On High-Dimensional Graph Learning Under Total Positivity

Jitendra K. Tugnait
Dept. of Electrical & Computer Eng.
Auburn University, Auburn, AL 36849, USA

2021 55th Asilomar Conference on Signals, Systems, and Computers | 978-1-6654-5828-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/IEEECONF53345.2021.9723191

***Abstract*—We consider the problem of estimating the structure of an undirected weighted sparse graphical model of multivariate data under the assumption that the underlying distribution is multivariate totally positive of order 2, or equivalently, all partial correlations are non-negative. Total positivity holds in several applications. The problem of Gaussian graphical model learning has been widely studied without the total positivity assumption where the problem can be formulated as estimation of the sparse precision matrix that encodes conditional dependence between random variables associated with the graph nodes. An approach that imposes total positivity is to assume that the precision matrix obeys the Laplacian constraints which include constraining the off-diagonal elements of the precision matrix to be non-positive. In this paper we investigate modifications to widely used penalized log-likelihood approaches to enforce total positivity but not the Laplacian structure. An alternating direction method of multipliers (ADMM) algorithm is presented for constrained optimization under total positivity and lasso as well as adaptive lasso penalties. Numerical results based on synthetic data show that the proposed constrained adaptive lasso approach significantly outperforms existing Laplacian-based approaches, both statistical and smoothness-based non-statistical.**

I. Introduction

An undirected simple weighted graph is denoted $\mathcal{G} = (V, \mathcal{E}, \boldsymbol{W})$ where $V = \{1, 2, \cdots, p\} = [p]$ is the set of p nodes, $\mathcal{E} \subseteq [p] \times [p]$ is the set of undirected edges, and $\boldsymbol{W} = \boldsymbol{W}^\top \in \mathbb{R}^{p\times p}$ stores the non-negative weights $W_{ij} \geq 0$ associated with the undirected edges. If there is an edge between nodes i and j, then edge $\{i,j\} \in \mathcal{E}$ and $W_{ij} > 0$. If there is no edge between nodes i and j, then edge $\{i,j\} \notin \mathcal{E}$ and $W_{ij} = 0$. In a simple graph there are no self-loops or multiple edges, so $\mathcal{E}$ consists of distinct pairs $\{i,j\}$, $i \neq j$ and $W_{ii} = 0$. In an undirected graph, if $\{i,j\} \in \mathcal{E}$, then $\{j,i\} \in \mathcal{E}$. In graphical models of data variables $x_1, x_1, \cdots, x_p$, ($\boldsymbol{x} = [x_1 \ x_2 \ \cdots \ x_p]^\top$), a weighted graph $\mathcal{G} = (V, \mathcal{E}, \boldsymbol{W})$ (or unweighted $\mathcal{G} = (V, \mathcal{E})$) with $|V| = p$ is used to capture relationships between the p variables x_is [1]–[3]. If $\{i,j\} \in \mathcal{E}$, then x_i and x_j are related (similar or dependent) in some sense, with higher W_{ij} indicating stronger similarity or dependence.

Graphical models provide a powerful tool for analyzing multivariate data [1]–[3]. In a statistical graphical model, the conditional statistical dependency structure among p random variables $x_1, x_1, \cdots, x_p$, is represented using an undirected graph $\mathcal{G} = (V, \mathcal{E})$. The graph $\mathcal{G}$ then is a conditional independence graph (CIG) where there is no edge between nodes i and j (i.e., $\{i,j\} \notin \mathcal{E}$) iff x_i and x_j are conditionally independent given the remaining p-2 variables x_ℓ, $\ell \in [p]$, $\ell \neq i$, $\ell \neq j$. In particular, Gaussian graphical models (GGMs) are CIGs where $\boldsymbol{x}$ is multivariate Gaussian. Suppose $\boldsymbol{x}$ has positive-definite covariance matrix $\boldsymbol{\Sigma}$ with inverse covariance matrix (also called precision matrix) $\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1}$. Then Ω_{ij}, the (i,j)-th element of $\boldsymbol{\Omega}$, is zero iff x_i and x_j are conditionally independent. Such models for $\boldsymbol{x}$ have been extensively studied. Given n samples of $\boldsymbol{x}$, in *high-dimensional settings* where $p \gg 1$ and/or n is of the order of p, one estimates $\boldsymbol{\Omega}$ under some sparsity constraints; see [4]–[8].

More recently, several authors have considered Gaussian graphical models under the constraint that the distribution is multivariate totally positive of order 2 (MTP_2), or equivalently, that all partial correlations are non-negative (see [9], [10] and references therein). Such models are also known as attractive Gaussian random fields [11]. Note that a Gaussian distribution is MTP_2 if and only if its precision matrix $\boldsymbol{\Omega}$ is an M-matrix, i.e., $\Omega_{ij} \leq 0$ for all $i \neq j$ [12]. As discussed in [9], MTP_2 is a strong form of positive dependence, which is relevant for modeling in various applications including phylogenetics or portfolio selection, where the shared ancestry or latent global market variable often lead to positive dependence among the observed variables.

On the other hand, graphical models for data variables have been inferred from consideration other than statistical, depending upon the intended application, nature of data and available prior information [1]. One class of graphical models are based on signal smoothness [1], [13]–[15]. Suppose we are given n samples $\{\boldsymbol{x}(t)\}_{t=1}^n$ of the p data variables $x_1, x_1, \cdots, x_p$, with $\boldsymbol{x}(t) = [x_1(t)\ x_2(t)\ \cdots\ x_p(t)]^\top$. Define the $p \times n$ matrix

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}(1) & \boldsymbol{x}(2) & \cdots & \boldsymbol{x}(n) \end{bmatrix}. \tag{1}$$

A measure of smoothness of signal $\boldsymbol{x}(t)$ under which the signal takes "similar" values at "neighboring" vertices of a given weighted undirected graph, is the function [1], [13], [14]

$$\frac{1}{2}\sum_{i,j=1}^{p} W_{ij}\|\boldsymbol{X}_{i\cdot} - \boldsymbol{X}_{j\cdot}\|_2^2 = \mathrm{tr}(\boldsymbol{X}^\top \boldsymbol{L}\boldsymbol{X}) \tag{2}$$

where $\boldsymbol{X}_{i\cdot}$ denotes the ith row of $\boldsymbol{X}$, $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$ is the (combinatorial) graph Laplacian (matrix), and $\boldsymbol{D}$ is the diagonal weighted degree matrix with $D_{ii} = \sum_{j=1}^p W_{ij}$.

