Sparse-Group Lasso for Graph Learning From Multi-Attribute Data

Jitendra K. Tugnait, Life Fellow, IEEE

Abstract—We consider the problem of inferring the conditional independence graph (CIG) of high-dimensional Gaussian vectors from multi-attribute data. Most existing methods for graph estimation are based on single-attribute models where one associates a scalar random variable with each node. In multi-attribute graphical models, each node represents a random vector. In this paper, we present a sparse-group lasso based penalized log-likelihood approach for graph learning from multi-attribute data. Existing works on multi-attribute graphical modeling have considered only group lasso penalty. The main objective of this paper is to explore the use of sparse-group lasso for multi-attribute graph estimation. An alternating direction method of multipliers (ADMM) algorithm is presented to optimize the objective function to estimate the inverse covariance matrix. Sufficient conditions for consistency and sparsity of the estimator are provided. Numerical results based on synthetic as well as real data are presented.

Index Terms—Graph learning, inverse covariance estimation, undirected graph, sparse-group lasso, multi-attribute data.

I. INTRODUCTION

GRAPHICAL models provide a powerful tool for analyzing multivariate data [1], [2]. A central concept is that of conditional independence. In graphical models, graphs display the conditional independence structure of the variables, and learning the graph structure is equivalent to learning a factorization of the joint probability distribution of these random variables. In an undirected graphical model, the conditional dependency structure among p random variables \(x_1, x_2, \ldots, x_p\), \((x = [x_1 \ x_2 \ \cdots \ x_p]^T)\), is represented using an undirected graph \(G = (V, E)\), where \(V = \{1, 2, \ldots, p\} = \{1, p\}\) is the set of p nodes corresponding to the p random variables \(x_i\), and \(E \subseteq V \times V\) is the set of undirected edges describing conditional dependencies among the components of \(X\). The graph \(G\) then is a conditional independence graph (CIG) where there is no edge between nodes \(i\) and \(j\) (i.e., \(\{i, j\} \notin E\) iff \(x_i\) and \(x_j\) are conditionally independent given the remaining \(p - 2\) variable [2, p. 60]. Graphical models based on gene expression data have been used for data visualization and biological hypothesis generation (i.e., exploratory data analysis) in [3]. Some other applications include classification and exploratory data analysis in intensive care monitoring [4], financial time series [5], [6], social networks [7], gene regulatory networks [8], [9], and analysis of fMRI (functional magnetic resonance imaging) data [10].

Gaussian graphical models (GGMs) are CIGs where \(x\) is multivariate Gaussian. Suppose \(x\) has positive-definite covariance matrix \(\Sigma\) with inverse covariance matrix \(\Omega = \Sigma^{-1}\). Then \(\Omega_{ij}\), the \((i, j)\)-th element of \(\Omega\), is zero iff \(x_i\) and \(x_j\) are conditionally independent. Given \(n\) samples of \(x\), in high-dimensional (data-starved) settings, one estimates \(\Omega\) under some sparsity constraints; see [3], [11], [12], [13], [14]. In these graphs each node represents a scalar random variable; we will call such a graph \(G\) a single-attribute graphical model for \(x\). In many applications, there may be more than one random variable associated with a node. This class of graphical models has been called multi-attribute graphical models in [8], [9], [15] where a focus is on application to biological regulatory networks. In [16], [17] image graphs for grayscale texture images are inferred for modeling dependence of a pixel on neighboring pixels; here one has one variable per pixel node. These approaches do not apply to color images where one has three variables (RGB — red, green, blue — color components) per pixel node. Image graphs for color images is an example of multi-attribute graphical models.

In this paper we consider \(p\) random vectors \(z_i \in \mathbb{R}^m, i = 1, 2, \ldots, p, m \geq 2\). We associate \(z_i\) with the \(i\)-th node of an undirected graph \(\mathcal{G} = (V, \mathcal{E})\) where \(V = \{1, 2, \ldots, p\}\) is the set of \(p\) nodes corresponding to the \(p\) random variables \(z_i\), and \(\mathcal{E} \subseteq V \times V\) is the set of undirected edges describing conditional dependencies among the vectors \(\{z_i, i \in V\}\). As in the scalar case (\(m = 1\)), there is no edge between node \(i\) and node \(j\) in \(\mathcal{G}\) iff random vectors \(z_i\) and \(z_j\) are conditionally independent given the remaining \(p - 2\) vectors.

We now consider some specific cases where use of multi-attribute models is relevant. Gene regulatory networks have been considered in [8], [9], [15], [18], [19]. Antibodies are proteins, and proteins are encoded by genes. Protein profiles and gene profiles have been used in [8], [9], [15], [18], [19] for graph modeling to investigate links between various proteins/gens based on their two profiles using data from the US national cancer institute NCI-60 database for 60 human tumor cell lines. In [15] a network with 91 nodes (\(p = 91\) genes), each with two (\(m = 2\)) attributes comprised of protein and gene profiles, is considered based on \(n = 60\) samples (cell lines). Since these molecular profiles are on the same set of biological samples, it is argued in [15] (also by others) that the multi-attribute graphical model “proposes a consensus version of the interactions at hand in the
cell, and one which is hopefully more robust to noise," compared with single-attribute models based on protein profile alone or gene profile alone. Similar conclusions are reached in [8], [9], [18], [19]. In [16], [17] image graphs for grayscale images are inferred for modeling dependence of a pixel on neighboring pixels; here one has one variable per pixel node. These approaches do not apply to color images where one has three variables (RGB color components) per pixel node. Image graphs for color images is an example of multi-attribute graphical models with a pixel represented by a graph node and three attributes (RGB components) per pixel. Graph-based transform coding and its potential application to signal/image compression is discussed in [20] which is a potential application of image graphs; see also [21], [22]. These contributions are restricted to single attribute models. Connections among different industries in the US are explored in [18] to see if the GDP (gross domestic product) of one industry has some effect on that of other industries. Regional GDP data available from U.S. Department of Commerce website for 8 regions (New England, Mideast, Great Lakes, etc.) and 20 industries (utilities, construction, manufacturing, etc.) are used resulting in a multi-attribute graph with \( p = 20 \) nodes (industries) and \( m = 8 \) attributes (regions). Exploration in [18] takes regions into consideration, since significant differences in relations may exist because of regional characteristics, which are not possible to capture using only national data. In another application in [23], daily time series data from Hong Kong to analyze air pollution of Hong Kong via single attribute graphical models is considered, following the earlier work of [24] based on dependent time series. The time series data of the daily average for four pollutants over three monitoring stations are used in [23] to construct a time series graph of \( p = 12 \) nodes with \( m = 1 \) attribute. If the objective is to study conditional dependencies between various pollutants, then a more appropriate model would be a multi-attribute model with \( p = 4 \) pollutant nodes, each with \( m = 3 \) attributes, reflecting measurements at the three monitoring stations.

A. Related Work

For high-dimensional linear regression problems with grouped covariates, it has been shown in [25], [26] that imposing an additional within group level sparsity constraint can lead to improved classification performance. This is the sparse-group lasso approach. These papers are concerned with algorithm development and do not offer any theoretical analysis, and also do not consider graphical models, single or multi-attribute. Multi-attribute graphical model learning has been considered in [8], [9], [15] and [18]. In [8], [9] a group lasso based penalized log-likelihood approach is investigated. A primal-dual optimization algorithm is given and a theoretical analysis of true graph recovery with high probability is provided following the single-attribute results of [27]. In [15], [18] group lasso based penalized pseudo-likelihood approaches are considered for multi-attribute graph estimation. While [15] offers no theoretical analysis, in [18] sufficient conditions for convergence in the Frobenius norm of the inverse covariance estimator to the true value are presented following the single-attribute results of [28]. Sparse-group lasso penalty has also been used in [3] for joint graphical lasso in the context of multiple classes. As noted in [8], [9], the approach of [3] can be used for multi-attribute graphical model learning. An alternating direction method of multipliers (ADMM) algorithm is presented in [3], but there is no theoretical analysis regarding graph estimator.

B. Our Contributions

In this paper, we present a sparse-group lasso based penalized log-likelihood approach for graph learning from multi-attribute data, whereas [8], [9], [15], [18] consider only group lasso which is a special case of sparse-group lasso. Our penalty is similar, but not identical, to the group graphical lasso penalty in [3]. In [3], correlations between data from different classes are ignored, whereas correlations between data from different attributes are central to our approach. In group lasso, sparsity penalty is imposed on all entries of \( \Omega \) associated with a pair of nodes, as a group. In sparse-group lasso, an additional sparsity penalty is imposed on each off-diagonal \( \Omega_{ij} \). An alternating direction method of multipliers (ADMM) algorithm is presented to optimize the objective function to estimate the inverse covariance matrix. We provide sufficient conditions for convergence in the Frobenius norm of the estimator to the true value, a rate of convergence, and also consider sparsistency; [8], [9] provide conditions only for consistent graph edge recovery, but not for consistent inverse covariance matrix estimation, and [15] offers no theoretical analysis. Related works of [25], [26] dealing with sparse-group lasso do not offer any theoretical analysis (such as our Theorems 1 and 2), and do not consider graphical models.

Our theoretical results follow the single-attribute method of [29] for consistency and the method of [30] for sparsistency, resulting in much simpler, and checkable, sufficient conditions, whereas [8], [9] require an “irrepresentable condition” ([9, condition (12)]) which is hard to verify. The sufficient conditions for convergence given in [18] also require an “irrepresentable (incoherence) condition” ([18, condition (C2)], [28, condition (C2)]) which is hard to verify. We require no such conditions.

We test the proposed approach on synthetic as well as real data. While the ground truth is unknown in the real data applications, requiring domain expert knowledge to interpret the results, the synthetic data examples clearly demonstrate the advantages of using sparse-group lasso instead of just group-lasso or just lasso.

C. Outline and Notation

The rest of the paper is organized as follows. The system model is presented in Sec. II where we describe the multi-attribute graphical model with \( m \) random variables per node, and also an associated larger single-attribute graph. A sparse-group lasso based penalized log-likelihood approach for graph learning from multi-attribute data is proposed in Sec. III. An ADMM algorithm is presented in Sec. III-A to optimize the objective function to estimate the inverse covariance matrix and the edges in the graph. In Sec. IV we analyze consistency (Theorem 1) and sparsistency (Theorem 2) of the proposed approach. Numerical results based on synthetic as well as real data are presented in
Sec. V to illustrate the proposed approach. Proofs of Theorems 1 and 2 are given in Appendices A and B, respectively.

