ESTIMATION OF DIFFERENTIAL GRAPHS VIA LOG-SUM PENALIZED D-TRACE LOSS

Jitendra K. Tugnait

Department of Electrical & Computer Engineering
Auburn University, Auburn, AL 36849, USA

ABSTRACT

We consider the problem of estimating differences in two Gaussian graphical models (GGMs) which are known to have similar structure. The GGM structure is encoded in its precision (inverse covariance) matrix. In many applications one is interested in estimating the difference in two precision matrices to characterize underlying changes in conditional dependencies of two sets of data. Most existing methods for differential graph estimation are based on a lasso penalized loss function. In this paper, we analyze a log-sum penalized D-trace loss function approach for differential graph learning. An alternating direction method of multipliers (ADMM) algorithm is presented to optimize the objective function. Theoretical analysis establishing consistency in estimation in high-dimensional settings is provided. We illustrate our approach using a numerical example where log-sum penalized D-trace loss significantly outperforms lasso-penalized D-trace loss as well as smoothly clipped absolute deviation (SCAD) penalized D-trace loss.

Keywords: Sparse graph learning; differential graph estimation; Gaussian graphical models; lasso; log-sum penalty.

1. INTRODUCTION

Graphical models provide a powerful tool for analyzing multivariate data [1,2]. 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 $\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1}$. 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 [3–6].

More recently there has been increasing interest in differential network analysis where one is interested in estimating the difference in two inverse covariance matrices [7–9]. Given observations $\boldsymbol{x}$ and $\boldsymbol{y}$ from two groups of subjects, one is interested in the difference $\boldsymbol{\Delta} = \boldsymbol{\Omega}_y - \boldsymbol{\Omega}_x$, where $\boldsymbol{\Omega}_x = (E\{\boldsymbol{x}\boldsymbol{x}^\top\})^{-1}$ and $\boldsymbol{\Omega}_y = (E\{\boldsymbol{y}\boldsymbol{y}^\top\})^{-1}$. The associated differential graph is $\mathcal{G}_\Delta = (V, \mathcal{E}_\Delta)$ where $\{i,j\} \in \mathcal{E}_\Delta$ iff $\boldsymbol{\Delta}_{ij} \neq 0$. It characterizes differences between the GGMs of the two sets of data. We use the term differential graph as in [10, 11] ([7–9] use the term differential network). As noted in [9], in biostatistics, the differential network/graph describes the changes in conditional dependencies between components under different environmental or genetic conditions. For instance, one may be interested in the differences in the graphical models of healthy and impaired subjects, or models under different disease states, given gene expression data or functional MRI signals [3,12,13].

This work was supported by NSF Grant ECCS-2040536. Author's email: tugnajk@auburn.edu

All existing methods for differential graph estimation except for [21], are based on a lasso penalized loss function. In this paper we consider a log-sum penalty (LSP) instead of lasso penalty to regularize the problem, motivated by [14]. For sparse solutions, ideal penalty is ℓ_0 which is non-convex and the problem is usually impossible to solve. So one relaxes the problem using ℓ_1 (lasso) penalty which is convex. Ref. [14] notes that a key difference between the ℓ_1 and ℓ_0 norms is the dependence on magnitude. Their solution to rectify this imbalance, is iterative reweighted ℓ_1 minimization, and to construct an analytical framework, [14] suggests the log-sum penalty. As in [8,15] (and others) we use a D-trace loss function for differential graph estimation where D-trace refers to difference-in-trace loss function, a term coined in [16] in the context of graphical model estimation.

Unlike lasso, log-sum penalty is non-convex. Although, non-convex penalties have been extensively used for graph estimation (see [17–20] and references therein), only [21] has investigated use of non-convex penalties SCAD (smoothly clipped absolute deviation) and MCP (minimax concave penalty) for differential graph estimation. Ref. [21] does not consider LSP, and in our numerical results, we show that our LSP-based differential graph estimator significantly outperforms both lasso and SCAD penalized methods of [8,15] and [21], respectively.