This work was supported by NSF Grants CCF-1617610 and ECCS-2040536. Author's email: tugnajk@auburn.edu

Graph learning from data $\boldsymbol{X}$ then becomes equivalent to estimation of the graph Laplacian matrix $\boldsymbol{L}$ [1], [13].

Another set of approaches are based on statistical considerations under the graph Laplacian constraint [1], [16]–[18] where Laplacian $\boldsymbol{L}$, after regularization, plays the role of inverse covariance $\boldsymbol{\Omega}$; $\boldsymbol{L}$ is a symmetric, non-negative-definite matrix but with non-positive off-diagonal entries. Thus, under Gaussian distribution we have an MTP_2 model.

Graph Laplacian matrix has been extensively used for embedding, manifold learning, clustering and semi-supervised learning [19]–[24].

In this paper we investigate modifications to widely used penalized log-likelihood approaches to enforce total positivity but not the Laplacian structure. An alternating direction method of multipliers (ADMM) algorithm is presented for constrained optimization under total positivity and lasso as well as adaptive lasso penalties. Numerical results based on synthetic data show that the proposed constrained adaptive lasso approach significantly outperforms existing Laplacian-based approaches, both statistical [17] and smoothness-based non-statistical [13].

Notation: Given $\boldsymbol{A} \in \mathbb{R}^{p\times p}$, $\mathrm{tr}(\boldsymbol{A})$ denotes its trace. For $\boldsymbol{B} \in \mathbb{R}^{p\times q}$, we define its Frobenius norm and the vectorized ℓ_1 norm, respectively, as $\|\boldsymbol{B}\|_F = \sqrt{\mathrm{tr}(\boldsymbol{B}^\top\boldsymbol{B})}$ and $\|\boldsymbol{B}\|_1 = \sum_{i,j}|B_{ij}|$ where B_{ij} is the (i,j)-th element of $\boldsymbol{B}$. We also denote B_{ij} by $[\boldsymbol{B}]_{ij}$. Given $\boldsymbol{A} \in \mathbb{R}^{p\times p}$, $\boldsymbol{A}^+ = \mathrm{diag}(\boldsymbol{A})$ is a diagonal matrix with the same diagonal as $\boldsymbol{A}$, $\boldsymbol{A}^- = \boldsymbol{A} - \boldsymbol{A}^+$ is $\boldsymbol{A}$ with all its diagonal elements set to zero, and $\boldsymbol{A} \succ \boldsymbol{0}$ denotes that $\boldsymbol{A}$ is positive-definite.

II. Some Existing Approaches

A. Smoothness-Based Graph Learning [13]

With reference to (1) and (2), it follows (see [13]) that $\mathrm{tr}(\boldsymbol{X}^\top\boldsymbol{L}\boldsymbol{X}) = \frac{1}{2}\mathrm{tr}(\boldsymbol{W}\hat{\boldsymbol{Z}})$ where $\boldsymbol{W}, \hat{\boldsymbol{Z}} \in \mathbb{R}^{p\times p}$, $\hat{Z}_{ij} = \|\boldsymbol{X}_{i\cdot} - \boldsymbol{X}_{j\cdot}\|_2^2$ and $\boldsymbol{W}$ is the weight matrix (or the weighted adjacency matrix) with $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$, $\boldsymbol{W} = \boldsymbol{W}^\top$, $W_{ij} \geq 0$ and $W_{ii} = 0$ for $1 \leq i, j \leq p$. Instead of performing a penalized minimization of $\mathrm{tr}(\boldsymbol{X}^\top\boldsymbol{L}\boldsymbol{X})$ to estimate $\boldsymbol{L}$, [13] minimizes a penalized $\mathrm{tr}(\boldsymbol{W}\hat{\boldsymbol{Z}})$ w.r.t. $\boldsymbol{W}$ for graph learning. Given $\boldsymbol{W}$, one has unique $\boldsymbol{L}$ and the edge-set $\mathcal{E}$. *In the rest of the paper, we will scale $\hat{\boldsymbol{Z}}$ as $\hat{\boldsymbol{Z}}/n$ and denote the latter as $\hat{\boldsymbol{Z}}$.*

Define the space $\mathcal{W}_p$ of all valid $p \times p$ weight matrices $\boldsymbol{W}$

$$\mathcal{W}_p = \Big\{\boldsymbol{W} \in \mathbb{R}^{p\times p} \,:\, \boldsymbol{W} = \boldsymbol{W}^\top,\ W_{ij} \geq 0,\ W_{ii} = 0\Big\} \quad (3)$$

In [13] one looks for $\min_{\boldsymbol{W}\in\mathcal{W}_p} f_s(\boldsymbol{W})$ where

$$f_s(\boldsymbol{W}) = \mathrm{tr}(\boldsymbol{W}\hat{\boldsymbol{Z}}) + \frac{\beta}{2}\|\boldsymbol{W}\|_F^2 - \alpha\sum_{i=1}^{p}\ln\Big(\sum_{j=1}^{p}W_{ij}\Big) \quad (4)$$

with parameters $\alpha > 0$ and $\beta \geq 0$ controlling the "shape.". In (4), $\mathrm{tr}(\boldsymbol{W}\hat{\boldsymbol{Z}})$ is the main cost but minimizing it alone w.r.t. $\boldsymbol{W}$ is ill-posed ($\boldsymbol{W} = \boldsymbol{0}$ minimizes it). Using only the logarithmic barrier ($\beta = 0$) leads to very sparse graphs, and changing α only changes the scale of the solution. The term $\frac{\beta}{2}\|\boldsymbol{W}\|_F^2$ controls graph sparsity. Note that the model of [13] unifies all prior smoothness-based models for estimation of Laplacian $\boldsymbol{L}$ [15], [25]–[27]. A forward-backward algorithm based on [28] is given in [13] to optimize (4), where optimization is carried for fixed $\alpha = 1$ and then one scales $\boldsymbol{W}$ to obtain a desired $\|\boldsymbol{W}\|$; a MATLAB implementation is in [29].