Given $A \in \mathbb{R}^{p \times p}$, we use $\phi_{\min}(A)$, $\phi_{\max}(A)$, $|A|$, $\text{tr}(A)$ and $\text{et}(A)$ to denote the minimum eigenvalue, maximum eigenvalue, determinant, trace, and exponential of trace of $A$, respectively. For a matrix $B \in \mathbb{R}^{p \times p}$, we define the operator norm, the Frobenius norm and the vectorized $\ell_1$ norm, respectively, as $\|B\| = \phi_{\max}(B^T B)$, $\|B\|_F = \text{tr}(B^T B)$ and $\|B\|_1 = \sum_{i,j} |B_{ij}|$ where $B_{ij}$ is the $(i,j)$-th element of $B$. We also denote $B_{ij}$ by $[B_{ij}]$. Given $A \in \mathbb{R}^{p \times p}$, $A^+ = \text{diag}(A)$ is a diagonal matrix with the same diagonal as $A$, and $A^- = A - A^+$ is $A$ with all its diagonal elements set to zero. Symbol $\otimes$ denotes the matrix Kronecker product and $1_n$ is the indicator norm, equaling one if $A$ is true, zero otherwise. For $y_n, x_n \in \mathbb{R}^p$, $y_n \sim x_n$ means that $y_n = \mathcal{O}(x_n)$ and $x_n = \mathcal{O}(y_n)$, where the latter means there exists $0 < M < \infty$ such that $\|x_n\| \leq M \|y_n\| \forall n \geq 1$. The notation $y_n = \mathcal{O}(p, x_n)$ for random vectors $y_n, x_n \in \mathbb{R}^p$ means that for any $\varepsilon > 0$, there exists $0 < M < \infty$ such that $P(\|y_n\| \leq M \|x_n\|) \geq 1 - \varepsilon \forall n \geq 1$.

II. SYSTEM MODEL

Consider $p$ jointly Gaussian random vectors $z_i \in \mathbb{R}^m$, $i = 1, 2, \ldots, p$. We associate $z_i$ with the $i$th node of an undirected graph $G = (V,E)$ where $V = [1, p]$ is the set of nodes, and $E \subseteq V \times V$ is the set of edges that describe the conditional dependencies among vectors $\{z_i, i \in V\}$. As in the scalar case (m = 1), there is no edge between node $i$ and node $j$ in $G$ (i.e., $i, j \notin E$) if random vectors $z_i$ and $z_j$ are conditionally independent given all the remaining random vectors $z_k$ corresponding to the remaining $p-2$ nodes in $V$, i.e., for $\ell \in V \setminus \{j, k\}$ [8], [9]. This is the multi-attribute Gaussian graphical model of interest in this paper. The term multi-attribute Gaussian graphical model has been used in [15] for such models. Define the $mp$-vector $x = [z_1^T, z_2^T, \ldots, z_p^T]^T \in \mathbb{R}^{mp}$.

Suppose we have $n$ i.i.d. observations $x(t), t = 0, 1, \ldots, n-1$, of zero-mean $x$. Our objective is to estimate the inverse covariance matrix $(\mathbb{E}\{xx^T\})^{-1}$ and to determine if edge $i, j \in E$, given data $\{x(t)\}_{t=0}^{n-1}$.

Let us associate $x$ with an “enlarged” graph $\hat{G} = (\hat{V}, \hat{E})$, where $\hat{V} = [1, mp]$ and $\hat{E} \subseteq V \times V$. Note that $\{z_q, q \in \hat{V}\}$, the $\ell$th component of $z_q$, associated with node $j \in \hat{G}$, is the random variable $z_q = [x_q]$, where $q = (j-1)m + \ell$, $j = 1, 2, \ldots, p$ and $\ell = 1, 2, \ldots, m$. The random variable $z_q$ is associated with node $q \in \hat{G} = (\hat{V}, \hat{E})$. Corresponding to the edge $\{j, k\} \in \hat{E}$ in the multi-attribute graph $\hat{G} = (\hat{V}, \hat{E})$, there are $m^2$ edges $\{q, r\} \in \hat{E}$ specified by $q = (j-1)m + s$ and $r = (k-1)m + t$, where $s = 1, 2, \ldots, m$ and $t = 1, 2, \ldots, m$. The graph $\hat{G} = (\hat{V}, \hat{E})$ is a single-attribute graph. In order for $\hat{G}$ to reflect the conditional independencies encoded in $G$, we must have the equivalence

$$\{j, k\} \notin E \Leftrightarrow \hat{E}(jk) \cap \hat{E} = \emptyset,$$

where

$$\hat{E}(jk) = \{q, r \mid q = (j-1)m + s, r = (k-1)m + t, s, t = 1, 2, \ldots, m\}.$$

Let $R_{xx} = \mathbb{E}\{xx^T\} > 0$ and $\Omega = R_{xx}^{-1}$. Define the $(j,k)$th $m \times m$ subblock $\Omega_{jk}$ of $\Omega$ as

$$(\Omega_{jk})_{st} = [\Omega]_{(j-1)m+s,(k-1)m+t}, s, t = 1, 2, \ldots, m. \quad (4)$$

It is established in [9, Sec. 2.1] that

$$\Omega_{jk} = 0 \iff z_j \text{ and } z_k \text{ are conditionally independent} \iff \{j, k\} \notin E. \quad (5)$$

Since $\Omega_{jk} = 0$ is equivalent to $[\Omega]_{qr} = 0$ for every $\{q, r\} \in \hat{E}(jk)$, and since, by [1, Proposition 5.2], $[\Omega]_{qr} = 0$ if $x_q$ and $x_r$ are conditionally independent, hence, if $\{q, r\} \notin \hat{E}$, it follows that equivalence (2) holds true.

III. SPARSE-GROUP GRAPHICAL LASSO

Given $n$ samples $\{x(t)\}_{t=0}^{n-1}$ of zero-mean $x$, define the sample covariance $\hat{\Sigma} = \frac{1}{n} \sum_{t=0}^{n-1} x(t)x(t)^T$. Let $X = [x(0) x(1) \cdots x(n-1)]^T \in \mathbb{R}^{n \times (mp)}$. We have the log-likelihood (up to some constants)

$$\ln f_X (X) = \ln |\Omega| - \text{tr}(\hat{\Sigma} \Omega). \quad (6)$$

To estimate sparse $\Omega$, consider minimization of a penalized version of the negative log-likelihood

$$L(X; \Omega) = -\ln f_X (X) + P(\Omega) \quad (7)$$

using a sparse-group lasso penalty [25], where

$$P(\Omega) = \lambda \Omega^{-1} (1 - \alpha) \sum_{j,k} ||\Omega_{jk}||_F, \quad \lambda > 0 \quad (8)$$

is a penalty (tuning) parameter used to control sparsity, and $0 \leq \alpha \leq 1$ yields a convex combination of lasso and group lasso penalties ($\alpha = 0$ gives the group-lasso fit while $\alpha = 1$ yields the lasso fit). In (8), an $\ell_1$ penalty term is applied to each off-diagonal element of $\Omega$ via $\alpha \lambda \left|\Omega_{jk}\right|_1$ (lasso), and to the off-block-diagonal group of $m^2$ terms in (4)–(5) via $(1 - \alpha)\lambda \sum_{j,k} ||\Omega_{jk}||_F$ (group lasso). The function $L(X; \Omega)$ is strictly convex in $\Omega > 0$.

A. Optimization

We will use the ADMM approach [31] with variable splitting. The method is similar, but not identical, to a method in [3]. Using variable splitting, consider

$$\min_{\Omega, W} \left\{ \text{tr}(\hat{\Sigma}\Omega) - \ln(|\Omega|) + P(W) \right\} \text{ subject to } \Omega = W. \quad (9)$$

The scaled augmented Lagrangian for this problem is [31]

$$L_\rho = \text{tr}(\hat{\Sigma}\Omega) - \ln(|\Omega|) + P(W) + \frac{\rho}{2} \|\Omega - W + U\|^2_F \quad (10)$$

where $U$ is the dual variable, and $\rho > 0$ is the penalty parameter. Given the results $\Omega^{(i)}, W^{(i)}$ of the $i$th iteration, in the $(i+1)$st iteration, an ADMM algorithm executes the following three updates:

a) $\Omega^{(i+1)} \leftarrow \arg \min_{\Omega} L_\rho(\Omega), \quad L_\rho(\Omega) := \text{tr}(\hat{\Sigma}\Omega) - \ln(|\Omega|) + \frac{\rho}{2} \|\Omega - W^{(i)}\|^2_F.$

b) $W^{(i+1)} \leftarrow \arg \min_{W} L_\rho(W), \quad L_\rho(W) := \alpha \lambda \|W - 1\|_1 + (1 - \alpha)\lambda \sum_{j,k} \|W_{jk}\|_F + \frac{\rho}{2} \|\Omega^{(i+1)} - W + U^{(i)}\|^2_F$.
c) \( \mathbf{U}^{(i+1)} \leftarrow \mathbf{U}^{(i)} + (\mathbf{\Omega}^{(i+1)} - \mathbf{W}^{(i+1)}) \)

A necessary and sufficient condition for a global optimum in update (a) is that the gradient of \( L_\alpha(\mathbf{\Sigma}) \) w.r.t. \( \mathbf{\Sigma} \), given by (11), vanishes, with \( \mathbf{\Omega} = \mathbf{\Omega} \rightarrow 0 \):

\[
0 = \left( \frac{\partial L_\alpha(\mathbf{\Sigma})}{\partial \mathbf{\Sigma}} \right) = \mathbf{\Sigma} - \mathbf{\Omega}^{-1} + \rho(\mathbf{\Omega} - \mathbf{W}^{(i)} + \mathbf{U}^{(i)}).
\]

(11)

The solution to (11) follows from [31, Sec. 6.5]. Rewrite (11) as

\[
\mathbf{\Sigma} - \rho(\mathbf{W}^{(i)} - \mathbf{U}^{(i)}) = \mathbf{\Omega}^{-1} - \rho \mathbf{\Omega}.
\]

(12)

Let \( \mathbf{V} \mathbf{D} \mathbf{V}^T \) denote the eigen-decomposition of the symmetric matrix \( \mathbf{\Sigma} - \rho(\mathbf{W}^{(i)} - \mathbf{U}^{(i)}) \) where \( \mathbf{D} \) is diagonal with real values on the diagonal, and \( \mathbf{V} \mathbf{V}^T = \mathbf{V}^T \mathbf{V} = \mathbf{I} \). Then \( \mathbf{\Omega}^{(i+1)} = \mathbf{V} \mathbf{D} \mathbf{V}^T \) where \( \mathbf{D} \) is the diagonal matrix with \( i \)th diagonal element

\[
D_{ii} = -D_{ii} + \sqrt{D_{ii}^2 + 4\rho}.
\]

Since eigenvalues \( D_{ii} > 0 \), \( \mathbf{\Omega}^{(i+1)} > 0 \), and as shown in [31, Sec. 6.5] for a similar problem, it satisfies (11).

