Sparse Inverse Covariance Estimation via the Split Bregman Method

Gui-Bo Ye and Jian-Feng Cai and Xiaohui Xie

School of Information and Computer Science, University of California, Irvine, CA 92697, USA
e-mail: yeg@uci.edu

Department of Mathematics, University of California, Los Angeles, CA 90095, USA
e-mail: cai@math.ucla.edu

School of Information and Computer Science, University of California, Irvine, CA 92697, USA
e-mail: xhx@ics.uci.edu

Abstract: We consider the problem of learning the structure of graphical models by estimating the inverse covariance matrix with sparsity regularization. We develop a novel first order method based on split Bregman to solve the sparse inverse covariance estimation problem. We show that our method is significantly faster than the widely used glasso method, which is based on blockwise coordinate descent. In addition, our method is easy to implement and can be applied to a general class of regularization terms.

Keywords and phrases: Inverse covariance matrix, Split Bregman, ℓ_1 norm, Graphical model.

1. Introduction

Graphical model is an efficient tool for modeling the relationship between a set of objects (or variables) in many application domains such as internet, social network, and biological networks [1, 2, 3]. Learning the structure of graphical models is, however, often very challenging due to the large number of nodes commonly seen in real applications and the enormous number of possible structures that these nodes can form. However, it has been noted that graphical models arising from many real-world applications are often sparse, which led to the idea of reformulating the structural learning problem as a covariance selection problem as first proposed by Dempster in [4]. The main observation is that nonzero entries of the inverse covariance matrix (also known as *concentration matrix* or *precision matrix*) correspond to nonzero partial correlations. If we assume the observations have a multivariate Gaussian distribution with mean μ and covariance matrix Σ, then nonzero entries of the concentration matrix Σ^{-1} will imply conditional dependency between corresponding variable pairs conditional on the rest of the variables [5, 6].

Regularization terms that encourage sparsity are increasingly being used for sparse graphical models. Based on the observation that the ij element of concentration matrix Σ^{-1} is, up to a positive scalar, the regression coefficient of variable j in the multiple regression of variable i on the rest, and vice versa [7, 6], Meinshausen and Bühlmann [8] estimate a sparse graphical model by fitting a collection of lasso regression models, using in turn each variable as the response, and the others as predictors. The component $\hat{\Sigma}^{-1}_{ij}$ is then estimated to be non-zero if either the estimated coefficient of variable i on j or the estimated coefficient of variable j on i is nonzero (alternatively, they use an AND rule). It is a computationally attractive method for covariance selection that can be used for very large Gaussian graphs. They showed that their method is consistent for sparse high-dimensional graphs. Despite of its efficiency and consistency, it does not take into account the intrinsic symmetry of the problem, which could result in contradictory neighborhoods and non-positive concentration matrix. Furthermore, if the same penalty parameter is used for all lasso regressions, as suggested by their article, more or less equal effort is placed on building each neighborhood, which is not so efficient for networks with skewed degree distributions. To overcome the asymmetry of Meinshausen and Bühlmann [8], Peng et al. [6] take a symmetric regression approach, called "SPACE", which performs well in both nonzero partial correlation selection and the identification of hub variables. However, the derived contradictory matrix could still be non-positive.

Another popular approach for covariance selection is to use regularized maximum likelihood with ℓ_1 penalty on the off-diagonal elements of the concentration matrix, as proposed by Yuan and Lin [2]. More

specifically, suppose we are given n samples independently drawn from a p-variate Gaussian distribution: $\mathbf{x}_1, \ldots, \mathbf{x}_n \sim \mathcal{N}(\mu, \Sigma)$. Let S be the empirical covariance matrix:

$$S := \frac{1}{n} \sum_{k=1}^{n} (\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T.$$

Denote $\Theta = \Sigma^{-1}$, the problem is to minimize

$$- \log \det \Theta + \mathrm{tr}(S\Theta) + \lambda \sum_{i \neq j} |\Theta_{ij}| \tag{1}$$

over nonnegative definite matrices Θ. Here, tr denotes the trace and $\Theta_i j$ is ij-th entry of Θ. Due to the ℓ_1 penalty and explicit positive definite constraint, the method lead to a sparse and shrinkage estimator of the concentration matrix that is positive definite. The simpler approach of Meinshausen and Bühlmann [8] can be viewed as an approximation to this exact problem [9].

The loss function in (1) is strictly convex, so a global optimal solution is guaranteed to exist. However, finding the optimal solution is computationally challenging due to complex log-likelihood function and the nondifferentiability of $\|\Theta\|_1$. To solve (1), Yuan and Lin [2] propose to use interior-point optimization methods for the "maxdet" problem, which are prohibitive for problems with more than tens of nodes due to its memory requirements and complexity. Banerjee et al. [9] develop blockwise coordinate descent for the optimization, which is a totally different framework. Following that, Friedman et al. [10] propose the "graphical lasso" procedure, an efficient implementation of blockwise coordinate descent, to solve the optimization problem. It is by far the most efficient algorithm for solving (1). However, graphical lasso cannot be extended to other penalties in a natural manner since it tied to the lasso formulation and block coordinate descent.

