Appl Intell (2018) 48:1933–1952
DOI 10.1007/s10489-016-0850-7

Comparative study of computational algorithms for the Lasso with high-dimensional, highly correlated data

Baekjin Kim[1] · **Donghyeon Yu**[2] · **Joong-Ho Won**[1]

Published online: 20 October 2016

Abstract Variable selection is important in high-dimensional data analysis. The Lasso regression is useful since it possesses sparsity, soft-decision rule, and computational efficiency. However, since the Lasso penalized likelihood contains a nondifferentiable term, standard optimization tools cannot be applied. Many computation algorithms to optimize this Lasso penalized likelihood function in high-dimensional settings have been proposed. To name a few, coordinate descent (CD) algorithm, majorization-minimization using local quadratic approximation, fast iterative shrinkage thresholding algorithm (FISTA) and alternating direction method of multipliers (ADMM). In this paper, we undertake a comparative study that analyzes relative merits of these algorithms. We are especially concerned with numerical sensitivity to the correlation between the covariates. We conduct a simulation study considering factors that affect the condition number of covariance matrix of the covariates, as well as the level of penalization. We apply the algorithms to cancer biomarker discovery, and compare convergence speed and stability.

✉ Joong-Ho Won
wonj@stats.snu.ac.kr

Baekjin Kim
skybackj@snu.ac.kr

Donghyeon Yu
dyu3@kmu.ac.kr

1 Department of Statistics, Seoul National University, Seoul, Korea

2 Department of Statistics, Keimyung University, Daegu, Korea

Keywords Lasso · Majorization-minimization · Coordinate descent · ADMM · FISTA

1 Introduction

Variable selection in high-dimensional data analysis is an important issue in various fields of science including genetics, compressed censing, and machine learning. For instance, if we want to detect a disease from a person's gene expressions measured by microarray, tens of thousands of genes are possible predictors while only a few of them are relevant to the disease. A systematic method to select these genes are in need.

Statistically, this is a variable or model selection problem in regression. A classical approach is to apply the best subset method, which selects the best model by examining the models created from all possible combinations of the predictor variables based on a predictive performance criterion such as the Akaike information criterion (AIC) [1], the Bayesian information criterion (BIC) [19], or the adjusted coefficient of determination (R^2). Applied to high-dimensional data, however, best subset method becomes erratic because of the unavoidable multicollinearity among the predictor variables. A reason for this instability is that the best subset method selects models in a discrete manner, hence the selected models may be very different in the presence of noise. Furthermore, the number of subsets of variables to be compared increases exponentially as the number of predictor variables becomes larger. To overcome these difficulties, "soft" versions of variable selection methods that continuously select models, have been extensively studied. Among these, the Lasso, which penalizes the sum of residual squares or other loss functions by the l_1 norm of the regression coefficients, has received a considerable attention [23]. The l_1 norm penalty makes many of the estimated coefficients exactly zero, which is why this method is broadly used for variable selection in high-dimensional data. Furthermore, the Lasso allows the coefficients to vary continuously with the level of regularization, rendering itself a stable soft model selector. The Lasso also shrinks the coefficients toward zero and thus exhibits a better predictive performance than ordinary least square estimators.

For these reasons, the Lasso has found many applications, including face recognition [26, 28], speech recognition [9], and estimation of gene networks [29].

Computationally, the Lasso problem is to minimize the objective function

$$f(\beta) = (1/2)\|y - X\beta\|_2^2 + \lambda\|\beta\|_1, \tag{1}$$

where y is an $n \times 1$ vector of the observations, X is an $n \times p$ data matrix of the levels of the predictor variables, and $\lambda > 0$ is the regularization parameter; β is a p-dimensional vector of the regression coefficients, which is the optimization variable. We wish to find the vector $\hat{\beta}$ that minimizes $f(\beta)$. The l_1 norm penalty function is non-differentiable at the origin, and standard optimization methods based on derivatives such as the Newton-Raphson and the gradient descent methods cannot be used. Various algorithms to compute the Lasso solutions have been studied. Among these, the homotopy algorithm [6], while widely used for problems of small to moderate dimensions, is often inapplicable to high-dimensional problems because it requires repetitive Cholesky decompositions. To meet the needs of efficient and stable computation, iterative methods, which reduce the original optimization problem to a series of simpler problems and successively refine solutions, have been of interest. Many algorithms have been developed, including the majorization-minimization using local quadratic approximation (MM-LQA) algorithm [12], alternating direction method of multipliers (ADMM) [4], fast iterative shrinkage thresholding algorithm (FISTA) [2]. For individual algorithm, its scalability for the high-dimensional data and applicability for various types of penalties are well-studied. There are a few of comparative studies for the computation times of the algorithms as well [8, 27, 28]. However, studies that compare the merits and the drawbacks of these algorithms in a systematic fashion are scarce.

In this paper, we conduct studies that compare computational algorithms for high-dimensional Lasso, to find the factors that affect the performance and convergence of the algorithms. In practice, covariates in regression analysis for the high-dimensional data usually contain highly correlated variables. While there exist other penalized statistical procedures, it is worth investigating the numerical sensitivity of algorithms to the correlation between the covariates.

We compare four computational algorithms suitable for high-dimensional Lasso, that is, the coordinate descent (CD) algorithm (shooting algorithm), MM-LQA, FISTA, and ADMM. We analyze the sensitivity of these algorithms to the correlation between the covariates to see the relative merits of each algorithm. We found that MM-LQA and ADMM are quite robust to correlation. These algorithms also show stable convergence for a wide range of the regularization parameter. On the contrary, CD and FISTA may behave unstably when the correlation becomes high; this unstability is improved as regularization parameter gets larger. We also report empirical conditions for the CD algorithm to fail to converge.

Section 2 reviews the four algorithms for solving Lasso. In Section 3 we conduct the comparative numerical study. In Section 4, we apply the four algorithms to a study for cancer biomarker discovery, and compare their performance.

2 Preliminaries

In this section, we provide an overview of the iterative optimization methods to be compared in the following sections.

2.1 Coordinate descent algorithm (CD)

CD is devised to minimize an objective function that is separable with respect to each coordinate of the variable. It iteratively updates the variable from $\beta^{(k)}$ to $\beta^{(k+1)}$ by choosing a single coordinate, and then performs univariate minimization with the other coordinates held fixed at the values of the previous step. In other words, write $f(\beta) = f(\beta_1, \cdots, \beta_p)$. At the k-th iteration, choose an appropriate $j \in \{1, \cdots p\}$, and set

$$\beta_j^{(k+1)} = \underset{\beta_j \in \mathbb{R}}{\operatorname{argmin}}\, f(\beta_1^{(k+1)}, ..., \beta_{j-1}^{(k+1)}, \beta_j, \beta_{j+1}^{(k)}, ..., \beta_p^{(k)}).$$

For the Lasso, the objective function is convex and separable because so are the sum of residual squared and the l_1 norm, rendering the following decomposition possible,

$$\begin{aligned} f(\beta) &= (1/2)\|y - X\beta\|^2 + \lambda\|\beta\|_1 \\ &= (1/2)\textstyle\sum_{i=1}^n (y_i - \sum_{j=1}^p X_{ij}\beta_j)^2 + \lambda \sum_{j=1}^p |\beta_j| \\ &= (1/2)\textstyle\sum_{i=1}^n (y_i - \sum_{l \neq j} X_{il}\beta_l - X_{ij}\beta_j)^2 \\ &\quad +\lambda \textstyle\sum_{l\neq j} |\beta_l| + \lambda|\beta_j|. \end{aligned} \tag{2}$$

If all the coefficients except for β_j are fixed at $\left\{\tilde{\beta}_l, l \neq j\right\}$, (2) becomes a convex, piecewise quadratic function in β_j. This univariate function is minimized by

$$\beta_j^* = \frac{S_\lambda(\sum_{i=1}^n X_{ij}(y_i - \tilde{y}_{i(j)}))}{\sum_{i=1}^n X_{ij}^2}$$

where $\tilde{y}_{i(j)} = \sum_{l \neq j} X_{il}\tilde{\beta}_l$, and $S_\lambda(\cdot)$ is the soft-thresholding operator:

$$S_\lambda(x) = \text{sign}(x) \cdot \max(|x| - \lambda, 0). \tag{3}$$

This coordinate-wise minimization step is repeated with respect to all j $(1 \leq j \leq p)$ until relative error is below determined tolerance, making a full iteration. The resulting algorithm for the Lasso is as follows.

Algorithm 1 CD algorithm

```
1: Initialize $\beta^{(0)}, \lambda > 0, \tilde{\beta}^{(0)} = \beta^{(0)}$
2: **while** termination condition unsatisfied **do**
3:     Choose index $j \in \{1, 2, \cdots, p\}$
4:     **for** $i = 1, \cdots, n$ **do**
5:         $\tilde{y}_{i(j)}^{(k+1)} = \sum_{l \neq j} X_{il}\tilde{\beta}_l^{(k)}$
6:     **end for**
7:     $\beta_j^{(k+1)} = S_\lambda(\sum_{i=1}^n X_{ij}(y_i - \tilde{y}_{i(j)}^{(k+1)}))/\sum_{i=1}^n X_{ij}^2$
8:     $\beta^{(k+1)} = \beta^{(k)} + (\beta_j^{(k+1)} - \beta_j^{(k)})e_j$, where $e_j$ is the
       $j$th column of the $p \times p$ identity matrix
9:     $\tilde{\beta}^{(k+1)} = \beta^{(k+1)}$
10: **end while**
```

Since each coordinate-wise minimization step is very efficiently computed, Friedman et al. [8], Wu and Lange [27] devised path algorithms based on CD with the warm start strategy that successively use the previous solution as an initial value of the current solution in the path. They report the CD-based path algorithms show better performance than LARS [6] in high dimensional settings.

CD can be accelerated by the active shooting strategy [17], which reduces the computational cost by maintaining a set of nonzero solutions, called an active set. The detailed steps are described as follows.

Algorithm 2 CD algorithm with active shooting

```
1: Initialize $\beta^{(0)}, \lambda > 0, \tilde{\beta}^{(0)} = \beta^{(0)}$
2: **while** termination condition unsatisfied **do**
3:     Define an active set $\Lambda = \{j | \beta_j^{(k)} \neq 0\}$
4:     Apply the CD algorithm with the indices in $\Lambda$
5:     Repeat Step 4 until convergence occurs
6:     Apply the CD algorithm with the whole indices
7: **end while**
```

2.2 Majorization-Minimization using Local Quadratic Approximation (MM-LQA)

The majorization-minimization (MM) algorithm is based on the idea of iteratively minimizing series of surrogate functions, which majorizes the objective function $f(\beta)$ at the current iteration point. A surrogate function at the current point $\beta^{(k)}$, denoted by $f^*(\beta, \beta^{(k)})$, must majorize $f(\beta)$. Majorization is defined as the following property: a function $f^*(\cdot, \cdot)$ majorizes $f(\cdot)$ at $\beta^{(k)}$ if

$$f(\beta) \leq f^*(\beta, \beta^{(k)}), \forall \beta \in \mathbb{R}^p$$

with equality holding when $\beta = \beta^{(k)}$.

Let $\beta^{(k+1)}$ denote a minimizer of $f^*(\beta, \beta^{(k)})$. The mapping $\beta^{(k)} \to \beta^{(k+1)}$ generates a sequence $\{\beta^{(k)}\}$ for which the sequence of objective function values $\{f(\beta^{(k)})\}$ is non-increasing. In particular, we have

$$f(\beta^{(k+1)}) \leq f^*(\beta^{(k+1)}, \beta^{(k)}) \leq f^*(\beta^{(k)}, \beta^{(k)}) = f(\beta^{(k)})$$

This descent property provides a basis for the numerical stability of the MM algorithm. Choosing an appropriate majorizing function is crucial for the success of an MM algorithm [11].