Now we turn to update (b), where we note that \( L_\alpha(\mathbf{W}) \) is completely separable w.r.t. each \( m \times m \) subblock \( \mathbf{W}^{(jk)} \) (defined as in (4)). Therefore, we define \( (\mathbf{W}^{(jk)} + 1) - \arg\min \mathbf{W}^{(jk)} \mathbf{W}^{(jk)} \) for subblock indexed by \( (j,k) \), where

\[
J_{jk}(\mathbf{W}^{(jk)}) := \alpha \lambda(1_{j=k}\|\mathbf{W}^{(jk)}\|_1 + 1_{j\neq k}\|\mathbf{W}^{(jk)}\|_1) + 1_{j\neq k}(1-\alpha)\lambda\|\mathbf{W}^{(jk)}\|_F + \rho \left( \mathbf{\Omega}^{(i+1)} - \mathbf{W}^{(i)} \right)^{(jk)}_2
\]

Note that sparse lasso penalty applies only to off-diagonal elements of \( \mathbf{W} \), hence to off-diagonal elements of \( \mathbf{W}^{(jk)} \) and all elements of \( \mathbf{W}^{(jk)} \), \( j \neq k \). The group-lasso penalty applies only to off-diagonal subblocks \( \mathbf{W}^{(jk)} \), \( j \neq k \). Therefore, for \( j = k \), \( 1,2,\cdots,p \) (that is, diagonal sub-blocks), we have

\[
[(\mathbf{W}^{(jk)})^{(i+1)}]_{st} = \begin{cases} 
\left( [\Omega^{(jk)}]^{(i+1)} \right)_{ss} & \text{if } s = t \\
S\left( [\Omega^{(jk)}]^{(i+1)} \right)_{st}, \alpha \lambda \rho & \text{if } s \neq t 
\end{cases}
\]

where

\[
S(\alpha, \beta) := (1 - \beta/|a|)_+, \quad (a)_+: = \max(0,a),
\]
denotes scalar soft thresholding. For \( j \neq k \), following [25, 26], the solution to update (b) is given by

\[
[(\mathbf{W}^{(jk)})^{(i+1)}]_{st} = \begin{cases} 
\left( [\Omega^{(jk)}]^{(i+1)} \right)_{ss} & \text{if } s = t \\
S\left( [\Omega^{(jk)}]^{(i+1)} \right)_{st}, \alpha \lambda / \rho & \text{if } s \neq t 
\end{cases}
\]

where

\[
\mathbf{A} = (\mathbf{\Omega}^{(jk)})^{(i+1)} - (\mathbf{U}^{(jk)})^{(i)} \quad \text{and} \quad \mathbf{S}(\mathbf{A}, \alpha) \quad \text{denotes elementwise matrix soft thresholding, specified by } \mathbf{S}(\mathbf{A}, \alpha)_{st} := S(\mathbf{A}_{st}, \alpha).
\]

Finally, update (c) is

\[
\mathbf{U}^{(i+1)} = \mathbf{U}^{(i)} + (\mathbf{\Omega}^{(i+1)} - \mathbf{W}^{(i+1)})
\]

Using a pseudocode for the ADMM algorithm used in this paper is given in Algorithm 1 where we use the stopping (convergence) criterion following [31, Sec. 3.3.1] and varying penalty parameter \( \rho \) following [31, Sec. 3.4.1]. The stopping criterion is based on primal and dual residuals being small, where, in our case, at \( (i+1) \)st iteration, the primal residual is given by \( \mathbf{\Omega}^{(i+1)} - \mathbf{W}^{(i+1)} \), and the dual residual by \( \rho(\mathbf{W}^{(i+1)} - \mathbf{W}^{(i)}) \). The convergence criterion is met when the norms of these residuals are below primary and dual tolerances \( \tau_{pri} \) and \( \tau_{dual} \), respectively; see line 8 of Algorithm 1. In turn, \( \tau_{pri} \) and \( \tau_{dual} \) are chosen using an absolute and relative criterion as in line 8 of Algorithm 1 where \( \tau_{abs} \) and \( \tau_{rel} \) are user chosen absolute and relative tolerances, respectively. As stated in [31, Sec. 3.4.1], one may use “possibly different penalty parameters \( \rho^{(i)} \) for each iteration, with the goal:

\[
\text{Algorithm 1: ADMM Algorithm for Sparse-Group Graphical Lasso.}
\]

**Input:** Number of samples \( n \), number of nodes \( p \), number of attributes \( m \), data \( \{x(t)\}_{t=0}^{n-1}, x \in \mathbb{R}^{m \times p} \), regularization and penalty parameters \( \lambda, \alpha \) and \( \rho_0 \), tolerances \( \tau_{abs} \) and \( \tau_{rel} \), variable penalty factor \( \mu \), maximum number of iterations \( i_{max} \).

**Output:** estimated inverse covariance \( \mathbf{\Sigma} \) and edge-set \( \hat{E} \).

1. Calculate sample covariance \( \hat{\mathbf{\Sigma}} = \frac{1}{n} \sum_{t=0}^{n-1} x(t)x^T(t) \) (after centering \( x(t) \)).
2. Initialize: \( \mathbf{U}^{(0)} = 0, \mathbf{\Omega}^{(0)} = (\mathbf{\Sigma})^{-1}, \) where \( \mathbf{U}, \mathbf{\Omega} \in \mathbb{R}^{mp \times mp}, \rho(0) = \rho_0 \)
3. converged = false, \( i = 0 \)
4. while converged = false and \( i \leq i_{max} \), do
5. \( \tau_{pri} = \mu \tau_{abs}, \tau_{dual} = \mu \tau_{rel} \)
6. Define soft thresholding scalar operator \( S(a, \beta) := (1 - \beta/|a|)_+, \quad (a)_+ := \max(0,a) \)
7. \( \mathbf{W}^{(i+1)} = \mathbf{U}^{(i+1)} + \mathbf{\Omega}^{(i+1)} - \mathbf{W}^{(i+1)} \)
8. Check convergence. Set tolerances
9. \( \tau_{pri} = \mu \tau_{abs}, \tau_{dual} = \mu \tau_{rel} \)
10. Define penalty parameter \( \rho \)
11. \( \mathbf{U}^{(i+1)} = U^{(i+1)} + (\mathbf{\Omega}^{(i+1)} - \mathbf{W}^{(i+1)} - \mathbf{U}^{(i+1)})
12. end while

13. for \( j \neq k \), if \( \|\mathbf{W}^{(jk)}\|_p > 0 \), assign edge \( \{j,k\} \in \hat{E} \), else \( \{j,k\} \notin \hat{E} \). Inverse covariance estimate \( \mathbf{\Omega} = \mathbf{W} \).
of improving the convergence in practice, as well as making performance less dependent on the initial choice of the penalty parameter.” Line 9 of Algorithm 1 follows typical choices given in [31, Sec. 3.4.1]. For all numerical results presented later, we used \( \rho_0 = 2, \mu = 10, \) and \( \tau_{adv} = \tau_{red} = 10^{-4}. \)

The objective function \( L(\mathbf{X}; \Omega) \), given by (7), is strictly convex, and its domain is the set of strictly positive definite matrices (because of \( -\ln(|\Omega|) \)) and using the log-determinant barrier function [27]). It is also closed, proper and lower semi-continuous. Hence, for any fixed \( \rho > 0 \), the ADMM algorithm is guaranteed to converge [31, Sec. 3.2], in the sense that we have primal residual convergence to 0, dual residual convergence to 0, and objective function convergence to the optimal value. For varying \( \rho \), the convergence of ADMM has not been proven [31, Sec. 3.4.1].

### B. Parameter Tuning, Model Selection and Debiasing

Now we briefly discuss some practical aspects of applying the ADMM algorithm, such as how to select the tuning parameters \( \lambda \) and \( \alpha \). In this, in turn, dictates how many edges are connected in the graph. It is also well-known that lasso and related approaches yield biased estimates of inverse covariance [32], [33]. To debias (approximately), for real data, we will mimic the adaptive lasso approach of [32] to propose an adaptive sparse-group lasso approach, with the following exception. We select \( \lambda_{sm} \) as before, using sparse-group lasso. Then for adaptive sparse-group lasso (comprised of two steps of sparse-group lasso followed by adaptive sparse-group lasso), we set \( \lambda_u = \lambda_{sm}/6 \) and \( \lambda_\ell = \lambda_u/10 \). Notice that we now have the upper limit \( \lambda_u \) one-third of the upper limit for sparse-group lasso. This is based on empirical evidence, and the following observation. Elements of \( \Omega \) estimated as zero in the first step of sparse-group lasso will stay zero in the next adaptive step. Too high a value of \( \lambda \) in sparse-group lasso stage results in higher number of zero elements in \( \Omega \), which in adaptive version will remain as zeros. Reduced \( \lambda \) allows these elements to stay non-zero, thereby allowing adaptive sparse-group lasso to “properly” process such elements. Computation of BIC is done after the second, adaptive step.

### 2) Debiasing: Adaptive Sparse-Group Lasso:

Lasso and related approaches yield biased estimates [32], [33]. To approximately debias, we will mimic the adaptive lasso approach of [32] to propose an adaptive sparse-group lasso approach where we replace \( \hat{P}(\Omega) \) in (8) with \( \tilde{P}(\Omega) \) given below. Given estimates \( \hat{\Omega}_{ij} \) obtained from the proposed non-adaptive sparse-group lasso approach, we define

\[
\hat{P}(\Omega) = \alpha \lambda \sum_{i \neq j=1}^{mp} \frac{\hat{\Omega}_{ij}}{\Omega_{ij}} + (1 - \alpha) \lambda \sum_{i \neq j=1}^{p} \frac{||\hat{\Omega}^{(k)} ||_F}{||\Omega^{(k)} ||_F}.
\]

Thus, we replace \( \alpha \lambda / \hat{\Omega}_{ij} \) and \( (1 - \alpha) \lambda / \Omega^{(k)} \). Now run the ADMM algorithm using adaptive weights (with “obvious” modifications) using the previously selected values of \( \lambda \) and \( \alpha \). Higher \( |\hat{\Omega}_{ij}| \) values decrease the penalty, while lower values increase the penalty. We will illustrate this approach later when exploring both synthetic and real data networks.

Selection of \( \lambda \) and \( \alpha \) is done via BIC, as in Sec. III-B1 for sparse-group lasso, with the following exception. We select \( \lambda_{sm} \) as before, using sparse-group lasso. Then for adaptive sparse-group lasso (comprised of two steps of sparse-group lasso followed by adaptive sparse-group lasso), we set \( \lambda_u = \lambda_{sm}/6 \) and \( \lambda_\ell = \lambda_u/10 \). Notice that we now have the upper limit \( \lambda_u \) one-third of the upper limit for sparse-group lasso. This is based on empirical evidence, and the following observation. Elements of \( \Omega \) estimated as zero in the first step of sparse-group lasso will stay zero in the next adaptive step. Too high a value of \( \lambda \) in sparse-group lasso stage results in higher number of zero elements in \( \Omega \), which in adaptive version will remain as zeros. Reduced \( \lambda \) allows these elements to stay non-zero, thereby allowing adaptive sparse-group lasso to “properly” process such elements. Computation of BIC is done after the second, adaptive step.

