LETTER **Communicated by James Saunderson**

Alternating Direction Methods for Latent Variable Gaussian Graphical Model Selection

Shiqian Ma
sqma@se.cuhk.edu.hk
Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Shatin, N. T., Hong Kong

Lingzhou Xue
lzxue@princeton.edu
Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ 08536, U.S.A.

Hui Zou
zouxx019@umn.edu
School of Statistics, University of Minnesota, Minneapolis, MN 55455, U.S.A.

Chandrasekaran, Parrilo, and Willsky (2012) proposed a convex optimization problem for graphical model selection in the presence of unobserved variables. This convex optimization problem aims to estimate an inverse covariance matrix that can be decomposed into a sparse matrix minus a low-rank matrix from sample data. Solving this convex optimization problem is very challenging, especially for large problems. In this letter, we propose two alternating direction methods for solving this problem. The first method is to apply the classic alternating direction method of multipliers to solve the problem as a consensus problem. The second method is a proximal gradient-based alternating-direction method of multipliers. Our methods take advantage of the special structure of the problem and thus can solve large problems very efficiently. A global convergence result is established for the proposed methods. Numerical results on both synthetic data and gene expression data show that our methods usually solve problems with 1 million variables in 1 to 2 minutes and are usually 5 to 35 times faster than a state-of-the-art Newton-CG proximal point algorithm.

1 Introduction

In this letter, we consider two efficient algorithms with a global convergence guarantee for solving the convex program for latent variable graphical model selection proposed by Chandrasekaran, Parrilo, and Willsky (2012). Graphical model selection is closely related to the inverse covariance matrix

Neural Computation **25, 2172–2198** (2013)
doi:10.1162/NECO_a_00379

estimation problem, which is of fundamental importance in multivariate statistical inference. In particular, when data $X = (X_1, \ldots, X_p)'$ follow a p-dimensional joint normal distribution with some unknown covariance matrix Σ, the precision matrix $\Theta = \Sigma^{-1}$ can be directly translated into a gaussian graphical model. The zero entries in the precision matrix $\Theta = (\theta_{ij})_{1 \le i,j \le p}$ precisely capture the desired conditional independencies in the gaussian graphical model (Lauritzen, 1996; Edwards, 2000): $\theta_{ij} = 0$ if and only if $X_i \perp\!\!\!\perp X_j | X_{-(i,j)}$. gaussian graphical models have been successfully used to explore complex systems consisting of Gaussian random variables in many research fields, including gene expression genomics (Friedman, 2004; Wille et al., 2004), macroeconomics determinant studies (Dobra, Eicher, & Lenkoski, 2009), and social studies (Ahmed & Xing, 2009; Kolar, Song, Ahmed, & Xing, 2010).

Massive high-dimensional data are now being routinely generated with rapid advances of modern high-throughput technology (e.g., microarray and functional magnetic resonance imaging). Estimation of a sparse graphical model has become increasingly important in the high-dimensional regime, and some well-developed penalization techniques have received considerable attention in the statistical literature. Meinshausen and Bühlmann (2006) were the first to study the high-dimensional sparse graphical model selection problem and proposed the neighborhood penalized regression scheme, which performs the lasso (Tibshirani, 1996) to fit each neighborhood regression and summarizes the sparsity pattern by aggregation via union or intersection. Peng, Wang, Zhou, and Zhu (2009) proposed the joint sparse regression model to jointly estimate all neighborhood lasso penalized regressions. Yuan (2010) considered the Dantzig selector (Candès & Tao, 2007) as an alternative to the lasso in each neighborhood regression. Cai, Liu, and Luo (2011) proposed a constrained ℓ_1 minimization estimator called CLIME for estimating sparse precision matrices and established rates of convergence under both the entrywise ℓ_∞ norm and the Frobenius norm. Computationally, CLIME can be further decomposed into a series of vector minimization problems. Recently, Xue and Zou (2012) and Liu, Han, Yuan, Lafferty, and Wasserman (2012) proposed the rank-based neighborhood Dantzig selector and the rank-based CLIME to account for the nonnormality in the graphical modeling.

The ℓ_1-penalized maximum normal likelihood method (Yuan & Lin, 2007) is another popular method for graphical model selection. Rothman, Bickel, Levina, and Zhu (2008) established its rate of convergence under the Frobenius norm. Under the irrepresentable condition, Ravikumar, Wainwright, Raskutti, and Yu (2008) obtained its convergence rates under the entrywise ℓ_∞ norm and the spectral norm. Xue and Zou (2012) and Liu et al. (2012) considered the regularized rank scheme under the semiparametric gaussian copula models. Define the entrywise ℓ_1 norm of S as the sum of absolute values of the entries of S: $\|S\|_1 := \sum_{ij} |S_{ij}|$. For a given sample

covariance matrix $\hat{\Sigma} \in \mathbb{R}^{p\times p}$, the ℓ_1-penalized maximum normal likelihood estimation can be formulated as the following convex optimization problem.

$$\min_S \langle S, \hat{\Sigma}\rangle - \log\det S + \rho\|S\|_1, \tag{1.1}$$

where the first part, $\langle S, \hat{\Sigma}\rangle - \log\det S$, gives the negative log-likelihood function of S, and the entrywise ℓ_1 norm, $\|S\|_1$, is used to promote the sparsity of the resulting matrix. Note that in the literature, the ℓ_1-penalized maximum normal likelihood usually uses the so-called one-off absolute penalty $\|S\|_{1,\text{off}} := \sum_{i\neq j}|S_{ij}|$. However, $\|S\|_1$ and $\|S\|_{1,\text{off}}$ cause no difference when using our algorithm. Chandrasekaran et al. (2012) have used $\|S\|_1$ in defining their convex optimization problem, and hence we follow their convention in this letter.

The gaussian graphical model selection methods were proposed under the ideal setting without missing variables. Chandrasekaran et al. (2012) considered a more realistic scenario where the full data consist of both observed variables and missing (hidden) variables. Let $X_{p\times 1}$ be the observed variables. Suppose that there are some hidden variables $Y_{r\times 1}$ ($r \ll p$) such that (X, Y) jointly follow a multivariate normal distribution. Denote the covariance matrix by $\Sigma_{(X,Y)}$ and the precision matrix by $\Theta_{(X,Y)}$. Then we can write $\Sigma_{(X,Y)} = [\Sigma_X, \Sigma_{XY}; \Sigma_{YX}, \Sigma_Y]$ and $\Theta_{(X,Y)} = [\Theta_X, \Theta_{XY}; \Theta_{YX}, \Theta_Y]$. Given the hidden variables Y, the conditional precision matrix of observed variables, Θ_X, is sparse for a sparse graphical model. However, the marginal precision matrix of observed variables, $\Sigma_X^{-1} = \Theta_X - \Theta_{XY}\Theta_Y^{-1}\Theta_{YX}$, might not be a sparse matrix but a difference between the sparse term Θ_X and the low-rank term $\Theta_{XY}\Theta_Y^{-1}\Theta_{YX}$. The problem of interest is to recover the sparse conditional matrix Θ_X and the low-rank term $\Theta_{XY}\Theta_Y^{-1}\Theta_{YX}$ based on observed variables X. Chandrasekaran et al. (2012) accomplished this goal by solving a convex optimization problem under the assumption that $\Sigma_X^{-1} = S - L$ for some sparse matrix S and low-rank matrix L. The low-rank assumption on L holds naturally since r is much less than p. Motivated by the success of the convex relaxation for the rank-minimization problem, Chandrasekaran et al. (2012) introduced a regularized maximum normal likelihood decomposition framework, the latent variable graphical model selection (LVGLASSO):

$$\begin{aligned}&\min_{S,L} \langle S - L, \hat{\Sigma}_X\rangle - \log\det(S - L) + \alpha\|S\|_1 + \beta\mathrm{Tr}(L),\\ &\quad \text{s.t. } S - L \succ 0, L \succeq 0,\end{aligned} \tag{1.2}$$

where $\hat{\Sigma}_X$ is the sample covariance matrix of X and $\mathrm{Tr}(L)$ denotes the trace of matrix L. In the high-dimensional setting, Chandrasekaran et al. (2012)

established the consistency theory for equation 1.2 concerning its recovery of the support and sign pattern of S and the rank of L.