B. Graphical Lasso: Penalized Log-Likelihood [7]

With $\hat{\boldsymbol{S}}$ denoting the sample covariance (assume zero-mean: $\hat{\boldsymbol{S}} = \frac{1}{n}\sum_{t=1}^{n}\boldsymbol{x}(t)\boldsymbol{x}^\top(t)$), seek $\boldsymbol{\Omega}$ to yield $\min_{\boldsymbol{\Omega}\succ\boldsymbol{0}} f_L(\boldsymbol{\Omega})$ where

$$f_L(\boldsymbol{\Omega}) = \mathrm{tr}(\boldsymbol{\Omega}\hat{\boldsymbol{S}}) - \ln(|\boldsymbol{\Omega}|) + \lambda\|\boldsymbol{\Omega}^-\|_1\,, \quad (5)$$

$\lambda\|\boldsymbol{\Omega}^-\|_1$ is the lasso penalty and $\lambda > 0$. Unlike Laplacian $\boldsymbol{L}$, off-diagonal entries of $\boldsymbol{\Omega}$ may not be non-positive.

C. Generalized Graph Laplacian Estimation [17]

In [1], [16]–[18] approaches that make $\boldsymbol{\Omega} = \boldsymbol{L}$ (Laplacian, or some regularized version) in (9) have been considered. In particular, [17] considers

$$\min_{\boldsymbol{\Theta}\succ\boldsymbol{0}} \mathrm{tr}(\boldsymbol{\Theta}\hat{\boldsymbol{K}}) - \ln(|\boldsymbol{\Theta}|) \text{ where } \hat{\boldsymbol{K}} = \hat{\boldsymbol{S}} + \lambda(\boldsymbol{I} - \boldsymbol{1}_p\boldsymbol{1}_p^\top) \quad (6)$$

with $\boldsymbol{\Theta}$ restricted to be a generalized graph Laplacian matrix. Software implementation of this algorithm is available in [30].

D. Adaptive Lasso [31]

With $\hat{\boldsymbol{\Omega}} = \arg\min_{\boldsymbol{\Omega}\succ\boldsymbol{0}} f_L(\boldsymbol{\Omega})$ from Sec. II-B, modify (5) as

$$\min_{\boldsymbol{\Omega}\succ\boldsymbol{0}} \mathrm{tr}(\boldsymbol{\Omega}\hat{\boldsymbol{S}}) - \ln(|\Omega|) + \lambda\sum_{i,j=1,\,i\neq j}^{p}\frac{\Omega_{ij}}{|\hat{\Omega}_{ij}|}\,, \quad (7)$$

i.e., use penalty varying with (i,j) as $\lambda/|\hat{\Omega}_{ij}|$.

III. Proposed Approach

Define the space $\mathcal{V}_p$ of all $p \times p$ matrices $\boldsymbol{V}$ that are symmetric with non-positive off-diagonal elements

$$\mathcal{V}_p = \Big\{\boldsymbol{V} \in \mathbb{R}^{p\times p} \,:\, \boldsymbol{V} = \boldsymbol{V}^\top,\ V_{ij} \leq 0,\ i \neq j\Big\} \quad (8)$$

A. Proposed Constrained Lasso: Penalized Log-Likelihood and Total Positivity

We propose to choose $\boldsymbol{\Omega}$ as

$$\min_{\boldsymbol{\Omega}\succ\boldsymbol{0},\ \boldsymbol{\Omega}\in\mathcal{V}_p} f_L(\boldsymbol{\Omega})\,. \quad (9)$$

We will use ADMM [32] after variable splitting to minimize $f_L(\boldsymbol{\Omega})$. Using variable splitting, consider

$$\min_{\substack{\boldsymbol{\Omega}\succ\boldsymbol{0}\\ \boldsymbol{V}\in\mathcal{V}_p}} \Big\{\mathrm{tr}(\hat{\boldsymbol{\Sigma}}\boldsymbol{\Omega}) - \ln(|\boldsymbol{\Omega}|) + \lambda\|\boldsymbol{V}^-\|_1\Big\} \text{ subject to } \boldsymbol{\Omega} = \boldsymbol{V}\,. \quad (10)$$

The scaled augmented Lagrangian for this problem is [32]

$$L_\rho = \mathrm{tr}(\hat{\boldsymbol{\Sigma}}\boldsymbol{\Omega}) - \ln(|\boldsymbol{\Omega}|) + \lambda\|\boldsymbol{V}^-\|_1 + \frac{\rho}{2}\|\boldsymbol{V} - \boldsymbol{\Omega} + \boldsymbol{U}\|_F^2 \quad (11)$$

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

(a) $\boldsymbol{\Omega}^{(k+1)} \leftarrow \arg\min_{\boldsymbol{\Omega}} L_a(\boldsymbol{\Omega}), \quad L_a(\boldsymbol{\Omega}) := \mathrm{tr}(\hat{\boldsymbol{\Sigma}}\boldsymbol{\Omega}) - \ln(|\boldsymbol{\Omega}|) + \frac{\rho}{2}\|\boldsymbol{V}^{(k)} - \boldsymbol{\Omega} + \boldsymbol{U}^{(k)}\|_F^2$