Notation: For a set V, $|V|$ or card(V) denotes its cardinality. Given $\boldsymbol{A} \in \mathbb{R}^{p\times p}$, we use $\phi_{\min}(\boldsymbol{A})$, $\phi_{\max}(\boldsymbol{A})$, $|\boldsymbol{A}|$ and tr$(\boldsymbol{A})$ to denote the minimum eigenvalue, maximum eigenvalue, determinant and trace of $\boldsymbol{A}$, respectively. For $\boldsymbol{B} \in \mathbb{R}^{p\times q}$, we define $\|\boldsymbol{B}\| = \sqrt{\phi_{\max}(\boldsymbol{B}^\top\boldsymbol{B})}$, $\|\boldsymbol{B}\|_F = \sqrt{\text{tr}(\boldsymbol{B}^\top\boldsymbol{B})}$, $\|\boldsymbol{B}\|_1 = \sum_{i,j}|B_{ij}|$, where B_{ij} is the (i,j)-th element of $\boldsymbol{B}$ (also denoted by $[\boldsymbol{B}]_{ij}$), $\|\boldsymbol{B}\|_\infty = \max_{i,j}|B_{ij}|$ and $\|\boldsymbol{B}\|_{1,\infty} = \max_i \sum_j |B_{ij}|$. The symbol $\otimes$ denote the Kronecker product. Let $S = \mathcal{E}_\Delta = \{\{k,\ell\} : |\boldsymbol{\Delta}_{(k\ell)}| \neq 0\}$ where $\boldsymbol{\Delta} = [\boldsymbol{\Delta}_{(k\ell)}] \in \mathbb{R}^{p\times p}$. Then $\boldsymbol{\Delta}_S$ denotes the submatrix of $\boldsymbol{\Delta}$ with rows and columns indexed by S, i.e., $\boldsymbol{\Delta}_S = [\boldsymbol{\Delta}]_{(k,\ell)\in S}$. Suppose $\boldsymbol{\Gamma} = \boldsymbol{A} \otimes \boldsymbol{B}$ for some matrices $\boldsymbol{A} = [A_{ij}]$ and $\boldsymbol{B} = [B_{ij}]$. For any two subsets T_1 and T_2 of $[p]\times[p]$, $\boldsymbol{\Gamma}_{T_1,T_2}$ denotes the submatrix of $\boldsymbol{\Gamma}$ with rows and columns indexed by T_1 and T_2, i.e., $\boldsymbol{\Gamma}_{T_1,T_2} = [A_{j\ell}B_{kq}]_{(j,k)\in T_1,(\ell,q)\in T_2}$.

2. LOG-SUM PENALIZED PENALIZED D-TRACE LOSS

Let $\boldsymbol{x}$, $\boldsymbol{y} \in \mathbb{R}^p$ be zero-mean, Gaussian, and independent of each other. Given i.i.d. samples $\boldsymbol{x}(t)$, $t = 1, 2, \cdots, n_x$, of $\boldsymbol{x}$, and similarly given i.i.d. samples $\boldsymbol{y}(t)$, $t = 1, 2, \cdots, n_y$, of $\boldsymbol{y}$, form the sample covariance estimates

$$\hat{\boldsymbol{\Sigma}}_x = \frac{1}{n_x}\sum_{t=1}^{n_x} \boldsymbol{x}(t)\boldsymbol{x}^\top(t)\,, \quad \hat{\boldsymbol{\Sigma}}_y = \frac{1}{n_y}\sum_{t=1}^{n_y} \boldsymbol{y}(t)\boldsymbol{y}^\top(t)\,. \quad (1)$$

and denote their true values as $\boldsymbol{\Sigma}_x^* = \boldsymbol{\Omega}_x^{-*} (= (\boldsymbol{\Omega}_x^*)^{-1})$ and $\boldsymbol{\Sigma}_y^* = \boldsymbol{\Omega}_y^{-*}$. We wish to estimate $\boldsymbol{\Delta} = \boldsymbol{\Omega}_y^* - \boldsymbol{\Omega}_x^*$ and graph $\mathcal{G}_\Delta = (V, \mathcal{E}_\Delta)$, based on $\hat{\boldsymbol{\Sigma}}_x$ and $\hat{\boldsymbol{\Sigma}}_y$. As in [8] (see also [15, Sec. 2.1]), we will use a convex D-trace loss function given by

$$L(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y) = \frac{1}{2}\text{tr}(\hat{\boldsymbol{\Sigma}}_x \boldsymbol{\Delta} \hat{\boldsymbol{\Sigma}}_y \boldsymbol{\Delta}^\top) - \text{tr}(\boldsymbol{\Delta}(\hat{\boldsymbol{\Sigma}}_x - \hat{\boldsymbol{\Sigma}}_y)) \quad (2)$$

where D-trace refers to difference-in-trace loss function, a term coined in [16] in the context of graphical model estimation. The function $L(\boldsymbol{\Delta}, \boldsymbol{\Sigma}_x^*, \boldsymbol{\Sigma}_y^*)$ is strictly convex in $\boldsymbol{\Delta}$ and has a unique minimum at $\boldsymbol{\Delta}^* = \boldsymbol{\Omega}_y^* - \boldsymbol{\Omega}_x^*$ [8, 15]. When sample covariances are used, [15] proposed to estimate $\boldsymbol{\Delta}$ by minimizing the lasso penalized loss function

$$L_\lambda(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y) = L(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y) + \lambda \sum_{i,j=1}^{p} |\boldsymbol{\Delta}_{ij}| \quad (3)$$

where $\lambda > 0$ is a tuning parameter. In [8] a symmetrized version of $L(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y)$ is used with lasso penalty. In this paper we will replace the lasso penalty with a log-sum penalty, motivated by [14].

With $0 < \delta \ll 1$, following [14], define the log penalty for $\theta \in \mathbb{R}$,

$$p_\lambda(\theta) = \lambda \ln(1 + |\theta|/\delta)\ . \quad (4)$$

Replace the lasso penalty in (3) with $p_\lambda(\theta)$, to yield

$$L_{LSP}(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y) = L(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y) + \sum_{i,j=1}^{p} p_\lambda(\Delta_{ij}), \quad (5)$$

the log-sum penalized D-trace loss. Unlike $L_\lambda(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y)$, we now have a non-convex function of $\boldsymbol{\Delta}$ in $L_{LSP}(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y)$.