### IV. THEORETICAL ANALYSIS

In this section we analyze consistency (Theorem 1) and sparsity (Theorem 2) of the proposed approach. For consistency we follow the method of [29] which deals with single attribute models and lasso penalty in a high-dimensional setting where we allow \( p \) to be a function of sample size \( n \), denoted as \( p_n \). High-dimensional setting allows consideration of the case where number of unknowns \( mp(mp+1)/2 \) in \( \Omega \) is much greater than (or comparable to) the sample size \( n \) [34], and as \( n \) increases, \( p_n \) may increase too maintaining more unknowns than sample size. We also allow \( \lambda \) to be a function of sample size \( n \), denoted as \( \lambda_n \).

Assume

A1) Define the true edge set \( E_0 = \{ (i,j) : \Omega_0^{(ij)} \neq 0, i \neq j \} \) where \( \Omega_0 \) denotes the true inverse covariance of \( x \). Assume that card(\( E_0 \)) = |\( (E_0) \)| \( \leq s_0 \).

A2) The minimum and maximum eigenvalues of \( \Sigma_0 = \Omega_0^{-1} \succ 0 \) satisfy

\[
0 < \beta_{\min} \leq \phi_{\min}(\Sigma_0) \leq \phi_{\max}(\Sigma_0) \leq \beta_{\max} < \infty.
\]

Here \( \beta_{\min} \) and \( \beta_{\max} \) are not functions of \( n \).
Let $\Omega_\lambda = \arg \min_{\Omega_0} L(X; \Omega)$. Theorem 1, proved in Appendix A, establishes consistency of $\Omega_\lambda$.

**Theorem 1. (Consistency):** For $\tau > 2$, let
$$C_0 = 40 \max_k (\Sigma_{\text{obj}{k}})^{1/2} (\tau + \ln(4)/\ln(\text{mp}_n)),$$

(14)

Given real numbers $\delta_1 \in (0, 1)$, $\delta_2 > 0$ and $C_1 > 0$, let $C_2 = \sqrt{m} + 1 + C_1$, and
$$M = \{(1 + \delta_1)^2 (2C_2 + \delta_2) / \delta_2^2\}.$$  

(15)

$$r_n = \sqrt{(\text{mp}_n + m^2 s_{n0}) \ln(\text{mp}_n)} = o(1),$$

(16)

$$N_1 = 2 (\ln(4) + \tau \ln(\text{mp}_n)),$$

(17)

$$N_2 = \arg \min \left\{n : r_n \leq \frac{\delta_1 \beta_{\min}}{(1 + \delta_1)^2 (2C_2 + \delta_2) C_0} \right\}. \tag{18}$$

Suppose the regularization parameter $\lambda_n$ and $\alpha \in [0, 1]$ satisfy
$$\frac{C_1 C_0}{1 + \alpha (m - 1)} \left(\frac{1 + p_n/m s_{n0}}{m s_{n0}}\right) \frac{\ln(\text{mp}_n)}{n} \geq \lambda_n \geq \frac{C_0 \ln(\text{mp}_n)}{n}. \tag{19}$$

Then if the sample size $n > \max\{N_1, N_2\}$ and assumptions (A1)-(A2) hold true, $\Omega_\lambda$ satisfies
$$\|\Omega_\lambda - \Omega_0\|_F \leq M r_n$$

(20)

with probability greater than $1 - (\text{mp}_n)^{-2}$. In terms of rate of convergence, $\|\Omega_\lambda - \Omega_0\|_F = O_P(r_n)$.

**Comments on Theorem 1.**

- In Theorem 1, the number of attributes $m$ are fixed (not a function of sample size $n$) whereas number of nodes $p$ and hence, number of connected edges $s_0$ ($s_{n0}$) of $\Phi_0$ are allowed to be a function of $n$. For Theorem 1 to hold, $\lim_{n \to \infty} r_n = 0$. Clearly the results hold if $p$ is fixed, independent of $n$.

- The bounds on $\lambda_n$ in (19) could restrict maximum value of $\alpha$ for larger $m$ values, if $C_1$ is chosen to be too small. Upperbound of (19) always works for $\alpha = 0$ so long as $C_1 \geq 1$. If we pick $C_1 \geq m$, then the lower bound in (19) is less than the upper bound for every $\alpha \in [0, 1]$.

Sparsistency refers to the property that all parameters that are zero are actually estimated as zero with probability tending to one, as $n \to \infty$ [30]. Theorem 2, stated below and proved in Appendix B, deals with sparsistency of $\Omega_\lambda$. Its proof follows that of [30, Theorem 2] pertaining to lasso and a larger class of penalty functions (including some non-convex functions).

**Theorem 2. (Sparsistency):** Suppose all assumptions and conditions of Theorem 1 hold true so that (20) holds. In addition, suppose that there exists a sequence $\eta_n \to 0$ such that $\|\Omega_\lambda - \Omega_0\| = O_P(\eta_n)$ and $\sqrt{\ln(\text{mp}_n)} / n + \eta_n = O(\lambda_n)$. Then with probability tending to one, $\Omega_{\lambda\text{ikh}} = 0$ for all $\{i, k\} \in \mathcal{E}_0^c$ and $\Omega_{\lambda\text{ikh}} = 0$ for all $\{j, \ell\} \in \mathcal{E}_0^c$.

**Remark 1:** For both consistency and sparsistency to be satisfied, the chosen regularization parameters $\lambda_n$ need to be compatible. Theorem 1 imposes upper and lower bounds on the rate of $\lambda_n$ and Theorem 2 specifies a lower bound. Therefore, for both consistency and sparsistency to be satisfied, we must have
$$\sqrt{\ln(\text{mp}_n)} / n + \eta_n \leq \lambda_n \leq m \left(1 + \frac{p_n}{m s_{n0}}\right) \frac{\ln(\text{mp}_n)}{n}.$$

(21)

Its consequences depend upon $\eta_n$ required to attain $\|\Omega_\lambda - \Omega_0\| = O_P(\eta_n)$. As in [30], we consider two cases, using the inequalities $\|A\|_F / \sqrt{\text{mp}_n} \leq \|A\|_F$ for $A \in \mathbb{R}^{(\text{mp}_n) \times (\text{mp}_n)}$.

- a) Since $\|\Omega_\lambda - \Omega_0\|_F \leq \|\hat{\Omega}_\lambda - \Omega_0\|_F$, in the worst case where the two have the same order, $\|\Omega_\lambda - \Omega_0\| = O_P\left(\sqrt{\ln(\text{mp}_n)} / \sqrt{\text{mp}_n}\right)$ so that $\eta_n = \sqrt{\ln(\text{mp}_n)} / \sqrt{\text{mp}_n}$.

- b) If $\|\Omega_\lambda - \Omega_0\|_F / \sqrt{\text{mp}_n} < \|\Omega_\lambda - \Omega_0\|_F$, the optimistic case where the two have the same order, $\|\Omega_\lambda - \Omega_0\| = O_P\left(\sqrt{1 + \ln(\text{mp}_n)} / \sqrt{\text{mp}_n}\right)$ so that $\eta_n = \sqrt{1 + \ln(\text{mp}_n)} / \sqrt{\text{mp}_n}$.

Then for (21) to hold true, we should have $1 + \sqrt{1 + \ln(\text{mp}_n)} / \sqrt{\text{mp}_n}$ which holds only if $s_{n0} = O(1)$.

V. NUMERICAL EXAMPLES

We now present numerical results for both synthetic and real data to illustrate the proposed approach. In synthetic data examples, the ground truth is known and this allows for assessment of the efficacy of various approaches. In real data examples, where the ground truth is unknown, our goal is visualization and exploration of the dependency structures underlying the data, similar to [3], [9], [16], [17].

Various aspects of single-attribute versus multi-attribute graphical modeling have been well-covered in [8], [9], [15], [18]. So we will focus more on sparse-group lasso versus group lasso comparisons for multi-attribute graph estimation. For our numerical examples, in Algorithm 1, we used $\rho_0 = 2$, $\mu = 10$, $\tau_{abs} = \tau_{rel} = 10^{-4}$, with maximum number of iterations $i_{\max}$ set at 300 and 2000 for synthetic and real data, respectively.

A. Synthetic Data: Chain Graph

We consider a chain graph (an example in [9]) where $p$ nodes are connected in succession. In the upper triangular $\Phi$, using the notation of (4), we set $\Phi_{(jk)}^{(st)} = 0.5^{s-t}$ for $j = k = 1, \cdots, p$, $s, t = 1, \cdots, m$. For $j \neq k$, if the two nodes are not connected, we have $\Phi_{(jk)} = 0$, and if nodes $j$ and $k$ are connected in the chain graph, then $\Phi_{(jk)} = 1$ uniformly distributed over $[0, 1]$. Now add $\gamma I$ to $\Phi$ with $\gamma$ picked to make minimum eigenvalue of $\Phi + \gamma I$ equal to 0.5. This is similar to simulation example 3 in [9, Sec. 5.1]. With $\Phi_{(jk)}^{(st)} = (\Phi + \gamma I)^{-1}$, we generate $x = \Phi w$ with $w \in \mathbb{R}^{mp}$ as Gaussian $w \sim N(0, I)$. We generate $n$ i.i.d. observations from $x$, with $m = 3$, $p \in \{100, 400\}$, $n \in \{100, 200, 400, 800\}$. The true value of edgest cardinality $|\mathcal{E}| = 2(p - 1)$.

Simulation results based on 100 runs are shown in Figs. 1–5 for the following approaches: Lasso (proposed ADMM with
Fig. 1. $F_1$-score (mean ± 1 std) vs sample size $n$, for chain graph with $m = 3$.

Fig. 2. Error norm $||\hat{\Omega}_\lambda - \Omega_0||_F / ||\Omega_0||_F$ (mean ± 1 std) vs sample size $n$, for chain graph with $m = 3$.

Fig. 3. $F_1$-score and Hamming distance (mean ± 1 std) vs $\alpha$, for chain graph with $m = 3$, $p = 100$. $\hat{\lambda}_\alpha$ as for Fig. 1. $(i, j)$ and $(j, i)$ are counted as one edge in computing the Hamming distance.

Fig. 4. True and estimated precision matrices, chain graph, $p = 100$, $m = 3$, $n = 200$. Top left is the ground truth, top right the lasso estimate, bottom left is group lasso estimate and bottom right is the sparse-group lasso estimate. All entries that are exactly zero, are color coded as white.

Fig. 5. Subsets of true and estimated precision matrices of Fig. 4, for ease of viewing.