For the Lasso, a majorizing function can be constructed from a local quadratic approximation (LQA) of the l_1 norm. Recall that

$$\|\beta\|_1 = \sum_{j=1}^p |\beta_j|.$$

Then it can be easily seen that the quadratic function in x

$$p^*(x, x_0) = |x_0| + \frac{x^2 - x_0^2}{2|x_0|}, \quad x, x_0 \in \mathbb{R}$$

majorizes $|x|$ at $x_0 \neq 0$, hence $\sum_{j=1}^p p^*(\beta_j, \beta_j^{(k)})$ majorizes $\|\beta\|_1$. It immediately follows that

$$f^*(\beta, \beta^{(k)}) = \frac{1}{2}\|y - X\beta\|_2^2 + \lambda \sum_{j=1}^p p^*(\beta_j, \beta_j^{(k)})$$

majorizes $f(\beta)$ at $\beta^{(k)} \neq 0$. Unfortunately, this majorizing function is undefined at $\beta^{(k)} = 0$. To overcome this difficulty, Hunter and Li [12] propose to minimize a slightly perturbed version of (1):

$$f_\epsilon(\beta) = \frac{1}{2}\|y - X\beta\|_2^2 + \lambda \sum_{j=1}^p p_\epsilon(\beta_j)$$

where

$$p_\epsilon(\beta_j) = |\beta_j| - \epsilon \int_0^{|\beta_j|} \frac{1}{\epsilon + t} dt, \quad \epsilon > 0$$

It can be shown that

$$f^{**}(\beta, \beta^{(k)}) = \frac{1}{2}\|y - X\beta\|_2^2 + \lambda \sum_{j=1}^p p_\epsilon^{**}(\beta_j, \beta_j^{(k)})$$

majorizes $f_\epsilon(\beta)$ at $\beta^{(k)}$, where

$$p_\epsilon^{**}(\beta_j, \beta_j^{(k)}) = p_\epsilon(\beta_j^{(k)}) + \frac{\beta_j^2 - (\beta_j^{(k)})^2}{2(|\beta_j^{(k)}| + \epsilon)}.$$

This majorizing function is defined for all $\beta \in \mathbb{R}^p$. The optimality condition to minimize $f^{**}(\beta, \beta^{(k)})$ for β is given by

$$(X^T X + D^{(k)})\beta = X^T y, \tag{4}$$

where $D^{(k)}$ is a diagonal matrix, $D_{ii}^{(k)} = \lambda/(|\beta_i^{(k)}| + \epsilon)$.

To solve this linear equation, the preconditioned conjugate gradient (PCG) algorithm is utilized due to its well-known efficiency. The PCG algorithm is also used for the ADMM introduced in the sequel; see Appendix for the details of the PCG algorithm.

Algorithm 3 MM-LQA algorithm

```
1: Initialize $\beta^{(0)} \in \mathbb{R}^p, \epsilon > 0$
2: while termination condition unsatisfied do
3:     for $i = 1, \cdots, p$ do
4:         $d_i^{(k)} = \lambda/(|\beta_i^{(k)}| + \epsilon)$
5:     end for
6:     $M^{(k)} = X^T X + \mathrm{diag}(d_1^{(k)}, \cdots, d_p^{(k)})$
7:     Solves $M^{(k)}\beta^{(k+1)} = X^T y$
8: end while
```

2.3 Fast iterative shrinkage thresholding algorithm (FISTA)

FISTA is a variant of the proximal gradient algorithm [16] applied to the Lasso [2]. Proximal algorithms minimize an objective function of the form of $f(\beta) = g(\beta) + h(\beta)$, $\beta \in \mathbb{R}^p$. Typically, $h(\beta)$ is convex and non-smooth and $g(\beta)$ is smooth convex function whose gradient is Lipschitz continuous with Lipschitz constant L. These algorithms have been applied to problems difficult to solve with standard smooth optimization methods since it can deal with non-smooth function [16].

The proximal gradient algorithm successively applies the proximal operator, defined below, to a quadratic majorizing function $Q_L(\beta, \beta^{(k)})$ of $f(\beta)$ at $\beta = \beta^{(k)}$:

$$\begin{aligned} Q_L(\beta, \beta^{(k)}) &= g(\beta^{(k)}) + \langle \beta - \beta^{(k)}, \nabla g(\beta^{(k)})\rangle \\ &\quad + (L/2)\|\beta - \beta^{(k)}\|^2 + h(\beta) \end{aligned}$$

The proximal operator for Q_L is its minimizer in β:

$$\begin{aligned} p_L(\beta^{(k)}) &= \operatorname{argmin} Q_L(\beta, \beta^{(k)}) \\ &= \operatorname{argmin}_\beta h(\beta) \\ &\qquad + (L/2)\|\beta - (\beta^{(k)} - (1/L)\nabla g(\beta^{(k)}))\|^2 \end{aligned}$$

Applying the map $\beta^{(k+1)} = p_L(\beta^{(k)})$ iteratively generates a sequence of non-increasing objective function values $\{f(\beta^{(k)})\}$, yielding the proximal gradient algorithm [16]. This is another instance of the MM algorithm.

Recall that for the Lasso, the objective function

$$f(\beta) = \frac{1}{2}\|y - X\beta\|^2 + \lambda\|\beta\|_1,$$

can be divided into two parts:

$$g(\beta) = \frac{1}{2}\|y - X\beta\|^2,$$

and

$$h(\beta) = \lambda\|\beta\|_1.$$

Accordingly, the proximal operator p_L reduces to the soft-thresholding operator and the updating mechanism for β is given by

$$\beta^{(k+1)} = \mathbf{S}_{\lambda/L}(\beta^{(k)} - (1/L)X^T(X\beta^{(k)} - y)), \tag{5}$$

where $\mathbf{S}_{\lambda/L}$ is the vector version of the soft-thresholding operator in which (3) is applied element-wisely. This is the iterative shrinkage thresholding algorithm (ISTA) [16]. FISTA is a variation of ISTA with an accelerated convergence rate. In this algorithm, a real sequence $\{t_k\}$ is updated by the formula

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2}$$

and $\beta^{(k+1)}$ is updated by a point interpolating $\beta^{(k)}$ and $\beta^{(k-1)}$ depending on the value of $t^{(k)}$. In ISTA, the update for β depends only upon its previous value $\beta^{(k)}$, while in FISTA it depends on the two previous values. In general, ISTA iteration converges at the rate of $O(1/k)$, whereas, FISTA converges at the rate of $O(1/k^2)$, achieving faster convergence.

In practice, the Lipschitz constant L is difficult to find or expensive to compute, and backtracking is used to determine a reasonable upper bound. The resulting is given in Algorithm 4.

Algorithm 4 FISTA with backtracking

```
1: Initialize $L^{(0)} > 0$, some $\eta > 1$, and $\beta^{(0)} \in \mathbb{R}^p$. Set
   $\gamma^{(1)} = \beta^{(0)}, t_1 = 1$
2: while termination condition unsatisfied do
3:     Find the smallest nonnegative integers $i_k$ such that
       $\bar{L} = \eta^{i_k} L^{k-1}$ and
       $f(p_{\bar{L}}(\gamma^{(k)})) \le Q_{\bar{L}}(p_{\bar{L}}(\gamma^{(k)}), \gamma^{(k)})$
4:     Set $L^{(k)} = \eta^{i_k} L^{k-1}$
5:     $\beta^{(k+1)} = \mathbf{S}_{\lambda/L^{(k)}}(\beta^{(k)} - (1/L^{(k)})X^T(X\gamma^{(k)} - y))$
6:     $t_{k+1} = (1 + \sqrt{1 + 4t_k^2})/2$
7:     $\gamma^{(k+1)} = \beta^{(k)} + (t_k - 1)/t_{k+1}(\beta^{(k)} - \beta^{(k-1)})$
8: end while
```

2.4 Alternating direction method of multipliers (ADMM)

ADMM is devised to minimize objective function that can be represented as a sum of two convex functions, similar to FISTA. In general, one of the two convex functions is smooth and the other is non-smooth. For the Lasso,

$$f(\beta) = \frac{1}{2}\|y - X\beta\|^2 + \lambda\|\beta\|_1 = g(\beta) + h(\beta),$$

where

$$\begin{aligned} g(\beta) &= \frac{1}{2}\|y - X\beta\|_2^2 = \frac{1}{2}\beta^T X^T X\beta - y^T X\beta + \frac{1}{2}y^T y, \\ h(\beta) &= \lambda\|\beta\|_1. \end{aligned}$$

Minimizing $f(\beta)$ is equivalent to the following constrained optimization problem

$$\text{minimize } g(\beta) + h(\gamma) \quad \text{subject to } \beta = \gamma$$

To solve this, the augmented Lagrangian

$$L_\mu(\alpha, \beta, \gamma) = g(\beta) + h(\gamma) + \alpha^T(\gamma - \beta) + \frac{\mu}{2}\|\beta - \gamma\|_2^2$$

is constructed. Here $\mu > 0$ is a small fixed parameter. The quadratic term involving μ enforces the constraint in a smoother fashion than the ordinary Lagrangian ($\mu = 0$). The multiplier α is the dual variable.

The saddle point condition is given by

$$\begin{aligned} (\beta^*, \gamma^*) &= \operatorname*{argmin}_{\beta,\gamma} L_\mu(\alpha^*, \beta, \gamma) \quad \text{(primal)} \\ \alpha^* &= \operatorname*{argmax}_{\alpha} L_\mu(\alpha, \beta^*, \gamma^*) \quad \text{(dual)} \end{aligned} \tag{6}$$

ADMM solves (6) by a fixed-point iteration scheme, for which the dual variable is updated by gradient ascent. Specifically, the scheme consists of

- β-update

$$\begin{aligned} \beta^{(k+1)} &= \operatorname{argmin}_\beta L_\mu(\alpha^{(k)}, \beta, \gamma^{(k)}) \\ &= \operatorname{argmin}_\beta \beta^T X^T X\beta/2 - y^T X\beta \\ &\qquad -\alpha^{(k)T}\beta + (\mu/2)\|\beta - \gamma^{(k)}\|_2^2 \\ &= (X^T X + \mu I)^{-1}(X^T y + \mu\gamma^{(k)} + \alpha^{(k)}) \end{aligned} \tag{7}$$

  In computing the final equation, PCG algorithm in Appendix can be employed.
- γ-update

$$\begin{aligned} \gamma^{(k+1)} &= \operatorname{argmin}_\gamma L_\mu(\alpha^{(k)}, \beta^{(k+1)}, \gamma) \\ &= \operatorname{argmin}_\gamma \lambda\|\gamma\|_1 + \alpha^{(k)T}\gamma \\ &\qquad +(\mu/2)\|\beta^{(k+1)} - \gamma\|_2^2 \\ &= \mathbf{S}_{\lambda/\mu}(\beta^{(k+1)} + \alpha^{(k)}/\mu) \end{aligned} \tag{8}$$

- α-update

$$\alpha^{(k+1)} = \alpha^{(k)} + \mu(\beta^{(k+1)} - \gamma^{(k+1)}) \tag{9}$$

Algorithm 5 ADMM algorithm

```
1: Initialize : $\mu > 0, \alpha^{(0)}, \beta^{(0)}, \gamma^{(0)} \in \mathbb{R}^p$.
2: while termination condition unsatisfied do
3:     Solves $(X^T X + \mu I)\beta^{(k+1)} = (X^T y + \mu\gamma^{(k)} + \alpha^{(k)})$
4:     $\gamma^{(k+1)} = \mathbf{S}_{\lambda/\mu}(\beta^{(k+1)} + \alpha^{(k)}/\mu)$
5:     $\alpha^{(k+1)} = \alpha^{(k)} + \mu(\beta^{(k+1)} - \gamma^{(k+1)})$
6: end while
```

3 Numerical study

3.1 Method

In this section, we conduct a numerical study comparing the sensitivity of the five algorithms of Section 2 (including CD with the active shooting strategy) to the correlation between the covariates and to the regularization parameter λ, and the ratio between the number of variables (p) and the sample size (n). We measure the sensitivity of an algorithm by the number of iterations required for numerical convergence, the corresponding computation time, and the converged objective function value. Note that the warm start strategy is not considered because our primary goal is to compare the algorithms for a fixed value of λ; the warm start strategy can be applied to any algorithm by simply setting the initial value of β for the new λ as the final (optimized) value for the previous λ.

3.1.1 Design of numerical study

To study the effect of correlations between the covariates, we control the condition number of population covariance matrix of the covariates, denoted by Σ. The condition number of a positive definite matrix is defined as the ratio of the largest eigenvalue to the smallest eigenvalue of the matrix. The larger the condition number is, the more correlated the covariates are. The data matrix, X, sampled from a p-variate distribution with covariance matrix Σ, is likely to be highly correlated if Σ is so. For the numerical study, we consider the following four types of population covariance matrices:

(Σ_1) *Smallest condition number*: an identity matrix (I_p) of size $p \times p$ whose condition number is 1.

(Σ_2) *Small condition number*: a symmetric banded matrix of bandwidth 1,

$$\Sigma_2 = \begin{pmatrix} 1 & 0.45 & 0 & \cdots & 0 \\ 0.45 & 1 & 0.45 & \cdots & 0 \\ 0 & 0.45 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0.45 & 1 \end{pmatrix}.$$

The condition number of Σ_2 is approximately 19 for all the dimensions considered in this study.