(b) $\boldsymbol{V}^{(k+1)} \leftarrow \arg\min_{\boldsymbol{V}\in\mathcal{V}_p} L_b(\boldsymbol{V}), \;\; L_b(\boldsymbol{V}) := \lambda\,\|\boldsymbol{V}^-\|_1 + \frac{\rho}{2}\|\boldsymbol{V} - \boldsymbol{\Omega}^{(k+1)} + \boldsymbol{U}^{(k)}\|_F^2$

(c) $\boldsymbol{U}^{(k+1)} \leftarrow \boldsymbol{U}^{(k)} + \left(\boldsymbol{V}^{(k+1)} - \boldsymbol{\Omega}^{(k+1)}\right)$

Solution to update (a) follows from [32, Sec. 6.5] and is given in Algorithm 1.

In update (b) notice that $L_b(\boldsymbol{V})$ is completely separable w.r.t. each element V_{ij}. Therefore, we solve $V_{ij}^{(k+1)} \leftarrow \arg\min_{V_{ij}\le 0,\, i\neq j} J_{ij}(V_{ij})$, where

$$J_{ij}(V_{ij}) := \lambda|V_{ij}|\mathbf{1}_{i\neq j} + \frac{\rho}{2}(V_{ij} - [\boldsymbol{\Omega}^{(k+1)} - \boldsymbol{U}^{(k)}]_{ij})^2$$

We claim that the solution is given by

$$V_{ij}^{(k+1)} = \begin{cases} [\boldsymbol{\Omega}^{(k+1)} - \boldsymbol{U}^{(k)}]_{ii} & \text{if } i = j \\ S_{neg}([\boldsymbol{\Omega}^{(k+1)} - \boldsymbol{U}^{(k)}]_{ij}, \frac{\lambda}{\rho}) & \text{if } i \neq j \end{cases} \tag{12}$$

where, with $(a)_+ := \max(0, a)$ and $(a)_- := \min(0, a)$,

$$S_{neg}(a, \beta) := (1 - \beta/|a|)_+ a_-$$

denotes scalar soft thresholding for negative values of a and hard thersholding for $a > 0$. When $i = j$, we need to minimize only $(V_{ij} - [\boldsymbol{\Omega}^{(k+1)} - \boldsymbol{U}^{(k)}]_{ij})^2$ w.r.t. $V_{ii} = V_{ij}$, thus the given solution follows. For constrained optimization under $V_{ij} \le 0$, after setting $A_{ij} = [\boldsymbol{\Omega}^{(k+1)} - \boldsymbol{U}^{(k)}]_{ij}$, consider the Lagrangian L_v

$$L_v = \lambda|V_{ij}| + \frac{\rho}{2}(V_{ij} - A_{ij})^2 + \nu V_{ij} \tag{13}$$

where $\nu \ge 0$ is the Lagrange multiplier for the inequality constraint $V_{ij} \le 0$. With v^* denoting an optimal solution, the KKT conditions for minimization are

$$0 \in \partial L_v = \lambda t + \rho(v^* - A_{ij}) + \nu \tag{14}$$

$$\nu v^* = 0 \tag{15}$$

$$\nu \ge 0 \tag{16}$$

$$v^* \le 0 \tag{17}$$

where ∂L_v denotes the subdifferential of L_v at v^* and

$$t = \begin{cases} v^*/|v^*| & \text{if } v^* \neq 0 \\ \in \{u : |u| \le 1,\, u \in \mathbb{R}\} & \text{if } v^* = 0\,. \end{cases} \tag{18}$$

When $A_{ij} > 0$, our claimed solution is $v^* = 0$. We need to check if $\nu \ge 0$ and $0 \in \partial L_v$ for some $|t| \le 1$. The choice $t = 0$ and $\nu = \rho A_{ij} > 0$ satisfies the KKT conditions. When $A_{ij} \le 0$, our claimed solution is the well-known soft-thresholding solution which satisfies the KKT conditions with $\nu = 0$. If $|A_{ij}) \le \rho/\lambda$, then $v^* = 0$ and $t = \rho A_{ij}/\lambda$ satisfies the KKT conditions since $|t| \le 1$. If $|A_{ij}) > \rho/\lambda$, then the given solution with $t = A_{ij}/|A_{ij}|$ satisfies the KKT conditions. This proves that the solution (12) minimizes $J_{ij}(V_{ij})$.

A pseudocode for the ADMM algorithm used in this paper is given in Algorithm 1 where we use the stopping (convergence) criterion following [32, Sec. 3.3.1] and varying penalty parameter ρ following [32, Sec. 3.4.1]. For constrained lasso we take $\lambda_{ij} = \lambda$ for all (i, j) in Algorithm 1.

B. Proposed Constrained Adaptive Lasso

With $\min_{\boldsymbol{\Omega}\succ\mathbf{0},\, \boldsymbol{\Omega}\in\mathcal{V}_p} f_L(\boldsymbol{\Omega})$ from constrained lasso optimization, modify (9) as

$$\min_{\boldsymbol{\Omega}\succ\mathbf{0},\, \boldsymbol{\Omega}\in\mathcal{V}_p} \mathrm{tr}(\boldsymbol{\Omega}\hat{\boldsymbol{S}}) - \ln(|\Omega|) + \lambda \sum_{i,j=1,\, i\neq j}^{p} \frac{\Omega_{ij}}{|\hat{\Omega}_{ij}|}\,, \tag{19}$$

i.e., use penalty varying with (i, j) as $\lambda_{ij} = \lambda/|\hat{\Omega}_{ij}|$ where $\hat{\Omega}_{ij}$ is obtained from proposed constrained lasso. Here we follow [31]. The solution given in Algorithm 1 applies.

