{151}$$

By the fact that $\mathbf{z}_1^0$ is the minimizer of $g(\mathbf{x}_1|\mathbf{z}^{-1})$ and $\bar{\mathbf{x}} = \mathbf{z}^0 = \mathbf{z}^{-1}$, we have

$$g(\bar{\mathbf{x}}_1|\bar{\mathbf{x}}) \leq g(\mathbf{x}_1|\bar{\mathbf{x}}), \quad \forall\ \mathbf{x}_1 \in \mathcal{X}_1. \tag{152}$$

which implies

$$g'(\mathbf{x}_1|\bar{\mathbf{x}})|_{\mathbf{x}_1=\bar{\mathbf{x}}_1} = \mathbf{0}. \tag{153}$$

By the majorized function property (24d) and $F(\cdot)$ is differentiable, we can get

$$F'(\mathbf{x}_1, \bar{\mathbf{x}}_2, \bar{\mathbf{x}}_3, \bar{\mathbf{x}}_4, \bar{\mathbf{x}}_5)|_{\mathbf{x}_1=\bar{\mathbf{x}}_1} = \mathbf{0}. \tag{154}$$

Similarly, we can also obtain

$$F_2(\bar{\mathbf{x}}_2|\bar{\mathbf{x}}) \leq F_2(\mathbf{x}_2|\bar{\mathbf{x}}), \quad \forall\ \mathbf{x}_2 \in \mathcal{X}_2. \tag{155}$$

implying

$$F'(\bar{\mathbf{x}}_1, \mathbf{x}_2, \bar{\mathbf{x}}_3, \bar{\mathbf{x}}_4, \bar{\mathbf{x}}_5)|_{\mathbf{x}_2=\bar{\mathbf{x}}_2} = \mathbf{0}. \tag{156}$$

By the fact that $\mathbf{z}_4^3$ is the minimizer of $F_4(\mathbf{x}_4|\mathbf{z}^0)$, and Eq. (151), we get

$$F_4(\bar{\mathbf{x}}_4|\bar{\mathbf{x}}) \leq F_4(\mathbf{x}_4|\bar{\mathbf{x}}), \quad \forall\ \mathbf{x}_4 \in \mathcal{X}_4. \tag{157}$$

and thus

$$F'(\bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2, \mathbf{x}_3, \bar{\mathbf{x}}_4, \bar{\mathbf{x}}_5)|_{\mathbf{x}_3=\bar{\mathbf{x}}_3} = \mathbf{0}. \tag{158}$$

For $\mathbf{x}_3$, we have

$$F(\mathbf{x}^{k_j+2}) \leq F_3(\mathbf{x}_3^{k_j+2}|\mathbf{x}^{k_j+1}) \leq F_3(\mathbf{x}_3^{k_j+1}|\mathbf{x}^{k_j+1}) = F(\mathbf{x}^{k_j+1}). \tag{159}$$

By the continuity of $F_3(\cdot)$, we take the limit $j \to \infty$ and get

$$F_3(\mathbf{z}_3^2|\mathbf{z}^1) = F_3(\mathbf{z}_3^1|\mathbf{z}^1). \tag{160}$$

Since $\mathbf{z}_3^1$ is the minimizer of $F_3(\mathbf{x}_3|\mathbf{z}^1)$. By $\mathbf{z}^1 = \mathbf{z}^0 = \bar{\mathbf{x}}$, we obtain

$$F_3(\bar{\mathbf{x}}_3|\bar{\mathbf{x}}) \leq F_3(\mathbf{x}_3|\bar{\mathbf{x}}), \quad \forall\ \mathbf{x}_3 \in \mathcal{X}_3. \tag{161}$$

implying

$$F'(\bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2, \bar{\mathbf{x}}_3, \mathbf{x}_4, \bar{\mathbf{x}}_5)|_{\mathbf{x}_4=\bar{\mathbf{x}}_4} = \mathbf{0}. \tag{162}$$

Similarly, we can also have

$$F_5(\bar{\mathbf{x}}_5|\bar{\mathbf{x}}) \leq F_5(\mathbf{x}_5|\bar{\mathbf{x}}), \quad \forall\ \mathbf{x}_5 \in \mathcal{X}_5. \tag{163}$$

and thus

$$F'(\bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2, \bar{\mathbf{x}}_3, \bar{\mathbf{x}}_4, \mathbf{x}_5)|_{\mathbf{x}_5=\bar{\mathbf{x}}_5} = \mathbf{0}. \tag{164}$$

Together with Eq. (154), (156), (158), (162), and (164), we can conclude that $\bar{\mathbf{x}}$ is a stationary point of (65). Next, we only need to prove that each limit of the sequence $\mathbf{x}^k$ satisfies KKT conditions of (65). The proof is very similar to that for Theorem 5 and thus we omit it here. □

References

P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

C. Ambroise, J. Chiquet, C. Matias, et al. Inferring sparse gaussian graphical models with latent structure. *Electronic Journal of Statistics*, 3:205–238, 2009.