(Σ_3) *Moderate condition number*: a modification of Σ_2 so that the largest eigenvalue is 100. Specifically,

$$\Sigma_3 = \sum_{i=1}^{p-1} \lambda_{(i)} u_i u_i^T + \tilde{\lambda} u_p u_p^T, \tag{10}$$

where $\lambda_{(i)}$ is the ith smallest eigenvalue of Σ_2, u_i is an eigenvector of Σ_2 corresponding to $\lambda_{(i)}$, and $\tilde{\lambda} = 100$. The condition number of Σ_3 is approximately 1000.

(Σ_4) *Large condition number*: similar to Σ_3, but $\tilde{\lambda}$ is set to 10000 so that the condition number is approximately 100000.

For the dimensions, we consider $p = 100$, 500, and 2500, and set the number of samples as 500. Thus, the ratio p/n takes values of 0.2, 1, and 5, respectively. Note this ratio affects the condition number of the sample covariance matrix $X^T X$.

To take into account the influence of the regularization parameter λ, we consider $\lambda = 0.1$, 1, 10, and 100 for all the cases.

3.1.2 Data generation

For each combination of the two factors p/n and Σ, we generate $n = 500$ observations of a p-dimensional covariate vector $\mathbf{x}$ from the multivariate normal distribution with mean 0 and covariance matrix $\Sigma_i, i = 1, 2, 3, 4$. To simulate sparse regression models, pre-selected 10 % of the coordinates of the p-dimensional coefficient vector β are set to be nonzero. Nonzero coefficients in β is sampled from the standard normal distribution. The response y is generated from the linear model

$$y = \mathbf{x}^T \beta + \epsilon,$$

where ϵ is from the standard normal distribution.

3.1.3 Algorithm parameters

For a fair comparison of the algorithms, we use $\beta^{(0)} = \left((\sum_{i=1}^n X_{ij} y_i)/(\sum_{i=1}^n X_{ij}^2)\right)_{1 \le j \le p}$ as the common initial value and impose a common convergence criterion. The four algorithms stop when the relative error

$$\frac{|f(\beta^{(k)}) - f(\beta^{(k-1)})|}{|f(\beta^{(k-1)})|}$$

is below the pre-specified tolerance 10^{-7}. For each combinations of the factors, each algorithm is repeated 10 times with different random seeds and then the means and the standard errors of the objective function values, the number of iterations and the computation times are taken.

For algorithm-specified parameters, we set the perturbation constant $\epsilon = 10^{-10}$ for MM-LQA, backtracking factor $\eta = 2$ for FISTA, and augmented Lagrange multiplier $\mu = 10$ for ADMM. For PCG used with MM-LQA and ADMM, we use a diagonal matrix whose diagonal entries are those of the matrix to solve, i.e, $\text{diag}(X^T X + D^{(k)})$ for MM-LQA, and $\text{diag}(X^T X + \mu I)$ for ADMM, as the preconditioner.

All algorithms we consider are implemented in MATLAB and the computation times are measured on a desktop PC (Intel i7-4770 (3.40 GHz) and 16.0 GB RAM). The code that implements the algorithms in this paper is available at https://sites.google.com/site/dhyeonyu/software.

3.2 Results of numerical study

In this section, we summarize the results of comparisons of the sensitivity of the MM-LQA, CD, ADMM and FISTA algorithms to the correlations between covariates, the ratio p/n, and the regularization parameter λ. We report the averages and the standard errors of the numbers of iterations, the computation times, the objective function values (F-value), and the number of nonzero solutions (sparsity) for the combinations of the covariance matrices and the regularization parameters with $p = 100$, 500, and 2500 in Tables 1–3, respectively. Note that MM-LQA and ADMM utilize the PCG method to solve the inner linear system; the average and standard errors of the inner iterations are reported for these algorithms only.

It is interesting to note that CD with the active shooting strategy (referred to as "CD-act") accelerated the CD for all the cases although CD-act needed much larger iterations than CD. Other than this, CD-act is basically equivalent to the CD except for the order of updates and we omit the comparison of the CD-act in this section.

To visualize the convergence properties, we plot the relative error of each algorithm with respect to the iteration for $\lambda = 10$ and $p = 100$, 500, and 2500 in Figs. 1, 2, and 3. For a fair comparison, the relative error curves for MM-LQA and ADMM are represented as step functions to take into account the inner iterations for each outer iteration. Note that the plots are taken from one of the 10 simulations, as the other 9 exhibit similar patterns.

3.2.1 Sensitivity to the condition number of population covariance matrices

It is worth noting that overall, MM-LQA and ADMM show relatively stable convergence compared to CD and FISTA. The convergence speed of CD and FISTA considerably slows down as the condition number of the population covariance matrix increases, which means that these algorithms are sensitive to the correlation between the covariates. Notably, CD fails to converge for the case of $p = 2500$,

Table 1 Convergence result, $p = 100$

Σ	λ	Method	Iteration	Time	F-value	Inner iteration	Sparsity	Σ	λ	Method	Iteration	Time	F-value	Inner iteration	Sparsity
Σ_1	0.1	MM-LQA	3.1 (0.1)	0.003 (0.001)	193.8 (5.1)	8.7 (0.2)	99.7 (0.0)	Σ_2	0.1	MM-LQA	4.2 (0.1)	0.003 (0.000)	194.0 (5.1)	18.8 (0.5)	99.5 (0.0)
		CD	10.2 (0.2)	0.011 (0.001)	193.8 (5.1)	·	99.5 (0.3)			CD	40.3 (0.9)	0.03 (0.001)	194.0 (5.1)	·	99.3 (0.2)
		CD-Act	10.4 (0.3)	0.008 (0.001)	193.8 (5.1)	·	99.5 (0.3)			CD-Act	42.8 (2.5)	0.028 (0.002)	194.0 (5.1)	·	99.2 (0.2)
		ADMM	4.7 (0.2)	0.011 (0.004)	193.8 (5.1)	8.5 (0.2)	99.8 (0.1)			ADMM	6.6 (0.5)	0.004 (0.000)	194.0 (5.1)	17.9 (0.7)	99.9 (0.1)
		FISTA	43.9 (4.4)	0.009 (0.001)	193.8 (5.1)	·	99.7 (0.2)			FISTA	213.4 (5.2)	0.02 (0.000)	194.0 (5.1)	·	99.2 (0.2)
	1	MM-LQA	8.5 (0.3)	0.003 (0.000)	203.9 (5.0)	6.5 (0.2)	96.9 (0.5)		1	MM-LQA	12.6 (0.6)	0.007 (0.000)	205.2 (5.0)	12.8 (0.5)	95.0 (0.9)
		CD	9.9 (0.2)	0.008 (0.000)	203.9 (5.0)	·	96.5 (0.4)			CD	35.9 (0.5)	0.03 (0.001)	205.2 (5.0)	·	94.3 (1.1)
		CD-Act	15.4 (1.1)	0.011 (0.001)	203.9 (5.0)	·	96.5 (0.4)			CD-Act	52.7 (4.1)	0.037 (0.003)	205.2 (5.0)	·	93.9 (1.0)
		ADMM	48.1 (2.9)	0.043 (0.028)	203.9 (5.0)	4.4 (0.2)	99.4 (0.2)			ADMM	49.2 (4.3)	0.02 (0.001)	205.2 (5.0)	9.7 (0.6)	97.9 (0.6)
		FISTA	42.6 (4.3)	0.007 (0.002)	203.9 (5.0)	·	96.5 (0.4)			FISTA	189.7 (4.3)	0.016 (0.001)	205.2 (5.0)	·	94.3 (1.1)
	10	MM-LQA	39.4 (1.1)	0.011 (0.001)	294.1 (7.3)	3.3 (0.1)	67.4 (1.6)		10	MM-LQA	45.5 (1.3)	0.01 (0.001)	296.3 (7.4)	5.3 (0.1)	60.6 (1.0)
		CD	7.9 (0.1)	0.006 (0.000)	294.1 (7.3)	·	64.9 (1.7)			CD	19.4 (0.8)	0.014 (0.001)	296.3 (7.4)	·	58.8 (1.0)
		CD-Act	14.8 (0.4)	0.009 (0.001)	294.1 (7.3)	·	64.9 (1.7)			CD-Act	31.1 (1.2)	0.018 (0.001)	296.3 (7.4)	·	58.7 (0.9)
		ADMM	373.0 (8.2)	0.180 (0.100)	294.1 (7.3)	3.0 (0.1)	64.9 (1.7)			ADMM	405.8 (16.1)	0.13 (0.012)	296.3 (7.4)	5.3 (0.2)	59.2 (1.1)
		FISTA	30.4 (3.3)	0.003 (0.000)	294.1 (7.3)	·	64.9 (1.7)			FISTA	103.2 (5.0)	0.01 (0.000)	296.3 (7.4)	·	59.0 (1.0)
	100	MM-LQA	36.3 (5,8)	0.012 (0.003)	913.0 (69.0)	2.5 (0.1)	8.3 (0.4)		100	MM-LQA	52.9 (5.8)	0.02 (0.002)	904.0 (69.3)	2.9 (0.1)	8.6 (0.4)
		CD	4.7 (0.15)	0.006 (0.002)	913.0 (69.0)	·	8.1 (0.5)			CD	6.5 (0.7)	0.005 (0.001)	904.0 (69.3)	·	8.2 (0.5)
		CD-Act	5.3 (0.5)	0.001 (0.000)	913.0 (69.0)	·	8.1 (0.5)			CD-Act	8.5 (1.0)	0.002 (0.000)	904.0 (69.3)	·	8.2 (0.5)
		ADMM	825.0 (11.8)	0.25 (0.006)	913.0 (69.0)	2.7 (0.1)	8.1 (0.5)			ADMM	1066.9 (21.0)	0.36 (0.01)	904.0 (69.3)	4.0 (0.1)	8.2 (0.5)
		FISTA	10.2 (0.2)	0.002 (0.000)	913.0 (69.0)	·	8.1 (0.5)			FISTA	32.2 (4.0)	0.004 (0.000)	904.0 (69.3)	·	8.3 (0.4)
Σ_3	0.1	MM-LQA	4.7 (0.3)	0.004 (0,000)	194.2 (5.1)	21.2 (1.4)	99.1 (0.0)	Σ_4	0.1	MM-LQA	10.8 (0.5)	0.009 (0.000)	198.0 (5.2)	24.9 (1.4)	96.6 (0.0)
		CD	687.8 (43.0)	0.49 (0.031)	194.2 (5.1)	·	98.5 (0.3)			CD	60769.9 (2350.2)	58.2 (3.1)	198.1 (5.2)	·	95.0 (0.6)
		CD-Act	690.1 (43.5)	0.436 (0.025)	194.2 (5.1)	·	98.5 (0.3)			CD-Act	59531.9 (2326.9)	36.76 (1.59)	198.1 (5.2)	·	94.5 (0.6)
		ADMM	9.6 (0.5)	0.006 (0.000)	194.2 (5.1)	17.5 (0.9)	100.0 (0.0)			ADMM	131.2 (8.0)	0.044 (0.004)	198.0 (5.2)	7.2 (0.2)	95.8 (0.5)
		FISTA	4431.0 (155.3)	0.40 (0.01)	194.2 (5.1)	·	98.7 (0.3)			FISTA	90007.4 (2949.0)	43.79 (2.70)	199.0 (5.2)	·	98.0 (0.4)
	1	MM-LQA	15.2 (0.4)	0.008 (0.000)	206.8 (5.1)	13.9 (0.8)	94.2 (1.0)		1	MM-LQA	47.9 (2.4)	0.027 (0.002)	225.4 (5.5)	15.8 (0.7)	65.7 (1.2)
		CD	660.2 (40.1)	0.46 (0.03)	206.8 (5.1)	·	93.0 (1.1)			CD	38267.3 (1636.1)	31.31 (1.66)	225.4 (5.5)	·	62.4 (1.2)
		CD-Act	674.8 (37.0)	0.420 (0.023)	206.8 (5.1)	·	93.0 (1.1)			CD-Act	34193.4 (1361.0)	18.13 (0.92)	225.4 (5.5)	·	62.6 (1.2)
		ADMM	36.6 (2.6)	0.016 (0.001)	206.8 (5.1)	11.2 (0.9)	97.0 (0.7)			ADMM	54.8 (4.2)	0.02 (0,002)	225.4 (5.5)	8.7 (0.3)	64.9 (1.6)
		FISTA	3720.1 (127.3)	0.33 (0.01)	206.9 (5.1)	·	93.4 (1.1)			FISTA	39520.6 (1658.9)	7.95 (0.71)	225.7 (5.5)	·	65.6 (1.1)
	10	MM-LQA	59.0 (1.9)	0.018 (0.001)	299.6 (7.4)	5.6 (0.2)	52.6 (1.0)		10	MM-LQA	94.2 (8.9)	0.030 (0.003)	282.1 (7.5)	6.4 (0.3)	14.0 (1.5)
		CD	397.7 (34.5)	0.29 (0.02)	299.6 (7.4)	·	49.6 (1.0)			CD	4077.4 (473.0)	2.93 (0.35)	282.1 (7.5)	·	12.2 (1.3)
		CD-Act	425.5 (26.0)	0.253 (0.017)	299.6 (7.4)	·	49.4 (0.9)			CD-Act	4077.4 (473.0)	2.54 (0.31)	282.1 (7.5)	·	12.2 (1.3)
		ADMM	263.8 (16.5)	0.105 (0.020)	299.6 (7.4)	6.7 (0.2)	51.1 (1.0)			ADMM	148.5 (14.4)	0.05 (0.004)	282.10 (7.5)	6.7 (0.5)	23.8 (7.8)
		FISTA	1880.4 (118.6)	0.159 (0.008)	299.6 (7.4)	·	49.7 (1.0)			FISTA	13738.0 (2027.8)	1.54 (0.28)	282.3 (7.5)	·	13.7 (1.5)
	100	MM-LQA	70.0 (8.7)	0.02 (0.002)	849.6 (67.3)	3.4 (0.1)	7.6 (0.5)		100	MM-LQA	355.2 (48.1)	0.09 (0.01)	484.5 (28.8)	3.3 (0.1)	8.9 (1.7)
		CD	85.3 (14.7)	0.060 (0.010)	849.6 (67.3)	·	7.0 (0.4)			CD	670.6 (113.0)	0.47 (0.08)	484.5 (28.8)	·	4.8 (1.0)
		CD-Act	87.2 (15.2)	0.050 (0.009)	849.6 (67.3)	·	7.0 (0.4)			CD-Act	693.0 (129.3)	0.43 (0.08)	484.5 (28.8)	·	4.8 (0.9)
		ADMM	1830.3 (146.2)	0.71 (0.05)	849.8 (67.3)	4.5 (0.4)	11.0 (4.1)			ADMM	927.1 (143.1)	0.25 (0.04)	484.7 (28.8)	4.6 (0.4)	35.4 (13.3)
		FISTA	674.4 (69.4)	0.079 (0.011)	849.6 (67.3)	·	7.1 (0.4)			FISTA	9194.0 (1600.8)	0.96 (0.20)	484.7 (28.9)	·	6.7 (1.4)