Similar to the SCAD (smoothly clipped absolute deviation) penalty in [17,18], we solve the problem $\min_{\boldsymbol{\Delta}}\ L_{LSP}(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y)$ iteratively, where in each iteration, the problem is convex. Using $\partial p_\lambda(|\theta|)/\partial|\theta| = \lambda/(|\theta| + \delta)$, a local linear approximation (LLA) to $p_\lambda(|\theta|)$ around θ_0 yields

$$p_\lambda(|\theta|) \approx P_\lambda(|\theta_0|) + \frac{\lambda}{|\theta_0| + \delta}(|\theta| - |\theta_0|) \Rightarrow \frac{\lambda}{|\theta_0| + \delta}|\theta|, \quad (6)$$

therefore, with θ_0 fixed, we consider only the last term above for optimization w.r.t. θ. Suppose we have a "good" initial solution $\bar{\boldsymbol{\Delta}}$ to the problem (from e.g., using $L_\lambda(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y)$). Then, given $\bar{\boldsymbol{\Delta}}$, a local linear approximation (LLA) to $p_\lambda(\Delta_{ij})$ yields the convex function to be minimized

$$\tilde{L}_{LSP}(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y) = L(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y) + \sum_{i,j=1}^{p} \frac{\lambda\,|\Delta_{ij}|}{|[\bar{\boldsymbol{\Delta}}]_{ij}| + \delta}\,, \quad (7)$$

$$= L(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y) + \sum_{i,j=1}^{p} \lambda_{ij}|\Delta_{ij}|\,, \quad \lambda_{ij} = \frac{\lambda}{|[\bar{\boldsymbol{\Delta}}]_{ij}| + \delta}\,. \quad (8)$$

This is then quite similar to adaptive lasso [22] with adaptive lasso-like penalty λ_{ij}, except that [22] has $\delta = 0$. By [17, Theorem 1], the LLA of LSP provides a majorization of LSP, therby yielding a majorization-minimization approach. In fact, by [17, Theorem 2], the LLA is the best convex majorization of the LSP (which is concave since $\partial^2 p_\lambda(|\theta|)/\partial|\theta|^2 = -\lambda/(|\theta| + \delta)^2$).

Suppose

$$\hat{\boldsymbol{\Delta}} = \arg\min_{\boldsymbol{\Delta}} \tilde{L}_{LSP}(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y)\,. \quad (9)$$

Even though $\boldsymbol{\Delta}$ is symmetric, $\hat{\boldsymbol{\Delta}}$ is not. We can symmetrize it by setting $\hat{\boldsymbol{\Delta}}_{sym} = \frac{1}{2}(\hat{\boldsymbol{\Delta}} + \hat{\boldsymbol{\Delta}}^\top)$, after obtaining $\hat{\boldsymbol{\Delta}}$.

3. OPTIMIZATION

Instead of minimizing $L_{LSP}(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y)$ in (5), we will iteratively minimize its LLA $\tilde{L}_{LSP}(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y)$. In each minimization of $\tilde{L}_{LSP}(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y)$, we we use an alternating direction method of multipliers (ADMM) approach [23] with variable splitting. We can use the solution of [15] (see also [8]) modified for adaptive penalty λ_{ij} ([15] uses $\lambda_{ij} = \lambda$ for all edges $\{i, j\}$). Using variable splitting, consider

$$\min_{\boldsymbol{\Delta}, \boldsymbol{W}} \Big\{ L(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y) + \sum_{i,j=1}^{p} \lambda_{ij}|W_{ij}| \Big\} \text{ subject to } \boldsymbol{\Delta} = \boldsymbol{W}\,. \quad (10)$$

The **scaled** augmented Lagrangian for this problem is [23]

$$L_\rho = L(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y) + \sum_{i,j=1}^{p} \lambda_{ij}|W_{ij}| + \frac{\rho}{2}\|\boldsymbol{\Delta} - \boldsymbol{W} + \boldsymbol{U}\|_F^2 \quad (11)$$

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

(a) $\boldsymbol{\Delta}^{(k+1)} \leftarrow \arg\min_{\boldsymbol{\Delta}} L_a(\boldsymbol{\Delta}),\ \ L_a(\boldsymbol{\Delta}) := L(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y) + \frac{\rho}{2}\|\boldsymbol{\Delta} - \boldsymbol{W}^{(k)} + \boldsymbol{U}^{(k)}\|_F^2$

(b) $\boldsymbol{W}^{(k+1)} \leftarrow \arg\min_{\boldsymbol{W}} L_b(\boldsymbol{W}),\ \ L_b(\boldsymbol{W}) := \sum_{i,j=1}^{p} \lambda_{ij}|W_{ij}| + \frac{\rho}{2}\|\boldsymbol{\Delta}^{(k+1)} - \boldsymbol{W} + \boldsymbol{U}^{(k)}\|_F^2$

(c) $\boldsymbol{U}^{(k+1)} \leftarrow \boldsymbol{U}^{(k)} + \Big(\boldsymbol{\Delta}^{(k+1)} - \boldsymbol{W}^{(k+1)}\Big)\,. \quad (12)$