Solving the convex optimization problem 1.2 is very challenging, especially in the high-dimensional setting. Chandrasekaran et al. (2012) considered equation 1.2 as a log-determinant semidefinite programming (SDP) problem and used a Newton–CG–based proximal point algorithm (LogdetPPA) proposed by Wang, Sun, and Toh (2010) to solve it. However, LogdetPPA does not take advantage of the special structure of the problem, and we argue that it is inefficient for solving large-scale problems. To illustrate our point, let us consider the special case of equation 1.2 with $L = 0$, and then the latent variable graphical model selection 1.2 exactly reduces to the gaussian graphical model selection, equation 1.1. Note that equation 1.1 can be rewritten as

$$\min_{S} \max_{\|W\|_\infty \le \rho} -\log\det S + \langle \hat{\Sigma}_X + W, S\rangle,$$

where $\|W\|_\infty$ is the largest absolute value of the entries of W. The dual problem of equation 1.1 can be obtained by exchanging the order of max and min,

$$\max_{\|W\|_\infty \le \rho} \min_{S} -\log\det S + \langle \hat{\Sigma}_X + W, S\rangle,$$

which is equivalent to

$$\max_{W} \{\log\det W + p : \|W - \hat{\Sigma}_X\|_\infty \le \rho\}. \tag{1.3}$$

Both the primal and the dual graphical lasso problems, 1.1 and 1.3, can be viewed as semidefinite programming problems and solved via interior point methods (IPMs) in polynomial time (Boyd & Vandenberghe, 2004). However, the per-iteration computational cost and memory requirements of an IPM are prohibitively high for equations 1.1 and 1.3, especially when the size of the matrix is large. Customized SDP-based methods such as the ones studied in Wang et al. (2010) and Li and Toh (2010) require a reformulation of the problem that increases the size of the problem and thus makes them impractical for solving large-scale problems. Therefore, most of the methods developed for solving equations 1.1 and 1.3 are first-order methods. These methods include block-coordinate-descent–type methods (Banerjee, El Ghaoui, & d'Aspremont, 2008; Friedman, Hastie, & Tibshirani, 2008; Scheinberg & Rish, 2009; Witten, Friedman, & Simon, 2011), projected gradient method (Duchi, Gould, & Koller, 2008), and variants of Nesterov's accelerated method (D'Aspremont, Banerjee, & El Ghaoui, 2008; Lu, 2009). Recently, alternating direction methods have been applied to solve equation

1.1 and shown to be very effective (Yuan, 2012; Scheinberg, Ma, & Goldfarb, 2010).

In this letter, we propose two alternating direction–type methods to solve the latent variable graphical model selection. The first method is to apply the alternating direction method of multipliers to solve this problem. This is due to the fact that the latent variable graphical model selection can be seen as a special case of the consensus problem discussed in Boyd, Parikh, Chu, Peleato, and Eckstein (2011). The second method we propose is an alternating direction method with proximal gradient steps. To apply the second method, we first group the variables into two blocks and then apply the alternating direction method with one of the subproblems being solved inexactly by taking a proximal gradient step. Our methods take advantage of the special structure of the problem and thus can solve large problems very efficiently. Although the convergence results of the proposed methods are not very different from the existing results for alternating direction–type methods, we still include the convergence proof for the second method in the appendix for completeness. We apply the proposed methods to solving problems from both synthetic data, and gene expression data and show that our methods outperform the state-of-the-art Newton-CG proximal point algorithm LogdetPPA significantly on both accuracy and CPU times.

The rest of this letter is organized as follows. In section 2, we give some preliminaries on alternating direction method of multipliers and proximal mappings. In section 3, we propose solving LVGLASSO, equation 1.2, as a consensus problem using the classical alternating direction method of multipliers. We propose the proximal gradient based alternating direction method for solving equation 1.2 in section 4. In section 5, we apply our proposed alternating direction method to solving equation 1.2 using both synthetic data and gene expression data. Section 6 offers concluding remarks.

2 Preliminaries

Problem 1.2 can be rewritten in the following equivalent form by introducing a new variable R:

$$\begin{aligned} &\min \ \langle R, \hat{\Sigma}_X \rangle - \log\det R + \alpha \|S\|_1 + \beta \mathrm{Tr}(L) \\ &\text{s.t.} \ \ R = S - L, R \succ 0, L \succeq 0, \end{aligned} \tag{2.1}$$

which can be further reduced to

$$\begin{aligned} &\min \ \langle R, \hat{\Sigma}_X \rangle - \log\det R + \alpha \|S\|_1 + \beta \mathrm{Tr}(L) + \mathcal{I}(L \succeq 0), \\ &\quad \text{s.t.} \ \ R - S + L = 0, \end{aligned} \tag{2.2}$$

where the indicator function $\mathcal{I}(L \succeq 0)$ is defined as

$$\mathcal{I}(L \succeq 0) := \begin{cases} 0, & \text{if } L \succeq 0 \\ +\infty, & \text{otherwise} \end{cases}. \tag{2.3}$$

Note that we have dropped the constraint $R \succ 0$ since it is already implicitly imposed by the $\log\det R$ function.

Now since the objective function involves three separable convex functions and the constraint is simply linear, Problem 2.2 is suitable for the alternating direction method of multipliers (ADMM). ADMM is closely related to the Douglas-Rachford and Peaceman-Rachford operator-splitting methods for finding zero of the sum of two monotone operators that have been studied extensively (Douglas & Rachford, 1956; Peaceman & Rachford, 1955; Lions & Mercier, 1979; Eckstein, 1989; Eckstein & Bertsekas, 1992; Combettes & Pesquet, 2007; Combettes & Wajs, 2005). ADMM has been revisited recently due to its success in the emerging applications of structured convex optimization problems arising from image processing, compressed sensing, machine learning, semidefinite programming and statistics (see Glowinski & Le Tallec, 1989; Gabay, 1983; Wang, Yang, Yin, & Zhang, 2008; Yang & Zhang, 2011; Goldstein & Osher, 2009; Qin, Goldfarb, & Ma, 2011; Yuan, 2012; Scheinberg et al., 2010; Goldfarb & Ma, 2012; Goldfarb, Ma, & Scheinberg, in press; Malick, Povh, Rendl, & Wiegele, 2009; Wen, Goldfarb, & Yin, 2010; Boyd et al., 2011; Parikh & Boyd, 2011; Ma, 2011; Xue, Ma, & Zou, 2012). Problem 2.2 is suitable for alternating direction methods because the three convex functions involved in the objective function,

$$f(R) := \langle R, \hat{\Sigma}_X \rangle - \log\det R, \tag{2.4}$$

$$g(S) := \alpha \|S\|_1, \tag{2.5}$$

$$h(L) := \beta \mathrm{Tr}(L) + \mathcal{I}(L \succeq 0), \tag{2.6}$$

have easy proximal mappings. Note that the proximal mapping of function $c : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$ for given $\xi > 0$ and $Z \in \mathbb{R}^{m\times n}$ is defined as

$$\mathrm{Prox}(c, \xi, Z) := \underset{X\in\mathbb{R}^{m\times n}}{\mathrm{argmin}} \ \frac{1}{2\xi}\|X - Z\|_F^2 + c(X). \tag{2.7}$$

The proximal mapping of $f(R)$ defined in equation 2.4 is

$$\mathrm{Prox}(f, \xi, Z) := \underset{R}{\mathrm{argmin}} \ \frac{1}{2\xi}\|R - Z\|_F^2 + \langle R, \hat{\Sigma}_X \rangle - \log\det R. \tag{2.8}$$

The first-order optimality conditions of equation 2.8 are given by

$$R + \xi\hat{\Sigma}_X - Z - \xi R^{-1} = 0. \tag{2.9}$$

It is easy to verify that

$$R := U\text{diag}(\gamma)U^\top \tag{2.10}$$

satisfies equation 2.9 and thus gives the optimal solution of equation 2.8, where $U\text{diag}(\sigma)U^\top$ is the eigenvalue decomposition of matrix $\xi\hat{\Sigma}_X - Z$ and

$$\gamma_i = \left(-\sigma_i + \sqrt{\sigma_i^2 + 4\xi}\right)/2, \quad \forall i = 1, \ldots, p. \tag{2.11}$$

Note that equation 2.11 guarantees that the solution of equation 2.8 given by equation 2.10 is a positive-definite matrix. The proximal mapping of $g(S)$ defined in equation 2.5 is

$$\text{Prox}(g, \xi, Z) := \underset{S}{\text{argmin}} \ \frac{1}{2\xi}\|S - Z\|_F^2 + \alpha\|S\|_1. \tag{2.12}$$

It is well known that equation 2.12 has a closed-form solution that is given by the ℓ_1 shrinkage operation,

$$S := \text{Shrink}(Z, \alpha\xi),$$

where $\text{Shrink}(\cdot, \cdot)$ is defined as

$$[\text{Shrink}(Z, \tau)]_{ij} := \begin{cases} Z_{ij} - \tau, & \text{if } Z_{ij} > \tau \\ Z_{ij} + \tau, & \text{if } Z_{ij} < -\tau \\ 0, & \text{if } -\tau \le Z_{ij} \le \tau \end{cases}. \tag{2.13}$$

The proximal mapping of $h(L)$ defined in equation 2.6 is

$$\text{Prox}(h, \xi, Z) := \underset{L}{\text{argmin}} \ \frac{1}{2\xi}\|L - Z\|_F^2 + \beta\text{Tr}(L) + \mathcal{I}(L \succeq 0). \tag{2.14}$$

It is easy to verify that the solution of equation 2.14 is given by

$$L := U\text{diag}(\gamma)U^\top, \tag{2.15}$$

where $Z = U\text{diag}(\sigma)U^\top$ is the eigenvalue decomposition of Z and γ is given by

$$\gamma_i := \max\{\sigma_i - \xi\beta, 0\}, \quad i = 1, \ldots, p. \tag{2.16}$$

Note that equation 2.16 guarantees that L, given in equation 2.15, is a positive-semidefinite matrix.