In this paper, we propose a new algorithm based on split Bregman method to solve regularized maximum likelihood graphical model with different penalties, including ℓ_1 penalty on the off-diagonal elements of the concentration matrix for edge sparsity problems [2]. Although the Bregman iteration was an old technique proposed in the sixties [11, 12], it gained significant interest only recently after Osher and his coauthors demonstrated its high efficiency for image restoration [13, 14, 15]. Most recently, it has also been shown to be an efficient tool for compressed sensing [16, 17, 18], matrix completion [19], low rank matrix recovery [20] and general fused lasso problem [21]. In the following, we will show that the general regularized maximum likelihood for sparse graphical model can be reformulated so that split Bregman method can be readily applied. The derived algorithm consists of three steps for each iteration. We noticed that the first update is the main time consuming step and proposed an iterative Newton method to solve it. The contribution is vital and makes the derived algorithm from split Bregman method very easy to implement and quite efficient. According to our numerical experiments using time trial, our proposed algorithm for solving the problem (1) saves more than 30% computational time from graphical lasso for large scale problems.

The rest of the paper is organized as follows. In Section 2, we proposed SBGM based on split Bregman method to solve the general regularized maximum likelihood graphical model. We present the general regularized maximum likelihood graphical model as well as its reformulation in subsection 2.1. The framework of SBGM for solving the general regularized maximum likelihood graphical model is proposed in subsection 2.2. In subsection 2.3, we give the explicit formula for each update and propose an efficient Newton method to solve the first update. The convergence property of this algorithm is also given. Section 3 gives an explicit implementation of algorithm SBGM for the special case proposed by Yuan and Lin [2]. We call this algorithm SBGLasso for simplicity. Numerical experiments to show the performance of our proposed algorithms are given in Section 4.

2. Split Bregman method for general regularized maximum likelihood graphical model (SBGM)

Split Bregman method was first proposed by S. Osher and his coauthors for solving total variation based image restoration problems [14]. It is later found to be equivalent or closely related to many other algorithms, such as Douglas-Rachford splitting [22], alternating direction method of multipliers [23, 24] [14] and the method of multipliers [25]. It is becoming an efficient algorithm to solve many large-scale problems [20, 15, 21].

In this section, we will propose an efficient algorithm based on split Bregman method to solve regularized maximum likelihood graphical model with different penalties. It is very efficient and applicable for large-scale data. For the special case with ℓ_1 penalty (1), it saves more than 30% computational time from graphical lasso for large scale problems.

2.1. General Problem and its reformulation

We first describe our algorithm in a more general setting than the one described in (1). Instead of using ℓ_1 norm penalty, we would use a combination of an ℓ_2 penalty and a general non-smooth penalty $\phi(\Theta)$ which is convex and satisfying $\phi(\Theta) = \phi(\Theta^T)$. More specifically, we find Θ^* by solving the following unconstrained optimization problem

$$\min_{\Theta \succ 0} -\log\det\Theta + \mathrm{tr}(S\Theta) + \lambda_1\phi(\Theta) + \frac{\lambda_2}{2}\|\Theta\|_F^2, \tag{2}$$

where $\Theta \succ 0$ denotes Θ is positive semi-definite, $\lambda_1, \lambda_2 \in \mathbb{R}_+$ are two regularization parameters and $\|\Theta\|_F$ is the Frobenius norm of Θ. If we choose $\phi(\Theta) = \sum_{i\neq j}|\Theta_{ij}|$ and $\lambda_2 = 0$, the problem is exactly the minimization problem (1).

In order to use split Bregman method, we reformulate (2) into an equivalent constraint problem

$$\begin{aligned} \min\ & -\log\det\Theta + \mathrm{tr}(S\Theta) + \lambda_1\phi(A) + \frac{\lambda_2}{2}\|\Theta\|_F^2 \\ \text{s.t.}\quad & A = \Theta \\ & \Theta \succ 0. \end{aligned} \tag{3}$$

Note that the introduction of the new variable of A is very important and will make the subproblems derived by split Bregman method very easy to solve.

2.2. Framework of split Bregman method for the problem (3)

Although split Bregman method originates from Bregman iterations [26, 15, 27], it has been demonstrated to be equivalent to alternating direction method of multipliers (ADMM) [23, 24, 28, 29]. For convenience, here we derive the split Bregman method from augmented lagrangian method [30, 25].

Ignoring the constraint $\Theta \succ 0$, the augmented Lagrangian function of (3) is

$$\mathcal{L}(\Theta, A, M) = -\log\det\Theta + \mathrm{tr}(S\Theta) + \lambda_1\phi(A) + \frac{\lambda_2}{2}\|\Theta\|_F^2 + \mathrm{tr}(M^T(\Theta - A)) + \frac{\mu}{2}\|\Theta - A\|_F^2, \tag{4}$$

where matrix M is a dual variable corresponding to the linear constraint $\Theta = A$ and $\mu > 0$ is a prameter. Compared with standard Lagrangian function, we have the extra term $\frac{\mu}{2}\|\Theta - A\|_F^2$, which is used to penalize the violation of the linear constraint $\Theta = A$.