Table 2 Convergence result, $p = 500$

Σ	λ	Method	Iteration	Time	F-value	Inner iteration	Sparsity	Σ	λ	Method	Iteration	Time	F-value	Inner iteration	Sparsity
Σ_1	0.1	MM-LQA	39.3 (2.8)	0.20 (0.02)	14.9 (0.7)	53.1 (2.9)	485.9 (0.0)	Σ_2	0.1	MM-LQA	49.4 (4.6)	0.28 (0.01)	17.8 (0.8)	65.1 (3.3)	482.3 (0.0)484.3 (1.1)
		CD	2794.7 (135.9)	9.56 (0.45)	14.9 (0.7)	·	486.3 (1.1)			CD	3345.3 (214.3)	11.3 (0.7)	17.8 (0.8)	·	484.3 (1.1)
		CD-Act	3230.4 (275.5)	9.9 (0.80)	14.9 (0.7)	·	485.9 (1.2)			CD-Act	4007.1 (602.2)	12.40 (1.87)	17.8 (0.80)	·	484.2 (1.1)
		ADMM	137.1 (9.4)	0.36 (0.03)	14.9 (0.7)	17.7(0.5)	487.6 (1.1)			ADMM	164.9 (14.1)	0.53 (0.14)	17.8 (0.8)	19.4 (0.5)	483.7 (0.9)
		FISTA	12458.2 (769.8)	2.7 (0.18)	14.9 (0.7)	·	489.2 (1.0)			FISTA	24135.8 (1665.6)	5.86 (0.49)	17.8 (0.8)	·	486.5 (0.9)
	1	MM-LQA	75.2 (2.8)	0.19 (0.01)	84.9 (2.0)	16.5 (0.5)	437.8 (1.4)		1	MM-LQA	76.6 (4.8)	0.22 (0.01)	94.7 (2.2)	21.0 (0.7)	425.2 (1.3)
		CD	315.30 (11.0)	1.1 (0.04)	84.9 (2.0)	·	437.1 (1.5)			CD	377.1 (14.4)	1.29 (0.05)	94.7 (2.2)	·	424.2 (1.5)
		CD-Act	611.1 (50.6)	1.77 (0.16)	84.9 (2.0)	·	436.7 (1.5)			CD-Act	749.0 (89.3)	2.22 (0.29)	94.7 (2.2)	·	424.0 (1.4)
		ADMM	61.6 (4.8)	0.21 (0.05)	84.9 (2.04)	21.1 (1.9)	445.5 (5.8)			ADMM	58.7 (1.8)	0.19 (0.02)	94.67 (2.24)	23.3 (0.6)	427.5 (1.7)
		FISTA	1562.8 (140.6)	0.31 (0.03)	84.9 (2.0)	·	437.8 (1.4)			FISTA	3143.7 (133.7)	0.65 (0.03)	94.7 (2.2)	·	426.3 (1.7)
	10	MM-LQA	77.4 (2.2)	0.14 (0.01)	544.8 (10.3)	5.9 (0.1)	269.0 (2.8)		10	MM-LQA	83.5 (2.6)	0.16 (0.007)	551.9 (10.0)	6.9 (0.1)	250.4 (2.1)
		CD	35.7 (0.8)	0.12 (0.004)	544.8 (10.3)	·	26.17 (2.9)			CD	47.2 (1.2)	0.16 (0.006)	551.9 (10.0)	·	242.3 (2.1)
		CD-Act	78.4 (5.5)	0.19 (0.01)	544.8 (10.3)	·	261.1 (2.9)			CD-Act	103.0 (7.7)	0.243 (0.017)	551.9 (10.0)	·	242.4 (2.1)
		ADMM	433.0 (39.6)	1.21 (0.37)	544.9 (10.3)	10.7 (0.8)	265.0 (3.9)			ADMM	356.4 (20.3)	0.99 (0.22)	551.9 (10.0)	13.9 (0.6)	251.0 (1.4)
		FISTA	195.7 (20.3)	0.040 (0.005)	544.8 (10.3)	·	261.5 (3.0)			FISTA	444.1 (10.5)	0.09 (0.004)	551.9 (10.0)	·	242.5 (2.0)
	100	MM-LQA	67.9 (1.6)	0.14 (0.01)	3662.6 (107.7)	3.2 (0.0)	46.7 (1.5)		100	MM-LQA	79.9 (2.6)	0.13 (0.004)	3638.0 (105.1)	3.8 (0.1)	3.6 (0.6)
		CD	8.1 (0.1)	0.027 (0.001)	3662.6 (107.7)	·	44.4 (1.4)			CD	11.8 (0.4)	0.041 (0.001)	3638.0 (105.1)	·	3.0 (0.4)
		CD-Act	15.3 (0.58)	0.026 (0.002)	3662.6 (107.7)	·	44.4 (1.4)			CD-Act	23.9 (1.1)	0.033 (0.001)	3638.0 (105.1)	·	49.4 (1.7)
		ADMM	1330.4 (9.1)	3.16 (0.05)	3662.7 (107.7)	6.3 (0.1)	44.4 (1.4)			ADMM	1475.2 (60.8)	2.77 (0.08)	3638.3 (105.0)	7.6 (0.5)	3.0 (0.4)
		FISTA	38.3 (0.7)	0.016 (0.001)	3662.6 (107.7)	·	44.4 (1.4)			FISTA	109.3 (3.7)	0.024 (0.001)	3638.0 (105.1)	·	3.0 (0.4)
Σ_3	0.1	MM-LQA	44.0 (4.6)	0.25 (0.01)	18.7 (0.9)	63.4 (3.3)	481.5 (0.0)	Σ_4	0.1	MM-LQA	67.7 (13.7)	0.23 (0.02)	37.8 (1.6)	41.1 (7.9)	464.3 (0.0)
		CD	11172.3 (1506.3)	38.8 (5.2)	18.7 (0.9)	·	482.4 (1.1)			CD	380028.1 (30756.9)	2196.2 (252.5)	37.9 (1.6)	·	456.7 (1.8)
		CD-Act	11803.2 (1761.0)	36.14 (5.16)	18.7 (0.9)	·	482.6 (1.2)			CD-Act	330293.5 (29645.7)	1004.44 (92.97)	37.8 (1.6)	·	457.0 (1.6)
		ADMM	162.9 (13.5)	0.43 (0.05)	18.7 (0.9)	18.3 (0.8)	482.9 (1.5)			ADMM	707.2 (258.7)	1.3 (0.37)	37.8 (1.6)	8.6 (1.0)	462.9 (1.8)
		FISTA	178914.7 (12291.7)	202.7 (25.7)	18.8 (0.9)	·	491.8 (0.9)			FISTA	1032160.8 (46298.3)	6713.5 (559.2)	41.2 (1.9)	·	483.3 (0.9)
	1	MM-LQA	80.3 (4.5)	0.23 (0.01)	97.5 (2.3)	20.5 (0.7)	420.1 (0.9)		1	MM-LQA	101.6 (16.7)	0.23 (0.02)	150.7 (2.8)	14.5 (1.6)	335.4 (2.8)
		CD	1545.0 (165.7)	5.4 (0.58)	97.5 (2.3)	·	419.4 (0.9)			CD	281250.8 (18284.5)	1575.1 (134.4)	150.7 (2.8)	·	328.0 (3.1)
		CD-Act	1601.7 (140.8)	4.90 (0.45)	97.5 (2.3)	·	419.3 (0.8)			CD-Act	235105.8 (14951.5)	623.50 (50.02)	150.7 (2.8)	·	328.1 (2.9)
		ADMM	64.7 (3.6)	0.18 (0.02)	97.5 (2.3)	19.6 (1.2)	420.0 (1.2)			ADMM	83.4 (9.1)	0.19 (0.02)	150.4 (2.8)	10.4 (0.9)	337.0 (7.5)
		FISTA	30412.3 (1402.4)	7.91 (0.53)	97.6 (2.3)	·	427.3 (2.0)			FISTA	231004.6 (14323.0)	399.6 (63.6)	152.0 (2.9)	·	356.5 (3.0)
	10	MM-LQA	83.2 (2.2)	0.16 (0.006)	555.3 (10.2)	7.5 (0.1)	236.7 (2.4)		10	MM-LQA	101.8 (3.2)	0.19 (0.007)	516.8 (10.1)	6.1 (0.3)	101.0 (2.5)
		CD	786.5 (61.8)	2.73 (0.23)	555.3 (10.2)	·	227.9 (2.2)			CD	76467.5 (9638.8)	296.5 (41.1)	516.4 (10.1)	·	96.1 (2.7)
		CD-Act	878.8 (66.2)	2.60 (0.25)	555.3 (10.2)	·	227.9 (2.2)			CD-Act	68498.2 (8464.7)	171.06 (23.33)	516.8 (10.1)	·	96.8 (2.6)
		ADMM	356.2 (23.9)	0.90 (0.19)	555.4 (10.2)	12.2 (0.7)	248.5 (10.9)			ADMM	222.9 (12.4)	0.48 (0.07)	516.9 (10.1)	6.2 (0.4)	103.2 (3.7)
		FISTA	5594.7 (134.4)	1.15 (0.03)	555.4 (10.2)	·	230.9 (2.5)			FISTA	66646.0 (4528.1)	32.7 (3.7)	517.8 (10.1)	·	100.9 (2.5)

Table 2 (continued)

Σ	λ	Method	Iteration	Time	F-value	Inner iteration	Sparsity	Σ	λ	Method	Iteration	Time	F-value	Inner iteration	Sparsity
	100	MM-LQA	80.6 (3.1)	0.18 (0.010)	3544.1 (104.0)	4.3 (0.1)	50.6 (1.6)		100	MM-LQA	173.0 (16.0)	0.29 (0.04)	1573.3 (107.9)	3.2 (0.1)	22.2 (2.7)
		CD	227.3 (26.0)	0.81 (0.09)	3544.0 (104.0)	·	47.1 (1.5)			CD	9137.6 (1751.9)	32.9 (6.7)	1573.3 (107.9)	·	18.3 (2.4)
		CD-Act	246.9 (23.8)	0.69 (0.08)	3544.1 (104.0)	·	47.1 (1.5)			CD-Act	8895.4 (1646.6)	25.77 (4.54)	1573.3 (107.9)	·	18.2 (2.4)
		ADMM	1624.4 (110.9)	4.00 (0.19)	3545.8 (104.2)	7.4 (0.6)	80.7 (27.8)			ADMM	1741.6 (284.8)	3.4 (0.60)	1576.4 (108.0)	4.7 (0.4)	331.9 (61.5)
		FISTA	1428.0 (35.8)	0.49 (0.01)	3544.1 (104.0)	·	47.4 (1.5)			FISTA	20235.7 (2971.5)	5.4 (0.97)	1574.3 (108.0)	·	21.4 (2.8)

$\Sigma = \Sigma_4$, and $\lambda = 0.1, 1, 10$ (Table 3). For a relatively high level of regularization ($\lambda = 100$), however, the CD and the FISTA algorithms show relatively stable performance. While not shown in the Tables, for an even larger value of regularization parameter ($\lambda = 1000$), the CD converges within 124570.6 iterations.