A. Anandkumar, V. Y. Tan, F. Huang, and A. S. Willsky. High-dimensional gaussian graphical model selection: Walk summability and local separation criterion. *Journal of Machine Learning Research*, 13(Aug):2293–2337, 2012.

O. Banerjee, L. E. Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *Journal of Machine Learning Research*, 9(Mar):485–516, 2008.

A.-L. Barabási et al. *Network Sciene*. Cambridge University Press, 2016.

R. E. Barlow and H. D. Brunk. The isotonic regression problem and its dual. *Journal of the American Statistical Association*, 67(337):140–147, 1972.

D. J. Bartholomew. Isotonic inference. *Encyclopedia of Statistical Scienes*, 6, 2004.

S. Basu, A. Shojaie, and G. Michailidis. Network granger causality with inherent grouping structure. *Journal of Machine Learning Research*, 16(1):417–453, 2015.

K. Benidis, Y. Sun, P. Babu, and D. P. Palomar. Orthogonal sparse pca and covariance estimation via procrustes reformulation. *IEEE Transactions on Signal Processing*, 64(23): 6211–6226, 2016.

A. Berman and M. Farber. A lower bound for the second largest laplacian eigenvalue of weighted graphs. *Electronic Journal of Linear Algebra*, 22(1):79, 2011.

M. J. Best and N. Chakravarti. Active set algorithms for isotonic regression; a unifying framework. *Mathematical Programming*, 47(1-3):425–439, 1990.

A. Bogdanov, E. Mossel, and S. Vadhan. The complexity of distinguishing markov random fields. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 331–342. Springer, 2008.

T. T. Cai, H. Li, W. Liu, and J. Xie. Joint estimation of multiple high-dimensional precision matrices. *Statistica Sinica*, 26(2):445, 2016.

E. J. Candès, M. B. Wakin, and S. P. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications*, 14(5):877–905, 2008.

V. Chandrasekaran, P. A. Parrilo, and A. S. Willsky. Latent variable graphical model selection via convex optimization. In *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1610–1613. IEEE, 2010.

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

Y.-T. Chow, W. Shi, T. Wu, and W. Yin. Expander graph and communication-efficient decentralized optimization. In *Signals, Systems and Computers, 2016 50th Asilomar Conference on*, pages 1715–1720. IEEE, 2016.

F. R. Chung. *Spectral graph theory*. Number 92. American Mathematical Soc., 1997.

M. Coutino, E. Isufi, T. Maehara, and G. Leus. State-space network topology identification from partial observations. *arXiv preprint arXiv:1906.10471*, 2019.

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 (Statistical Methodology)*, 76(2):373–397, 2014.

K. C. Das and R. Bapat. A sharp upper bound on the largest laplacian eigenvalue of weighted graphs. *Linear Algebra and its Applications*, 409:153–165, 2005.

M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3844–3852, 2016.

A. P. Dempster. Covariance selection. *Biometrics*, pages 157–175, 1972.

D. den Hertog, C. Roos, and T. Terlaky. The linear complimentarity problem, sufficient matrices, and the criss-cross method. *Linear Algebra and its Applications*, 187:1–14, 1993.

D. Dheeru and E. Karra Taniskidou. UCI machine learning repository, 2017. URL `https://archive.ics.uci.edu/ml/datasets/gene+expression+cancer+RNA-Seq#`.

I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 269–274. ACM, 2001.

I. S. Dhillon and J. A. Tropp. Matrix nearness problems with bregman divergences. *SIAM Journal on Matrix Analysis and Applications*, 29(4):1120–1146, 2007.

M. Drton and T. S. Richardson. Graphical methods for efficient likelihood inference in gaussian covariance models. *Journal of Machine Learning Research*, 9(May):893–914, 2008.

J. Duchi, S. Gould, and D. Koller. Projected subgradient methods for learning sparse gaussians. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*, page 153–160, 2008.

H. E. Egilmez, E. Pavez, and A. Ortega. Graph learning from data under laplacian and structural constrints. *IEEE Journal of Selected Topics in Signal Processing*, 11(6):825–841, 2017.