$\alpha = 1$, labeled “L”), Group Lasso (proposed ADMM with $\alpha = 0$, labeled “GL”), Sparse-Group Lasso (proposed ADMM with $\alpha$ as a parameter, labeled “SGL”), adaptive sparse-group lasso (as detailed in Sec. III-B2, labeled “AdpSGL”), and approaches labeled “BIC-SGL” and “BIC-AdpSGL” which are approaches SGL and AdpSGL for which the tuning parameters $(\lambda, \alpha)$ were selected in each run via BIC as discussed in Sec. III-B. For $p = 100$ and each value of sample size $n$, for approaches L, GL, SGL and AdpSGL, we selected the tuning parameters $(\lambda, \alpha)$ by searching over a two-dimensional grid to maximize the $F_1$-score (averaged over 100 runs), where $F_1$-score is defined as...
and for other values of $F$ the $0$-score is $0.6632$, $0.7988$, $0.8203$ and $0.914$ for L(QUIC), respectively. As seen in Fig. 1, as sample size becomes large (e.g., $n=800$), the $F_1$-score of approach L becomes comparable to GL and SGL, therefore, one may wish to use just lasso for multi-attribute models using fast lasso solvers such as QUIC. We implemented QUIC in MATLAB. The main computational requirement in ADMM is eigen-decomposition in line 5 of the ADMM algorithm whose complexity is $O((mp)^3)$. Therefore, comparing the results for $p=100$ and $p=400$, one would expect to see an increase in timing of the order of approximately 64, but the times displayed in Table I scale by less than 64. While the fast algorithm QUIC is indeed faster than ADMM Algorithm 1 for $p=100$, it is not so for $p=400$. We do note that the timing comparisons between QUIC and ADMM given in [35] (for a different example) are based on QUIC implemented in C++ with a MATLAB interface but ADMM implemented in MATLAB and without using variable penalty $\rho$.

Note that QUIC speeds up the algorithm by processing only a subset of variables (”free variables” [35]) for Newton direction computation at any given iteration, with the remaining variables (”fixed variables”) in the precision matrix left unprocessed. The number of free variables depends upon $\lambda$, with higher $\lambda$ leading to fewer free variables (sparser estimate), and vice versa. Numerical results in [35, Table 2] show that if one selects optimum $\lambda$ (the one that results “in the discovery of the correct number of non-zeros” in the precision matrix [35, p. 2936]), then QUIC converges the fastest. If the chosen $\lambda$ results in denser estimated precision matrix, one has higher number of free variables, and consequently larger time to convergence. Since, using (19), we scaled $\lambda$ values for $p=400$ from $\lambda$ values chosen to optimize performance for $p=100$, they are not necessarily optimal for $p=400$ (because (19) and Theorem 1 hold for large $n$). For optimal $\lambda$’s one expects the timings to scale as $O((mp)^3)$, in particular, by a factor of 64 from $p=100$ to $p=400$. But for non-optimal $\lambda$ values this factor can be much larger, as seen in [35, Table 2] for a different problem. Note that ADMM does not make use of this partition of variables into free and fixed sets, therefore, its timings are not as dependent upon choice of $\lambda$ values. (Another fast gradient-based lasso solver is given in [36] which we have not tested.)

Table II shows the results ($F_1$-score and timings) of using the approaches of [9] and [3] on chain graph with $p=100$ runs for the approaches of Kolar [9] and Danaher et al. [3], implemented in MATLAB table

**Table I**

<table>
<thead>
<tr>
<th>n</th>
<th>L (ADMM)</th>
<th>L (QUIC)</th>
<th>GL</th>
<th>SGL</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0.735 ± 0.043</td>
<td>0.316 ± 0.050</td>
<td>1.16 ± 0.206</td>
<td>1.42 ± 0.233</td>
</tr>
<tr>
<td>200</td>
<td>0.784 ± 0.045</td>
<td>0.340 ± 0.040</td>
<td>1.44 ± 0.205</td>
<td>1.20 ± 0.218</td>
</tr>
<tr>
<td>400</td>
<td>0.925 ± 0.061</td>
<td>0.378 ± 0.029</td>
<td>1.37 ± 0.098</td>
<td>1.31 ± 0.083</td>
</tr>
<tr>
<td>800</td>
<td>0.974 ± 0.027</td>
<td>0.541 ± 0.019</td>
<td>1.38 ± 0.033</td>
<td>1.32 ± 0.029</td>
</tr>
</tbody>
</table>

**Table II**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0.466 ± 0.079</td>
<td>0.079 ± 0.036</td>
</tr>
<tr>
<td>200</td>
<td>0.802 ± 0.056</td>
<td>0.179 ± 0.052</td>
</tr>
<tr>
<td>400</td>
<td>0.941 ± 0.033</td>
<td>0.353 ± 0.065</td>
</tr>
<tr>
<td>800</td>
<td>0.989 ± 0.008</td>
<td>0.332 ± 0.048</td>
</tr>
</tbody>
</table>

Note 1:

$$F_1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}.$$

$$\text{precision} = \frac{|\hat{E} \cap E_0|}{|\hat{E}|}, \quad \text{recall} = \frac{|\hat{E} \cap E_0|}{|E_0|}.$$
solution, and was implemented similar to our Algorithm 1, and we search for \((\lambda, \alpha)\) over a two-dimensional grid to maximize the \(F_1\)-score. Since for connected nodes \(j\) and \(k\) in the chain graph, we pick \(\Omega^{(jk)}\) if \(s = t\), (an example in [9]), and since for a given edge \(\{j, k\}\), [3] exploits \(\Omega^{(jk)}\), only for \(s = t\), and not \(s \neq t\) (unlike ours and Kolar’s approach), one would not expect [3] to do well for this example (as was also noted in [9]). The \(F_1\)-scores of Kolar are quite close to the SGL results (for \(n = 100, 200, 400, 800\), the \(F_1\)-scores are \(0.466, 0.802, 0.941, 0.989\) for Kolar compared to \(0.549, 0.780, 0.939, 0.989\) for GL) but timings are significantly higher (for \(n = 100, 200, 400, 800\), the timings are \(205.0, 46.0, 23.5, 15.6\) for Kolar compared to \(1.163, 1.442, 1.372, 1.375\)s for GL). The approach of [3] yields quite poor \(F_1\) scores, as seen in Table II.

Fig. 3 shows \(F_1\)-score vs \(\alpha\) for \(p = 100\) and \(n = 100\) or 400, with \(\lambda\) values equal to the optimized values for SGL, as used for the results of Fig. 1. For example, we have \((\lambda, \alpha) = (0.3253, 0.05)\) for \(n = 100\) and \((\lambda, \alpha) = (0.18, 0.1)\) for \(n = 400\). Parameter \(\alpha = 0\) leads to group-lasso while \(\alpha = 1\) is purely lasso. For \(n = 100\), the peak empirical \(F_1\)-score is 0.551 at \(\alpha = 0.05\) (compared to 0.525 at \(\alpha = 0\) (group lasso) and 0.115 at \(\alpha = 1\) (lasso)), and the minimum Hamming distance between \(\hat{e}_0\) and \(\hat{e}\) is 72.3 at \(\alpha = 0.15\) (compared to 108.2 at \(\alpha = 0\) and 93.1 at \(\alpha = 1\)). For \(n = 400\), the peak empirical \(F_1\)-score is 0.948 at \(\alpha = 0.10\) (compared to 0.893 at \(\alpha = 0\) and 0.676 at \(\alpha = 1\)), and the minimum Hamming distance is 10.08 at \(\alpha = 0.10\) (compared to 23.5 at \(\alpha = 0\) and 47.7 at \(\alpha = 1\)). Fig. 3 highlights possible advantages of using sparse-group lasso instead of just group-lasso or just lasso: performance could be significantly improved by allowing \(\alpha \neq 0\) or 1; however, the gains may not be as significant if one optimizes w.r.t. \(\lambda\) for each \(\alpha\) separately. While group-lasso enforces multi-attribute graph modeling explicitly, there are some gains to be had by also incorporating sparsity within the groups if such is the case; see also [25], [26].

For the chain graph, \(p = 100, m = 3, n = 200\), we show the true (ground truth) and estimated \((\hat{m}p)\times(\hat{m}p)\) inverse covariance matrices for a single run in Figs. 4 and 5 (the latter is a subset of the former, scaled for ease of viewing), using approaches lasso \((L, \alpha = 1)\), group lasso \((GL, \alpha = 0)\) and sparse group lasso \((SGL, \alpha = 0.05)\), implemented with the corresponding optimized tuning parameters as for Fig. 1. In these figures all matrix entries that are exactly zero, are color coded as white, other matrix elements follow the displayed colorbar coding. Lasso does not impose group penalty, hence, for \(j = k\), estimate of the \(m \times m\) subblock \([\Omega^{(j k)}]_{s t}\), \(s, t = 1, 2, \cdots, m\), yields nonzero diagonals \((s \neq t)\) and several zero off-diagonals, unlike GL and SGL. On the other hand, GL imposes only group penalty which can lead to errors in “full” \(m \times m\) subblocks (notice the “thick” nonzero subblocks off the diagonal in Fig. 5) whereas errors in SGL result in “partial” subblocks since individual elements are also penalized. The \(F_1\)-scores for the given run were 0.690, 0.809 and 0.833 for \(L\), GL and SGL, respectively.

### B. Synthetic Data: Erdős-Rényi Graph

Now we consider an Erdős-Rényi graph where \(p\) nodes are connected to each other with probability \(p_{ex} = 0.05\). In the upper triangular \(\Omega\), using the notation of (4), we set \([\Omega^{(j k)}]_{s t} = 0.5^{s-t}\) for \(j = k = 1, \cdots, p, s, t = 1, \cdots, m\). For \(j \neq k\), if the two nodes are not connected, we have \(\Omega^{(j k)} = 0\), and if nodes \(j\) and \(k\) are connected in the chain graph, then \([\Omega^{(j k)}]_{s t}\) is uniformly distributed over \([-0.4, -0.1]\) \(\cup\) \([0.1, 0.4]\) if \(s \neq t\) and \([\Omega^{(j k)}]_{s t} = 0\) if \(s = t\). Now add \(\gamma I\) to \(\Omega\) with \(\gamma\) picked to make minimum eigenvalue of \(\Omega + \gamma I\) equal to 0.5. With \(\Phi \Phi^T = (\Omega + \gamma I)^{-1}\), we generate \(x = \Phi w\) with \(w \in \mathbb{R}^m\) as Gaussian \(w \sim \mathcal{N}(0, I)\). We generate \(n\) i.i.d. observations from \(x\), with \(m = 3, p \in \{100, 400\}, n \in \{100, 200, 400, 800\}\). We then have \(\frac{1}{n} \mathbb{E}\{\|\hat{e}\|_1\} = 247.5\) and 3990 for \(p = 100\) and 400, respectively.