Update (a): The solution is as given in [8, 15]. Carry out eigen-decomposition of $\hat{\boldsymbol{\Sigma}}_x$ and $\hat{\boldsymbol{\Sigma}}_y$ as $\hat{\boldsymbol{\Sigma}}_x = \boldsymbol{Q}_x \boldsymbol{D}_x \boldsymbol{Q}_x^\top$, $\boldsymbol{Q}_x \boldsymbol{Q}_x^\top = \boldsymbol{I}$ and $\hat{\boldsymbol{\Sigma}}_y = \boldsymbol{Q}_y \boldsymbol{D}_y \boldsymbol{Q}_y^\top$, $\boldsymbol{Q}_y \boldsymbol{Q}_y^\top = \boldsymbol{I}$, where $\boldsymbol{D}_x$ and $\boldsymbol{D}_y$ are diagonal matrices. Then $\hat{\boldsymbol{\Delta}}$ that minimizes $L_a(\boldsymbol{\Delta})$ is given by

$$\hat{\boldsymbol{\Delta}} = \boldsymbol{Q}_x \Big[\boldsymbol{B} \circ [\boldsymbol{Q}_x^\top \big(\hat{\boldsymbol{\Sigma}}_x - \hat{\boldsymbol{\Sigma}}_y + \rho(\boldsymbol{W} - \boldsymbol{U})\big)\boldsymbol{Q}_y]\Big]\boldsymbol{Q}_y^\top \quad (13)$$

where the symbol $\circ$ denotes the Hadamard product and $\boldsymbol{B} \in \mathbb{R}^{p \times p}$ organizes the diagonal of $(\boldsymbol{D}_y \otimes \boldsymbol{D}_x + \rho \boldsymbol{I})^{-1}$ in a matrix with $\boldsymbol{B}_{ij} = 1/([\boldsymbol{D}_x]_{ii}[\boldsymbol{D}_y]_{jj} + \rho)$. Note that the eigen-decomposition of $\hat{\boldsymbol{\Sigma}}_x$ and $\hat{\boldsymbol{\Sigma}}_y$ has to be done only once. Thus

$$\boldsymbol{\Delta}^{(k+1)} = \boldsymbol{Q}_x \Big[\boldsymbol{B} \circ [\boldsymbol{Q}_x^\top \big(\hat{\boldsymbol{\Sigma}}_x - \hat{\boldsymbol{\Sigma}}_y + \rho(\boldsymbol{W}^{(k)} - \boldsymbol{U}^{(k)})\big)\boldsymbol{Q}_y]\Big]\boldsymbol{Q}_y^\top \quad (14)$$

Update (b): With $\boldsymbol{A} = \boldsymbol{\Delta}^{(k+1)} + \boldsymbol{U}^{(k)}$, we have the lasso solution

$$(\boldsymbol{W}_{ij})^{(k+1)} = \Big(1 - \frac{(\lambda_{ij}/\rho)}{|[\boldsymbol{A}]_{ij}|}\Big)_+ [\boldsymbol{A}]_{ij} \quad (15)$$

where $(a)_+ = \max(0, a)$.

Our optimization algorithm is as follows.

1. Calculate sample estimates $\hat{\boldsymbol{\Sigma}}_x$ and $\hat{\boldsymbol{\Sigma}}_y$. Compute $\boldsymbol{B}$ as in ADMM update (a). Initialize $m = 1$, $\check{\boldsymbol{\Delta}}^{(0)} = \mathbf{0}$, $\bar{\boldsymbol{\Delta}} = \check{\boldsymbol{\Delta}}^{(0)}$ in (8) resulting in $\lambda_{ij} = \lambda/\delta$.
2. Set $k = 1$, $\boldsymbol{\Delta}^{(k)} = \check{\boldsymbol{\Delta}}^{(m-1)}$ and $\boldsymbol{W}^{(k)} = \boldsymbol{U}^{(k)} = \mathbf{0}$ to start the ADMM algorithm.
   (i) In the ADMM step (a), calculate $\boldsymbol{\Delta}^{(k+1)}$ via (14).
   (ii) In ADMM step (b), calculate $\boldsymbol{W}^{(k+1)}$ via (15).
   (iii) Execute ADMM step (c), via (12).
   (iv) Let $k \leftarrow k + 1$ and repeat steps (i)-(iii) until convergence. Denote the converged value as $\hat{\boldsymbol{\Delta}}$.
3. Set $\check{\boldsymbol{\Delta}}^{(m)} = \hat{\boldsymbol{\Delta}}$, $\bar{\boldsymbol{\Delta}} = \check{\boldsymbol{\Delta}}^{(m)}$ in (8), hence, $\lambda_{ij} = \lambda/(|[\check{\boldsymbol{\Delta}}^{(m)}]_{ij}| + \delta)$. Let $m \leftarrow m + 1$.
4. Repeat steps 2 and 3 until convergence. The converged $\check{\boldsymbol{\Delta}}$ is the final estimate $\hat{\boldsymbol{\Delta}}$. (For simulation results shown in Sec. 5, we terminated after two iterations of steps 2 and 3, similar to [17–20].)