M. Farber and I. Kaminer. Upper bound for the laplacian eigenvalues of a graph. *arXiv preprint arXiv:1106.0769*, 2011.

S. Fattahi and S. Sojoudi. Graphical lasso and thresholding: Equivalence and closed-form solutions. *Journal of Machine Learning Research*, 20(10):1–44, 2019.

M. A. T. Figueiredo and A. K. Jain. Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (3):381–396, 2002.

M. Fiori, P. Musé, and G. Sapiro. Topology constraints in graphical models. In *Advances in Neural Information Processing Systems*, pages 791–799, 2012.

C. Fraley and A. E. Raftery. Bayesian regularization for normal mixture estimation and model-based clustering. *Journal of Classification*, 24(2):155–181, 2007.

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

A. Fu, B. Narasimhan, and S. Boyd. CVXR: An R package for disciplined convex optimization. Technical report, Stanford University, 2017a. URL `https://web.stanford.edu/~boyd/papers/cvxr_paper.html`.

X. Fu, K. Huang, M. Hong, N. D. Sidiropoulos, and A. M.-C. So. Scalable and flexible multiview max-var canonical correlation analysis. *IEEE Transactions on Signal Processing*, 65(16):4150–4165, 2017b.

M. Gavish, B. Nadler, and R. R. Coifman. Multiscale wavelets on trees, graphs and high dimensional data: Theory and applications to semi supervised learning. In *International Conference on Machine Learning*, pages 367–374, 2010.

C. D. Godsil and B. McKay. Constructing cospectral graphs. *Aequationes Mathematicae*, 25(1):257–268, 1982.

M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.1, 2014.

J. Guo, E. Levina, G. Michailidis, and J. Zhu. Joint estimation of multiple graphical models. *Biometrika*, 98(1):1–15, 2011.

B. Hao, W. W. Sun, Y. Liu, and G. Cheng. Simultaneous clustering and estimation of heterogeneous graphical models. *Journal of Machine Learning Research*, 18(217):1–58, 2018.

O. Heinävaara, J. Leppä-Aho, J. Corander, and A. Honkela. On the inconsistency of ℓ_1-penalised sparse precision matrix estimation. *BMC Bioinformatics*, 17(16):448, 2016.

B. Huang and T. Jebara. Maximum likelihood graph structure estimation with degree distributions. In *Analyzing Graphs: Theory and Applications, NIPS Workshop*, volume 14, 2008.

P. Jeziorski and I. Segal. What makes them click: Empirical analysis of consumer demand for search advertising. *American Economic Journal: Microeconomics*, 7(3):24–53, 2015.

V. Kalofolias. How to learn a graph from smooth signals. In *Artificial Intelligence and Statistics*, pages 920–929, 2016.

A. Karatzoglou, A. Smola, K. Hornik, and A. Zeileis. kernlab-an s4 package for kernel methods in r. *Journal of Statistical Software*, 11(9):1–20, 2004.

J. K. Kim and S. Choi. Clustering with r-regular graphs. *Pattern Recognition*, 42(9): 2020–2028, 2009.

I. Koutis, G. L. Miller, and D. Tolliver. Combinatorial preconditioners and multilevel solvers for problems in computer vision and image processing. *Computer Vision and Image Understanding*, 115(12):1638–1646, 2011.

S. Kumar, J. Ying, J. V. d. M. Cardoso, and D. P. Palomar. Structured graph learning via Laplacian spectral constraints. In *Advances in Neural Information Processing Systems*, 2019. URL `arXivpreprintarXiv:1909.11594`.

B. Lake and J. Tenenbaum. Discovering structure by learning sparse graphs. In *Proceedings of the 33rd Annual Cognitive Science Conference.*, 2010.

C. Lam and J. Fan. Sparsistency and rates of convergence in large covariance matrix estimation. *The Annals of statistics*, 37(6B):4254, 2009.

S. L. Lauritzen. *Graphical models*. Oxford Statistical Science Series 17. Oxford University Press, 1996.

Y. LeCun. The mnist database of handwritten digits. 1998. URL `http://yann.lecun.com/exdb/mnist/`.

C.-I. C. Lee. The quadratic loss of isotonic regression under normality. *The Annals of Statistics*, 9(3):686–688, 1981.

W. Lee and Y. Liu. Joint estimation of multiple precision matrices with common structures. *Journal of Machine Learning Research*, 16(1):1035–1062, 2015.

H. Lin, R. Liu, and J. Shu. Some results on the largest and least eigenvalues of graphs. *Electronic Journal of Linear Algebra*, 27(1):259, 2014.

Q. Liu and A. Ihler. Learning scale free networks by reweighted l1 regularization. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 40–48, 2011.

O. E. Livne and A. Brandt. Lean algebraic multigrid (lamg): Fast graph laplacian linear solver. *SIAM Journal on Scientific Computing*, 34(4):B499–B522, 2012.