The discussions above suggest that the following natural ADMM for solving equation 2.2 is efficient:

$$\begin{cases} R^{k+1} := \underset{R}{\operatorname{argmin}}\ \mathcal{L}_\mu(R, S^k, L^k; \Lambda^k) \\ S^{k+1} := \underset{S}{\operatorname{argmin}}\ \mathcal{L}_\mu(R^{k+1}, S, L^k; \Lambda^k) \\ L^{k+1} := \underset{L}{\operatorname{argmin}}\ \mathcal{L}_\mu(R^{k+1}, S^{k+1}, L; \Lambda^k) \\ \Lambda^{k+1} := \Lambda^k - (R^{k+1} - S^{k+1} + L^{k+1})/\mu, \end{cases} \tag{2.17}$$

where the augmented Lagrangian function is defined as

$$\begin{aligned} \mathcal{L}_\mu(R, S, L; \Lambda) := \langle R, \hat{\Sigma}_X \rangle - \log\det R + \alpha\|S\|_1 + \beta \mathrm{Tr}(L) + \mathcal{I}(L \succeq 0) \\ - \langle \Lambda, R - S + L \rangle + \frac{1}{2\mu}\|R - S + L\|_F^2, \end{aligned} \tag{2.18}$$

with Λ as the Lagrange multiplier and $\mu > 0$ as the penalty parameter. Note that the three subproblems in equation 2.17 correspond to the proximal mappings of f, g, and h defined in equations 2.4, 2.5, and 2.6, respectively. Thus they are all easy to solve. However, the global convergence of ADMM, equation 2.17, with three blocks of variables was ambiguous. Recently, Hong and Luo (2012) showed that a variant of equation 2.17, which uses a different step size depending on some error-bound parameter of the objective function to update the Lagrange multiplier Λ, is globally convergent. It should be noted, however, that the error-bound condition required in Hong and Luo (2012) is strong and only a few classes of convex function are known to satisfy this condition.

3 ADMM for Solving Equation 2.2 as a Consensus Problem

Problem 2.2 can be rewritten as a convex minimization problem with two blocks of variables and two separable functions as follows:

$$\begin{aligned} &\min\ \phi(Z) + \psi(\tilde{Z}), \\ &\text{s.t.}\ \ Z - \tilde{Z} = 0, \end{aligned} \tag{3.1}$$

where $Z = (R, S, L)$, $\tilde{Z} = (\tilde{R}, \tilde{S}, \tilde{L})$, and

$$\phi(Z) := f(R) + g(S) + h(L), \quad \psi(\tilde{Z}) = \mathcal{I}(\tilde{R} - \tilde{S} + \tilde{L} = 0),$$

with f, g and h defined in equations 2.4, 2.5, and 2.6, respectively. The ADMM applied to solving equation 3.1 can be described as follows:

$$\begin{cases} Z^{k+1} := \underset{Z}{\operatorname{argmin}}\ \phi(Z) - \langle \Lambda^k, Z - \tilde{Z}^k \rangle + \dfrac{1}{2\mu}\|Z - \tilde{Z}^k\|_F^2, \\ \tilde{Z}^{k+1} := \underset{\tilde{Z}}{\operatorname{argmin}}\ \psi(\tilde{Z}) - \langle \Lambda^k, Z^{k+1} - \tilde{Z} \rangle + \dfrac{1}{2\mu}\|Z^{k+1} - \tilde{Z}\|_F^2, \\ \Lambda^{k+1} := \Lambda^k - (Z^{k+1} - \tilde{Z}^{k+1})/\mu, \end{cases} \tag{3.2}$$

where Λ is the Lagrange multiplier associated with the equality constraint. The two subproblems in equation 3.2 are both easy to solve. The first subproblem in it can be equivalently written as

$$Z^{k+1} := \underset{Z}{\operatorname{argmin}}\ \phi(Z) + \frac{1}{2\mu}\|Z - \tilde{Z}^k - \mu\Lambda^k\|_F^2, \tag{3.3}$$

by discarding a constant term $-\frac{\mu}{2}\|\Lambda^k\|_F^2$ in the objective function. By partitioning the matrix $W^k := \tilde{Z}^k + \mu\Lambda^k$ into three blocks $W^k = (W_R^k, W_S^k, W_L^k)$ in the same form as $Z = (R, S, L)$, and partitioning $\Lambda^k = (\Lambda_R^k, \Lambda_S^k, \Lambda_L^k)$, equation 3.3 is actually decomposable into three problems:

$$R^{k+1} := \underset{R}{\operatorname{argmin}}\ f(R) + \frac{1}{2\mu}\|R - W_R^k\|_F^2, \tag{3.4}$$

$$S^{k+1} := \underset{S}{\operatorname{argmin}}\ g(S) + \frac{1}{2\mu}\|S - W_S^k\|_F^2, \tag{3.5}$$

$$L^{k+1} := \underset{L}{\operatorname{argmin}}\ h(L) + \frac{1}{2\mu}\|L - W_L^k\|_F^2. \tag{3.6}$$

Obviously these three problems correspond to applying the proximal mappings of f, g, and h, respectively, to the three components of W^k. Thus they are easy to solve. As for the second subproblem in equation 3.2, by noticing $\tilde{Z} = (\tilde{R}, \tilde{S}, \tilde{L})$, it is equivalent to

$$\min -\langle \Lambda^k, Z^{k+1} - \tilde{Z} \rangle + \frac{1}{2\mu}\|Z^{k+1} - \tilde{Z}\|_F^2, \ \text{ s.t., } \tilde{R} - \tilde{S} + \tilde{L} = 0. \tag{3.7}$$

By partitioning the matrix $T^k := Z^{k+1} - \mu\Lambda^k$ into three blocks $T^k = (T_R^k, T_S^k, T_L^k)$ in the same form as $\tilde{Z} = (\tilde{R}, \tilde{S}, \tilde{L})$, and by discarding a constant term $\frac{\mu}{2}\|\Lambda^k\|_F^2$ in the objective function of equation 3.7, this problem can be rewritten equivalently as

$$\begin{aligned} &\min\ \frac{1}{2}\|(\tilde{R}, \tilde{S}, \tilde{L}) - (T_R^k, T_S^k, T_L^k)\|_F^2 \\ &\text{s.t. } \ \tilde{R} - \tilde{S} + \tilde{L} = 0. \end{aligned} \tag{3.8}$$

Algorithm 1: ADMM for Solving Equation 3.1.

1: **for** k=0,1,... **do**
2: Update R^{k+1} by solving equation 3.4.
3: Update S^{k+1} by solving equation 3.5.
4: Update L^{k+1} by solving equation 3.6.
5: Update $\tilde{R}^{k+1}$, $\tilde{S}^{k+1}$ and $\tilde{L}^{k+1}$ by equation 3.11
6: Update $\Lambda^{k+1} := \Lambda^k - (Z^{k+1} - \tilde{Z}^{k+1})/\mu$, where $Z^{k+1} := (R^{k+1}, S^{k+1}, L^{k+1})$ and $\tilde{Z}^{k+1} := (\tilde{R}^{k+1}, \tilde{S}^{k+1}, \tilde{L}^{k+1})$
7: **end for**

The first-order optimality conditions of equation 3.8 are given by

$$(\tilde{R}, \tilde{S}, \tilde{L}) - (T_R^k, T_S^k, T_L^k) - (\Gamma, -\Gamma, \Gamma) = 0, \tag{3.9}$$

where Γ is the Lagrange multiplier associated with equation 3.8. Thus, we get

$$\tilde{R} = T_R^k + \Gamma, \quad \tilde{S} = T_S^k - \Gamma, \quad \tilde{L} = T_L^k + \Gamma.$$

Substituting them into the equality constraint in equation 3.8, we get

$$\Gamma = -(T_R^k - T_S^k + T_L^k)/3. \tag{3.10}$$

By substituting equation 3.10 into 3.9, we get the solution to equation 3.8:

$$\begin{aligned} &\tilde{R} = T_R^k - (T_R^k - T_S^k + T_L^k)/3, \quad \tilde{S} = T_S^k + (T_R^k - T_S^k + T_L^k)/3, \\ &\tilde{L} = T_L^k - (T_R^k - T_S^k + T_L^k)/3. \end{aligned} \tag{3.11}$$

The ADMM, equation 3.2, for solving equation 3.1 can be summarized as algorithm 1.

The ADMM solves problem 3.1 with two blocks of variables. It can be seen as a special case of the consensus problem discussed in Boyd et al. (2011). The global convergence result of equation 3.2 has also been well studied in the literature (see Eckstein, 1989; Eckstein & Bertsekas, 1992).