Convergence. For the ADMM algorithm, a stopping (convergence) criterion following [23, Sec. 3.3.1] can be devised. The stopping criterion is based on primal and dual residuals being small where, in our case, at $(k+1)$st iteration, the primal residual is given by $\boldsymbol{\Delta}^{(k+1)} - \boldsymbol{W}^{(k+1)}$ and the dual residual by $\rho(\boldsymbol{W}^{(k+1)} - \boldsymbol{W}^{(k)})$. Convergence criterion is met when the norms of these residuals are below some threshold. The objective function $\tilde{L}_{LSP}(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y)$, given by (8), is strictly convex. It is also closed, proper and lower semi-continuous. Hence, for any fixed $\rho > 0$, the ADMM algorithm is guaranteed to converge [23, 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. Since the LLA of LSP provides a majorization of LSP, therby yielding a majorization-minimization approach, the overall iterative minimization approach yileds a local minimum of $L_{LSP}(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y)$.

Model Selection. Following the lasso penalty work of [8] (who invokes [12]), we will use the following criterion for selection of λ in the LSP for a given δ:

$$\begin{aligned} BIC(\lambda) =& (n_x + n_y)\, \|\hat{\boldsymbol{\Sigma}}_x \hat{\boldsymbol{\Delta}} \hat{\boldsymbol{\Sigma}}_y - (\hat{\boldsymbol{\Sigma}}_x - \hat{\boldsymbol{\Sigma}}_y)\|_F \\ &+ \ln(n_x + n_y)\, |\hat{\boldsymbol{\Delta}}|_0 \end{aligned} \tag{16}$$

where $|\boldsymbol{A}|_0$ denotes number of nonzero elements in $\boldsymbol{A}$ and $\hat{\boldsymbol{\Delta}}$ obeys (9). Choose λ to minimize $BIC(\lambda)$. Following [8] we term it BIC (Bayesian information criterion) even though the cost function used is not negative log-likelihood although $\ln(n_x + n_y)\, |\hat{\boldsymbol{\Delta}}|_0$ penalizes over-parametrization as in BIC. It is based on the fact that true $\boldsymbol{\Delta}^*$ satisfies $\boldsymbol{\Sigma}_x^* \boldsymbol{\Delta}^* \boldsymbol{\Sigma}_y^* - (\boldsymbol{\Sigma}_x^* - \boldsymbol{\Sigma}_y^*) = \mathbf{0}$. In our simulations we search over $\lambda \in [\lambda_\ell, \lambda_u]$, where λ_ℓ and λ_u are selected via a heuristic as in [24]. Find the smallest λ, labeled λ_{sm} for which we get a no-edge model; then we set $\lambda_u = \lambda_{sm}/2$ and $\lambda_\ell = \lambda_u/10$.

4. THEORETICAL ANALYSIS

Here we analyze the properties of $\check{\boldsymbol{\Delta}}^{(m)}$, the minimizer of the LLA $\tilde{L}_{LSP}(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y)$ at iteration m, $m \geq 1$, by following the general framework of [25]. In the lasso-penalized approaches of [8, 15, 16], the general method of [26] is used which requires an irrepresentability condition which we do not impose.

Define the true differential edgeset

$$S = \mathcal{E}_{\Delta^*} = \{\{k, \ell\} : |\Delta_{k\ell}^*| > 0\}, \quad s = |S|. \tag{17}$$

In rest of this section we allow p, s and λ to be a functions of sample size n, denoted as p_n, s_n and λ_n, respectively. Lemma 1 follows from [26, Lemma 1] (specific form is based on [24, Lemma 2]).

Lemma 1: Let $\hat{\boldsymbol{\Sigma}}_x$ and $\hat{\boldsymbol{\Sigma}}_y$ be as in (1). Define $n = \min(n_x, n_y)$, $\bar{\sigma}_{xy} = \max\{\max_i \Sigma_{x,ii}^*, \max_i \Sigma_{y,ii}^*\}$ and

$$\begin{aligned} \mathcal{A} =& \max\left\{\|\hat{\boldsymbol{\Sigma}}_x - \boldsymbol{\Sigma}_x^*\|_\infty, \|\hat{\boldsymbol{\Sigma}}_y - \boldsymbol{\Sigma}_y^*\|_\infty\right\} \\ C_0 =& 40\, \bar{\sigma}_{xy} \sqrt{2\big(\tau + \ln(4)/\ln(p_n)\big)}\,. \end{aligned} \tag{18}$$

Then for any $\tau > 2$ and $n > 2(\ln(4) + \tau \ln(p_n))$,

$$P\Big(\mathcal{A} > C_0 \sqrt{\ln(p_n)/n}\Big) \leq 2/p_n^{\tau-2} \quad \bullet \tag{19}$$

Define

$$M_{xy} = \max\{\|\boldsymbol{\Sigma}_x^*\|_\infty, \|\boldsymbol{\Sigma}_y^*\|_\infty\}, \tag{20}$$

$$\phi_{min}^* = \phi_{min}(\boldsymbol{\Sigma}_x^*)\phi_{min}(\boldsymbol{\Sigma}_y^*)\,. \tag{21}$$

Let $\hat{\boldsymbol{\Delta}} = \arg\min_{\boldsymbol{\Delta}} \tilde{L}_{LSP}(\boldsymbol{\Delta}, \hat{\boldsymbol{\Sigma}}_x, \hat{\boldsymbol{\Sigma}}_y)$ for some choice of finite $\bar{\boldsymbol{\Delta}}$ in (7)-(8).