A. Loukas and P. Vandergheynst. Spectrally approximating large graphs with smaller graphs. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 3237–3246, 2018.

L. Lovász. Eigenvalues of graphs. Technical report, Eotvos Lorand University, 2007.

R. Luss and S. Rosset. Generalized isotonic regression. *Journal of Computational and Graphical Statistics*, 23(1):192–210, 2014.

H. P. Maretic, D. Thanou, and P. Frossard. Graph learning under sparsity priors. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2017*, pages 6523–6527. IEEE, 2017.

B. M. Marlin and K. P. Murphy. Sparse gaussian graphical models with unknown block structure. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 705–712. ACM, 2009.

R. Mazumder and T. Hastie. The graphical lasso: New insights and alternatives. *Electronic Journal of Statistics*, 6:2125, 2012.

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

Z. Meng, B. Eriksson, and A. Hero. Learning latent variable gaussian graphical models. In *International Conference on Machine Learning*, pages 1269–1277, 2014.

K. Mohan, P. London, M. Fazel, D. Witten, and S.-I. Lee. Node-based learning of multiple gaussian graphical models. *Journal of Machine Learning Research*, 15(1):445–488, 2014.

B. Mohar. Some applications of Laplace eigenvalues of graphs. In *Graph Symmetry: Algebraic Methods and Applications.*, pages 225–275. Springer, 1997.

S. K. Narang and A. Ortega. Perfect reconstruction two-channel wavelet filter banks for graph structured data. *IEEE Transactions on Signal Processing*, 60(6):2786–2799, 2012.

A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, pages 849–856, 2002.

F. Nie, X. Wang, M. I. Jordan, and H. Huang. The constrained laplacian rank algorithm for graph-based clustering. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

F. Nie, X. Wang, C. Deng, and H. Huang. Learning a structured optimal bipartite graph for co-clustering. In *Advances in Neural Information Processing Systems*, pages 4132–4141, 2017.

M. Nikolova and M. K. Ng. Analysis of half-quadratic minimization methods for signal and image recovery. *SIAM Journal on Scientific Computing*, 27(3):937–966, 2005.

A. Ortega, P. Frossard, J. Kova**v**cević, J. M. Moura, and P. Vandergheynst. Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE*, 106(5): 808–828, 2018.

D. N. Osherson, J. Stern, O. Wilkie, M. Stob, and E. E. Smith. Default probability. *Cognitive Science*, 15(2):251–269, 1991.

E. Pavez and A. Ortega. Generalized laplacian precision matrix estimation for graph signal processing. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 6350–6354. IEEE, 2016.

E. Pavez, H. E. Egilmez, and A. Ortega. Learning graphs with monotone topology properties and multiple connected components. *IEEE Transactions on Signal Processing*, 66(9): 2399–2413, 2018.

G. A. Pavlopoulos, P. I. Kontou, A. Pavlopoulou, C. Bouyioukos, E. Markou, and P. G. Bagos. Bipartite graphs in systems biology and medicine: a survey of methods and applications. *GigaScience*, 7(4):giy014, 2018.

A. Prabhu, G. Varma, and A. Namboodiri. Deep expander networks: Efficient deep networks from graph theory. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 20–35, 2018.

K. Rajawat and S. Kumar. Stochastic multidimensional scaling. *IEEE Transactions on Signal and Information Processing over Networks*, 3(2):360–375, 2017.

P. Ravikumar, M. J. Wainwright, and J. D. Lafferty. High-dimensional ising model selection using ℓ_1-regularized logistic regression. *The Annals of Statistics*, 38(3):1287–1319, 2010.

M. Razaviyayn, M. Hong, and Z.-Q. Luo. A unified convergence analysis of block successive minimization methods for nonsmooth optimization. *SIAM Journal on Optimization*, 23 (2):1126–1153, 2013.

H. Rue and L. Held. *Gaussian Markov random fields: theory and applications*. CRC press, 2005.

S. Sardellitti, S. Barbarossa, and P. Di Lorenzo. Graph topology inference based on sparsifying transform learning. *IEEE Transactions on Signal Processing*, 67(7):1712–1727, 2019.

S. E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.

U. Schulte. Constructing trees in bipartite graphs. *Discrete Mathematics*, 154(1-3):317–320, 1996.

S. Segarra, A. G. Marques, G. Mateos, and A. Ribeiro. Network topology inference from spectral templates. *IEEE Transactions on Signal and Information Processing over Networks*, 3(3):467–483, 2017.