In summary, convergence speed of the MM-LQA and the ADMM algorithms are less sensitive to the correlation between the covariates than those of the CD and the FISTA algorithms. This can be attributed to the fact that the MM-LQA and ADMM algorithms employ PCG to solve a linear system involving the sample covariance matrix $X^T X$. For a high-dimensional setting ($p/n \gg 1$), this matrix becomes ill-conditioned, and the surface of objective function is flat. This makes coordinate-wisely moving algorithms like CD to slow down due to the reduced step size. In FISTA, the step size is inversely proportional to the Lipschitz constant L which is related to the maximum eigenvalue of $X^T X$, hence FISTA also needs more iterations to converge. On the contrary, preconditioning alleviates ill-conditioning of the matrix $X^T X$ in MM-LQA.

3.2.2 Sensitivity to the ratio p/n

Here we examine the effect of the sample covariance matrix $X^T X$ which becomes ill-conditioned as the dimension increases. The ill-conditioning is most severe when $p/n > 1$, because this matrix is singular. Thus, we expect that MM-LQA and ADMM are more stable than CD and FISTA. Simulation results meet the expectation. For instance, for $\Sigma = \Sigma_1$ and $\lambda = 10$, as the dimension varies from 100 to 500 and to 2500 (Tables 1–3), average iterations of the CD algorithm sharply increases from 7.9 to 35.7 and to 1070.2. For FISTA those are 30.4, 195.7, and 8620.6. On the contrary, the average iteration counts for MM-LQA are 39.4, 77.4 and 317.6.

To examine the effect of the dimensions more carefully, we conducted an additional numerical study with $p = 1000$, 1500 and 2000 for $\Sigma = \Sigma_1$ and $\lambda = 10$. We chose this combination of Σ and λ to avoid the effect of the underlying covariance structure and that of too small or too large regularization parameter. We report the averages and standard errors of the the number of iterations and computation time for $p/n = 0.2, 1, 2, 3, 4, 5$ in Table 4. While both the number of iterations and the computation time for MM-LQA, CD, and FISTA increase as the dimension increases, ADMM does not show this tendency: the number of iterations are 373.0, 433.0, 847.0, 583.5, 406.6, and 379.0. MM-LQA shows an almost linear increase in convergence speed: the average numbers of iterations are 39.4, 77.4, 126.1, 195.9, 270.7, and 317.6. Compared to MM-LQA, CD and FISTA have a fast drop in the performance. For instance,

Table 3 Convergence result, $p = 2500$

Σ	λ	Method	Iteration	Time	F-value	Inner iteration	Sparsity	Σ	λ	Method	Iteration	Time	F-value	Inner iteration	Sparsity
Σ_1	0.1	MM-LQA	85.7 (1.3)	5.09 (0.1)	17.9 (0.2)	80.4 (1.14)	588.3 (3.6)	Σ_2	0.1	MM-LQA	88.1 (2.0)	5.31 (0.10)	18.0 (0.3)	84.0 (1.5)	591.3 (0.0)
		CD	46148.9 (1235.2)	794.29 (24.19)	18.0 (0.2)	·	558.0 (1.3)			CD	45150.4 (1120.1)	785.1 (12.1)	18.0 (0.3)	·	564.3 (1.6)
		CD-Act	73275.1 (1611.6)	500.88 (19.13)	18.0 (0.2)	·	554.0 (0.9)			CD-Act	74723.2 (1572.8)	498.0 (9.6)	18.0 (0.3)	·	552.0 (2.1)
		ADMM	861.3 (36.6)	14.41 (1.73)	18.0 (0.2)	7.9 (0.1)	569.2 (2.9)			ADMM	807.6 (97.9)	13.99 (1.78)	18.0 (0.3)	8.8 (0.3)	587.0 (6.1)
		FISTA	187989.5 (2740.5)	553.7 (9.81)	18.1 (0.2)	·	654.2 (3.4)			FISTA	189482.3 (4450.1)	571.1 (20.0)	18.2 (0.3)	·	656.7 (2.6)
	1	MM-LQA	258.7 (12.0)	6.46 (0.22)	176.7 (2.3)	24.7 (0.7)	529.2 (2.0)		1	MM-LQA	248.1 (10.2)	6.5 (0.23)	177.2 (3.0)	27.0 (0.7)	530.7 (2.0)
		CD	7137.5 (236.7)	122.6 (3.44)	176.7 (2.3)	·	527.2 (1.2)			CD	7380.1 (167.3)	126.1 (2.5)	177.2 (3.0)	·	527.3 (1.2)
		CD-Act	7678.8 (329.7)	110.1 (3.87)	176.7 (2.3)	·	526.0 (1.1)			CD-Act	7858.6 (413.0)	114.1 (3.7)	177.2 (3.0)	·	527.8 (1.0)
		ADMM	338.0 (38.2)	5.79 (0.62)	176.9 (2.3)	9.3 (0.3)	972.9 (149.6)			ADMM	378.7 (71.0)	6.8 (1.0)	177.3 (3.0)	10.5 (0.5)	990.7 (92.9)
		FISTA	43944.0 (673.4)	83.6 (1.97)	177.1 (2.3)	·	574.1 (1.8)			FISTA	42694.8 (820.3)	82.3 (2.58)	177.6 (3.1)	·	578.6 (1.8)
	10	MM-LQA	317.6 (16.4)	5.58 (0.23)	1742.1 (22.9)	12.3 (0.2)	505.7 (1.2)		10	MM-LQA	317.2 (22.1)	5.65 (0.31)	1745.0 (30.4)	13.0 (0.29)	504.7 (2.1)
		CD	1070.2 (41.3)	18.42 (0.66)	1742.1 (22.9)	·	493.3 (0.9)			CD	1047.4 (22.0)	18.04 (0.57)	1745.0 (30.4)	·	491.1 (1.9)
		CD-Act	2008.6 (176.6)	18.65 (8.4)	1742.1 (22.9)	·	493.9 (1.0)			CD-Act	2062.4 (137.9)	19.18 (0.76)	1745.0 (30.4)	·	489.7 (1.4)
		ADMM	379.0 (32.9)	6.83 (0.68)	1745.0 (22.9)	10.4 (0.2)	1430.9 (152.8)			ADMM	353.0 (50.1)	6.68 (0.73)	1749.7 (29.2)	12.3 (0.4)	1513.2 (219.3)
		FISTA	8620.6 (207.3)	14.69 (0.37)	1742.6 (22.9)	·	514.8 (1.5)			FISTA	8852.1 (213.6)	15.4 (0.57)	1745.5 (30.4)	·	513.8 (2.7)
	100	MM-LQA	194.2 (4.8)	2.66 (0.07)	15423.1 (223.6)	6.0 (0.1)	396.8 (2.8)		100	MM-LQA	195.9 (3.8)	2.71 (0.05)	15362.0 (303.7)	6.2 (0.1)	391.9 (2.6)
		CD	89.7 (2.8)	1.54 (0.03)	15423.0 (223.6)	·	370.8 (3.2)			CD	90.8 (2.2)	1.59 (0.05)	15362.0 (303.7)	·	367.3 (2.1)
		CD-Act	209.3 (4.7)	1.49 (0.03)	15423.0 (223.6)	·	370.6 (3.2)			CD-Act	209.6 (7.1)	1.54 (0.05)	15362.0 (303.7)	·	366.9 (2.2)
		ADMM	1188.8 (86.8)	20.35 (1.59)	15435.7 (222.8)	9.1 (0.3)	944.1 (112.4)			ADMM	1077.7 (88.1)	20.4 (2.57)	15393.7 (298.9)	11.0 (0.5)	1007.1 (198.6)
		FISTA	1025.1 (26.8)	1.74 (0.04)	15423.3 (223.6)	·	373.5 (3.0)			FISTA	1021.5 (31.8)	1.7 (0.06)	15362.3 (303.7)	·	370.9 (2.3)
Σ_3	0.1	MM-LQA	95.9 (10.8)	5.37 (0.10)	18.0 (0.3)	81.7 (6.1)	594.7 (0.0)	Σ_4	0.1	MM-LQA	700.9 (217.6)	8.96 (2.55)	17.1 (0.3)	7.8 (2.1)	745.1 (0.0)
		CD	304744.0 (89745.6)	6283.27 (2166.26)	18.0 (0.3)	·	556.3 (1.8)			CD	X	X	X	·	X
		CD-Act	289210.1 (84957.7)	1919.8 (546.7)	18.0(0.3)	·	551.6 (1.7)			CD-Act	X	X	X	·	X
		ADMM	824.7 (44.1)	14.74 (0.80)	18.0 (0.3)	9.5 (0.3)	579.6 (2.8)			ADMM	476.9 (32.1)	7.36 (0.56)	17.1 (0.3)	6.3 (0.4)	755.9 (60.4)
		FISTA	583379.7 (31997.3)	3408.0 (316.8)	18.9 (0.3)	·	829.7 (16.3)			FISTA	1152791.5 (35330.9)	11798.9 (880.9)	22.1 (0.6)	·	1663.9 (64.7)
	1	MM-LQA	247.2 (10.1)	6.7 (0.28)	177.3 (3.0)	27.8 (0.4)	530.4 (2.9)		1	MM-LQA	1447.3 (384.7)	17.6 (4.3)	165.3 (3.2)	5.5 (1.6)	542.3 (4.5)
		CD	33079.0 (9006.5)	598.8 (179.8)	177.4 (3.0)	·	526.6 (1.8)			CD	X	X	X	·	X
		CD-Act	41979.5 (9011.0)	334.2 (58.8)	177.4 (3.00)	·	524.3 (2.2)			CD-Act	X	X	X	·	X
		ADMM	296.9 (27.4)	5.9 (1.0)	177.5 (3.0)	12.1 (0.2)	1124.5 (124.2)			ADMM	379.9 (43.1)	5.53 (0.53)	165.3 (3.2)	7.0 (0.8)	694.5 (99.7)
		FISTA	171123.1 (11769.2)	498.6 (49.2)	179.1 (3.0)	·	647.5 (6.8)			FISTA	739028.0 (38652.4)	5034.6 (445.9)	177.3 (3.4)	·	885.9 (17.6)
	10	MM-LQA	318.5 (11.8)	5.89 (0.21)	1745.4 (30.0)	13.6 (0.3)	505.5 (1.1)		10	MM-LQA	293.1 (42.1)	4.1 (0.48)	1555.7 (33.0)	6.7 (0.6)	461.0 (2.8)
		CD	3594.1 (834.6)	64.4 (16.4)	1746.1 (29.9)	·	500.6 (6.1)			CD	X	X	X	·	X
		CD-Act	4557.0 (817.9)	57.2 (12.2)	1745.5 (30.0)	·	492.1 (1.5)			CD-Act	X	X	X	·	X
		ADMM	372.4 (28.5)	7.67 (0.90)	1748.1 (30.1)	13.3 (0.4)	1418.8 (143.2)			ADMM	342.6 (30.6)	5.5 (0.6)	1557.0 (32.9)	8.4 (0.7)	823.5 (151.7)
		FISTA	37952.6 (3054.8)	72.07 (6.91)	1748.3 (30.0)	·	545.6 (3.5)			FISTA	230375.9 (17728.2)	769.6 (87.0)	1575.2 (33.3))	·	561.5 (5.1)

Table 3 (continued)

Σ	λ	Method	Iteration	Time	F-value	Inner iteration	Sparsity	Σ	λ	Method	Iteration	Time	F-value	Inner iteration	Sparsity
	100	MM-LQA	190.1 (4.1)	2.75 (0.06)	15297.9 (300.0)	6.9 (0.1)	391.4 (2.4)		100	MM-LQA	152.9 (3.4)	1.93 (0.05)	10265.7 (290.4)	3.8 (0.1)	208.7 (3.7)
		CD	352.8 (91.1)	6.3(1.79)	15297.8 (299.9)	.	363.6 (2.6)			CD	124570.6 (28017.3)	2302.9 (585.6)	10266.1 (290.4)	.	191.0 (3.8)
		CD-Act	486.8 (91.6)	5.46 (1.35)	15297.8 (299.9)	.	362.8 (2.6)			CD-Act	108633.2 (23877.3)	761.59 (140.51)	10265.9 (290.4)	.	191.8 (3.8)
		ADMM	1120.2 (139.9)	21.5 (2.5)	15337.1 (300.8)	12.1 (0.8)	1123.2 (220.6)			ADMM	1030.3 (63.2)	15.5 (1.39)	10284.9 (290.5)	7.0 (0.4)	1005.3 (222.9)
		FISTA	5939.5 (624.0)	10.2 (1.1)	15300.1 (299.9)	.	374.8 (2.9)			FISTA	39686.1 (4051.7)	76.3 (8.5)	10279.8 (290.6)	.	204.1 (4.4)

when dimension increases from 500 to 1000, the average number of iterations for CD are 35.7 and 119.8, and that of FISTA are 195.7 and 959.3, while that of MM-LQA are 77.4 and 126.1.