4 A Proximal Gradient–Based Alternating Direction Method

In this section, we propose another alternating direction type method to solve equation 2.2. In section 3, we managed to reduce the original problem with three blocks of variables, equation 2.2, to a new problem with two

blocks of variables equation 3.1. As a result, we can use ADMM for solving problems with two blocks of variables, whose convergence has been well studied. Another way to reduce problem 2.2 into a problem with two blocks of variables is to group two variables (say, S and L) as one variable. This leads to the new equivalent form of equation 2.2:

$$\begin{aligned} &\min\ f(R) + \varphi(W) \\ &\text{s.t.}\ \ R - [I, -I]W = 0, \end{aligned} \tag{4.1}$$

where $W = (S; L)$ and $\varphi(W) = g(S) + h(L)$. Now the ADMM for solving equation 4.1 can be described as

$$\begin{cases} R^{k+1} := \underset{R}{\operatorname{argmin}}\ f(R) - \langle \Lambda^k, R - [I, -I]W^k \rangle + \dfrac{1}{2\mu}\|R - [I, -I]W^k\|_F^2 \\ W^{k+1} := \underset{W}{\operatorname{argmin}}\ \varphi(W) - \langle \Lambda^k, R^{k+1} - [I, -I]W \rangle \\ \qquad\qquad + \dfrac{1}{2\mu}\|R^{k+1} - [I, -I]W\|_F^2 \\ \Lambda^{k+1} := \Lambda^k - (R^{k+1} - [I, -I]W^{k+1})/\mu, \end{cases} \tag{4.2}$$

where Λ is the Lagrange multiplier associated with the equality constraint and $\mu > 0$ is a penalty parameter. The first subproblem in equation 4.2 is still easy, and it corresponds to the proximal mapping of function f. However, the second subproblem is not easy, because the two parts of $W = (S; L)$ are coupled together in the quadratic penalty term. To overcome this difficulty, we solve the second subproblem inexactly by one step of a proximal gradient method. Note that the second subproblem can be equivalently written as

$$W^{k+1} := \underset{W}{\operatorname{argmin}}\ \varphi(W) + \frac{1}{2\mu}\left\|R^{k+1} - [I, -I]W - \mu\Lambda^k\right\|_F^2 - \frac{\mu}{2}\left\|\Lambda^k\right\|_F^2. \tag{4.3}$$

One step of proximal gradient method solves the following problem,

$$\min_W \varphi(W) + \frac{1}{2\mu\tau}\left\|W - \left(W^k + \tau\begin{pmatrix} I \\ -I \end{pmatrix}(R^{k+1} - [I, -I]W^k - \mu\Lambda^k)\right)\right\|_F^2, \tag{4.4}$$

where $\tau > 0$ is a step size for the proximal gradient step. Since the two parts of $W = (S; L)$ are separable in the quadratic part, equation 4.4 is

Algorithm 2: A Proximal Gradient–Based Alternating Direction Method.

1: **for** k=0,1,... **do**

2: Solve $R^{k+1} := \operatorname{argmin}_R f(R) - \langle \Lambda^k, R - S^k + L^k \rangle + \frac{1}{2\mu}\|R - S^k + L^k\|_F^2$ by computing the proximal mapping of f: $\operatorname{Prox}(f, \mu, S^k - L^k + \mu\Lambda^k)$ as defined in equation 2.8.

3: Solve $S^{k+1} := \operatorname{argmin}_S g(S) + \frac{1}{2\mu\tau}\|S - (S^k + \tau G_R^k)\|_F^2$ by computing the proximal mapping of g: $\operatorname{Prox}(g, \mu\tau, S^k + \tau G_R^k)$ as defined in equation 2.12.

4: Solve $L^{k+1} := \operatorname{argmin}_L h(L) + \frac{1}{2\mu\tau}\|L - (L^k - \tau G_R^k)\|_F^2$ by computing the proximal mapping of h: $\operatorname{Prox}(h, \mu\tau, L^k - \tau G_R^k)$ as defined in equation 2.14.

5: $\Lambda^{k+1} := \Lambda^k - (R^{k+1} - S^{k+1} + L^{k+1})/\mu$

6: **end for**

decomposable to two problems,

$$\min_S g(S) + \frac{1}{2\mu\tau}\|S - (S^k + \tau G_R^k)\|_F^2 \tag{4.5}$$

and

$$\min_L h(L) + \frac{1}{2\mu\tau}\|L - (L^k - \tau G_R^k)\|_F^2, \tag{4.6}$$

where $G_R^k = R^{k+1} - S^k + L^k - \mu\Lambda^k$. Both equations 4.5 and 4.6 are easy to solve as they correspond to the proximal mappings of g and h, respectively. Thus, our proximal gradient–based alternating-direction method (PGADM) can be summarized as algorithm 2.

Remark 1. The idea of incorporating proximal step into the alternating direction method of multipliers has been suggested by Eckstein (1994) and Chen and Teboulle (1994). This idea has then been generalized by He, Liao, Han, and Yang (2002) to allow varying penalty and proximal parameters. Recently this technique has been used for sparse and low-rank optimization problems (see Yang & Zhang, 2011; Tao & Yuan, 2011). More recently, some convergence properties of alternating direction methods with proximal gradient steps have been studied by Deng and Yin (2012), Hong and Luo (2012), Fazel, Pong, Sun, and Tseng (2012), and Ma (2012). For completeness, we include a global convergence proof for algorithm 2 in the appendix.

Remark 2. In algorithm 2, we grouped S and L as one block of variables. We also implemented the other two ways of grouping the variables: group R and S as one block of variable and group R and L as one block of variable. We

found from the numerical experiments that these two alternatives yielded similar practical performance as algorithm 2.

Remark 3. If we use the one-off absolute penalty $\|S\|_{1,\text{off}} := \sum_{i\neq j} |S_{ij}|$ to replace $\|S\|_1$, our algorithm basically remains the same except that we modify Shrink$(\cdot,\cdot)$ as follows

$$[\text{Shrink}(Z,\tau)]_{ij} := \begin{cases} Z_{ii}, & \text{if } i = j \\ Z_{ij} - \tau, & \text{if } i \neq j \text{ and } Z_{ij} > \tau \\ Z_{ij} + \tau, & \text{if } i \neq j \text{ and } Z_{ij} < -\tau \\ 0, & \text{if } i \neq j \text{ and } -\tau \leq Z_{ij} \leq \tau. \end{cases} \tag{4.7}$$

5 Numerical Experiments

In this section, we present numerical results on both synthetic and real data to demonstrate the efficiency of the proposed methods: ADMM (algorithm 1) and PGADM (algorithm 2). Our codes were written in Matlab. All numerical experiments were run in Matlab 7.12.0 on a laptop with Intel Core I5 2.5 GHz CPU and 4 GB of RAM.

We first compared ADMM (algorithm 1) with PGADM (algorithm 2) on some synthetic problems. We compared them using two different ways of choosing μ. One set of comparisons used a fixed $\mu = 10$, and the other set used a continuation scheme to dynamically change μ. The continuation scheme we used was to set the initial value of μ as the size of the matrix p and then multiply μ by 1/4 after every 10 iterations.

We then compared the performance of PGADM (with continuation on μ) with LogdetPPA proposed by Wang et al. (2010) and used in Chandrasekaran et al. (2012) for solving equation 2.2.

5.1 Comparison of ADMM and PGADM on Synthetic Data.

We observed from the numerical experiments that the step size τ of the proximal gradient step in PGADM (algorithm 2) can be slightly larger than 1/2 and the algorithm produced very good results. We thus chose the step size τ to be 0.6 in our experiments.

We randomly created test problems using a procedure proposed by Scheinberg and Rish (2009) and Scheinberg et al. (2010) for the classical graphical lasso problems. Similar procedures were used by Wang et al. (2010) and Li and Toh (2010). For a given number of observed variables p and a given number of latent variables r, we first created a sparse matrix $U \in \mathbb{R}^{(p+r)\times(p+r)}$ with sparsity around 10% (i.e., 10% of the entries are nonzeros). The nonzero entries were set to -1 or 1 with equal probability. Then we computed $K := (U * U^\top)^{-1}$ as the true precision matrix. We then chose the submatrix of K, $\hat{S} := K(1:p, 1:p)$ as the ground truth matrix

of the sparse matrix S and chose $\hat{L} := K(1:p, p+1:p+r)K(p+1:p+r, p+1:p+r)^{-1}K(p+1:p+r, 1:p)$ as the ground truth matrix of the low-rank matrix L. We then drew $N = 5p$ i.i.d. vectors, $Y_1, \ldots, Y_N$, from the gaussian distribution $\mathcal{N}(\mathbf{0}, (\hat{S} - \hat{L})^{-1})$ by using the *mvnrnd* function in Matlab, and computed a sample covariance matrix of the observed variables $\Sigma_X := \frac{1}{N}\sum_{i=1}^{N} Y_i Y_i^\top$.

We computed the relative infeasibility of the sequence (R^k, S^k, L^k) using

$$\text{infeas} := \frac{\|R^k - S^k + L^k\|_F}{\max\{1, \|R^k\|_F, \|S^k\|_F, \|L^k\|_F\}}. \tag{5.1}$$

To compare the performance of ADMM and PGADM, we first ran ADMM until infeas $< 10^{-4}$ or the relative difference of the objective function values in successive two iterations was less than 10^{-6}. We denoted the solution generated according to this stopping criterion as $[\hat{R}, \hat{S}, \hat{L}]$ and denoted the corresponding objective function value as $\hat{F}$. We believe that $[\hat{R}, \hat{S}, \hat{L}]$ is a relatively accurate solution that can be produced by ADMM, because either its infeasibility is very small or ADMM cannot improve it too much. We then ran both ADMM and PGADM until one of the following stopping criteria held:

$$\text{infeas} < 10^{-4} \quad \text{and} \quad F^k < \hat{F}$$

or

$$\text{infeas} < 10^{-4} \quad \text{and} \quad \frac{\|[R^k, S^k, L^k] - [\hat{R}, \hat{S}, \hat{L}]\|_F}{\|[\hat{R}, \hat{S}, \hat{L}]\|_F} < 10^{-4},$$

where $[R^k, S^k, L^k]$ is the solution produced by ADMM or PGADM at the kth iteration and F^k is the corresponding objective function value. In other words, we terminated ADMM and PGADM when the infeasibility was less than 10^{-4} and the objective value was smaller than $\hat{F}$, or the relative difference of the solution with $[\hat{R}, \hat{S}, \hat{L}]$ was less than 10^{-4}.