Algorithm 1 ADMM Algorithm for Constrained Lasso and Constrained Adaptive Lasso

Input: Number of samples n, number of nodes p, data $\{\boldsymbol{x}(t)\}_{t=1}^n$, $\boldsymbol{x} \in \mathbb{R}^p$, regularization and penalty parameters λ_{ij} and ρ_0, tolerances τ_{abs} and τ_{rel}, variable penalty factor μ, maximum number of iterations k_{max}. $\lambda_{ij} = \lambda$ for lasso and $\lambda_{ij} = \lambda/\hat{\Omega}_{ij}$ for adaptive lasso where $\hat{\Omega}_{ij}$ is the result of lasso.

Output: estimated inverse covariance $\hat{\boldsymbol{\Omega}}$ and edge-set $\hat{\mathcal{E}}$

1: Calculate sample covariance $\hat{\boldsymbol{S}} = \frac{1}{n}\sum_{t=1}^n \boldsymbol{x}(t)\boldsymbol{x}^\top(t)$ (after centering $\boldsymbol{x}(t)$).

2: Initialize: $\boldsymbol{U}^{(0)} = \boldsymbol{V}^{(0)} = \mathbf{0}$, $\boldsymbol{\Omega}^{(0)} = (\mathrm{diag}(\hat{\boldsymbol{S}}))^{-1}$, where $\boldsymbol{U}, \boldsymbol{V} \in \mathbb{R}^{(p)\times(p)}$, $\rho^{(0)} = \rho_0$

3: converged = FALSE, $k = 0$

4: **while** converged = FALSE AND $k \le k_{max}$, **do**

5: Eigen-decompose $\hat{\boldsymbol{S}} - \rho^{(k)}\left(\boldsymbol{V}^{(k)} + \boldsymbol{U}^{(k)}\right)$ as $\hat{\boldsymbol{S}} - \rho^{(k)}\left(\boldsymbol{V}^{(k)} + \boldsymbol{U}^{(k)}\right) = \boldsymbol{Q}\boldsymbol{D}\boldsymbol{Q}^\top$ with diagonal matrix $\boldsymbol{D}$ consisting of eigenvalues. Define diagonal matrix $\tilde{\boldsymbol{D}}$ with ℓth diagonal element $\tilde{\boldsymbol{D}}_{\ell\ell} = (-\boldsymbol{D}_{\ell\ell} + \sqrt{\boldsymbol{D}_{\ell\ell}^2 + 4\rho^{(k)}})/(2\rho^{(k)})$. Set $\boldsymbol{\Omega}^{(k+1)} = \boldsymbol{Q}\tilde{\boldsymbol{D}}\boldsymbol{Q}^\top$.

6: Define thresholding operator $S_{neg}(a, \beta) := (1 - \beta/|a|)_+ a_-$ where $(a)_+ := \max(0, a)$ and $(a)_- := \min(0, a)$. The (i, j)th element of $\boldsymbol{V}$ is updated as

$$V_{ij}^{(k+1)} = \begin{cases} [\boldsymbol{\Omega}^{(k+1)} - \boldsymbol{U}^{(k)}]_{ii} & \text{if } i = j \\ S_{neg}([\boldsymbol{\Omega}^{(k+1)} - \boldsymbol{U}^{(k)}]_{ij}, \frac{\lambda_{ij}}{\rho}) & \text{if } i \neq j \end{cases}$$

7: Dual update $\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} + \left(\boldsymbol{V}^{(k+1)} - \boldsymbol{\Omega}^{(k+1)}\right)$.

8: Check convergence. Set tolerances

$$\tau_{pri} = p\,\tau_{abs} + \tau_{rel}\,\max(\|\boldsymbol{\Omega}^{(k+1)}\|_F, \|\boldsymbol{V}^{(k+1)}\|_F)$$
$$\tau_{dual} = p\,\tau_{abs} + \tau_{rel}\,\|\boldsymbol{U}^{(k+1)}\|_F/\rho^{(k)}\,.$$

Define $d_p = \|\boldsymbol{\Omega}^{(k+1)} - \boldsymbol{V}^{(k+1)}\|_F$ and $d_d = \rho^{(k)}\|\boldsymbol{V}^{(k+1)} - \boldsymbol{V}^{(k)}\|_F$. If $(d_p \le \tau_{pri})\; AND\; (d_d \le \tau_{dual})$, set converged = TRUE .

9: Update penalty parameter ρ :

$$\rho^{(k+1)} = \begin{cases} 2\rho^{(k)} & \text{if } d_p > \mu d_d \\ \rho^{(k)}/2 & \text{if } d_d > \mu d_p \\ \rho^{(k)} & \text{otherwise}\,. \end{cases}$$

We also need to set $\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k+1)}/2$ for $d_p > \mu d_d$ and $\boldsymbol{U}^{(k+1)} = 2\boldsymbol{U}^{(k+1)}$ for $d_d > \mu d_p$.