It is easy to see that solving (3) can be transferred to finding a solution on its dual problem

$$\max_M E(M), \quad \text{with} \quad E(M) = \min_{\Theta\succ 0, A} \mathcal{L}(\Theta, A, M). \tag{5}$$

Note that the gradient $\nabla E(M)$ can be calculated by the following [31]

$$\nabla E(M) = \Theta(M) - A(M), \quad \text{with} \quad (\Theta(M), A(M)) = \arg\min_{\Theta\succ 0, A} \mathcal{L}(\Theta, A, M). \tag{6}$$

Applying gradient ascent on the dual problem (5) and using equation (6), we get the method of multipliers [25] to solve (3)

$$\begin{cases} (\Theta^{k+1}, A^{k+1}) = \arg\min_{\Theta\succ 0, A} \mathcal{L}(\Theta, A, M^k), \\ M^{k+1} = M^k + \mu(\Theta^{k+1} - A^{k+1}). \end{cases} \tag{7}$$

Here we have used μ as the step size of gradient ascent. It is easy to see that the efficiency of the iterative algorithm (7) largely lies on whether the first equation of (7) can be solved quickly. Note that the augmented

Lagrangian function $\mathcal{L}(\Theta, A, M^k)$ still contains nondifferentiable term $\phi(A)$. But different from the original objective function (2), the function ϕ induced nondifferentiable terms has now been transferred from terms involving Θ to terms involving A. Thus we can solve the first equation of (7) by alternating minimization of Θ and A,

$$\begin{cases} \Theta^{k+1} = \arg\min_{\Theta \succ 0} -\log\det\Theta + \mathrm{tr}(S\Theta) + \frac{\lambda_2}{2}\|\Theta\|_F^2 + \mathrm{tr}((M^k)^T(\Theta - A^k)) + \frac{\mu}{2}\|\Theta - A^k\|_F^2, \\ A^{k+1} = \arg\min \lambda_1\phi(A) + \mathrm{tr}((M^k)^T(\Theta^{k+1} - A)) + \frac{\mu}{2}\|\Theta^{k+1} - A\|_F^2, \end{cases} \tag{8}$$

For the method of multipliers, we need to run the alternate minimization multiple times until converge. Note that the first equation of (7) is only one step of the overall iterative algorithm, we don't need to solve it completely. It is called alternating direction method of multipliers [23] or split Bregman method if we use one alternation. That is, we use the following iterations to solve (3)

$$\begin{cases} \Theta^{k+1} = \arg\min_{\Theta \succ 0} -\log\det\Theta + \mathrm{tr}(S\Theta) + \frac{\lambda_2}{2}\|\Theta\|_F^2 + \mathrm{tr}((M^k)^T(\Theta - A^k)) + \frac{\mu}{2}\|\Theta - A^k\|_F^2, \\ A^{k+1} = \arg\min \lambda_1\phi(A) + \mathrm{tr}((M^k)^T(\Theta^{k+1} - A)) + \frac{\mu}{2}\|\Theta^{k+1} - A\|_F^2, \\ M^{k+1} = M^k + \mu(\Theta^{k+1} - A^{k+1}). \end{cases} \tag{9}$$

2.3. Algorithm

2.3.1. Explicit formulas for the iterations (9)

For the second equation of (9), it is easy to get the explicit solution if we know the proximal operator of ϕ. More specifically, denote the *proximal operator* with respect to ϕ as

$$\mathrm{prox}_\phi(Y) = \arg\min_{Z \in \mathbf{R}^{p\times p}} \left(\phi(Z) + \frac{1}{2}\|Y - Z\|_F^2\right).$$

Note that we have $\phi(Z) = \phi(Z^T)$, $\mathrm{prox}_\phi(Y)$ is a symmetric matrix if Y is symmetric. In addition, we introduce $\mathrm{prox}_{\phi_\lambda}$ to be

$$\mathrm{prox}_{\phi_\lambda}(Y) = \arg\min_{Z \in \mathbf{R}^{p\times p}} \left(\phi(Z) + \frac{\lambda}{2}\|Y - Z\|_F^2\right).$$

Then the optimal solution of A in the second equation of (9) is

$$A^{k+1} = \mathrm{prox}_{\phi_{\frac{\lambda_1}{\mu}}}(\Theta^{k+1} + \mu^{-1}M^k). \tag{10}$$

If we can get the update A^{k+1} very quickly, this is always the case for many applications, the efficiency of the iterative algorithm (9) lies greatly on whether the first equation of (9) can be solved quickly.