The number of iterations (iter), CPU times (cpu), infeas, and objective function values (obj) for both ADMM and PGADM were reported in Table 1. From Table 1, we see that for fixed $\mu = 10$, ADMM was faster than PGADM when α and β are both small, and PGADM was faster than ADMM when α and β are both large. Note that α and β are parameters that control the sparsity and low-rankness of the solution. Thus, our numerical results in Table 1 show that for fixed $\mu = 10$, PGADM was faster than ADMM when the solution is very sparse and low rank and ADMM was faster than PGADM when the solution is not very sparse and low rank.

Table 1: Comparison of ADMM with PGADM (for Fixed μ) on Synthetic Data.

		PGADM				ADMM			
α	β	obj	iter	cpu	infeas	obj	iter	cpu	infeas
0.005	0.025	−1.7113e+002	370	654.6	2.3e-006	−1.7113e+002	177	336.7	7.0e-006
0.005	0.05	−9.3484e+001	269	498.8	3.5e-006	−9.3484e+001	131	249.7	7.6e-006
0.01	0.05	−4.4763e+001	209	345.3	1.5e-006	−4.4763e+001	117	207.0	2.5e-006
0.01	0.1	5.4574e+001	99	162.1	1.8e-005	5.4573e+001	52	85.3	7.4e-005
0.02	0.1	1.1881e+002	112	186.9	1.2e-006	1.1881e+002	53	93.2	3.6e-005
0.02	0.2	2.3717e+002	64	105.3	3.3e-006	2.3718e+002	50	87.4	8.1e-005
0.04	0.2	3.2417e+002	47	77.1	5.4e-006	3.2415e+002	47	82.3	8.7e-005
0.04	0.4	4.5701e+002	25	40.8	4.9e-005	4.5701e+002	51	90.8	9.9e-005
0.08	0.4	5.7507e+002	24	39.8	8.7e-005	5.7506e+002	54	95.6	8.1e-005
0.08	0.8	7.1683e+002	41	65.9	9.6e-005	7.1683e+002	73	127.0	9.4e-005

Table 2: Comparison of ADMM with PGADM (with Continuation for μ) on Synthetic Data.

		PGADM				ADMM			
α	β	obj	iter	cpu	infeas	obj	iter	cpu	infeas
0.005	0.025	−1.711329e+002	32	51.5	5.7e-006	−1.711334e+002	61	103.7	6.7e-006
0.005	0.05	−9.348245e+001	41	65.8	4.2e-006	−9.348589e+001	62	103.2	6.6e-006
0.010	0.05	−4.476323e+001	41	67.8	2.8e-006	−4.476499e+001	62	105.7	9.8e-006
0.010	0.10	5.456790e+001	41	65.7	5.1e-006	5.456724e+001	71	119.6	6.0e-006
0.020	0.10	1.188077e+002	41	64.9	3.4e-006	1.188054e+002	71	115.1	8.1e-006
0.020	0.20	2.371659e+002	45	76.4	8.7e-006	2.371687e+002	75	125.6	7.0e-006
0.040	0.20	3.241688e+002	44	72.7	8.3e-006	3.241684e+002	75	126.6	8.5e-006
0.040	0.40	4.570019e+002	50	77.5	7.5e-006	4.570058e+002	78	126.5	9.7e-006
0.080	0.40	5.750627e+002	51	89.0	5.8e-006	5.750665e+002	81	154.8	8.6e-006
0.080	0.80	7.168308e+002	55	97.8	7.4e-006	7.168313e+002	91	173.4	4.8e-006

We then compared ADMM and PGADM on synthetic data with the continuation scheme for μ discussed above. We terminated both ADMM and PGADM when infeas $< 10^{-5}$. The results are in Table 2.

From Table 2, we see that the continuation scheme used helped to speed up the convergence and produced much better results. Also, when this continuation scheme was used, PGADM was faster than ADMM with comparable infeasibilities (infeas) and objective function values. However, PGADM was faster than ADMM using the specific continuation scheme. If other continuation schemes were adopted, the results could be quite different. In the comparison with LogdetPPA in the following sections, we compare LogdetPPA with PGADM only with this continuation scheme.

5.2 Comparison of PGADM and LogdetPPA on Synthetic Data. In this section, we compare PGADM with LogdetPPA on synthetic data created the same way as in the last section. LogdetPPA, proposed by Wang et al. (2010), is a proximal point algorithm for solving semidefinite programming problems with log det$(\cdot)$ function. The specialized Matlab code of LogdetPPA for solving equation 2.2 was downloaded from http://ssg.mit.edu/∼venkatc/latent-variable-code.html. This specialized code solves the following equivalent form of equation 2.1:

$$\min_{R,L} \langle R, \hat{\Sigma}_X \rangle - \log\det R + \alpha\|R + L\|_1 + \beta \mathrm{Tr}(L), \quad \text{s.t.,} \ L \succeq 0. \tag{5.2}$$

Note that compared to equation 2.1, there is no variable S and thus no constraint $R - S + L = 0$ in equation 5.2. The sparse matrix S can be obtained by

$$S := \hat{R} + \hat{L}, \tag{5.3}$$

where $(\hat{R}, \hat{L})$ is the optimal solution to equation 5.2.

We compared PGADM (with continuation on μ) with LogdetPPA with different α and β. We reported the comparison results on objective function value, CPU time, sparsity of S (sp), and infeas in Table 3. The sparsity of S is denoted as

$$sp := \frac{\#\{(i, j) : S_{ij} \neq 0\}}{p^2}$$

(the percentage of nonzero entries). Since matrix S generated by LogdetPPA (i.e., obtained by equation 5.3) is always dense (usually with a sparsity 100% according to our numerical tests) but with many small entries, we measure its sparsity by truncating small entries that are less than 10^{-4} as zeros:

$$sp1 := \frac{\#\{(i, j) : |S_{ij}| > 10^{-4}\}}{p^2}.$$

In Table 3, we report the sparsity in the percentage values that are denoted as spl (%) and sp (%). Note that we do not report the infeas for LogdetPPA, because there is no equality constraint $R - S + L = 0$ in problem 5.2 solved by LogdetPPA. All CPU times reported are in seconds. We report the speed-up of PGADM over LogdetPPA in Table 4.

From Table 3 we see that the solutions produced by PGADM always have comparable objective function values compared to the solutions produced by LogdetPPA. However, our PGADM is always much faster than LogdetPPA, as shown in both Tables 3 and 4. In fact, PGADM is usually

Table 3: Results of PGADM and LogdetPPA on Synthetic Data.

dim	LogdetPPA			PGADM			
p	obj	cpu	sp1 (%)	obj	cpu	sp (%)	infeas
$\alpha = 0.005, \beta = 0.025$							
200	1.914315e+2	7.2	19.17	1.910379e+2	1.0	18.90	4.3e-6
500	1.898418e+2	235.2	5.78	1.898275e+2	7.3	5.63	7.2e-6
1000	−1.711293e+2	1706.0	0.52	−1.711329e+2	48.2	0.49	5.7e-6
2000	−1.430010e+3	5001.2	0.06	−1.435605e+3	427.2	0.05	4.9e-6
$\alpha = 0.005, \beta = 0.05$							
200	1.926376e+2	29.1	43.63	1.924829e+2	2.5	48.66	6.6e-6
500	2.051884e+2	358.3	12.04	2.051425e+2	10.5	11.42	8.8e-6
1000	−9.347297e+1	1076.4	4.88	−9.348245e+1	71.6	4.72	4.2e-6
2000	−1.229323e+3	5191.5	0.21	−1.230238e+3	445.0	0.15	9.5e-6
$\alpha = 0.01, \beta = 0.05$							
200	2.030390e+2	20.8	20.46	2.026586e+2	2.6	11.06	9.0e-6
500	2.394720e+2	146.0	4.25	2.394631e+2	10.7	4.14	3.7e-6
1000	−4.476078e+1	740.6	0.24	−4.476323e+1	72.8	0.23	2.8e-6
2000	−1.101454e+3	6433.4	0.05	−1.111504e+3	453.7	0.05	6.4e-6
$\alpha = 0.01, \beta = 0.1$							
200	2.050879e+2	29.1	42.16	2.048359e+2	1.6	35.83	7.0e-6
500	2.639548e+2	235.2	8.62	2.638565e+2	11.6	8.19	5.8e-6
1000	5.456825e+1	932.1	2.26	5.456790e+1	74.9	2.26	5.1e-6
2000	−8.541802e+2	4813.6	0.14	−8.712916e+2	516.6	0.07	8.6e-6

Table 4: Speed-Up of PGADM over LogdetPPA on Synthetic Data.

p	$\alpha = 0.005, \beta = 0.025$	$\alpha = 0.005, \beta = 0.05$	$\alpha = 0.01, \beta = 0.05$	$\alpha = 0.01, \beta = 0.1$
200	7.1	11.7	7.9	18.7
500	32.0	34.2	13.6	20.3
1000	35.4	15.0	10.2	12.4
2000	11.7	11.7	14.2	9.3