Model:	**Chain Graph**: number of nodes p=100				
sample size n	50	100	200	400	2000
Approach	F_1 score ($\pm\sigma$)				
Const. Lasso	0.4058 $\pm$0.0115	0.4353 $\pm$0.0113	0.4630 $\pm$0.0107	0.4921 $\pm$0.0095	0.5681 $\pm$0.0102
Kalofolias [13]	0.2631 $\pm$0.0919	0.2656 $\pm$0.0997	0.2753 $\pm$0.1049	0.2770 $\pm$0.1068	0.2941 $\pm$0.1139
GGL [17]	0.6727 $\pm$0.0218	0.6753 $\pm$0.0204	0.6715 $\pm$0.0192	0.6760 $\pm$0.0217	0.6758 $\pm$0.0215
Const. Adap. Lasso	**0.9851** $\pm$0.0089	**0.9977** $\pm$0.0035	**0.9999** $\pm$0.0007	**1.000** $\pm$0.0000	**1.0000** $\pm$0.0000
	Frobenius Error Norm ($\pm\sigma$)				
Const. Lasso	0.4752 $\pm$0.0116	0.4041 $\pm$0.0103	0.3405 $\pm$0.0084	0.2785 $\pm$0.0077	0.1620 $\pm$0.0053
Kalofolias [13]	0.5814 $\pm$0.0211	0.5564 $\pm$0.0206	0.5445 $\pm$0.0192	0.5357 $\pm$0.0196	0.5262 $\pm$0.0180
GGL [17]	0.2599 $\pm$0.0216	0.1799 $\pm$0.0135	0.1252 $\pm$0.0099	0.0881 $\pm$0.0066	0.0387 $\pm$0.0031
Const. Adap Lasso	0.2390 $\pm$0.0210	0.1623 $\pm$0.0126	0.1113 $\pm$0.0089	0.0785 $\pm$0.0066	0.0345 $\pm$0.0030
	Time(s) ($\pm\sigma$)				
Const. Lasso	2.4289 $\pm$0.3891	1.9707$\pm$0.0868	1.5710 $\pm$0.0601	1.2978 $\pm$0.0441	0.8413 $\pm$0.0286
Kalofolias [13]	0.2706 $\pm$0.0070	0.2706 $\pm$0.0172	0.2789 $\pm$0.0763	0.2652 $\pm$0.0030	0.2662 $\pm$0.0019
GGL [17]	0.0668 $\pm$0.0055	0.0688 $\pm$0.0086	0.0682 $\pm$0.0051	0.0687 $\pm$0.0064	0.0679 $\pm$0.0022
Const. Adap Lasso	8.1710 $\pm$0.8587	5.7741 $\pm$0.2202	4.8850 $\pm$1.2391	3.4469 $\pm$0.3290	1.5416 $\pm$0.1274
Model:	**Erdös-Rènyi Graph**: number of nodes p=100				
	F_1 score ($\pm\sigma$)				
Const. Lasso	0.2363 $\pm$0.0135	0.2462 $\pm$0.0141	0.2585 $\pm$0.0174	0.2745 $\pm$0.0152	0.3119 $\pm$0.0205
Kalofolias [13]	0.1363 $\pm$0.0398	0.1420 $\pm$0.0397	0.1491 $\pm$0.0399	0.1666 $\pm$0.0495	0.1586 $\pm$0.0592
GGL [17]	0.4609 $\pm$0.0232	0.4901 $\pm$0.0232	0.4984 $\pm$0.0197	0.5032 $\pm$0.0175	0.4927 $\pm$0.0218
Const. Adap Lasso	**0.7165** $\pm$0.0575	**0.8756** $\pm$0.0391	**0.9584** $\pm$0.0183	**0.9859** $\pm$0.0087	**0.9991** $\pm$0.0022
	Frobenius Error Norm ($\pm\sigma$)				
Const. Lasso	0.9964 $\pm$0.0043	0.9898 $\pm$0.0098	0.6367 $\pm$0.0914	0.4743 $\pm$0.0378	0.3720 $\pm$0.0212
Kalofolias [13]	0.7621 $\pm$0.0316	0.7028 $\pm$0.0236	0.6756 $\pm$0.0224	0.6562 $\pm$0.0200	0.6475 $\pm$0.0166
GGL [17]	0.6062 $\pm$0.0615	0.4204 $\pm$0.0492	0.2866 $\pm$0.0317	0.1963 $\pm$0.0234	0.0848 $\pm$0.0117
Const. Adap Lasso	0.6309 $\pm$0.0609	0.4454 $\pm$0.0501	0.3119 $\pm$0.0379	0.2265 $\pm$0.0301	0.1039 $\pm$0.0186
	Time(s) ($\pm\sigma$)				
Const. Lasso	3.4578 $\pm$0.6695	2.6559 $\pm$0.2382	2.1640 $\pm$0.1588	1.8420 $\pm$0.0908	1.2341 $\pm$0.0584
Kalofolias [13]	0.2587 $\pm$0.0068	0.2570 $\pm$0.0031	0.2568 $\pm$0.0023	0.2557 $\pm$0.0024	0.2582 $\pm$0.0021
GGL [17]	0.0652 $\pm$0.0047	0.0639 $\pm$0.0013	0.0637 $\pm$0.0012	0.0636 $\pm$0.0015	0.0633 $\pm$0.0014
Const. Adap Lasso	9.3087 $\pm$0.3813	8.5631 $\pm$0.4493	7.0382 $\pm$0.4599	5.3631 $\pm$0.3213	2.8511 $\pm$0.2075

TABLE I: *Results for Chain and Erdos-Renyi graphs. Frobenius error norm is* $\|c\hat{\mathbf{\Omega}}^- - \mathbf{\Omega}_0^-\|_F / \|\mathbf{\Omega}_0^-\|_F$ *where scalar* c *is picked to minimize* $\|c\hat{\mathbf{\Omega}}^- - \mathbf{\Omega}_0^-\|_F$*. "Const. Lasso" stands for the proposed constrained lasso approach that enforces total positivity, and "Const. Adap lasso" denotes its adaptive lasso version.*

10: $k \leftarrow k+1$
11: **end while**
12: For $i \neq j$, if $|V_{ij}| > 0$, assign edge $\{i,j\} \in \hat{\mathcal{E}}$, else $\{i,j\} \notin \hat{\mathcal{E}}$. Inverse covariance estimate is $\hat{\mathbf{\Omega}}$.