Now we focus on the first equation of (9). Taking the derivative of its objective function and setting it to be zero, we get

$$-\Theta^{-1} + (\lambda_2 + \mu)\Theta = \mu A^k - S - M^k. \tag{11}$$

Note that $\mu A^k - S - M^k$ will be symmetric from (9) by induction if Θ^k is. Assume that $\mu A^k - S - M^k$ is symmetric. Then it shares the same eigenvectors as Θ and Θ^{-1} according to (11). We take the orthogonal eigenvalue decomposition of the matrix $\mu A^k - S - M^k$,

$$\mu A^k - S - M^k = U\Lambda U^T, \tag{12}$$

where $UU^T = U^TU = I$ and $\Lambda = \mathrm{diag}(\widetilde{\lambda}_1, \ldots, \widetilde{\lambda}_p)$ with each $\widetilde{\lambda}_i$ being the eigenvalue of $\mu A^k - S - M^k$. As Θ and Θ^{-1} shares the same eigenvectors as the ones of $\mu A^k - S - M^k$, we have

$$\Theta = U\widetilde{\Sigma}U^T \qquad \text{and} \qquad \Theta^{-1} = U\widetilde{\Sigma}^{-1}U^T, \tag{13}$$

where $\widetilde{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_p)$ with each σ_i being the eigenvalue of Θ. Replacing (12) and (13) into (11), we get

$$U((\lambda_2 + \mu)\widetilde{\Sigma} - \widetilde{\Sigma}^{-1} - \Lambda)U^T = 0. \tag{14}$$

Together with the fact that $UU^T = U^TU = I$ yields

$$(\lambda_2 + \mu)\widetilde{\Sigma} - \widetilde{\Sigma}^{-1} - \Lambda = 0,$$

or equivalently

$$(\lambda_2 + \mu)\sigma_i - \sigma_i^{-1} - \widetilde{\lambda}_i = 0, \quad i = 1, \ldots, p.$$

As $\widetilde{\lambda}_i$ are known and Θ is supposed to be positive definite (which implies each $\sigma_i > 0$), we can easily get the solutions of σ_i,

$$\sigma_i = \frac{\widetilde{\lambda}_i + \sqrt{\widetilde{\lambda}_i^2 + 4(\lambda_2 + \mu)}}{2(\lambda_2 + \mu)}, \quad i = 1, \ldots, p. \tag{15}$$

Thus we get the explicit solution of (11) by taking use of (12), (13) and (15),

$$\Theta^{k+1} = \frac{\mu A^k - S - M^k + \sqrt{(\mu A^k - S - M^k)^2 + 4(\lambda_2 + \mu)I}}{2(\lambda_2 + \mu)}. \tag{16}$$

With the explicit solution for the update of A^{k+1} in (10) and update of Θ^{k+1} in (16), we get the explicit iteration algorithm for (3) (**Algorithm 1**). Furthermore, the derived Θ^k for each step is positive definite.

Algorithm 1 Split Bregman method for general regularized maximum likelihood graphical model (2) (SBGM)

Initialize Θ^0, A^0, and M^0.
repeat
1) $\Theta^{k+1} = \frac{\mu A^k - S - M^k + \sqrt{(\mu A^k - S - M^k)^2 + 4(\lambda_2 + \mu)I}}{2(\lambda_2 + \mu)}$
2) $A^{k+1} = \text{prox}_{\phi \frac{\lambda_1}{\mu}}(\Theta^{k+1} + \mu^{-1} M^k)$
3) $M^{k+1} = M^k + \mu(\Theta^{k+1} - A^{k+1})$
until
Convergence

2.3.2. Newton method for deriving the square root of a positive definite matrix

From the derivation of the solution of (11), we need to calculate all the eigenvalues and eigenvectors of matrix $\mu A^k - S - M^k$, which is computational demanding when the size of the matrix is large. Note that the main step to get the update of Θ^{k+1} is calculating the square root of the positive definite matrix $(\mu A^k - S - M^k)^2 + 4(\lambda_2 + \mu)I$. In this subsection, we will use Newton method to calculate the square root of a positive definite matrix directly instead of using eigenvalue decomposition of $\mu A^k - S - M^k$. Our numerical experiments shows that our proposed algorithm for calculating the square root of a positive definite matrix saves about half of computational time from the one using eigenvalue decomposition.

We begin with finding the positive root of the equation $x^2 - a = 0 (a > 0)$ using Newton method [32, 33]. That is,

$$x^{k+1} = \frac{1}{2}(x^k + \frac{a}{x^k}). \tag{17}$$

The following lemma ensures that (17) with $x^0 > 0$ always converges to $\sqrt{a}$ and the convergence rate is quadratic.