Theorem 1 : For the system model of Sec. 2, assume that initialization satisfies $\|\bar{\boldsymbol{\Delta}}\|_\infty \leq M_{init}$ and let $\bar{M}_{init} = M_{init} + \delta$. If

$$\lambda_n \;\geq\; 2\bar{M}_{init}\big(3M_{xy}\, \|\boldsymbol{\Delta}^*\|_1 + 2\big)C_0 \sqrt{\frac{\ln(p_n)}{n}} \tag{22}$$

$$n = \min(n_x, n_y) > \frac{96 C_0 M_{xy} \bar{M}_{init}^2\, s_n \sqrt{\ln(p_n)}}{\phi_{min}^*\, \delta^2}\,, \tag{23}$$

then with probability $> 1 - 2/p_n^{\tau-2}$, for any $\tau > 2$, we have

$$\|\hat{\boldsymbol{\Delta}} - \boldsymbol{\Delta}^*\|_F \leq \frac{12\lambda_n \sqrt{s_n}}{\phi_{min}^*\, \delta} \qquad \bullet \tag{24}$$

The proof of Theorem 1 is omitted for lack of space. It uses the restricted strong convexity framework of [25] together with a weighted ℓ_1-norm penalty.

Remark 1: Convergence Rate. For any positive-definite matrix $\boldsymbol{A}$, we have $\|\boldsymbol{A}\|_\infty \leq \max_\ell |A_{\ell\ell}| \leq \phi_{max}(\boldsymbol{A})$. Assume that

$$\begin{aligned} 0 <& \beta_{min} \leq \min\{\phi_{min}(\boldsymbol{\Sigma}_y^*), \phi_{min}(\boldsymbol{\Sigma}_x^*)\} \\ &\leq \max\{\phi_{max}(\boldsymbol{\Sigma}_y^*), \phi_{max}(\boldsymbol{\Sigma}_x^*)\} \leq \beta_{max} < \infty \end{aligned} \tag{25}$$

where β_{min} and β_{max} do not depend upon on n (or p_n and s_n). Hence $M_{xy} \leq \max\{\phi_{max}(\boldsymbol{\Sigma}_x^*), \phi_{max}(\boldsymbol{\Sigma}_y^*)\} \leq \beta_{max}$ and

$$\begin{aligned} \|\boldsymbol{\Delta}^*\|_1 \leq& s_n \|\boldsymbol{\Delta}^*\|_\infty \leq s_n\big[\|(\boldsymbol{\Sigma}_y^*)^{-1}\|_\infty + \|(\boldsymbol{\Sigma}_x^*)^{-1}\|_\infty\big] \\ \leq& s_n\big(\phi_{min}^{-1}(\boldsymbol{\Sigma}_y^*) + \phi_{min}^{-1}(\boldsymbol{\Sigma}_x^*)\big) \leq 2s_n/\beta_{min}\,. \end{aligned} \tag{26}$$

Then

$$\lambda_n = C_1 s_n \sqrt{\frac{\ln(p_n)}{n}} \geq \text{ right side of (22)} \tag{27}$$

for some C_1 that does not depend upon n, p_n or s_n. Then we have

$$\|\hat{\boldsymbol{\Delta}} - \boldsymbol{\Delta}^*\|_F \leq C_2\, s_n^{1.5} \sqrt{\ln(p_n)/n} \tag{28}$$

for some C_2 that does not depend upon n, p_n or s_n. Thus, $\|\hat{\boldsymbol{\Delta}} - \boldsymbol{\Delta}^*\|_F = \mathcal{O}_P(s_n^{1.5} \sqrt{\ln(p_n)/n})$. Therefore, for $\|\hat{\boldsymbol{\Delta}} - \boldsymbol{\Delta}^*\|_F \to 0$ as $n \to \infty$, we must have $s_n^{1.5} \sqrt{\ln(p_n)/n} \to 0$. The results in [8] need $s_n^{2.5} \sqrt{\ln(p_n)/n} \to 0$. Recall that $s_n = |S| = |\mathcal{E}_{\Delta^*}|$, number of edges in the differential graph. $\square$