IV. Simulation Examples

We consider Gaussian graphical models based on two graphs: a **chain graph** where p nodes are connected in succession, and an **Erdös-Rènyi** graph where nodes are connected with probability $p_{er} = 0.03$. In each model, in the upper triangular Ω (inverse covariance), $\Omega_{ij} = 0$ if $\{i,j\} \notin \mathcal{E}$, and Ω_{ij} is uniformly distributed over $[-0.3, -0.1]$ if $\{i,j\} \in \mathcal{E}$. With $\Omega = \Omega^\top$, we take $\Omega_{ii} = -\sum_{j=1}^{p} \Omega_{ij}$ for every i, yielding the combinatorial Laplacian matrix $\boldsymbol{L} = \mathbf{\Omega}$. Now add $\kappa \boldsymbol{I}$ to $\mathbf{\Omega}$ with κ picked to make minimum eigenvalue of $\mathbf{\Omega}+\kappa\boldsymbol{I}$ equal to 0.001, and with $\mathbf{\Phi}\mathbf{\Phi}^\top = (\mathbf{\Omega} + \kappa \boldsymbol{I})^{-1}$, we generate $\boldsymbol{x} = \mathbf{\Phi}\boldsymbol{w}$ with $\boldsymbol{w} \in \mathbb{R}^p$ as Gaussian $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. We generate n i.i.d. observations from $\boldsymbol{x}$ using $p = 100$. Addition of $\kappa \boldsymbol{I}$ yields a generalized Laplacian matrix $\boldsymbol{L} = \mathbf{\Omega} + \kappa \boldsymbol{I}$ [17]. Given choice of κ leads to a precision matrix (and corresponding covariance matrix) with (matrix inversion) condition number > 1000 for both models, making it a challenging problem.

We apply four methods for estimating the true edgeset $\mathcal{E}_0$ and true (off-diagonal) inverse covariance $\mathbf{\Omega}_0^-$. One of the methods is the signal smoothness-based method of [13] which yields the weighted adjacency matrix $\boldsymbol{W}$, equaling $-\mathbf{\Omega}_0^-$ (off-diagonal $\mathbf{\Omega}_0$) under the Laplacian assumption, hence one of our performance criterion is error in estimating $\mathbf{\Omega}_0^-$. The chosen methods are

(1) Proposed constrained lasso approach, labeled "Const. Lasso" in Table I. We use Algorithm 1 with constant λ for all (i,j).
(2) Smoothness-based graph learning [13] solved via the forward-backward algorithm available in [29] (MATLAB function gsp_learn_graph_log_degrees.m), labeled "smooth [13]" in Table I. It requires one to set small values in estimated $\boldsymbol{W}$ to be set to zero; following [13], [29], all $\hat{W}_{ij} \leq 10^{-4}$ are set to zero.
(3) Generalized graph Laplacian (GGL) method of [17], using MATLAB function estimate_ggl.m from [30], labeled "GGL [17]" in Table I.
(4) Proposed constrained adaptive lasso approach, labeled "Const. Adap Lasso" in Table I. We use Algorithm 1 with edge-dependent λ, $\lambda_{ij} = \lambda/|\hat{\Omega}_{ij}|$, where $\hat{\Omega}_{ij}$ is obtained from proposed constrained lasso.

A performance measure is F_1-score for efficacy in edge detection. The F_1-score is defined as $F_1 = 2 \times \text{precision} \times \text{recall}/(\text{precision} + \text{recall})$ where $\text{precision} = |\hat{\mathcal{E}} \cap \mathcal{E}_0|/|\hat{\mathcal{E}}|$, $\text{recall} = |\hat{\mathcal{E}} \cap \mathcal{E}_0|/|\mathcal{E}_0|$, and $\mathcal{E}_0$ and $\hat{\mathcal{E}}$ denote the true and estimated edge sets, respectively. Table I shows the simulation results where the run time in seconds was calculated via MATLAB tic-toc functions on a Window 10 operating system with processor Intel(R) Core(TM) i5-6400T CPU @2.20 GHz with 12 GB RAM. For each of the four schemes, tuning parameter λ (β for [13]) was picked for $n = 200$ via simulations to maximize F_1-score, then λ was scaled as $\propto \sqrt{\ln(p)/n}$ [8], [17] while β was kept fixed. The performance measures are F_1-score for efficacy in edge detection, and normalized Frobenius error norm in estimating $\mathbf{\Omega}_0^-$ (off-diagonal true $\mathbf{\Omega}_0$), defined as $\|c\hat{\mathbf{\Omega}}^- - \mathbf{\Omega}_0^-\|_F / \|\mathbf{\Omega}_0^-\|_F$ where c is selected as follows. We scale estimated $\mathbf{\Omega}_0^-$ (when signal-smoothing is used, set $\hat{\mathbf{\Omega}}_0^- = -\hat{\boldsymbol{W}}_0^-$), by a scalar c chosen to minimize mean-square error $\|\mathbf{\Omega}_0^- - c\hat{\mathbf{\Omega}}_0^-\|_F^2$, resulting in $c = \text{tr}(\mathbf{\Omega}_0^- \hat{\mathbf{\Omega}}_0^-)/\text{tr}(\hat{\mathbf{\Omega}}_0^- \hat{\mathbf{\Omega}}_0^-)$. In practice, $\mathbf{\Omega}_0^-$ is unknown. The above scaling preserves relative weighting among Ω_{ij}'s which is what is relevant in applications [13] and is available without knowing $\mathbf{\Omega}_0^-$.

The condition number > 1000 of covariance and precision matrices for both models makes it a challenging problem. We see that only constrained adaptive lasso performs well, with high F_1 scores.

V. Conclusions

We considered the problem of estimating the structure of an undirected weighted sparse graphical model of multivariate data under the assumption of total positivity where all partial correlations are non-negative. We investigated modifications to widely used penalized log-likelihood approaches to enforce total positivity but not the Laplacian structure. An ADMM algorithm was presented for constrained optimization under total positivity and lasso as well as adaptive lasso penalties. Numerical results show that the proposed constrained adaptive lasso approach significantly outperforms existing Laplacian-based approaches.