Lemma 1. *If we choose $x^0 > 0$, then x^k generated by (17) is well-defined and satisfies*

$$|x^{k+1} - \sqrt{a}| \leq \min\left\{\frac{1}{2\sqrt{a}}|x^k - \sqrt{a}|^2, \frac{1}{2}|x^k - \sqrt{a}|\right\}.$$

Proof. Since $x^0 > 0$, x^1 is well-defined and $x^1 = \frac{1}{2}(x^0 + \frac{a}{x^0}) \geq \sqrt{a}$. By induction, x^k is well-defined and $x^k \geq \sqrt{a}$. Moreover, using the iteration (17),

$$|x^{k+1} - \sqrt{a}| = \left|\frac{1}{2}(x^k + \frac{a}{x^k}) - \sqrt{a}\right| = \frac{1}{2|x^k|}|x^k - \sqrt{a}|^2 \leq \frac{1}{2\sqrt{a}}|x^k - \sqrt{a}|^2,$$

and

$$|x^{k+1} - \sqrt{a}| = \frac{1}{2|x^k|}|x^k - \sqrt{a}|^2 = \frac{1}{2}(1 - \frac{\sqrt{a}}{x^k})|x^k - \sqrt{a}| \leq \frac{1}{2}|x^k - \sqrt{a}|.$$

□

Now we apply Newton method to calculate the square root of a positive definite matrix of the form $K+\alpha I$, where K is positive definite and α is a positive constant. Let $X^0 = \sqrt{\alpha}I$ and

$$X^{k+1} = \frac{1}{2}\left(X^k + (X^k)^{-1}(K + \alpha I)\right). \tag{18}$$

Then the iteration (18) always converges quadratically to $\sqrt{K + \alpha I}$. Similar algorithms has been proposed in [34, 35, 33] to compute polar decompositions.

Theorem 1. *If we choose $X^0 = \sqrt{\alpha}I$, then X^k generated by (18) is well-defined and quadratically converges to $\sqrt{K + \alpha I}$. More specifically,*

$$\|X^k - \sqrt{K + \alpha I}\|_2 \leq \min\left\{\frac{1}{2\sqrt{\alpha + \lambda_{\min}(K)}}\|X^k - \sqrt{K + \alpha I}\|_2^2, \frac{1}{2}\|X^k - \sqrt{K + \alpha I}\|_2\right\}, \tag{19}$$

where $\lambda_{\min}(K)$ is the minimum eigenvalue of matrix K.

Proof. Let $K = U\widetilde{\Sigma}U^T$ with $\widetilde{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_p)$ and $UU^T = U^TU = I$. Since $X^0 = \sqrt{\alpha}I$, by induction, X^k can be written as $X^k = U\widetilde{\Sigma}^kU^T$ with $\widetilde{\Sigma}^k = \mathrm{diag}(\sigma_1^k, \ldots, \sigma_p^k)$ and $\sigma_i^k > 0$. Furthermore,

$$X^{k+1} = U\left(\frac{1}{2}\left(\widetilde{\Sigma}^k + (\widetilde{\Sigma}^k)^{-1}(\widetilde{\Sigma} + \alpha I)\right)\right)U^T.$$

Therefore, (18) changes only the singular values which are governed by (17). The Theorem follows immediately from Lemma 1. □

Note that the choice of the initial guess X^0 could be any diagonal matrix with positive entries and the convergence result is still true.

2.3.3. Convergence Analysis

The convergence of Algorithm 1 can be derived from the convergence theory of the alternating direction method of multipliers or the convergence of split Bregman method [23, 36, 15].

Theorem 2. *Suppose that there exists at least one minimizer Θ^* of (2). Then the following property for Algorithm 1 holds:*

$$\begin{aligned}\lim_{k\to\infty} &-\log\det\Theta^k + tr(S\Theta^k) + \lambda_1\phi(\Theta^k) + \frac{\lambda_2}{2}\|\Theta^k\|_F^2 \\ &= -\log\det\Theta^* + tr(S\Theta^*) + \lambda_1\phi(\Theta^*) + \frac{\lambda_2}{2}\|\Theta^*\|_F^2.\end{aligned}$$

Furthermore,

$$\lim_{k\to\infty}\|\Theta^k - \Theta^*\| = 0$$

whenever (2) has a unique solution.

From Theorem 2, the condition for the convergence of Algorithm 1 is quite mild and even irrelevant to the choice of the parameter μ in Algorithm 1. This might be another reason why the split Bregman method is so popular recently.

3. Example—Split Bregman method for ℓ_1 penalized maximum likelihood graphical model (SBGLasso)

The ℓ_1 penalized maximum likelihood graphical model (1) was first proposed by Yuan and Lin [2]. It is a special case of our general model (2) with $\phi(\Theta) = \sum_{i\neq j} |\Theta_{ij}|$ and $\lambda_2 = 0$. Algorithm 1 can be used here with minor changes. The update of Θ is exactly the same as (16) except that we choose $\lambda_2 = 0$. For the update of A, we only need to know the proximal operator of $\sum_{i\neq j} |\Theta_{ij}|$, which has an explicit formula and very easy to compute. More specifically, let $\mathcal{T}_\lambda$ be a soft thresholding operator defined on matrix space and satisfying

$$\mathcal{T}_\lambda(\Omega) = (t_\lambda(\omega_{ij}))_{i,j=1}^p, \quad \text{with} \quad t_\lambda(\omega_{ij}) = \text{sgn}(\omega_{ij}) \max\{0, |\omega_{ij}| - \lambda\}.$$

Then the update of A is

$$A^{k+1} = \mathcal{T}_{\frac{\mu}{\lambda_1}}(\Theta^{k+1} + \mu^{-1} M^k).$$

Combining altogether, we get Algorithm 2 for solving (1).