3.2.3 Sensitivity to the regularization parameter

It is interesting to observe that ADMM and FISTA exhibit opposite phenomena with respect to the regularization parameter λ. For large λ, ADMM shows slow convergence. For example, in p=2500, $\Sigma = \Sigma_4$ (Table 3), the iteration count increases steeply from 342.6, to 1030.3, and to 2213.9 as λ grows from 10 to 100, and to 1000. In contrast, FISTA needs less iterations to converge for large λ. For instance, as λ increases from 0.1 to 1000, iterations are diminishing as 1152791.5, 739028.0, 230375.9, 39686.1, and 13904.6. To understand these phenomena, look at the updating mechanism for ADMM in (8). The part related to regularization parameter is the soft-thresholding operation for updating the added primal variable γ, meant to be equal to the original primal variable β. However, the amount of shrinkage to zero increases as λ increases, pulling β and γ apart. Since the dual variable α converges only when difference of β and γ is extremely small, large λ hinders convergence of α. On the contrary, updating mechanism in (5) only involves the β. Since a threshold is proportional to λ in soft-thresholding operator, large λ makes many of the components of β to zero. That is, less iteration is needed for large λ.

3.2.4 Accuracy

In terms of the minimized objective function value, MM-LQA obtains the smallest objective function value for most cases considered and the other algorithms have slightly larger values than MM-LQA. In high dimensions (p=500, 2500), the objective function values FISTA significantly differ from those of the other algorithms even though they satisfy the convergence criterion. For instance, the objective function value of FISTA is 7.33 % larger than that of MM-LQA for $p = 2500$, $\Sigma = \Sigma_4$, and λ=10 (Table 3), and 8.8 % larger for $p = 500$, $\Sigma = \Sigma_4$, and $\lambda = 0.1$ (Table 2). The inaccuracy of FISTA becomes worse as the condition number of the population covariance matrix gets larger. In Table 1, the maximum relative differences of FISTA are 1.06 % with Σ_1, 1.01 % with Σ_2, 4.97 % with Σ_3, and 7.33 % with Σ_4.

3.2.5 Computation time

In lieu of Figs. 1–3, in which the number of inner-iterations for MM-LQA and ADMM is taken into account, CD performs the best for Σ_1 and Σ_2 and MM-LQA does the best

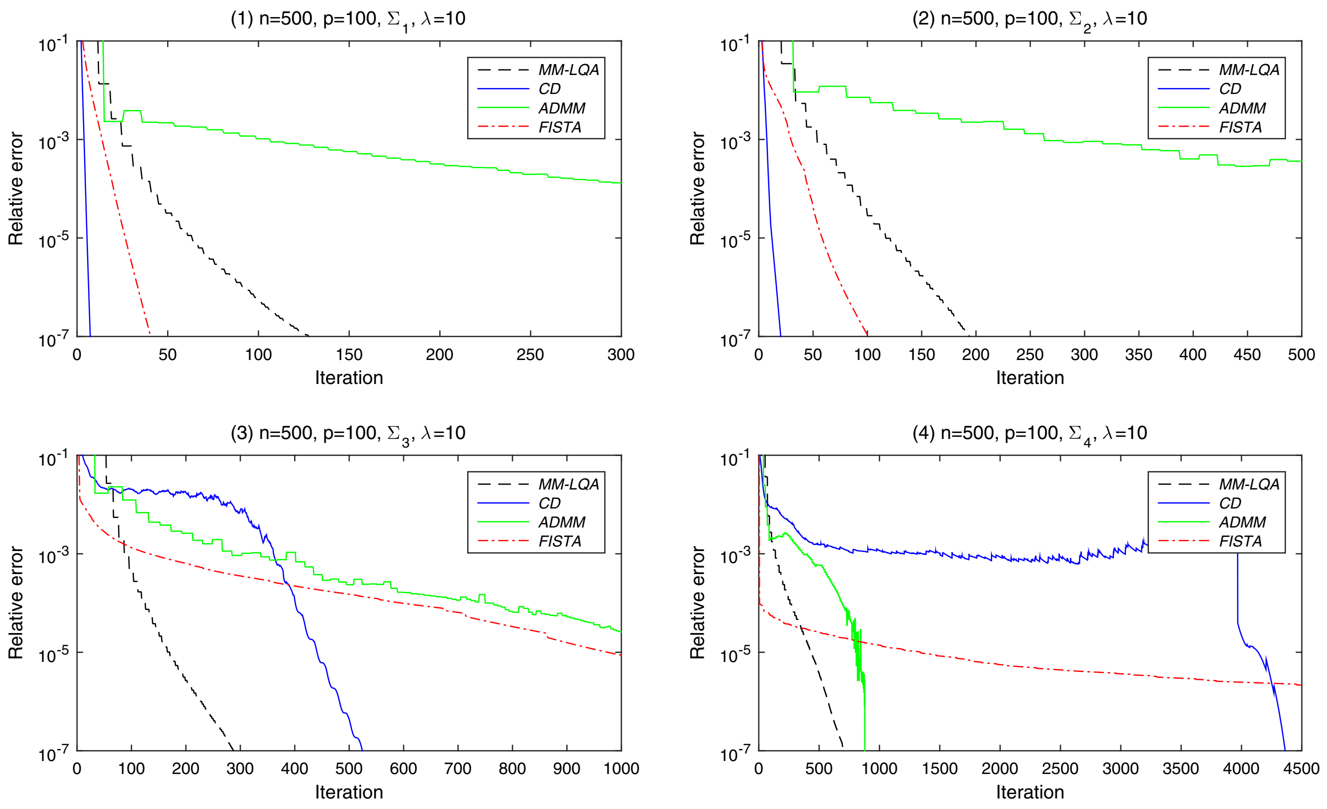


Fig. 1 Comparison of Convergence rate among four methods in $p = 100$ cases

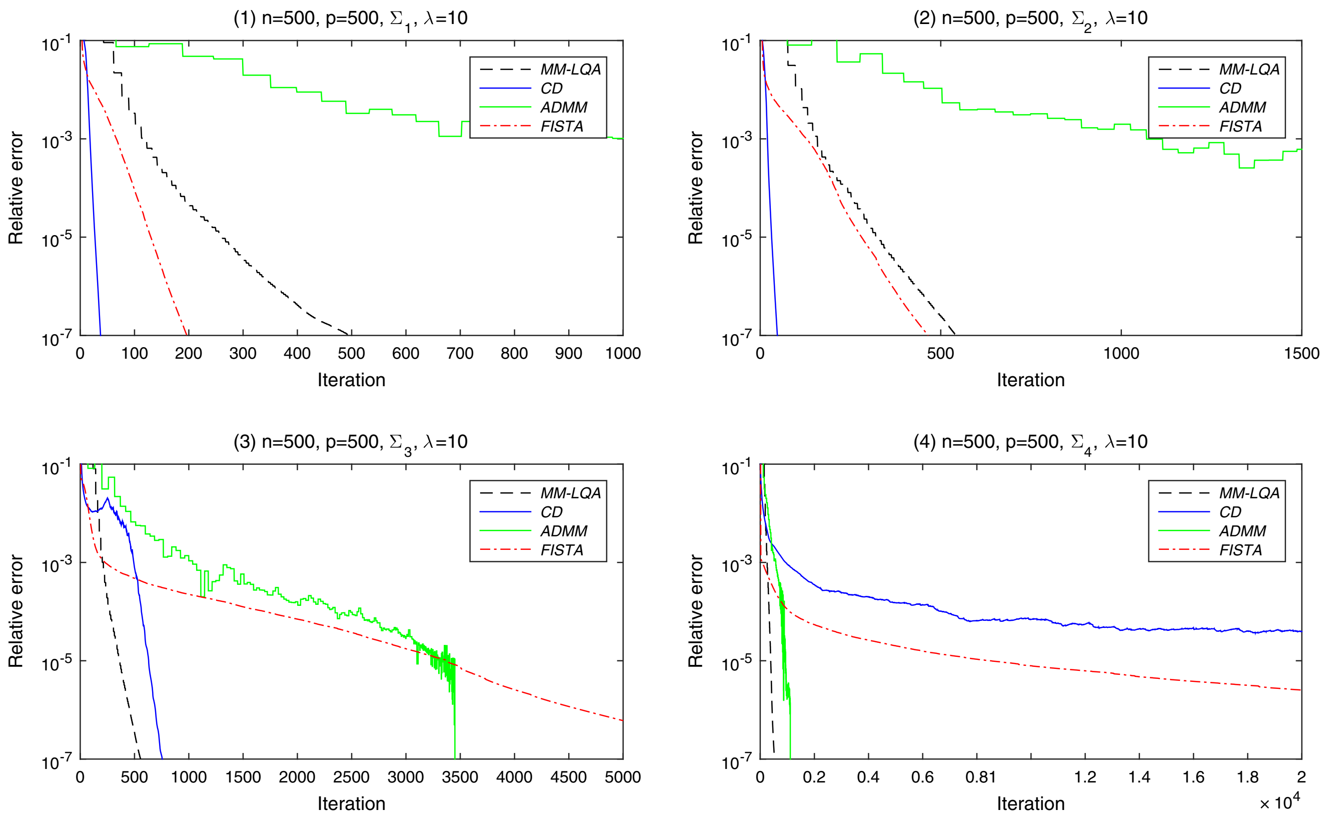


Fig. 2 Comparison of Convergence rate among four methods in $p = 500$ cases

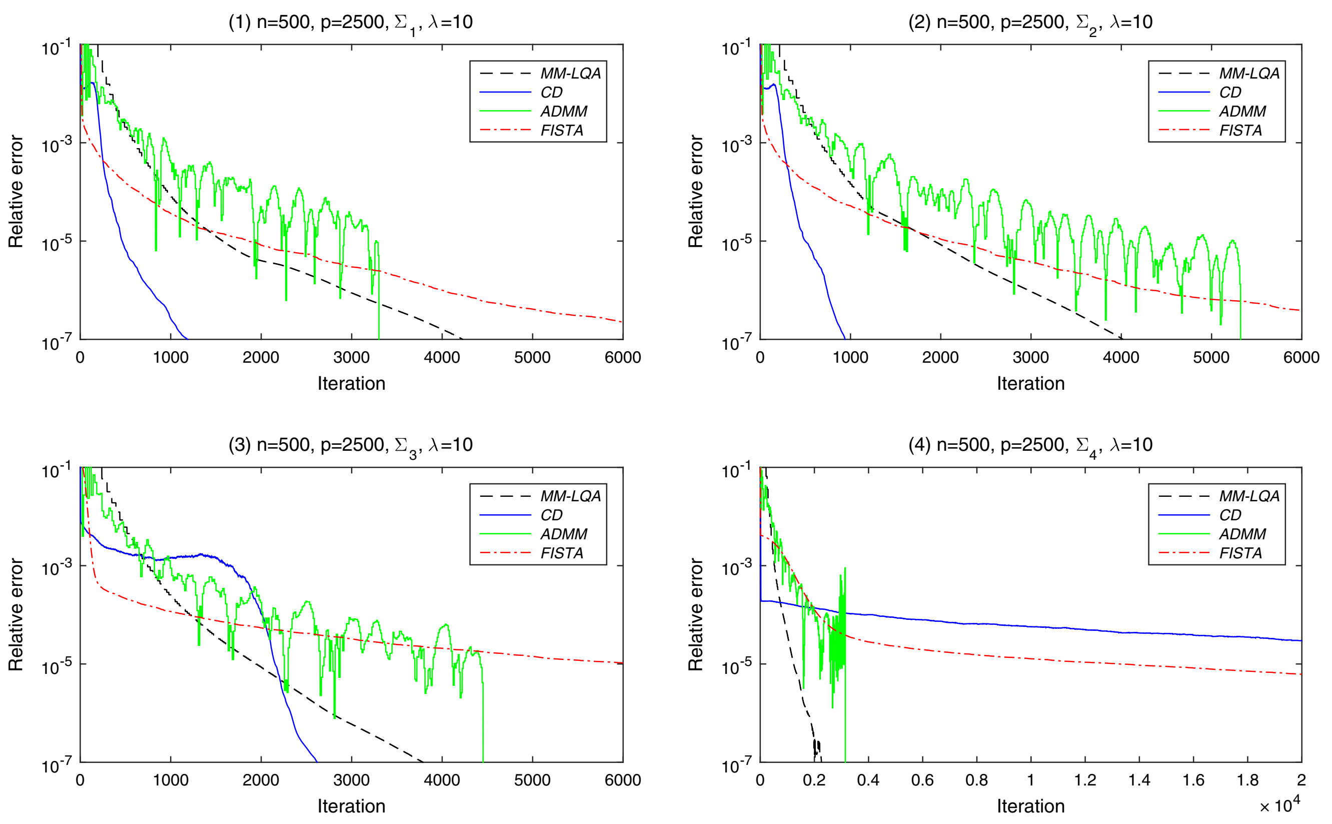


Fig. 3 Comparison of Convergence rate among four methods in $p = 2500$ cases

for Σ_3 and Σ_4. In terms of computation times, however, MM-LQA is faster than CD for all the covariance matrices and dimensions except for the cases of Σ_1 and Σ_2 and $p = 100, 500, 2500$ at $\lambda = 10, 100$ (Tables 1–3). Note that CD is at most 1.12 second faster than MM-LQA for the case of Σ_2 and $\lambda = 100$ (Table 3).