5. SIMULATION EXAMPLE

We consider an Erdös-Rènyi graph where p nodes are connected to each other with probability $p_{er} = 0.5$. In the upper triangular $\mathbf{\Omega}_x$, we set $[\mathbf{\Omega}_x]_{jk} = 0.5^{|j-k|}$ for $j = k = 1, \cdots, p$. For $j \neq k$, if the two nodes are not connected, we have $\Omega_{jk} = 0$, and if nodes j and k are connected, then Ω_{jk} is uniformly distributed over $[-0.4, -0.1] \cup [0.1, 0.4]$. Then add lower triangular elements to make $\mathbf{\Omega}_x$ a symmetric matrix. To generate $\mathbf{\Omega}_y$, we follow [8] and first generate a differential graph with $\mathbf{\Delta} \in \mathbb{R}^{p \times p}$ as an Erdös-Rènyi graph with connection probability $p_{er} = 0.05$ (sparse): if nodes j and k are connected, then Δ_{jk} is set to ± 0.9 with equal probabilities. Then $\mathbf{\Omega}_y = \mathbf{\Omega}_x + \mathbf{\Delta}$. Finally add $\gamma \boldsymbol{I}$ to $\mathbf{\Omega}_y$ and to $\mathbf{\Omega}_x$ and pick γ so that $\mathbf{\Omega}_y$ and $\mathbf{\Omega}_x$ are both positive definite. With $\mathbf{\Phi}_x \mathbf{\Phi}_x^\top = \mathbf{\Omega}_x^{-1}$, we generate $\boldsymbol{x} = \mathbf{\Phi} \boldsymbol{w}$ with $\boldsymbol{w} \in \mathbb{R}^p$ as Gaussian $\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, and similarly for independent $\boldsymbol{y}$. We generate $n = n_x = n_y$ i.i.d. observations for $\boldsymbol{x}$ and $\boldsymbol{y}$, with $p = 100$, $n \in \{100, 300, 800, 1600, 3200\}$.

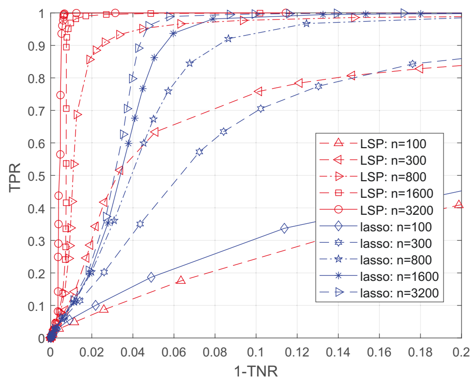

Fig. 1: ROC curves for LSP and Lasso penalties. TPR=true positive rate, TNR=true negative rate

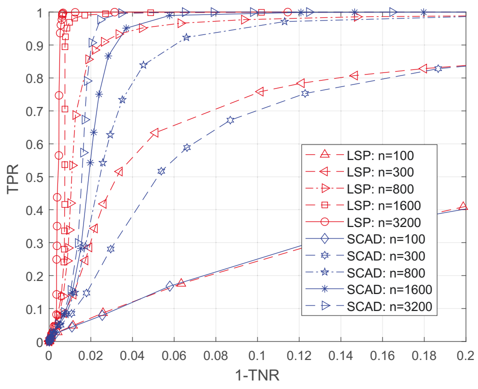

Fig. 2: ROC curves for LSP and SCAD penalties. TPR=true positive rate, TNR=true negative rate

Simulation results based on 100 runs are shown in Figs. 1-4. By changing the penalty parameter λ and determining the resulting edges, we calculated the true positive rate (TPR) and false positive rate 1-TNR (where TNR is the true negative rate) over 100 runs. The receiver operating characteristic (ROC) is shown in Fig. 1 for our LSP-based approach (labeled "LSP") as well as for a lasso-based approach (labeled "lasso"), based on [15]. Fig. 2 shows the ROC for our LSP-based approach (labeled "LSP") as well as for the SCAD-based approach (labeled "SCAD"), based on [21]. It is seen from Figs. 1-2 that our approach outperforms both the convex lasso-based approach and the non-convex SCAD-based approach.

For the results shown in Fig. 3, we picked the λ value (from a grid of λ values) that leads to the highest F_1 score averaged over 100 Monte Carlo runs, for a given method. The resulting F_1 scores are shown in Fig. 3 for varying n, for penalty functions Lasso, LSP and SCAD. It is seen that the LSP-based approach outperforms the other two approaches, lasso (convex penalty) and SCAD (non-convex penalty), and SCAD outperforms lasso, when the F_1 score is the performance measure.

In practice we do not know the ground truth, hence, we cannot pick λ to maximize the F_1 score. In Fig. 4 we show the results based on 100 runs for our approach when BIC parameter selection method (Sec. 3) is applied. Here we show the TPR, 1-TNR and F_1score values along with the $\pm\sigma$ error bars, for varying n. The proposed approach works well both in terms of F_1 score and TPR vs 1-TNR.

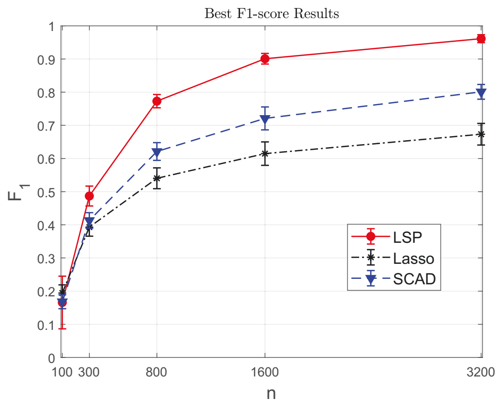

Fig. 3: Optimized F_1-scores for LSP, lasso and SCAD penalty based differential graph estimation methods.