Algorithm 2 Split Bregman method for ℓ_1 penalized maximum likelihood graphical model (SBGLasso)

Initialize Θ^0, A^0, and M^0.
repeat
1) Compute $K^k = \mu A^k - S - M^k$
1) Use Newton method to compute $X^k = \sqrt{(K^k)^2 + 4\mu I}$
1) $\Theta^{k+1} = \frac{K^k + X^k}{2\mu}$
2) $A^{k+1} = \mathcal{T}_{\frac{\lambda_1}{\mu}}(\Theta^{k+1} + \mu^{-1} M^k)$
3) $M^{k+1} = M^k + \mu(\Theta^{k+1} - A^{k+1})$
until
Convergence

4. Numerical experiments

In this section, we would use time trial to illustrate the efficiency of our proposed algorithms including the Newton method for computing the square root of a positive definite matrix and split Bregman method for ℓ_1 penalized maximum likelihood graphical model (SBGLasso). Time trials were generated on an Intel Core 2 Duo desktop PC (E7500, 2.93GHz).

4.1. Newton method versus eigenvalue decomposition method for computing the square root of a positive definite matrix

We first illustrate the efficiency of the Newton method for computing the square root of a positive definite matrix of form $M = K^2 + \mu I$, where $K = (B + B')/2$ and B is a $p \times p$ random Gaussian matrices with its entries randomly drawn from the standard Gaussian distribution. The form of the matrix M we choose is mostly motivated by the square root we need to solve in the equation (16). Our algorithm is implemented in Matlab using mex programming. We compare it with the method using matlab function "schur" on the matrix K. More specifically, we use the two-line code "$[U, D] = schur(K); SR = U * diag(sqrt(diag(D).\hat{}2 + \mu)) * U'$" to compute the square root of M. Note that the upper triangular matrix produced by schur decomposition is exactly diagonal since K is symmetric.

In Table 1, we have listed the number of iterations needed for our Newton method to solve the square root of M, where we stop the iterations whenever the relative change of two successive steps is less than 10^{-6}. We also calculated the relative difference between the result derived by our method and the one by schur decomposition. As shown in Table 1, the relative difference is of precision of order 10^{-9}, which shows our algorithm is quite accurate. Empirically, our algorithm only need 8 steps to converge and it keeps constant as the matrix size increases, which illustrate the quadratic convergence rate of our algorithm given in Theorem 1. Most important of all, our algorithm is much faster and saves more than 50% computational

time from the method via schur decomposition for all the examples we tested. For example, when $n = 2000$, the computational time for our algorithm is 10.86 seconds, and that for the method via schur decomposition is 38.23.

TABLE 1
Experimental results for computing the square root of of M. All results are averages of 10 runs.

p	Newton method		Schur decomposition	relative
	iters in Newton method	total time(s)	total time(s)	difference
1000	8	1.71	4.71	1.87×10^{-13}
1500	8	4.79	16.37	9.47×10^{-13}
2000	8	10.96	38.23	3.11×10^{-11}
2500	8	20.72	75.25	3.73×10^{-10}
3000	8	35.60	128.74	2.38×10^{-9}

4.2. SBGlasso versus graphical lasso for ℓ_1 penalized maximum likelihood graphical model

Here we illustrate the efficiency of the split Bregman method for ℓ_1 penalized maximum likelihood graphical model (SBGLasso) using time trials. As graphical lasso proposed by Friedman et al. [10] is the most efficient algorithm for solving (1) as far as we know, we focus on our comparison with it. Our algorithm was coded in C, linked to a Matlab function, thus it is fair to compare with graphical lasso which was coded in Fortran and linked to an R language function using time trials.

The stopping criteria of SBGLasso is specified as follows. Let $\Phi(\Theta^k) = -\log\det\Theta^k + \mathrm{tr}(S\Theta^k) + \lambda\sum_{i\neq j}|\Theta^k_{ij}|$. Since ℓ_1 penalized maximum likelihood graphical model (1) is a special case of model (2), we can get $\lim_{k\to\infty}\Phi(\Theta^k) = \Phi(\Theta^*)$ by Theorem 2. It is reasonable to terminate the algorithm when the relative change of the energy functional $\Phi(\Theta)$ falls below certain threshold δ. Furthermore, Algorithm 2 is also solving (3), the linear constraint $\Theta = A$ is satisfied when it converges. Therefore, we also expect the relative difference $\|\Theta - A\|_F/\|\Theta\|_F$ is less than δ. We used $\delta = 10^{-4}$ in our simulation. For the graphical lasso, we also set the termination threshold to be 10^{-4}.