10 times faster than LogdetPPA, and sometimes more than 35 times faster. For example, for the four large problems with matrices size 2000×2000, LogdetPPA needs 1 hour 23 minutes, 1 hour 26 minutes, 1 hour 47 minutes, and 1 hour 20 minutes, respectively, to solve them, while our PGADM needs about 7 minutes, 7 minutes, 8 minutes, and 9 minutes, respectively, to solve them. We also notice that the matrix S generated by PGADM is always a sparse matrix with many entries exactly equal to zero, but S generated by LogdetPPA is always a dense matrix, and only after we truncate the entries that are smaller than 10^{-4} to zeros does it become a sparse matrix with a similar level of sparsity. This is because in our PGADM, S is updated by the ℓ_1 shrinkage operation, which truncates the small entries to zeros, while

Table 5: Results of PGADM and LogdetPPA on Rosetta Data Set.

dim	LogdetPPA			PGADM			
p	obj	cpu	sp1 (%)	obj	cpu	sp (%)	infeas
200	−2.726188e+2	5.6	0.58	−2.726184e+2	0.7	0.58	7.1e-6
500	−8.662116e+2	68.0	0.20	−8.662113e+2	7.9	0.20	3.7e-6
1000	−1.970974e+3	490.9	0.10	−1.970973e+3	52.2	0.10	7.7e-6
2000	−4.288406e+3	4597.9	0.05	−4.288406e+3	422.3	0.05	5.4e-6

Table 6: Results of PGADM and LogdetPPA on Iconix Data Set.

dim	LogdetPPA			PGADM			
p	obj	cpu	sp1 (%)	obj	cpu	sp (%)	infeas
200	1.232884e+3	38.5	36.10	1.232744e+3	2.5	41.62	7.2e-6
500	1.842623e+3	98.8	2.27	1.839838e+3	17.0	0.77	1.0e-5
1000	1.439052e+3	1341.9	1.45	1.435425e+3	94.6	0.14	6.3e-6
2000	1.242966e+2	13,207.2	0.07	1.168757e+2	738.2	0.06	8.5e-6

LogdetPPA needs to replace $\|S\|_1$ with a smooth linear function that does not preserve sparsity.

5.3 Comparison of PGADM and LogdetPPA on Gene Expression Data. To further demonstrate the efficacy of PGADM, we applied PGADM to solving equation 2.2 with two gene expression data sets. One data set is the Rosetta Inpharmatics Compendium of gene expression data (denoted as Rosetta) (Hughes et al., 2000) profiles, which contains 301 samples with 6316 variables (genes). The other data set is the Iconix microarray data set (denoted as Iconix) from drug-treated rat livers (Natsoulis et al., 2005), which contains 255 samples with 10,455 variables.

For a given number of observed variables p, we created the sample covariance matrix Σ_X by the following procedure. We first computed the variances of all variables using all the sample data. We then selected the p variables with the highest variances and computed the sample covariance matrix Σ_X of these p variables using all the sample data. We reported the comparison results of PGADM and LogdetPPA in Tables 5 and 6 for the Rosetta and Iconix data sets, respectively. Table 7 summarizes the speed up of PGADM over LogdetPPA.

From Table 5, we again see that PGADM always generates solutions with comparable objective function values in much less time. For example, for $p = 2000$, LogdetPPA needs 1 hour 16 minutes to solve it, while PGADM takes just 7 minutes. From Table 6, we see that the advantage of PGADM is more obvious; for $p = 200, 500, 1000$, and 2000, PGADM always generates

Table 7: Speed-Up of PGADM over LogdetPPA on Rosetta and Iconix Data Sets.

dim	Rosetta Data	Iconix Data
200	8.0	15.4
500	8.6	5.8
1000	9.4	14.2
2000	10.9	17.9

solutions with much smaller objective function values, and it is always much faster than LogdetPPA. For example, for $p = 2000$, LogdetPPA takes 3 hours 40 minutes to solve it, while PGADM just takes about 12 minutes.

6 Conclusion

We have proposed two alternating direction methods for solving the latent variable gaussian graphical model selection problem. The global convergence results of our methods were established. We applied the proposed methods for solving large problems from both synthetic data and gene expression data. The numerical results indicate that our methods are 5 to 35 times faster than a state-of-the-art Newton-CG proximal point algorithm.

Appendix: Global Convergence Analysis of PGADM

In this section, we establish the global convergence result of PGADM (algorithm 2). This convergence proof is not much different from the one given by Yang and Zhang (2011) for compressed sensing problems. We include the proof here for completeness.

We introduce some notation first. We define $W = \begin{pmatrix} S \\ L \end{pmatrix}$. We define functions $F(\cdot)$ and $G(\cdot)$ as

$$F(R) := \langle R, \hat{\Sigma}_X \rangle - \log\det R$$

and

$$G(W) := \alpha\|S\|_1 + \beta \mathrm{Tr}(L) + \mathcal{I}(L \succeq 0).$$

Note that both F and G are convex functions. We also define matrix A as $A = [-I_{p\times p}, I_{p\times p}] \in \mathbb{R}^{p\times 2p}$. Now problem 2.2 can be rewritten as

$$\min\ F(R) + G(W), \quad \text{s.t. } R + AW = 0, \tag{A.1}$$

and our PGADM (algorithm 2) can be rewritten as

$$\begin{cases} R^{k+1} := \underset{R}{\operatorname{argmin}}\ F(R) + G(W^k) - \langle \Lambda^k, R + AW^k \rangle + \dfrac{1}{2\mu}\|R + AW^k\|_F^2 \\ W^{k+1} := \underset{W}{\operatorname{argmin}}\ F(R^{k+1}) + G(W) + \dfrac{1}{2\tau\mu}\|W \\ \qquad\qquad - (W^k - \tau A^\top (R^{k+1} + AW^k - \mu\Lambda^k))\|_F^2 \\ \Lambda^{k+1} := \Lambda^k - (R^{k+1} + AW^{k+1})/\mu. \end{cases} \tag{A.2}$$

Before we prove the global convergence result, we need to prove the following lemma:

Lemma 1. *Assume that* (R^*, W^*) *is an optimal solution of equation A.1 and* Λ^* *is the corresponding optimal dual variable associated with the equality constraint* $R + AW = 0$. *Assume the step size* τ *of the proximal gradient step satisfies* $0 < \tau < 1/2$. *Then there exists* $\eta > 0$ *such that the sequence* (R^k, W^k, Λ^k) *produced by equation A.2 satisfies*

$$\|U^k - U^*\|_H^2 - \|U^{k+1} - U^*\|_H^2 \geq \eta \|U^k - U^{k+1}\|_H^2, \tag{A.3}$$

where $U^* = \begin{pmatrix} W^* \\ \Lambda^* \end{pmatrix}$, $U^k = \begin{pmatrix} W^k \\ \Lambda^k \end{pmatrix}$ *and* $H = \begin{pmatrix} \frac{1}{\mu\tau} I_{p\times p} & 0 \\ 0 & \mu I_{p\times p} \end{pmatrix}$, *and the norm* $\|\cdot\|_H^2$ *is defined as* $\|U\|_H^2 = \langle U, HU \rangle$ *and the corresponding inner product* $\langle \cdot, \cdot \rangle_H$ *is defined as* $\langle U, V \rangle_H = \langle U, HV \rangle$.

Proof. Since (R^*, W^*) and Λ^* are a pair of primal and dual optimal solutions to equation A.1, it follows from the KKT conditions that the following equations hold:

$$0 \in \partial F(R^*) - \Lambda^*, \tag{A.4}$$

$$0 \in \partial G(W^*) - A^\top \Lambda^*, \tag{A.5}$$

$$0 = R^* + AW^*. \tag{A.6}$$

Note that the first-order optimality conditions for the first subproblem (i.e., the subproblem with respect to R) in equation A.2 are given by

$$0 \in \partial F(R^{k+1}) - \Lambda^k + \frac{1}{\mu}(R^{k+1} + AW^k - 0). \tag{A.7}$$

By using the updating formula for Λ^k,

$$\Lambda^{k+1} = \Lambda^k - (R^{k+1} + AW^{k+1})/\mu, \tag{A.8}$$

equation A.7 can be reduced to

$$0 \in \partial F(R^{k+1}) - \Lambda^{k+1} - \frac{1}{\mu}(AW^{k+1} - AW^k). \tag{A.9}$$

Combining equations A.4 and A.9 and using the fact that $\partial F(\cdot)$ is a monotone operator, we get

$$\left\langle R^{k+1} - R^*, \Lambda^{k+1} - \Lambda^* + \frac{1}{\mu}(AW^{k+1} - AW^k) \right\rangle \geq 0. \tag{A.10}$$

The first-order optimality conditions for the second subproblem (the subproblem with respect to W) in equation A.2 are given by

$$0 \in \partial G(W^{k+1}) + \frac{1}{\mu\tau}(W^{k+1} - (W^k - \tau A^\top (AW^k + R^{k+1} - \mu\Lambda^k))). \tag{A.11}$$

Using equation A.8, equation A.11 can be reduced to

$$0 \in \partial G(W^{k+1}) + \frac{1}{\mu\tau}(W^{k+1} - W^k + \tau A^\top (AW^k - AW^{k+1} - \mu\Lambda^{k+1})). \tag{A.12}$$

Combining equations A.5 and A.12 and using the fact that $\partial G(\cdot)$ is a monotone operator, we get

$$\left\langle W^{k+1} - W^*, \frac{1}{\mu\tau}(W^k - W^{k+1}) - \frac{1}{\mu}A^\top (AW^k - AW^{k+1}) + A^\top \Lambda^{k+1} - A^\top \Lambda^* \right\rangle \geq 0. \tag{A.13}$$