3.2.6 Oscillation of ADMM

What is noticeable in Figs. 1–3 is the oscillation of the relative error for ADMM. This oscillatory pattern is due to that ADMM allows $\beta^{(k)} \neq \gamma^{(k)}$, leaving the objective function infeasible before a sufficient number of iterations.

Table 4 Number of iterations and operation counts with respect to the dimension (p), n=500, Σ_1, λ=10

P	Value	MM-LQA	CD	CD-Act	ADMM	FISTA
100	Time	0.010 (0.000)	0.006 (0.000)	0.010 (0.002)	0.090 (0.003)	0.003 (0.000)
	Iteration	39.4 (1.1)	7.9 (0.1)	14.8 (0.4)	373.0 (8.2)	30.4 (3.3)
500	Time	0.14 (0.01)	0.13 (0.01)	0.19 (0.01)	0.90 (0.07)	0.04 (0.01)
	Iteration	77.4 (2.2)	35.7 (0.8)	78.4 (5.5)	433.0 (39.6)	195.7 (20.3)
1000	Time	0.56 (0.02)	0.86 (0.03)	1.09 (0.10)	3.65 (0.41)	0.34 (0.01)
	Iteration	126.1 (3.6)	119.8 (3.6)	244.7 (24.1)	847.0 (103.3)	959.3 (27.5)
1500	Time	1.48 (0.06)	3.54 (0.19)	4.06 (0.25)	4.51 (0.80)	1.39 (0.11)
	Iteration	195.9 (8.4)	328.6 (12.4)	698.1 (50.4)	583.5 (101.0)	2427.5 (95.1)
2000	Time	3.29 (0.14)	9.31 (0.48)	10.40 (0.64)	4.92 (0.27)	7.45 (0.56)
	Iteration	270.7 (12.3)	664.5 (26.0)	1423.0 (149.2)	406.6 (27.4)	6979.5 (152.4)
2500	Time	5.79 (0.25)	18.2 (0.82)	18.60 (0.80)	6.5 (0.52)	15.4 (0.60)
	Iteration	317.6 (16.4)	1070.2 (41.3)	2008.6 (176.6)	379.0 (32.8)	8654.7 (233.7)

3.2.7 Non-convergence

We found that CD does not converge when λ is small and condition number (κ) of covariance matrix is large (Table 3). For p=2500, $\Sigma = \Sigma_4$, CD does not satisfy the convergence criterion even after 2 million iterations. To see the pattern, we investigated similar cases. For p=2500, $\Sigma = \Sigma_2, \Sigma_3$, λ=0.1 (Table 3), CD algorithm spends 189482.4 and 304744.0 iterations to converge. Furthermore, for p=2500, $\Sigma = \Sigma_4$, λ=100 (Table 3), it needs 124570.6 iterations to converge. A similar phenomenon is found in low dimensions (p=100, 500). For p=500, $\Sigma = \Sigma_4$, λ=0.1, 1 (Table 2), 1032160.8 and 231004.6 iterations are needed to converge, and 90007.4, 39520.6 iterations are needed for p=100, $\Sigma = \Sigma_4$, λ=0.1, 1 (Table 1).

4 Application to cancer biomarker discovery

In this section, we apply the Lasso regression model to discover possible biomarkers for the lung cancer. Lung cancer is the leading cause of death from cancer and notorious for low survival rate; its 5-year survival rate is approximately 15 % [14]. Moreover, in the early stage, progression of the lung cancer varies greatly between patients. To make a proper treatment, it is important to classify the progression and metastasis for individual patient. Recently, cancer is considered as a disease of genomic alterations. Many prognostic or predictive biomarkers have been identified, which allow calculation of risk and precise classification of individual lung cancer patients [22]. For more accurate prognosis or treatments, it is useful to discover genes regulated by known biomarkers of lung cancer [5, 7]. The Lasso can be a useful tool for this purpose. To see the feasibility, we analyzed the gene expression dataset from the Lung Cancer Consortium [20].

4.1 Method

The dataset measures the gene levels in 442 lung cancer adenocarcinoma patients using the Affymetrix U133A platform. For preprocessing, we applied the robust multiarray average (RMA) algorithm and quantile-quantile normalization [13]. All gene expression values were log2 transformed. The Entrez IDs were used to map genes across microarray platforms. The annotated genes from this probe set, and their expression values were calculated using the average values of related probe sets.

To focus on the survival-related genes, we used univariate Cox regression to identify the gene sets that are significantly correlated with a patient's overall survival time. In order to deal with the multiple comparison issue, we controlled the false discovery rate (FDR) [3], by fitting p-values using a Beta-Uniform model [18]. We identified a subgroup of 3093 genes whose expression levels had been shown to be associated with patients' overall survival time (p-value < 0.0663, with estimated FDR $<$0.2).

Among the identified 3093 genes, the ROS1 gene is known as a biomarker for lung cancer [5, 7]. We consider the expression levels of the ROS1 gene as a response variable and the expression levels of the other 3092 genes as covariates. Both the response variable and covariates were standardized.

We chose the optimal regularization parameter λ that minimizes the BIC [19]

$$BIC(\lambda; \hat{\beta}_\lambda) = n \log\left(\sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{p}\hat{\beta}_{\lambda,j}X_{ij}\right)^2\right) + \log n \sum_{j=1}^{p} I(\hat{\beta}_{\lambda,j} \neq 0),$$

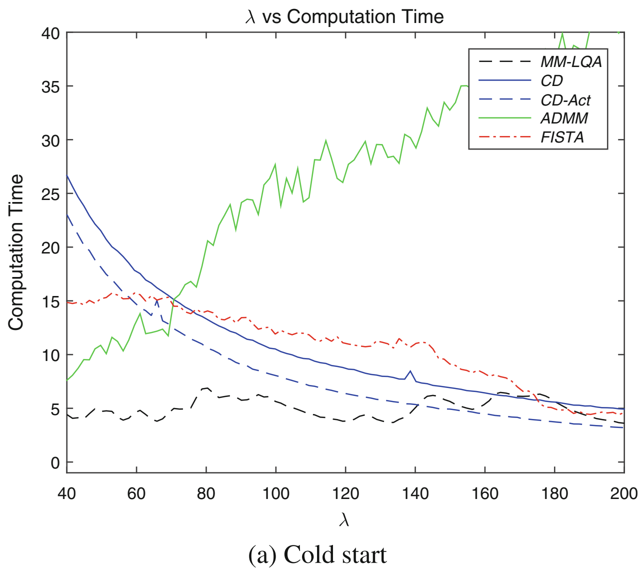
where $I(\cdot)$ denotes the indicator function, and $\hat{\beta}_\lambda$ refers to the estimated Lasso coefficients with the regularization parameter λ.

4.2 Results

For the purpose of the comparison of the computational algorithms, we measured the total computation time including the selection of λ using the BIC. As mentioned in

Table 5 Total computation times and number of iterations for calculating BIC

Algorithm	Total computation time		Average number of iterations	
	Cold start	Warm start	Cold start	Warm start
CD	1061.91	31.64	534.8	14.3
CD-Act	825.43	4.22	550.7	14.9
FISTA	1079.96	12.11	4802.7	48.1
MM-LQA	499.62	162.67	380.7	122.2
ADMM	2667.13	1165.6	1266.8	644.7



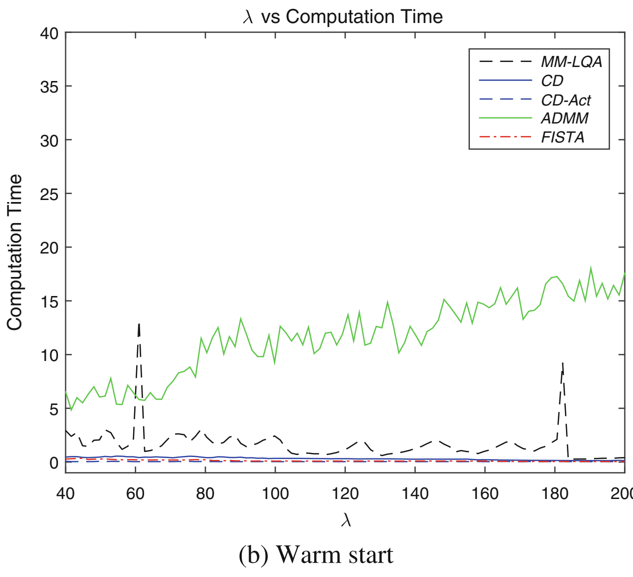


Fig. 4 Computation time vs. tuning parameter for the cold start (**a**) and warm start (**b**), for each algorithm

Section 2.1, the warm start strategy improves the overall computational efficiency when the solution path is sought. The results in Table 5 are made with both the warm and cold start strategies. In this Table, we see that MM-LQA shows the best performance both in convergence and computation time with the cold start strategy. On the other hand, CD with active shooting becomes the fastest when the warm start strategy is employed. This is because the active shooting strategy greatly reduces the number of variables to update for a high level of regularization.

We also plot the computation time for each tuning parameter in Fig. 4 for both strategies. It can be seen that the warm start strategy makes every algorithm faster. With increasing sparsity level, the speed of CD, CD-act, and FISTA increase; that of MM-LQA is almost static; and that of ADMM becomes worse.

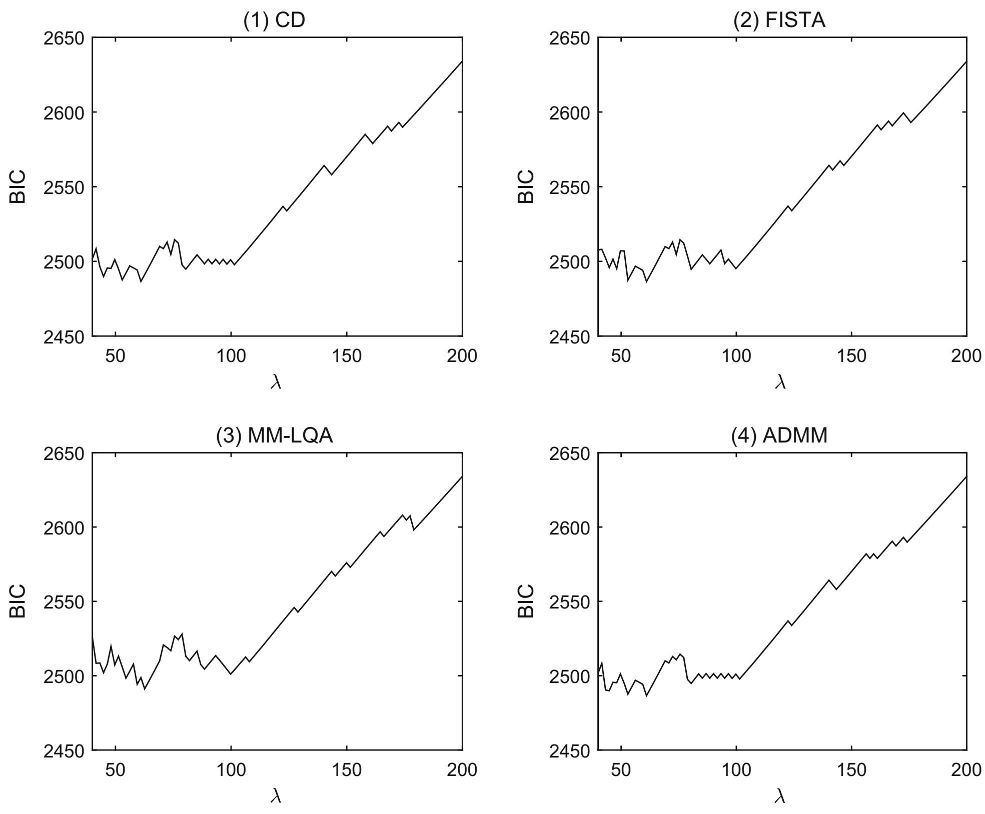


Fig. 5 Cross validation based on BIC criteria

In computing the BIC, each algorithm produced slightly different numbers due to the difference in convergence properties; see Fig. 5. Based on these plots, we chose $\lambda = 60$ since this value is close enough to the respective optima for all the algorithms. CD, FISTA, and ADMM selected exactly the same 24 variables, while MM-LQA selected two additional variables. We report the identified 24 genes and their regression coefficients in Table 6. Among the identified genes, the HLF, the PTPN13, the SFTPD, and the SLC34A2 genes were reported as lung cancer-related or lung injury-related genes in the literature [15, 25, 29]. Interestingly, these genes exhibit relatively strong interaction with ROS1 than the other genes in Table 6.