Note that the convergence of Algorithm 2 is guaranteed no matter what values of μ are chosen as shown in Theorem 2. However, the choice of μ can influence the speed of the algorithm since it would affect the number of iterations involved. In our implementation, we found empirically that choosing $\mu = 0.5$ works well for all the problems we tested.

We generate the underlying sparse inverse covariance matrices as the way described in [9]. We first randomly choose a $p \times p$ diagonal matrix with positive diagonal entries. Then approximately p nonzeros are inserted in the matrix at random locations symmetrically. Positive definiteness is ensured by adding a multiple of the identity to the matrix if needed. In our numerical experiments, we added $2I$ to the matrix. The penalty parameter is chosen such that the solution had about the actual number of nonzero elements. Table 2 shows computational time as well as relative error for SBGLasso and graphhical lasso. Our algorithm is faster than graphical lasso for large scale data ($p \geq 1000$) while achieving similar accuracy. It saves more than 30% of computational time from graphical lasso. For example, when $n = 2000, p = 3000$, it takes 458.66 seconds for our algorithm, while it takes 729.63 for graphical lasso. For the medium scale problem such as $p = 500$, the speed of our algorithm is still very fast and comparable with graphical lasso.

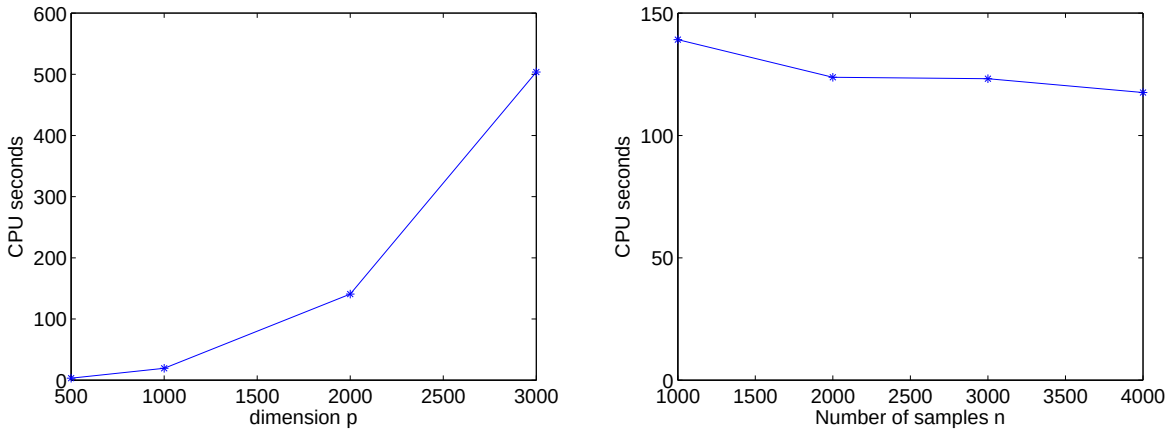
To evaluate how the performance of SBGLasso scales with problem size, we plotted the CPU time that SBGLasso took to solve the problem (1) as a function of p and n. We note that the CPU times is roughly quadratic in p and constant in n. This phenomena is reasonable since the unknown concentration matrix has $\frac{p(p+1)}{2}$ unknowns which is quadratic with respect to p, while the number of samples only appears in S which will not influence the speed of SBGLasso so much.

References

[1] S.L. Lauritzen. *Graphical models*. Oxford University Press, USA, 1996. ISBN 0198522193.

TABLE 2
Timings (seconds) and relative error for SBGLasso and graphical lasso

n,p	SBFGLasso		Graphical Lasso	
	total time (s)	relative error	total time(s)	relative error
$n = 1000, p = 500$	3.12	0.1135	1.94	0.1136
$n = 1000, p = 1000$	19.34	0.1192	27.08	0.1193
$n = 1000, p = 2000$	140.61	0.1170	217.88	0.1170
$n = 2000, p = 3000$	458.66	0.0980	729.63	0.0979



(a) (b)

Fig 1: CPU times for SBGLasso for the same problem as in Table 2, for different values of n and p. (a) n is fixed and equals to 1000; (b) p is fixed and equals to 2000.