References

[1] X. Dong, D. Thanou, M. Rabbat and P. Frossard, "Learning graphs from data," *IEEE Signal Process. Mag.*, pp. 44-63, May 2019.

[2] S.L. Lauritzen, *Graphical models*. Oxford, UK: Oxford Univ. Press, 1996.

[3] J. Whittaker, *Graphical Models in Applied Multivariate Statistics*. New York: Wiley, 1990.

[4] P. Danaher, P. Wang and D.M. Witten, "The joint graphical lasso for inverse covariance estimation across multiple classes," *J. Royal Statistical Society, Series B*, vol. 76, pp. 373-397, 2014.

[5] N. Meinshausen and P. Bühlmann, "High-dimensional graphs and variable selection with the Lasso," *Ann. Statist.*, vol. 34, no. 3, pp. 1436-1462, 2006.

[6] K. Mohan, P. London, M. Fazel, D. Witten and S.I. Lee, "Node-based learning of multiple Gaussian graphical models," *J. Machine Learning Research*, vol. 15, 2014.

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

[8] O. Banerjee, L.E. Ghaoui and A. d'Aspremont, "Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data," *J. Machine Learning Res.*, vol. 9, pp. 485-516, 2008.

[9] Y. Wang, U. Roy and C. Uhler, "Learning high-dimensional Gaussian graphical models under total positivity without adjustment of tuning parameters," in *Proc. 23rd Intern. Conf. Artificial Intelligence & Statistics (AISTATS)*, Palermo, Italy, 2020.

[10] S. Lauritzen, C. Uhler and P. Zwiernik, "Maximum likelihood estimation in Gaussian models under total positivity," *Annals of Statistics*, vol. 47, pp. 1835-1863, 2019.

[11] 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*, vol. 473, pp. 145-179, 2015.

[12] S. Karlin and Y. Rinott, "M-matrices as covariance matrices of multinormal distributions," *Linear Algebra and its Applications*, vol. 52, pp. 419-438, 1983.

[13] V. Kalofolias, "How to learn a graph from smooth signals," in *Proc. 19th Intern. Conf. Artificial Intelligence & Statistics (AISTATS)*, Cadiz, Spain, 2016.

[14] V. Kalofolias and N. Perraudin, "Large scale graph learning from smooth signals," in *7th Intern. Conf. Learning Representations (ICLR 2019)*, New Orleans, LA, USA, May 6-9, 2019.

[15] X. Dong, D. Thanou, P. Frossard and P. Vandergheynst "Learning Laplacian matrix in smooth graph signal representations," *IEEE Trans. Signal Process.*, vol. 64, no. 23, pp. 6160-6173, Dec. 1, 2016.

[16] E. Pavez and A. Ortega, "Generalized Laplacian precision matrix estimation for graph signal processing," in *Proc. IEEE ICASSP 2016*, Shanghai, China, March 2016, pp. 6350-6354.

[17] H.E. Egilmez, E. Pavez and A. Ortega, "Graph learning from data under Laplacian and structural constraints," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 6, pp. 825-841, Sept. 2017.

[18] E. Pavez, H.E. Egilmez and A. Ortega, "Learning graphs with monotone topology properties and multiple connected components," *IEEE Trans. Signal Process.*, vol. 66, no. 9, pp. 2399-2413, May 1, 2018.

[19] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering.," in *Proc. NIPS*, vol. 14, pp. 585-591, 2001.

[20] M. Belkin, P. Niyogi and V. Sindhwani "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *J. Machine Learning Research*, vol. 7, pp. 2399-2434, 2006.

[21] Y. Bengio, O. Delalleau and N. Le Roux, "Label propagation and quadratic criterion," in *Semi-Supervised Learning*, O. Chapelle, B. Schölkopf and A. Zien (Eds), pp. 193-216, MIT Press, 2006.

[22] U.V. Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395-416, 2007

[23] X. Zhu, Z. Ghahramani and J. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. 20th Intern. Conf. Machine Learning (ICML)*, vol. 3, pp. 912-919, 2003.

[24] X. Zhou and M. Belkin, "Semi-supervised learning by higher order regularization," in *Proc. 14th Intern. Conf. Artificial Intelligence & Statistics (AISTATS)*, Fort Lauderdale, FL, 2011.

[25] F. Wang and C. Zhang, "Label propagation through linear neighborhoods," *IEEE Trans. Knowledge and Data Engineering*, vol. 20, no. 1, pp. 55-67, 2008.

[26] S.T. Daitch, J.A. Kelner and D.A. Spielman, "Fitting a graph to vector data," in *Proc. 26th Intern. Conf. Machine Learning (ICML)*, pp. 201–208, Montreal, Canada, 2009.

[27] B. Lake and J. Tenenbaum, "Discovering structure by learning sparse graphs," *Proc. 32nd Annual Meeting of the Cognitive Science Society (CogSci 2010)*, Portland, Oregon, Aug. 2010, pp. 778-784.

[28] N. Komodakis and J.C. Pesquet, "Playing with duality: An overview of recent primal-dual approaches for solving large-scale optimization problems," *IEEE Signal Processing Mag.*, vol. 32, pp. 31-54, 2015.

[29] N. Perraudin, J. Paratte, D. Shuman, V. Kalofolias, P. Vandergheynst and D.K. Hammond, "GSPBOX: A toolbox for signal processing on graphs," *arXiv:1408.5781v2 [cs.IT]*, 15 March 2016.

[30] H.E. Egilmez, E. Pavez and A. Ortega, "GLL: Graph Laplacian learning package, version 1.0," [Online]. Available: https://github.com/STACUSC/Graph_Learning, 2017.

[31] H. Zou, "The adaptive lasso and its oracle properties," *J. American Statistical Assoc.*, vol. 101, pp. 1418-1429, 2006.

[32] S. Boyd, N. Parikh, E. Chu, B. Peleato and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1-122, 2010.