Simulation results based on 100 runs are shown in Figs. 6, 7 and 8 for \(p = 100\) and 400 nodes. Figs. 6, 7 and 8 are counterparts of Figs. 1, 2 and 3 pertaining to the chain graph. For \(p = 400\), we show results only for SGL in Figs. 6 and 7, similar to Figs. 1 and 2, where we first selected the tuning parameters \((\lambda, \alpha)\) by searching over a two-dimensional grid to maximize the \(F_1\)-score
for \((p, n) = (400, 400)\), resulting in \((\lambda, \alpha) = (0.08, 0.05)\). Then we scale \(\lambda\) for other values of \(n\) based on (19): \(\lambda_n = C_{400} / \sqrt{n}\) with \(C_{400} = 0.08\sqrt{400}\) for \(p = 400\). For \(p = 100\), for each \(n\), the tuning parameters were selected by exhaustive search or via BIC, exactly as for Figs. 1 and 2. We see that the performance or the practical case where the tuning parameters \((\lambda, \alpha)\) have to be selected based on data, the results for BIC-SGL and BIC-AdpSGL show a loss in \(F_1\)-score which is better than SGL which is better than GL, and all the three approaches are superior to L in terms of both the \(F_1\)-score and Frobenius error. For the practical case where the tuning parameters \((\lambda, \alpha)\) have to be selected based on data, the results for BIC-SGL and BIC-AdpSGL show a loss in \(F_1\)-score which significantly narrows with increasing \(n\). For instance, for \(n = 100, 200, 400, 800,\) and \(p = 100\), the \(F_1\) scores are \(0.3360, 0.5603, 0.8483, 0.9810\) for BIC-AdpSGL compared to \(0.4487, 0.7210, 0.9190, 0.9882\) for AdpSGL.

Fig. 8 shows \(F_1\)-score vs \(\alpha\) for \(p = 100\) and \(n = 100\) or 400, with \(\lambda_n\)’s as for Fig. 6, and Fig. 8 is the counterpart of Fig. 3 pertaining to the chain graph. The discussion pertaining to Fig. 3 applies here too.

C. Real Data Example: Gross Domestic Product (GDP) Network

As in [18], we consider regional GDP data obtained from the U.S. Department of Commerce (DoC) website.\(^1\) We picked data for the following 20 different industries with DoC labels: (1) utilities (uti), (2) construction (cons), (3) manufacturing (manu), (4) Durable goods manufacturing (durable), (5) nondurable goods manufacturing (nond), (6) wholesale trade (wholesale), (7) retail trade (retail), (8) transportation and warehousing (trans), (9) information (info), (10) finance and insurance (finance), (11) real estate and rental and leasing (real), (12) professional, scientific and technical services (prof), (13) management of companies and enterprises (manage), (14) administrative and waste management services (admin), (15) educational services (edu), (16) health care and social assistance (health), (17) arts, entertainment and recreation (arts), (18) accommodation and food services (food), (19) other services except government (other), and (20) government (gov). The quarterly data are available from the first quarter of 2005 to the first quarter of 2020, but we used the data from first quarter of 2005 to the first quarter of 2016 since some of the later data are undisclosed (confidential). We used data pertaining to 8 regions in the US: Far West (FWST: Alaska, California, Hawaii, Nevada, Oregon and Washington), Great Lakes (GLAK: Illinois, Indiana, Michigan, Ohio and Wisconsin), Mideast (MEST: Delaware, D.C., Maryland, New Jersey, New York and Pennsylvania), New England (NENG: Connecticut, Maine, Massachusetts, New Hampshire, Rhode Island and Vermont), Plains (PLNS: Iowa, Kansas, Minnesota, Missouri, Nebraska, North Dakota and South Dakota), Rocky Mountain (RKMN: Colorado, Idaho, Montana, Utah and Wyoming), Southeast (SEST: Alabama, Arkansas, Florida, Georgia, Kentucky, Louisiana, Mississippi, North Carolina, South Carolina, Tennessee, Virginia and West Virginia), and Southwest (SWST: Arizona, New Mexico, Oklahoma and Texas).

We model this GDP network as a multi-attribute graphical model with \(p = 20\) nodes (industries) and \(m = 8\) attributes (regions). The objective is visualization and exploration of the dependency structures (conditional dependencies) among these 20 industries. One could merge the region-wise data into nationwide data and then estimate the single-attribute graph for these industries; this would ignore regional differences and more granular information available in the data. Or, one could separately fit graphs for each of the eight regions, and then combine them somehow for a final graph for the entire nation; this ignores any group structure that may be present in the data, and moreover, it is not clear how to combine the various fitted graphs. Multi-attribute graphical modeling preserves/exploits the group structure while maintaining regional differences.

Thus we have a 160-dimensional \((m = 8, p = 20, np = 160)\) single-attribute time series with \(n=48\) samples. We pre-process the data by first detrending it (i.e., remove the best straight-line fit linear trend from each component series using the MATLAB function detrend). Then we normalized the entire dataset to have a mean-square value of one. To estimate the multi-attribute graph, we use the BIC-based method outlined in Sec. III-B to select the tuning parameters \(\lambda\) and \(\alpha\). The selected values turn out be \((\lambda, \alpha) = (1, 0.5)\) for sparse-group lasso and \((\lambda, \alpha) = (0.167, 0.1)\) for adaptive sparse-group lasso. The estimated graph using the sparse-group lasso ADMM algorithm has 46 distinct edges (out of possible 190) and the graph is shown in Fig. 9. For Fig. 9, we take \(\|\hat{\Omega}^{(jk)}\|_F\) as the edge weight for edge \((j, k)\), normalize maximum value to one, and show the resulting graph with colored edge weights and link thickness also reflecting edge weight. With maximum edge weight \(\|\hat{\Omega}^{(jk)}\|_F\) normalized to one, we quantize the interval \([0,1]\) to 4 link thicknesses. The estimated graph using adaptive sparse-group lasso has 54 edges and the graph is shown in Fig. 10. We repeated this procedure (BIC-based tuning parameter selection, for both lasso and adaptive lasso graphs) for single-attribute graphical modeling for each of the eight attributes and

\(^1\)https://www.bea.gov/index.html
nationwide data (pool all regional data). The selected \( \lambda \) values range from 0.04 to 0.075 for lasso, and from 0.013 to 0.018 for adaptive lasso. The resulting adaptive lasso graphs are shown in Fig. 11.

Some edge statistics are tabulated in Table III. Comparing multi-attribute graph of Fig. 10 with the single attribute graphs in Fig. 11, as well as examining the statistics in Table III, we see that multi-attribute graph captures edges not found in single attribute graphs while rejecting some edges in single attribute graphs. It is also interesting to note that there are no edges common to all eight single attribute regional graphs.

The average node degree in Fig. 10 is 5.4. In Fig. 10, there are two groups of most connected industries: the first group is comprised of construction, manufacturing, and finance and insurance, with 16, 17 and 13 edges, respectively, and the second group is comprised of durable goods manufacturing, non-durable goods manufacturing, information, and real estate and rental and leasing, with 8, 5, 6 and 7 edges, respectively. The hubs comprised of construction, manufacturing, and finance and insurance are not surprising as they are typical drivers of GDP. Educational services is connected to durable goods manufacturing, construction and manufacturing in Fig. 10 but is not connected to any node in Fig. 9. Noting that educational services sector includes food and accommodation services to the students, its conditional dependence on durable goods manufacturing, construction and manufacturing in Fig. 10 seems to be plausible given construction and maintenance of student housing and food services, in addition to that of classrooms. In view of the synthetic data results, adaptive sparse-group lasso graph of Fig. 10 would appear to be more accurate than the sparse-group lasso graph of Fig. 9 where educational services node sits in isolation. An in-depth analysis would require domain expertise.

### Table III: Number of Estimated Edges in Regional/Nationwide Single Attribute (SA) GDP Graphs and Multi-Attribute (MA) GDP Graph

<table>
<thead>
<tr>
<th>data</th>
<th>approach non-adaptive</th>
<th>adaptive</th>
<th>common to adaptive MA and adaptive SA</th>
</tr>
</thead>
<tbody>
<tr>
<td>MA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SA: FWST</td>
<td>32</td>
<td>37</td>
<td>13</td>
</tr>
<tr>
<td>SA: GLAK</td>
<td>53</td>
<td>45</td>
<td>11</td>
</tr>
<tr>
<td>SA: MEST</td>
<td>41</td>
<td>45</td>
<td>16</td>
</tr>
<tr>
<td>SA: NENG</td>
<td>51</td>
<td>47</td>
<td>17</td>
</tr>
<tr>
<td>SA: PLNS</td>
<td>30</td>
<td>44</td>
<td>14</td>
</tr>
<tr>
<td>SA: RKMT</td>
<td>51</td>
<td>46</td>
<td>16</td>
</tr>
<tr>
<td>SA: SEST</td>
<td>46</td>
<td>42</td>
<td>9</td>
</tr>
<tr>
<td>SA: SWST</td>
<td>38</td>
<td>43</td>
<td>16</td>
</tr>
<tr>
<td>SA: nationwide</td>
<td>37</td>
<td>29</td>
<td>10</td>
</tr>
</tbody>
</table>

D. Real Data Example: Graphs of Color Texture Images

Following [16], [17] where grayscale texture images from a University of Southern California (USC-SIPI) database are considered, we now consider color textures from the Amsterdam Library of Textures (ALOT)\(^2\) [37]. We use two versions of the image 108 (fake fur), images c111.png and c111r60.png (160 \times 160 patches shown in Fig. 12 where c111 is labeled as image 1 and c111r60 is labeled as image 2), photographed from different angles. These two images are 3072 \times 1536 RGB color pixels (1024 \times 1536 for each of RGB). For image 1 we extracted rows 1 through 160, and columns 186 through 345, and for image 2 we extracted rows 700 through 859, and columns 1 through 160, to create the 160 \times 160 patches used for inferring image graphs.