[2] M. Yuan and Y. Lin. Model selection and estimation in the gaussian graphical model. *Biometrika*, 94 (1):19–35, 2007. ISSN 0006-3444.
[3] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.
[4] A.P. Dempster. Covariance selection. *Biometrics*, 28(1):157–175, 1972. ISSN 0006-341X.
[5] D. Edwards. *Introduction to graphical modelling*. Springer Texts in Statistics. Springer-Verlag, New York, second edition, 2000. ISBN 0-387-95054-0.
[6] J. Peng, P. Wang, N. Zhou, and J. Zhu. Partial correlation estimation by joint sparse regression models. *Journal of the American Statistical Association*, 104(486):735–746, 2009. ISSN 0162-1459.
[7] T. Hastie, R. Tibshirani, and J.H. Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Verlag, 2009. ISBN 0387848576.
[8] N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the lasso. *The Annals of Statistics*, 34(3):1436–1462, 2006. ISSN 0090-5364.
[9] O. Banerjee, L. El Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *The Journal of Machine Learning Research*, 9: 485–516, 2008. ISSN 1532-4435.
[10] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008. ISSN 1465-4644.
[11] L. M. Brègman. A relaxation method of finding a common point of convex sets and its application to the solution of problems in convex programming. *Ž. Vyčisl. Mat. i Mat. Fiz.*, 7:620–631, 1967. ISSN 0044-4669.
[12] A. E. Çetin. Reconstruction of signals from Fourier transform samples. *Signal Process.*, 16(2):129–148, 1989. ISSN 0165-1684.
[13] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin. An iterative regularization method for total variation-based image restoration. *Multiscale Model. Simul.*, 4(2):460–489 (electronic), 2005. ISSN 1540-3459.
[14] T. Goldstein and S. Osher. The split Bregman method for $L1$-regularized problems. *SIAM J. Imaging Sci.*, 2(2):323–343, 2009. ISSN 1936-4954.
[15] J.-F. Cai, S. Osher, and Z. Shen. Split bregman methods and frame based image restoration. *Multiscale Model. Simul.*, 8(2):337–369, 2009.
[16] J.-F. Cai, S. Osher, and Z. Shen. Linearized Bregman iterations for compressed sensing. *Math. Comp.*, 78(267):1515–1536, 2009.
[17] S. Osher, Y. Mao, B. Dong, and W. Yin. Fast linearized Bregman iteration for compressive sensing and sparse denoising. *commu. Math. Sci.*, 8(2):93–111, 2010.
[18] W. Yin, S. Osher, D. Goldfarb, and J. Darbon. Bregman iterative algorithms for l_1-minimization with applications to compressed sensing. *SIAM J. Imaging Sci.*, 1(1):143–168, 2008. ISSN 1936-4954.
[19] J.-F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM J. Optim.*, 20(4):1956–1982, 2010. ISSN 1052-6234.
[20] E.J. Candes, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Arxiv preprint arXiv:0912.3599*, 2009.
[21] G.B. Ye and X. Xie. Split bregman method for large scale fused lasso. *Computational Statistics and Data Analysis*, 2010. .
[22] C. Wu and X.C. Tai. Augmented Lagrangian method, dual methods, and split Bregman iteration for ROF, vectorial TV, and high order models. *SIAM Journal on Imaging Sciences*, 3:300, 2010.
[23] D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers and Mathematics with Applications*, 2(1):17–40, 1976.
[24] R. Glowinski and A. Marroco. Sur l'approximation, par éléments finis d'ordre un, et la résolution, par penalisation-dualité, d'une classe de problèmes de dirichlet non linéaires. *Rev. Franc. Automat. Inform. Rech. Operat.*, pages 41–76, 1975.
[25] R. T. Rockafellar. A dual approach to solving nonlinear programming problems by unconstrained optimization. *Math. Programming*, 5:354–373, 1973. ISSN 0025-5610.
[26] J.-F. Cai, S. Osher, and Z. Shen. Linearized Bregman iterations for frame-based image deblurring. *SIAM J. Imaging Sci.*, 2(1):226–252, 2009.
[27] X. Zhang, M. Burger, X. Bresson, and S. Osher. Bregmanized nonlocal regularization for deconvolution

and sparse reconstruction. *SIAM Journal on Imaging Sciences*, 3:253–276, 2010.

[28] S. Setzer. Split bregman algorithm, douglas-rachford splitting and frame shrinkage. *Scale space and variational methods in computer vision*, pages 464–476, 2009.

[29] X. Zhang, M. Burger, and S. Osher. A unified primal-dual algorithm framework based on bregman iteration. *Journal of Scientific Computing*, 2010. .

[30] M. R. Hestenes. Multiplier and gradient methods. *J. Optimization Theory Appl.*, 4:303–320, 1969. ISSN 0022-3239.

[31] Dimitri P. Bertsekas. *Constrained optimization and Lagrange multiplier methods*. Computer Science and Applied Mathematics. Academic Press Inc. [Harcourt Brace Jovanovich Publishers], New York, 1982. ISBN 0-12-093480-9.

[32] R.L. Burden and J.D. Faires. *Numerical Analysis*. 8th edition, Brooks/Cole, 2004.

[33] J.F. Cai and S. Osher. Fast singular value thresholding without singular value decomposition. 2010.

[34] N.J. Higham. Computing the polar decomposition- with applications. *SIAM J. Sci. Stat. Comput.*, 7 (4):1160–1174, 1986.

[35] N.J. Higham and S.S. Robert. Fast polar decomposition of an arbitrary matrix. In *SIAM J. Sci. Stat. Comput*. Citeseer, 1990.

[36] J. Eckstein and D.P. Bertsekas. On the douglas rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1):293–318, 1992. ISSN 0025-5610.