Summing equations A.10 and A.13, and using $R^* = -AW^*$ and $R^{k+1} = \mu(\Lambda^k - \Lambda^{k+1}) - AW^{k+1}$, we obtain

$$\frac{1}{\mu\tau}\langle W^{k+1} - W^*, W^k - W^{k+1}\rangle + \mu\langle \Lambda^{k+1} - \Lambda^*, \Lambda^k - \Lambda^{k+1}\rangle \geq \langle \Lambda^k - \Lambda^{k+1}, AW^k - AW^{k+1}\rangle. \tag{A.14}$$

Using the notation of U^k, U^* and H, equation A.14 can be rewritten as

$$\langle U^{k+1} - U^*, U^k - U^{k+1}\rangle_H \geq \langle \Lambda^k - \Lambda^{k+1}, AW^k - AW^{k+1}\rangle, \tag{A.15}$$

which can be further written as

$$\langle U^k - U^*, U^k - U^{k+1}\rangle_H \geq \|U^k - U^{k+1}\|_H + \langle \Lambda^k - \Lambda^{k+1}, AW^k - AW^{k+1}\rangle. \tag{A.16}$$

Combining equation A.16 with the identity

$$\|U^{k+1} - U^*\|_H^2 = \|U^{k+1} - U^k\|_H^2 - 2\langle U^k - U^{k+1}, U^k - U^*\rangle_H + \|U^k - U^*\|_H^2,$$

we get

$$\begin{aligned}
&\|U^k - U^*\|_H^2 - \|U^{k+1} - U^*\|_H^2 \\
&\quad = 2\langle U^k - U^{k+1}, U^k - U^*\rangle_H - \|U^{k+1} - U^k\|_H^2 \\
&\quad \geq \|U^{k+1} - U^k\|_H^2 + 2\langle \Lambda^k - \Lambda^{k+1}, AW^k - AW^{k+1}\rangle.
\end{aligned} \tag{A.17}$$

Let $\xi := \tau + 1/2$. Then we know that $2\tau < \xi < 1$ since $0 < \tau < 1/2$. Let $\rho := \mu\xi$. Then from Cauchy-Schwartz inequality, we have

$$\begin{aligned}
&2\langle \Lambda^k - \Lambda^{k+1}, AW^k - AW^{k+1}\rangle \\
&\quad \geq -\rho\|\Lambda^k - \Lambda^{k+1}\|^2 - \frac{1}{\rho}\|AW^k - AW^{k+1}\|^2 \\
&\quad \geq -\rho\|\Lambda^k - \Lambda^{k+1}\|^2 - \frac{1}{\rho}\lambda_{\max}(A^\top A)\|W^k - W^{k+1}\|^2 \\
&\quad = -\rho\|\Lambda^k - \Lambda^{k+1}\|^2 - \frac{2}{\rho}\|W^k - W^{k+1}\|^2,
\end{aligned} \tag{A.18}$$

where the $\lambda_{\max}(A^\top A)$ denotes the largest eigenvalue of matrix $A^\top A$ and the equality is due to the fact that $\lambda_{\max}(A^\top A) = 2$. Combining equations A.17 and A.18, we get

$$\begin{aligned}
&\|U^k - U^*\|_H^2 - \|U^{k+1} - U^*\|_H^2 \\
&\quad \geq \left(\frac{1}{\mu\tau} - \frac{2}{\rho}\right)\|W^k - W^{k+1}\|^2 + (\mu - \rho)\|\Lambda^k - \Lambda^{k+1}\|^2 \\
&\quad \geq \eta\|U^k - U^{k+1}\|_H^2,
\end{aligned} \tag{A.19}$$

where $\eta := \min\{1 - \frac{2\mu\tau}{\rho}, 1 - \frac{\rho}{\mu}\} = \min\{1 - \frac{2\tau}{\xi}, 1 - \xi\} > 0$. This completes the proof.

We are now ready to give the main convergence result of the algorithm described in equation A.2:

Theorem 1. *The sequence* $\{(R^k, W^k, \Lambda^k)\}$ *produced by the algorithm described in equation A.2 from any starting point converges to an optimal solution to problem A.1.*

Proof. From lemma 1, we can easily get that

i. $\|U^k - U^{k+1}\|_H \to 0$.
ii. $\{U^k\}$ lies in a compact region.
iii. $\|U^k - U^*\|_H^2$ is monotonically nonincreasing and thus converges.

It follows from (i) that $\Lambda^k - \Lambda^{k+1} \to 0$ and $W^k - W^{k+1} \to 0$. Then equation A.8 implies that $R^k - R^{k+1} \to 0$ and $R^k + AW^k \to 0$. From (ii) we obtain that, U^k has a sub-sequence $\{U^{k_j}\}$ that converges to $\hat{U} = (\hat{W}, \hat{\Lambda})$—$\Lambda^{k_j} \to \hat{\Lambda}$ and $W^{k_j} \to \hat{W}$. From $R^k + AW^k \to 0$ we also get that $R^{k_j} \to \hat{R} := -A\hat{W}$. Therefore, $(\hat{R}, \hat{W}, \hat{\Lambda})$ is a limit point of $\{(R^k, W^k, \Lambda^k)\}$.

Note that equation A.9 implies that

$$0 \in \partial F(\hat{R}) - \hat{\Lambda}. \tag{A.20}$$

Note also that equation A.12 implies that

$$0 \in \partial G(\hat{W}) - A^\top \hat{\Lambda}. \tag{A.21}$$

Equations A.20 and A.21 and $\hat{R} + A\hat{W} = 0$ imply that $(\hat{R}, \hat{W}, \hat{\Lambda})$ satisfies the KKT conditions for equation A.1 and thus is an optimal solution to it. Therefore, we showed that any limit point of $\{(R^k, W^k, \Lambda^k)\}$ is an optimal solution to equation A.1.

To complete the proof, we need to show that the limit point is unique. Let $\{(\hat{R}_1, \hat{W}_1, \hat{\Lambda}_1)\}$ and $\{(\hat{R}_2, \hat{W}_2, \hat{\Lambda}_2)\}$ be any two limit points of $\{(R^k, W^k, \Lambda^k)\}$. As we have shown, both $\{(\hat{R}_1, \hat{W}_1, \hat{\Lambda}_1)\}$ and $\{(\hat{R}_2, \hat{W}_2, \hat{\Lambda}_2)\}$ are optimal solutions to equation A.1. Thus, U^* in equation A.19 can be replaced by $\hat{U}_1 := (\hat{R}_1, \hat{W}_1, \hat{\Lambda}_1)$ and $\hat{U}_2 := (\hat{R}_2, \hat{W}_2, \hat{\Lambda}_2)$. This results in

$$\|U^{k+1} - \hat{U}_i\|_H^2 \le \|U^k - \hat{U}_i\|_H^2, \quad i = 1, 2,$$

and we thus get the existence of the limits

$$\lim_{k\to\infty} \|U^k - \hat{U}_i\|_H = \eta_i < +\infty, \quad i = 1, 2.$$

Now using the identity

$$\|U^k - \hat{U}_1\|_H^2 - \|U^k - \hat{U}_2\|_H^2 = -2\langle U^k, \hat{U}_1 - \hat{U}_2\rangle_H + \|\hat{U}_1\|_H^2 - \|\hat{U}_2\|_H^2$$

and passing the limit, we get

$$\eta_1^2 - \eta_2^2 = -2\langle \hat{U}_1, \hat{U}_1 - \hat{U}_2\rangle_H + \|\hat{U}_1\|_H^2 - \|\hat{U}_2\|_H^2 = -\|\hat{U}_1 - \hat{U}_2\|_H^2$$

and

$$\eta_1^2 - \eta_2^2 = -2\langle \hat{U}_2, \hat{U}_1 - \hat{U}_2\rangle_H + \|\hat{U}_1\|_H^2 - \|\hat{U}_2\|_H^2 = \|\hat{U}_1 - \hat{U}_2\|_H^2.$$

Thus we must have $\|\hat{U}_1 - \hat{U}_2\|_H^2 = 0$ and hence the limit point of $\{(R^k, W^k, \Lambda^k)\}$ is unique.

We now immediately have the global convergence result for algorithm 2 for solving problem 2.2.

Corollary 1. *The sequence* $\{(R^k, S^k, L^k, \Lambda^k)\}$ *produced by algorithm 2 from any starting point converges to an optimal solution to problem 2.2.*

Acknowledgments

We thank Stephen Boyd for suggesting solving equation 2.2 as a consensus problem, equation 3.1, using ADMM. We also thank two anonymous referees for their constructive comments that greatly improved the presentation of this letter. H.Z.'s research was supported in part by grants from the National Science Foundation and the Office of Naval Research.

References

Ahmed, A., & Xing, E. P. (2009). Recovering time-varying networks of dependencies in social and biological studies. *Proceedings of the National Academy of Sciences*, *106*(29), 11878–11883.

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

Boyd, S., Parikh, N., Chu, E., Peleato, B., & Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, *3*(1), 1–122.

Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge: Cambridge University Press.

Cai, T., Liu, W., & Luo, X. (2011). A constrained ℓ_1 minimization approach to sparse precision matrix estimation. *Journal of the American Statistical Association*, *106*(494), 594–607.

Candès, E. J., & Tao, T. (2007). The Dantzig selector: Statistical estimation when p is much larger than n. *Annals of Statistics*, *35*(6), 2313–2351.