G. Shen and A. Ortega. Transform-based distributed data gathering. *IEEE Transactions on Signal Processing*, 58(7):3802–3815, 2010.

X. Shen, W. Pan, and Y. Zhu. Likelihood-based selection and sharp parameter estimation. *Journal of the American Statistical Association*, 107(497):223–232, 2012.

A. Shojaie and G. Michailidis. Penalized likelihood methods for estimation of sparse high-dimensional directed acyclic graphs. *Biometrika*, 97(3):519–538, 2010a.

A. Shojaie and G. Michailidis. Penalized principal component regression on graphs for analysis of subnetworks. In *Advances in Neural Information Processing Systems*, pages 2155–2163, 2010b.

D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013.

M. Slawski and M. Hein. Estimation of positive definite m-matrices and structure learning for attractive gaussian markov random fields. *Linear Algebra and its Applications*, 473: 145–179, 2015.

A. J. Smola and R. Kondor. Kernels and regularization on graphs. In *Learning Theory and Kernel Machines*, pages 144–158. Springer, 2003.

J. Song, P. Babu, and D. P. Palomar. Sparse generalized eigenvalue problem via smooth optimization. *IEEE Transactions on Signal Processing*, 63(7):1627–1642, 2015.

D. A. Spielman and S.-H. Teng. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40(4):981–1025, 2011.

S. Sun, H. Wang, and J. Xu. Inferring block structure of graphical models in exponential families. In *Artificial Intelligence and Statistics*, pages 939–947, 2015.

Y. Sun, P. Babu, and D. P. Palomar. Majorization-minimization algorithms in signal processing, communications, and machine learning. *IEEE Transactions on Signal Processing*, 65(3):794–816, Feb. 2016.

M. Sundin, A. Venkitaraman, M. Jansson, and S. Chatterjee. A connectedness constraint for learning sparse graphs. In *Signal Processing Conference (EUSIPCO), 2017 25th European*, pages 151–155. IEEE, 2017.

K. M. Tan, D. Witten, and A. Shojaie. The cluster graphical lasso for improved estimation of gaussian graphical models. *Computational Statistics and Data Analysis*, 85:23–36, 2015.

D. A. Tarzanagh and G. Michailidis. Estimation of graphical models through structured norm minimization. *Journal of Machine Learning Research*, 18(1):7692–7739, 2017.

O. Teke and P. Vaidyanathan. Uncertainty principles and sparse eigenvectors of graphs. *IEEE Transactions on Signal Processing*, 65(20):5406–5420, 2017.

P. Van Mieghem. *Graph spectra for complex networks*. Cambridge University Press, 2010.

B. Wang, A. Pourshafeie, M. Zitnik, J. Zhu, C. D. Bustamante, S. Batzoglou, and J. Leskovec. Network enhancement as a general method to denoise weighted biological networks. *Nature Communications*, 9(1):3108, 2018.

J. Wang. Joint estimation of sparse multivariate regression and conditional graphical models. *Statistica Sinica*, pages 831–851, 2015.

J. N. Weinstein, E. A. Collisson, G. B. Mills, K. R. M. Shaw, B. A. Ozenberger, K. Ellrott, I. Shmulevich, C. Sander, J. M. Stuart, C. G. A. R. Network, et al. The cancer genome atlas pan-cancer analysis project. *Nature Genetics*, 45(10):1113, 2013.

H. Yang, G. Hu, and Y. Hong. Bounds of spectral radii of weighted trees. *Tsinghua Science and Technology*, 8(5):517–520, 2003.

J. Ying, J.-F. Cai, D. Guo, G. Tang, Z. Chen, and X. Qu. Vandermonde factorization of hankel matrix for complex exponential signal recovery — application in fast *nmr* spectroscopy. *IEEE Transactions on Signal Processing*, 66(21):5520–5533, 2018.

A. Yu and M. Lu. Lower bounds on the (laplacian) spectral radius of weighted graphs. *Chinese Annals of Mathematics, Series B*, 35(4):669–678, 2014.

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

H. Zha, X. He, C. Ding, H. Simon, and M. Gu. Bipartite graph partitioning and data clustering. In *Proceedings of the 10th International Conference on Information and knowledge Management*, pages 25–32. ACM, 2001.

L. Zhao, Y. Wang, S. Kumar, and D. P. Palomar. Optimization algorithms for graph laplacian estimation via admm and mm. *IEEE Transactions on Signal Processing*, 67 (16):4231–4244, 2019.

T. Zhao, H. Liu, K. Roeder, J. Lafferty, and L. Wasserman. The huge package for high-dimensional undirected graph estimation in r. *Journal of Machine Learning Research*, 13 (1):1059–1062, 2012.