The 160 \times 160 patches were partitioned into non-overlapping 8 \times 8 blocks, vectorized into 64-pixel columns, 3 colors associated with each pixel. Thus, we have \( m = 3, p = 64 \) and \( n = 400 \). The data were centered and mean-square value normalized to one before processing. To select \( \lambda \) and \( \alpha \), we use the BIC-based method outlined in Sec. III-B. For texture shown in Fig. 12(a),

\(^2\)http://aloi.science.uva.nl/public_alot
the selected values turn out be \((\lambda, \alpha) = (0.0625, 0.25)\) for sparse-group lasso and \((\lambda, \alpha) = (0.0417, 0.1)\) for adaptive sparse-group lasso, and for texture shown in Fig. 12(d), the selected values are \((\lambda, \alpha) = (0.0775, 0.25)\) for sparse-group lasso and \((\lambda, \alpha) = (0.0517, 0.1)\) for adaptive sparse-group lasso. Using these values for graph estimation, we obtain \(\frac{1}{2} |\hat{\mathcal{E}}| = 554\) and \(\frac{1}{2} |\hat{\mathcal{E}}| = 208\) with sparse-group lasso and adaptive sparse-group lasso, respectively, for image 1 with estimated graphs shown in Figs. 12(b) and 12(c), respectively. For image 2, we obtain \(\frac{1}{2} |\hat{\mathcal{E}}| = 467\) and \(\frac{1}{2} |\hat{\mathcal{E}}| = 204\) with sparse-group lasso and adaptive sparse-group lasso, respectively, with estimated graphs shown in Figs. 12(e) and 12(f), respectively. We take \(\|\Omega_{jk}\|_F\) as the edge weight for edge \((j, k)\), normalize maximum value to one, and show the resulting graphs (arranged as \(8 \times 8\) nodes) with colored edge weights and link thickness also reflecting edge weight. (With maximum edge weight \(\|\Omega_{jk}\|_F\) normalized to one, we quantize the interval \([0,1]\) to 4 link thicknesses.) Compare Figs. 12(a), 12(b) and 12(c), and Figs. 12(d), 12(e) and 12(f), respectively, to note that the strong link weights follow the texture orientation: primarily vertical and some slanting left in Figs. 12(a), 12(b) and 12(c), and primarily slanting right and some horizontal in Figs. 12(d), 12(e) and 12(f). Weaker edge weights connect pixels in “other” directions. These observations provide “visual” support for fitted graphs, confirming our intuition; we do not know the ground truth. Observe also that the results obtained via adaptive sparse-group lasso are much sparser than those via sparse-group lasso. This is unlike the GDP graphs in Sec. V-C where the sample size is quite small \((n = 48\).
for \((p, m) = (20, 8)\) for GDP graphs compared to \(n = 400\) for \((p, m) = (64, 3)\) for image graphs.

VI. CONCLUSION

We proposed a sparse-group lasso based penalized log-likelihood approach for graph learning from multi-attribute data. Prior work of [8], [9], [15], [18] considers only group lasso which is a special case of sparse-group lasso. An ADMM algorithm was presented to optimize the objective function to estimate the inverse covariance matrix and the edges in the graph. We provided sufficient conditions for convergence in the Frobenius norm of the estimator to the true value, a rate of convergence, and also considered sparsistency.

We tested the proposed approach on synthetic as well as real data. While the ground truth is unknown in the real data applications, requiring domain expert knowledge to interpret the results (estimated graphs), the synthetic data examples clearly demonstrate the advantages of using sparse-group lasso instead of just group-lasso or just lasso.

It is of interest to perform theoretical analysis of the adaptive sparse-group lasso approach outlined in Sec. III-B2.

APPENDIX A

Proof of Theorem 1

Here we prove Theorem 1. First we need Lemmas 1 and 2. Lemma 1 below is specialization of [27, Lemma 1] to Gaussian random vectors. It follows from [27, Lemma 1] after setting the sub-Gaussian parameter \(\sigma\) in [27, Lemma 1] to 1.

Lemma 1: Consider a zero-mean Gaussian random vector \(z \in \mathbb{R}^p\) with covariance \(R > 0\). Given \(n\) i.i.d. samples \(z(t), t = 1, 2, \cdots, n\), of \(z\), let \(\hat{R} = (1/n) \sum_{t=1}^n zz^\top\) denote the sample covariance matrix. Then \(\hat{R}\) satisfies the tail bound

\[
P \left( \left| |\hat{R} - R|_{ij} \right| > \delta \right) \leq 4 \exp \left( -\frac{n \delta^2}{3200 \max_i (R_{ii})^2} \right) \tag{22}
\]

for all \(\delta \in (0, 40 \max_i (R_{ii}))\).

Now we state Lemma 2 and provide a proof.

Lemma 2: Under Assumption (A2), the sample covariance \(\hat{\Sigma}\) satisfies the tail bound

\[
P \left( \max_{k,l} |\hat{\Sigma} - \Sigma_{0}|_{kl} > \delta \right) \leq C_0 \exp \left( -\frac{n \delta^2}{3200 \max_{k,l} (\Sigma_{0kk})^2} \right) \tag{23}
\]

for \(\tau > 2\), if the sample size \(n > N_1\), where \(C_0\) is defined in (14) and \(N_1\) is defined in (17).

Proof: Applying Lemma 1 to our problem, we have

\[
P \left( \left| |\hat{\Sigma} - \Sigma_{0}|_{kl} \right| > \delta \right) \leq 4 \exp \left( -\frac{n \delta^2}{3200 (\Sigma_{0kk})_{\max}^2} \right) \tag{24}
\]

for all \(\delta \in (0, 40 \max_k (\Sigma_{0kk}))\) where \((\Sigma_{0kk})_{\max} = \max_{1 \leq k \leq m_p} (\Sigma_{0kk}^2)\). Applying the union bound over all \((m_p)^2\) entries of \(\Sigma - \Sigma_{0}\), we have

\[
P \left( \max_{k,l} |\hat{\Sigma} - \Sigma_{0}|_{kl} > \delta \right) \leq P_{tb}
\]

\[
= 4(m_p)^2 \exp \left( -\frac{n \delta^2}{3200 (\Sigma_{0kk})_{\max}^2} \right) \tag{25}
\]

for all \(\delta \in (0, 40 \max_k (\Sigma_{0kk}))\). Let \(c_* := 1/(40 \max_k (\Sigma_{0kk}))\). Suppose \(\delta\) is such that

\[
c_* \delta = \sqrt{\frac{N_1}{n}} \sqrt{\frac{2 \ln(4(m_p)^2)}{n}}
\]

where we have used the expression for \(N_1\) from (17). Then \(c_* \delta \leq 1\) for \(n > N_1\), and therefore, the bound (25) holds true since \(\delta \in (0, 40 \max_k (\Sigma_{0kk})) = (0, c_*^{-1})\). Using the definitions
of $c_s$ and $N_1$, $C_0$ specified in (14) can be expressed as
\[
C_0 = c_s^{-1} \sqrt{\frac{N_1}{\ln(mp_n)}}.
\]
Suppose we choose $\delta$ as
\[
\delta = C_0 \sqrt{\frac{\ln(mp_n)}{n}} \Rightarrow \delta = c_s^{-1} \sqrt{\frac{N_1}{n}},
\]
(26)
then $c_s \delta < 1$ for $n > N_1$, therefore, the bound (25) holds true. Now it remains to show that $P_{tb}$ equals $1/(mp_n)^{\tau - 2}$ for $\delta$ chosen as in (26). We have
\[
P_{tb} = n^2 (mp_n)^{2} \exp \left( \frac{-n^2}{3200 \max_k (\Sigma_{n}^{2})} \right) = (mp_n)^{2} \exp \left( \frac{-n^2}{2nc_s^2} \right)
\]
\[
= (mp_n)^2 \exp \left( -\frac{N_1/2}{\exp (\ln((mp_n)^{\tau - 2}))} \right)
\]
(27)
This proves the desired result. Proof of Theorem 1: Let $\Omega = \Omega_0 + \Delta$ with both $\Omega$, $\Omega_0 > 0$, and
\[
Q(\Omega) := L(X; \Omega) - L(X; \Omega_0).
\]
(28)
The estimate $\hat{\Omega}$, denoted by $\hat{\Omega}$ hereafter suppressing dependence upon $\lambda$, minimizes $Q(\Omega)$, or equivalently, $\Delta = \hat{\Omega} - \Omega_0$ minimizes $G(\Delta) := Q(\hat{\Omega} + \Delta)$. We will follow, for the most part, the method of proof of [29, Theorem 1] pertaining to lasso penalty. Consider the set
\[
\Theta_n(M) := \{ \Delta : \Delta = \Delta^\top, \| \Delta \|_F = M r_n \}
\]
where $M$ and $r_n$ are as in (15) and (16), respectively. Since $G(\Delta) \leq G(0) = 0$, if we can show that $\inf_{\Delta} G(\Delta) : \Delta \in \Theta_n(M) > 0$, then the minimizer $\Delta$ must be inside $\Theta_n(M)$, and hence $\| \Delta \|_F \leq M r_n$. It is shown in [29, (9)] that
\[
\ln | \Omega_0 + \Delta | - \ln | \Omega_0 | = \text{tr}(\Sigma_0 \Delta) - A_1
\]
(30)
with, $H(\Omega_0, \Delta, v) = (\Omega_0 + v \Delta)^{-1} \otimes (\Omega_0 + v \Delta)^{-1}$ and $v$ denoting a scalar,
\[
A_1 = \text{vec}(\Delta)^\top \left( \int_0^1 (1 - v) H(\Omega_0, \Delta, v) \, dv \right) \text{vec}(\Delta).
\]
(31)
Noting that $\Omega^{-1} = \Sigma$ and setting $\tilde{\lambda}_1 = \lambda \alpha_n$ and $\tilde{\lambda}_2 = (1 - \alpha) \lambda_n$, we can rewrite $G(\Delta)$ as
\[
G(\Delta) = A_1 + A_2 + A_3 + A_4,
\]
(32)
where
\[
A_2 = \text{tr} \left( \hat{\Sigma} - \Sigma_0 \Delta \right)
\]
\[
A_3 = \tilde{\lambda}_1 \left( \| \Omega_0^{i} \|_F + \Delta^{\top} |_{1} - \| \Omega_0^{i} \|_F \right),
\]
(34)
\[
A_4 = (1 - \alpha) \lambda_n \sum_{i,j=1:i\neq j}^n \left( \| \Sigma_0^{ij} \|_F + \Delta^{\top} |_{F} - \| \Sigma_0^{ij} \|_F \right).
\]
(35)
Following [29, p. 502], we have
\[
A_1 \geq \frac{|\Delta|_{F}^{2}}{2(\| \Omega_0^{i} \| + |\Delta|_{F}^{2})} \geq \frac{|\Delta|_{F}^{2}}{2(\beta_{\text{min}} + M r_n)^{2}}
\]
(36)
where we have used the fact that $\| \Omega_0^{i} \| = \| \Sigma_0^{i} \| = \phi_{\text{max}}(\Sigma_0^{i}) = (\phi_{\text{min}}(\Sigma_0^{i}))^{-1} \leq \beta_{\text{min}}^{-1}$ and $|\Delta|_{F} \leq |\Delta|_{F}$. We now consider $A_2$ in (33). We have
\[
A_2 = \sum_{i,j=1:i\neq j}^n \left[ |\Delta - \Sigma_0^{ij}|_{ij} \Delta_{ij} + \sum_{i=1}^{mp_n} |\hat{\Sigma} - \Sigma_0^{ij}|_{ii} \Delta_{ii} \right]
\]
(37)
To bound $L_1$, using Lemma 2, with probability $> 1 - 1/(mp_n)^{\tau - 2}$,
\[
|L_1| \leq \| \Delta^{\top} |_{1} \max_{i,j} \| |\Delta - \Sigma_0^{ij}|_{ij} \| \leq \| \Delta^{\top} |_{1} C_0 \sqrt{\ln(mp_n)/n}.
\]
(38)
Similarly, by Cauchy-Schwarz inequality, Lemma 2 and (16),
\[
|L_2| \leq \| \Delta^{\top} |_{1} C_0 \sqrt{\ln(mp_n)/n} \leq C_0 \sqrt{\ln(mp_n)/n} \| \Delta^{\top} |_{F}
\]
(39)
\[
\leq \| \Delta^{\top} |_{F} C_0 r_n.
\]
Therefore, with probability $> 1 - 1/(mp_n)^{\tau - 2}$,
\[
|A_2| \leq \| \Delta^{\top} |_{1} C_0 \sqrt{\ln(mp_n)/n} + \| \Delta^{\top} |_{F} C_0 r_n.
\]
(40)
We now derive a different bound on $A_2$. Define $\tilde{\Delta} \in R^{p \times p}$ with $(i,j)$-th element $\tilde{\Delta}_{ij} = \| \Delta^{(ij)} \|_F$, where $\Delta^{(ij)}$ is defined from $\Delta$ similar to (4). By Cauchy-Schwarz inequality,
\[
\| \Delta^{\top} |_{1} \leq \sum_{i,j=1:i\neq j} \| \Delta^{(ij)} \|_{1} \leq m \| \Delta^{\top} |_{1} + \| \Delta^{\top} |_{F} C_0 r_n.
\]
(41)
Then using $\sum_{k} \| \Delta^{(kk)} \|_{1} \leq m \sum_{k} \| \Delta_{kk} \|_{1} \leq m \sqrt{p_n} \| \Delta^{\top} |_{F}$, we have
\[
|L_2| + C_0 \sqrt{\ln(mp_n)/n} B \leq C_0 \sqrt{\ln(mp_n)/n} \left( \sum_{k=1}^{p_n} \| \Delta^{(kk)} \|_{1} \right)
\]
\[
\leq \| \Delta^{\top} |_{F} \sqrt{\ln(mp_n)/n} + \sqrt{m} \| \tilde{\Delta}^{\top} |_{F} C_0 r_n.
\]
Therefore, an alternative bound is
\[
|A_2| \leq m \| \Delta^{\top} |_{1} C_0 \sqrt{\ln(mp_n)/n} + \sqrt{m} \| \tilde{\Delta}^{\top} |_{F} C_0 r_n.
\]
(42)
We now bound $A_3$ in (34). Considering the true enlarged edge-set $\tilde{E}_0$ corresponding to $\tilde{E}_0$ (see Sec. II for $\tilde{E}_0$), $\tilde{E}_0$ denote its complement. For an index set $B$ and a matrix $C \in R^{p \times p}$, we write $C_B$ to denote a matrix in $R^{p_B \times p_B}$ such that $C_{B_{ij}} = C_{ij}$ if $(i,j) \in B$, and $C_{B_{ij}} = 0$ if $(i,j) \not\in B$. Then, by definition, $\tilde{\Delta} = \Delta^{\tilde{E}_0} + \Delta^{\tilde{E}_0}$, and $|\tilde{\Delta}^{\top} |_{1} = |\Delta^{\tilde{E}_0} |_{1} + |\Delta^{\tilde{E}_0} |_{1}$. We have
\[
A_3 = \tilde{\lambda}_1(\| \Omega_0^{i} + \Delta^{\tilde{E}_0} |_{1} - \| \Omega_0^{i} |_{1})
\]
\[
= \lambda_1(\| \Omega_0^{i} + \Delta^{\tilde{E}_0} |_{1} + | \Delta^{\tilde{E}_0} |_{1} - \| \Omega_0^{i} |_{1})
\]
(43)
where we have used the triangle inequality $\| \Omega_0^{i} + \Delta^{\tilde{E}_0} |_{1} \geq \Omega_0^{i} |_{1} - | \Delta^{\tilde{E}_0} |_{1}$. Next we bound $A_4$ in (35). Considering the true edge-set $E_0$ for the multi-attribute graph, let $E_0$ denote
its complement. If the edge \( \{i, j\} \in \mathcal{E}_0 \), then \( \Omega_{0}^{(ij)} = 0 \), therefore, \( \| \Omega_{0}^{(ij)} + \Delta^{(ij)} \|_F - \| \Omega_{0}^{(ij)} \|_F = \| \Delta^{(ij)} \|_F \). For \( \{i, j\} \in \mathcal{E}_0 \), by the triangle inequality, \( \| \Omega_{0}^{(ij)} + \Delta^{(ij)} \|_F - \| \Omega_{0}^{(ij)} \|_F \geq -\| \Delta^{(ij)} \|_F \).