Chandrasekaran, V., Parrilo, P., & Willsky, A. (2012). Latent variable graphical model selection via convex optimization. *Annals of Statistics*, *40*(4), 1935–1967.

Chen, G., & Teboulle, M. (1994). A proximal-based decomposition method for convex minimization problems. *Mathematical Programming*, *64*, 81–101.

Combettes, P. L., & Pesquet, J. C. (2007). A Douglas-Rachford splitting approach to nonsmooth convex variational signal recovery. *IEEE Journal of Selected Topics in Signal Processing*, *1*(4), 564–574.

Combettes, P. L., & Wajs, V. R. (2005). Signal recovery by proximal forward-backward splitting. *SIAM Journal on Multiscale Modeling and Simulation*, *4*(4), 1168–1200.

D'Aspremont, A., Banerjee, O., & El Ghaoui, L. (2008). First-order methods for sparse covariance selection. *SIAM Journal on Matrix Analysis and Its Applications*, *30*(1), 56–66.

Deng, W., & Yin, W. (2012). *On the global and linear convergence of the generalized alternating direction method of multipliers* (Tech. Rep.). Houston, TX: Rice University CAAM.

Dobra, A., Eicher, T. S., & Lenkoski, A. (2009). Modeling uncertainty in macroeconomic growth determinants using gaussian graphical models. *Statistical Methodology*, *7*, 292–306.

Douglas, J., & Rachford, H. H. (1956). On the numerical solution of the heat conduction problem in 2 and 3 space variables. *Transactions of the American Mathematical Society*, *82*, 421–439.

Duchi, J., Gould, S., & Koller, D. (2008). Projected subgradient methods for learning sparse gaussian. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence*. N.p.: AUAI Press.

Eckstein, J. (1989). *Splitting methods for monotone operators with applications to parallel optimization*. Unpublished doctoral dissertation, MIT.

Eckstein, J. (1994). Some saddle-function splitting methods for convex programming. *Optimization Methods and Software*, *4*(1), 75–83.

Eckstein, J., & Bertsekas, D. P. (1992). On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, *55*, 293–318.

Edwards, D. (2000). *Introduction to graphical modelling*. New York: Springer-Verlag.

Fazel, M., Pong, T., Sun, D., & Tseng, P. (2012). *Hankel matrix rank minimization with applications to system identification and realization*. Preprint.

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

Friedman, N. (2004). Inferring cellular networks using probabilistic graphical models. *Science*, *303*(5659), 799–805.

Gabay, D. (1983). Applications of the method of multipliers to variational inequalities. In M. Fortin & R. Glowinski (Eds.), *Augmented Lagrangian methods: Applications to the solution of boundary value problems*. Amsterdam: North-Holland.

Glowinski, R., & Le Tallec, P. (1989). *Augmented Lagrangian and operator-splitting methods in nonlinear mechanics*. Philadelphia: SIAM.

Goldfarb, D., & Ma, S. (2012). Fast multiple splitting algorithms for convex optimization. *SIAM Journal on Optimization*, *22*(2), 533–556.

Goldfarb, D., Ma, S., & Scheinberg, K. (in press). Fast alternating linearization methods for minimizing the sum of two convex functions. *Mathematical Programming*.

Goldstein, T., & Osher, S. (2009). The split Bregman method for L1-regularized problems. *SIAM J. Imaging Sci.*, *2*, 323–343.

He, B. S., Liao, L. Z., Han, D., & Yang, H. (2002). A new inexact alternating direction method for monotone variational inequalities. *Math. Program.*, *92*, 103–118.

Hong, M., & Luo, Z. (2012). *On the linear convergence of the alternating direction method of multipliers*. Preprint.

Hughes, T. R., Marton, M. J., Jones, A. R., Roberts, C. J., Stoughton, R., Armour, C. D., et al. (2000). Functional discovery via a compendium of expression profiles. *Cell*, *102*(1), 109–126.

Kolar, M., Song, L., Ahmed, A., & Xing, E. P. (2010). Estimating time-varying networks. *Annals of Applied Statistics*, *4*(1), 94–123.

Lauritzen, S. L. (1996). *Graphical models*. New York: Oxford University Press.

Li, L., & Toh, K. C. (2010). An inexact interior point method for L_1-regularized sparse covariance selection. *Mathematical Programming Computation*, *2*, 291–315.

Lions, P. L., & Mercier, B. (1979). Splitting algorithms for the sum of two nonlinear operators. *SIAM Journal on Numerical Analysis*, *16*, 964–979.

Liu, H., Han, F., Yuan, M., Lafferty, J., & Wasserman, L. (2012). High dimensional semiparametric Gaussian copula graphical models. *Annals of Statistics*, *40*, 2293–2326.

Lu, Z. (2009). Smooth optimization approach for sparse covariance selection. *SIAM J. Optim.*, *19*(4), 1807–1827.

Ma, S. (2011). *Alternating direction method of multipliers for sparse principal component analysis*. Preprint.

Ma, S. (2012). *Alternating proximal gradient method for convex minimization*. Preprint.

Malick, J., Povh, J., Rendl, F., & Wiegele, A. (2009). Regularization methods for semidefinite programming. *SIAM Journal on Optimization*, *20*, 336–356.

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

Natsoulis, G., Ghaoui, L. E., Lanckriet, G., Tolley, A., Leroy, F., Dunlea, S., et al. (2005). Classification of a large microarray data set: Algorithm comparison and analysis of drug signatures. *Genome Research*, *15*, 724–736.

Parikh, N., & Boyd, S. (2011). *Block splitting for large-scale distributed learning*. Presented at the Neural Information Workshop on Big Learning.

Peaceman, D. H., & Rachford, H. H. (1955). The numerical solution of parabolic elliptic differential equations. *SIAM Journal on Applied Mathematics*, *3*, 28–41.

Peng, J., Wang, P., Zhou, N., & Zhu, J. (2009). Partial correlation estimation by joint sparse regression models. *Journal of the American Statistical Association*, *104*(486), 735–746.

Qin, Z., Goldfarb, D., & Ma, S. (2011). *An alternating direction method for total variation denoising*. Preprint.

Ravikumar, P., Wainwright, M. J., Raskutti, G., & Yu, B. (2008). High-dimensional covariance estimation by minimizing ℓ_1-penalized log-determinant divergence. In D. Köller, D. Schuurmans, Y. Bengio, & L. Bottou (Eds.), *Advances in neural information processing systems, 21*. Cambridge, MA: MIT Press.

Rothman, A. J., Bickel, P. J., Levina, E., & Zhu, J. (2008). Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics*, *2*, 494–515.

Scheinberg, K., Ma, S., & Goldfarb, D. (2010). Sparse inverse covariance selection via alternating linearization methods. In J. Lafferty, C.K.I. Williams, J. Shawe-Taylor, R. S. Zernal, & A. Culotta (Eds.), *Advances in neural information processing systems, 23*. Red Hook, NY: Curran.

Scheinberg, K., & Rish, I. (2009). *Sinco: A greedy coordinate ascent method for sparse inverse covariance selection problem*. Preprint. http://www.optimization-online.org/DB_HTML/2009/07/2359.html

Tao, M., & Yuan, X. (2011). Recovering low-rank and sparse components of matrices from incomplete and noisy observations. *SIAM Journal on Optimization*, *21*, 57–81.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, *58*(1), 267–288.

Wang, C., Sun, D., & Toh, K. C. (2010). Solving log-determinant optimization problems by a Newton-CG primal proximal point algorithm. *SIAM Journal on Optimization*, *20*, 2994–3013.

Wang, Y., Yang, J., Yin, W., & Zhang, Y. (2008). A new alternating minimization algorithm for total variation image reconstruction. *SIAM Journal on Imaging Sciences*, *1*(3), 248–272.

Wen, Z., Goldfarb, D., & Yin, W. (2010). Alternating direction augmented Lagrangian methods for semidefinite programming. *Mathematical Programming Computation*, *2*, 203–230.

Wille, A., Zimmermann, P., Vranová, E., Fürholz, A., Laule, O., Bleuler, S., et al. (2004). Sparse graphical gaussian modeling of the isoprenoid gene network in *Arabidopsis thaliana*. *Genome Biology*, *5*(11), R92.

Witten, D. M., Friedman, J. H., & Simon, N. (2011). New insights and faster computations for the graphical lasso. *Journal of Computational and Graphical Statistics*, *20*(4), 892–900.

Xue, L., Ma, S., & Zou, H. (2012). Positive definite ℓ_1 penalized estimation of large covariance matrices. *Journal of the American Statistical Association*, *107*(500), 1480–1491.

Xue, L., & Zou, H. (2012). Regularized rank-based estimation of high-dimensional nonparanormal graphical models. *Annals of Statistics*, *40*, 2541–2571.

Yang, J., & Zhang, Y. (2011). Alternating direction algorithms for ℓ_1 problems in compressive sensing. *SIAM Journal on Scientific Computing*, *33*(1), 250–278.

Yuan, M. (2010). High dimensional inverse covariance matrix estimation via linear programming. *Journal of Machine Learning Research*, *11*, 2261–2286.

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

Yuan, X. (2012). Alternating direction methods for sparse covariance selection. *Journal of Scientific Computing*, *51*, 261–273.

Received September 24, 2012; accepted January 22, 2013.