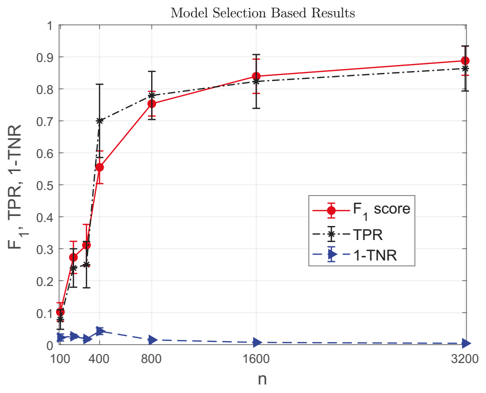

Fig. 4: BIC based results: F_1-scores, TPR and 1-TNR

6. CONCLUSIONS

A log-sum penalized D-trace loss function approach for differential graph learning was presented. An ADMM algorithm was presented to optimize the non-convex objective function. Theoretical analysis establishing consistency of the estimator in high-dimensional settings was performed. We illustrated our approach via a simulation example where the log-sum penalized D-trace loss significantly outperformed the lasso-penalized D-trace loss as well as the smoothly clipped absolute deviation (SCAD) penalized D-trace loss with ROC and/r the F_1 score as the performance metric.

7. REFERENCES

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

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

[3] 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 (Methodological)*, vol. 76, pp. 373-397, 2014.

[4] 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.

[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, pp. 445-488, 2014.

[7] Y. Wu, T. Li, X. Liu and L.| Chen, "Differential network inference via the fused D-trace loss with cross variables," *Electronic J. Statistics*, vol. 14, pp. 1269-1301, 2020.

[8] H. Yuan, R. Xi, C. Chen and M. Deng, "Differential network analysis via lasso penalized D-trace loss," *Biometrika*, vol. 104, pp. 755-770, 2017.

[9] Z. Tang, Z. Yu and C. Wang, "A fast iteraive algorithm for high-dimensional differential network," *Computational Statistics*, vol. 35, pp. 95-109, 2020.

[10] B. Zhao, Y.S. Wang and M. Kolar, "Direct estimation of differential functional graphical models," in *Proc. 33rd Conf. Neural Information Processing Systems (NeurIPS 2019)*, Vancouver, Canada, 2019.

[11] B. Zhao, Y.S. Wang and M. Kolar, "FuDGE: A method to estimate a functional differential graph in a high-dimensional setting," *J. Machine Learning Research*, vol. 23, pp. 1-82, 2022.

[12] S.D. Zhao, T.T. Cai and H. Li, "Direct estimation of differential networks," *Biometrika*, vol. 101, pp. 253-268, June 2014.

[13] E. Belilovsky, G. Varoquaux and M.B. Blaschko, "Hypothesis testing for differences in Gaussian graphical models: Applications to brain connectivity," *Advances in Neural Information Processing Systems (NIPS 2016)*, vol. 29, Dec. 2016.

[14] E.J. Candès, M.B. Wakin and S.P. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *J. Fourier Anal. Appl.*, vol. 14, pp. 877-905, 2008.

[15] B. Jiang, X. Wang and C. Leng, "A direct approach for sparse quadratic discriminant analysis," *J. Machine Learning Research*, vol. 19, pp. 1-37, 2018.

[16] T. Zhang and H. Zou, "Sparse precision matrix estimation via lasso penalized D-trace loss," *Biometrika*, vol. 101, pp. 103-120, 2014.

[17] H. Zou and R. Li, "One-step sparse estimates in nonconcave penalized likelihood models," *Ann. Statist.*, vol. 36, no. 4, pp. 1509-1533, 2008.

[18] C. Lam and J. Fan, "Sparsistency and rates of convergence in large covariance matrix estimation," *Ann. Statist.*, vol. 37, no. 6B, pp. 4254-4278, 2009.

[19] J.K. Tugnait, "Sparse graph learning under Laplacian-related constraints," *IEEE Access*, vol. 9, pp. 151067-151079, 2021.

[20] J.K. Tugnait, "Sparse-group log-sum penalized graphical model learning for time series," in *Proc. 2022 IEEE Intern. Conf. Acoustics, Speech & Signal Processing (ICASSP 2022)*, pp. 5822-5826, Singapore, May 22-27, 2022.

[21] P. Xu and Q. Gu, "Semiparametric differential graph models," in *Proc. 30th Conference on Neural Information Processing Systems (NIPS 2016)*, Barcelona, Spain, 2016.

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

[23] 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.

[24] J.K. Tugnait, "Sparse-group lasso for graph learning from multi-attribute data," *IEEE Trans. Signal Process.*, vol. 69, pp. 1771-1786, 2021. (Corrections, vol. 69, p. 4758, 2021.)

[25] S.N. Negahban, P. Ravikumar, M.J. Wainwright and B. Yu, "A unified framework for high-dimensional analysis of M-estimators with decomposable regularizers," *Statistical Science*, vol. 27, No. 4, pp. 538-557, 2012.

[26] P. Ravikumar, M.J. Wainwright, G. Raskutti and B. Yu, "High-dimensional covariance estimation by minimizing ℓ_1-penalized log-determinant divergence," *Electronic J. Statistics*, vol. 5, pp. 935-980, 2011.