Thus

\[
A_4 \geq \lambda_2(\| \Delta_{\mathcal{E}_0}^+ \|_1 - \| \Delta_{\mathcal{E}_0}^- \|_1).
\]

(44)

Split \( A_2 \) as \( A_2 = \alpha A_2 + (1 - \alpha) A_2 \), apply bound (40) to \( A_2 \) and (42) to \( 1 - A_2 \), use

\[
\| \Delta^- \|_1 = \| \Delta_{\mathcal{E}_0}^+ \|_1 + \| \Delta_{\mathcal{E}_0}^- \|_1, \quad \| \Delta^- \|_1 = \| \Delta_{\mathcal{E}_0}^+ \|_1 + \| \Delta_{\mathcal{E}_0}^- \|_1,
\]

and \( C_0 = C_0 \sqrt{n\log(m)} \), to yield

\[
A_2 + A_3 + A_4 \geq |A_2| + \lambda_1(\| \Delta_{\mathcal{E}_0}^+ \|_1 - \| \Delta_{\mathcal{E}_0}^- \|_1)
\]

\[
+ \lambda_2(\| \Delta_{\mathcal{E}_0}^+ \|_1 - \| \Delta_{\mathcal{E}_0}^- \|_1)
\]

\[
\geq - (\alpha \| \Delta_\mathcal{E}_0^+ \|_F + (1 - \alpha) \sqrt{\lambda_1 - \lambda_2} + 1)n \|
\]

\[
\geq - (\alpha \| \Delta_\mathcal{E}_0^+ \|_F + 1 - \alpha) \lambda_2 n \|
\]

\[
(\alpha \| \Delta_\mathcal{E}_0^+ \|_F + 1 - \alpha) \lambda_2 n \|
\]

(45)

where, for the last inequality above, we used the fact that for \( \lambda_n \) as in (19), \( \lambda_1 - \alpha \lambda_2 \geq 0 \) and \( \lambda_2 - (1 - \alpha) \lambda_0 \geq 0 \), and \( \| \Delta_\mathcal{E}_0^+ \|_F \geq \| \Delta_\mathcal{E}_0^+ \|_F \) by Cauchy-Schwarz

\[
\| \Delta_{\mathcal{E}_0}^+ \|_1 \leq \sqrt{m \lambda_{\mathcal{E}_0}} \| \Delta_{\mathcal{E}_0}^- \|_F \leq \lambda_{\mathcal{E}_0} \| \Delta_{\mathcal{E}_0}^+ \|_F .
\]

(46)

\[
\| \Delta_{\mathcal{E}_0}^+ \|_1 \leq \sqrt{m \lambda_{\mathcal{E}_0}} \| \Delta_{\mathcal{E}_0}^- \|_F \leq \lambda_{\mathcal{E}_0} \| \Delta_{\mathcal{E}_0}^+ \|_F \leq \| \Delta_{\mathcal{E}_0}^+ \|_F
\]

(47)

Using (45)–(47) and \( \alpha_m := (\alpha + (1 - \alpha) \sqrt{m}) \), we have

\[
A_2 + A_3 + A_4 \geq -[C_0^{1 - (1 - \alpha + \lambda_2/m) (1 + \alpha_m \sqrt{1 + p/n(\alpha - 1)/m \sigma_{\mathcal{E}_0})} + \lambda_1
\]

\[
+ \frac{(\lambda_2/m) \sqrt{m \lambda_{\mathcal{E}_0}} \| \Delta_{\mathcal{E}_0}^+ \|_F}{\lambda_{\mathcal{E}_0} \| \Delta_{\mathcal{E}_0}^+ \|_F}
\]

\[
\geq -C_2 C_0 r_n \| \Delta_{\mathcal{E}_0}^+ \|_F
\]

(48)

where in the last inequality above, we used the fact that for \( \lambda_n \) as in (19), \( m \sqrt{\lambda_{\mathcal{E}_0}} (\lambda_1 + \alpha_m \sqrt{1 + p/n(\alpha - 1)/m \sigma_{\mathcal{E}_0})} \geq C_1 (\mathcal{C}_{\mathcal{E}_0} r_n \), and \( \alpha_m \leq \sqrt{m} \). Using (32), the bound (36) on \( A_1 \) and (48) on \( A_2 + A_3 + A_4 \), and \( \| \Delta_{\mathcal{E}_0}^+ \|_F = M_{r_n} \), we have with \( \alpha > 1 - 1/(\alpha m) < 1 \)

\[
G(\Delta) \geq \| \Delta_{\mathcal{E}_0}^+ \|_F^2 \left[ \frac{1}{2(\beta^{1/2}_m + \frac{M_{r_n} \beta^2}{2})} - C_2 C_0 \right].
\]

(49)

For \( \alpha > N_2 \), if we pick \( M \) as specified in (15), we obtain \( M_{r_n} \leq M_{r_N} < \delta_{\beta/m} \). Then

\[
\frac{1}{2(\beta^{1/2}_m + \frac{M_{r_n} \beta^2}{2})} \geq \frac{4C^2 + \beta^2}{2(1 + \beta^2)} > C_2 C_0 \frac{M_{r_n}}{M}
\]

implying \( G(\Delta) > 0 \). This proves the desired result.

\[ \]
where we have used the fact that \( \| \Omega_\lambda - \Omega_0 \| = \mathcal{O}(\eta_n) \), 
\( \phi_{\min} (\Omega_\lambda - \Omega_0, \phi_{\min}(A + B) \geq \phi_{\min}(A) + \phi_{\min}(B) \). Hence, 
\[
\max_{i,k} |I_4| = \mathcal{O}(\sqrt{\frac{\ln(mp_n)}{n} + \eta_n}).
\] (54)

It then follows that
\[
|\hat{\Sigma}_{ik} - \Sigma \lambda_{ik}| \leq |I_3| + |I_4| = \mathcal{O}(\sqrt{\frac{\ln(mp_n)}{n} + \eta_n}).
\] (55)

Suppose \( \mathcal{O}(\lambda_n) = \sqrt{\frac{\ln(mp_n)}{n} + \eta_n} \). Then \( \alpha \lambda_n(\hat{\Omega}_{\lambda_{ik}}/|\hat{\Omega}_{\lambda_{ik}}|) \) dominates \( |\hat{\Sigma}_{ik} - \Sigma \lambda_{ik}| \) with probability tending to one. This completes the proof.

REFERENCES