In addition, we investigated the functionality of the 3092 genes and their association with lung-related diseases using the DAVID, a well-known gene annotation tool [10]. From this investigation we found that there are 149 genes reported to be associated with lung diseases. If we consider this as the gold standard, consequently, the recall (sensitivity) and accuracy of our analysis can be said to be 2.68 % and 94.66 %, respectively. Yet, the full functionality of the 3092 genes has not been completely understood and much research efforts are currently being made in order to establish their functions and association with the diseases. To the best of of our knowledge, the genes reported in our manuscript are the only genes known to be related with the lung cancer or lung injuries such as the pulmonary embolism. We also would like to remark that our regression analysis can only find genes associated with the known oncogene ROS1, and it is unlikely that all the genes associated with lung diseases are necessarily associated with this gene. This may explain the low true positive rate of the our study.

5 Discussion

We have compared numerical performance of the CD, MM-LQA, FISTA, and ADMM algorithms for solving high-dimensional Lasso problems. Among these, MM-LQA show stable performance in both convergence and computation time. Its convergence speed does not depend much on the numerical condition of the data matrix, or the correlations between the covariates, both of which can be high in the high-dimensional setting. This is because a preconditioner of PCG employed in the inner iteration of the algorithm reduces the effect of ill-conditioning of the matrix. We used a simple preconditioner; use of a tailored preconditioner specific to the application may yield a better

Table 6 List of the identified genes interacting with the gene ROS1 and their estimated coefficients from the LASSO

No.	Gene	Estimated coefficient
1	SFTPD	0.165
2	HLF	0.159
3	SLC34A2	0.096
4	MACF1	0.048
5	FLAD1	-0.048
6	PTPN13	0.043
7	KCNJ15	0.042
8	C12orf44	-0.034
9	ALPL	0.029
10	GULP1	-0.027
11	MARK4	-0.024
12	SRSF4	0.018
13	MDFIC	0.018
14	PIGR	0.017
15	PFN2	-0.016
16	DTYMK	-0.016
17	GGA2	0.015
18	SLC35A1	0.015
19	IL6ST	0.012
20	CPA3	0.012
21	NPC2	0.008
22	DRAM1	0.007
23	PIK3R1	0.006
24	ADH1B	0.005

performance. Still, relative insensitivity to the choice of the preconditioner makes MM-LQA an attractive option.

Similar to MM-LQA, ADMM is also immune to the covariance structure and the dimension-to-sample size ratio. However, its lack of the descent property raises a caution when applying this to problems that needs precision.

CD is simple to implement and can be considerably accelerated by the active shooting algorithm. With either relatively high level of regularization or the warm start strategy, CD is competitive when the covariance matrix is well-conditioned, but it may take too long to produce the solution when the covariance matrix is ill-conditioned.

Table 7 Comparison of the algorithms with and without applying the basic strong rule for $p = 2500$, $n = 500$, and $\Sigma = \Sigma_4$, $\lambda = 0.75\lambda_{\max}$ and $0.9\lambda_{\max}$

λ	Method	Strong rule	Time	F-value	Sparsity	Discarded
$0.75\lambda_{\max}$	MM-LQA		9.5	63386.7	26.7	
			(1.7)	(21593.6)	(7.4)	
		•	6.5	63386.7	26.7	330.8
			(1.0)	(21593.6)	(7.4)	(12.9)
	CD		62.0	63385.1	6.3	
			(3.6)	(21592.8)	(1.2)	
		•	56.7	63385.1	6.5	330.8
			(3.3)	(21592.9)	(1.3)	(12.9)
	CD-Act		50.3	63385.1	6.4	
			(2.4)	(21592.8)	(1.2)	
		•	46.3	63385.1	6.3	330.8
			(1.7)	(21592.8)	(1.2)	(12.9)
	ADMM		9.4	63391.9	6.1	
			(0.5)	(21595.0)	(1.1)	
		•	7.5	63402.1	204.8	330.8
			(0.4)	(21592.6)	(198.4)	(12.9)
	FISTA		23.6	63428.0	19.6	
			(5.8)	(21610.9)	(4.6)	
		•	8.4	63419.9	17.3	330.8
			(0.8)	(21604.9)	(3.5)	(12.9)
$0.9\lambda_{\max}$	MM-LQA		8.0	65769.7	18.2	
			(1.5)	(22771.7)	(5.4)	
		•	7.6	65769.7	18.3	938.2
			(1.5)	(22771.8)	(5.4)	(147.6)
	CD		50.8	65768.1	2.5	
			(2.9)	(22771.0)	(0.3)	
		•	19.7	65768.1	2.6	938.2
			(2.6)	(22771.0)	(0.4)	(147.6)
	CD-Act		41.7	65768.1	2.6	
			(1.9)	(22771.0)	(0.4)	
		•	16.0	65768.1	2.6	938.2
			(2.2)	(22771.0)	(0.4)	(147.6)
	ADMM		13.2	65776.3	2.7	
			(0.6)	(22772.2)	(0.4)	
		•	6.3	65776.7	2.7	938.2
			(0.8)	(22771.2)	(0.4)	(147.6)
	FISTA		11.7	65796.7	12.9	
			(1.3)	(22785.9)	(3.6)	
		•	2.1	65790.1	10.4	938.2
			(0.3)	(22782.2)	(2.6)	(147.6)

Numbers within parentheses represent the standard errors of performance measures considered

FISTA has the fastest per-iteration computation time, but is susceptible to the regularization parameter. In addition, the inaccuracy of the solution is a weak point, and this is worsened with ill-conditioned covariance matrices.

5.1 Active set strategies

While the focus of this paper is a comparative study of the considered algorithms *per se*, in practice these algorithms can be combined with an active set strategy, which reduces the number of variables before the algorithm runs by setting the coefficients of the discarded variables to zero. For example, the CD algorithm with the active shooting strategy (Algorithm 2) can be considered an instance of this approach specialized to the CD algorithm. On the other hand, the "strong rules" [24] are based on the Karush-Kuhn-Tucker (KKT) optimality conditions and are independent of a particular algorithm. Thus the strong rules can be applied to all of the algorithms considered in order to reduce computation time.

To see the potential of this active set strategy, we have conducted an additional numerical study with and without a strong rule for $p = 2500$, $n = 500$ and $\Sigma = \Sigma_4$ defined in Section 3.1.1. We consider the "basic strong rule", as our interest is a fixed value of the tuning parameter λ. The basic strong rule discards variables satisfying the condition $|X_j^T y| < 2\lambda - \lambda_{\max}$, where $\lambda_{\max} = \max_i |X_i^T y|$ is that with which all the coefficients are zero. Because the basic strong rule is valid for $\lambda_{\max}/2 < \lambda < \lambda_{\max}$, we consider two large tuning parameters, $0.75\lambda_{\max}$ and $0.9\lambda_{\max}$, which lead highly sparse estimates. We report the average computation times, objective function values, sparsity levels, and number of discarded variables by the basic strong rule in Table 7, which indicates that the basic strong rule indeed improves computational efficiency of all the algorithms.

Acknowledgments Donghyeon Yu was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP, No. 2015R1C1A1A02036312). Joong-Ho Won was supported by the National Research Foundation of Kor ea (NRF) grant funded by the Korean government (MSIP, Nos. 2013R1A1A1057949 and 2014R1A4A1007895).

Appendix

A Preconditioned conjugate gradient (PCG) method

Conjugate gradient (CG) is a method to resolve positive definite linear equations, $Ax = b$, applied to sparse system that has too large data to solve with Cholesky Decomposition. Instead of directly solving linear equations, this method is to minimize the following function $f(x)$,

$$f(x) = \frac{1}{2}x^T Ax - b^T x.$$

For a positive definite A, two nonzero vectors u, v are said to be conjugate with respect to A, if they satisfy

$$\langle u, v\rangle_A \triangleq u^T Av = 0.$$

If P is defined as

$$P = \{p_k : \forall i \neq k, \langle p_i, p_k\rangle_A = 0\},$$

it means the set of n number of mutually conjugate directional vectors. Thus, the set P becomes a basis of $\mathbb{R}^n$ and x is represented in the form of

$$x = \sum_{i=1}^{n} \alpha_i p_i.$$

By multiplying both sides by matrix A, b is decomposed by

$$b = Ax = \sum_{i=1}^{n} \alpha_i Ap_i.$$

Multiplying an arbitrary directional vector $p_k \in P$,

$$p_k^T b = p_k^T Ax = \sum_{i=1}^{n} \alpha_i p_k^T Ap_i = \alpha_k p_k^T Ap_k.$$

Accordingly, the explicit form of α_k can be derived as followed,

$$\alpha_k = \frac{p_k^T b}{p_k^T Ap_k} = \frac{\langle p_k, b\rangle}{\|p_k\|_A^2}.$$

If mutually conjugate directional vectors are not given, conjugate gradient (CG) solves the problem iteratively. Set x_0 as an initial value of x, and a linear equation given by

$$Az = b - Ax_0$$

becomes a target function to solve. If we regarding $r_k = b - Ax_k$ as k-th residual, r_k becomes a negative gradient of convex function $x = x_k$, $\nabla f(x)$ given by,

$$\nabla f(x_k) = Ax_k - b,$$

which means that conjugate gradient method moves toward the direction of r_k. Since all directional vectors should satisfy the condition that all vectors are conjugate with respect to A, then k-th direction p_k is given by,

$$p_k = r_k - \sum_{i>k} \frac{p_i^T Ar_k}{p_i^T Ap_i} p_i.$$

Following this direction, next value of x is updated as followed,

$$x_{k+1} = x_k + \alpha_k p_k,$$

where

$$\alpha_k = \frac{p_k^T b}{p_k^T A p_k} = \frac{p_k^T r_{k-1}}{p_k^T A p_k}.$$

Convergence rate of conjugate gradient method depends on condition number of A and especially eigenvalues of A [21]. Accordingly, $Ax = b$ problem can be regarded same as linear equation that multiply by inverse matrix of preconditioner given by

$$M^{-1}Ax = M^{-1}b.$$

In choosing an appropriate preconditioner, it should satisfy some necessary conditions.

- M is both symmetric and positive definite matrix.
- $M^{-1}A$ is well conditioned and hardly has extreme eigenvalues.
- $Mx = b$ is easy to solve.

Widely used preconditioners that satisfy these conditions are the followings;

1) Diagonal: $M = \text{diag}(1/A_{11}, ..., 1/A_{nn})$,
2) Incomplete(approximate) Cholesky factorization: $M = \hat{A}^{-1}$, where $\hat{A} = \hat{L}\hat{L}^T$.

Algorithm 6 PCG Algorithm

```
1: Initialize : $x = 0, r = b - Ax_0, p = r, z = Mr, \rho_1 = r^T z$
2: while $\sqrt{\rho_k} > \epsilon \|b\|_2$ or $\|r\| > \epsilon \|b\|_2$ do
3:     $w = Ap$
4:     $\alpha = \rho_k / (w^T p)$
5:     $x = x + \alpha p$
6:     $r = r - \alpha w$
7:     $z = Mr$
8:     $\rho_{k+1} = z^T r$
9:     $p = z + \frac{\rho_{k+1}}{\rho_k} p$
10: end while
```

References

1. Akaike H (1973) Information theory and an extension of the maximum likelihood principle. In: Second International Symposium on Information Theory, pp 267–281
2. Beck A, Teboulle M (2009) A Fast Iterative Shrinkage-Thresholding Algorithm fo Linear Inverse Problems. SIAM J. Imaging Sciences, doi:10.1137/080716542
3. Bejamini Y, Hochberg Y (1995) Controlling the false discovery rate: a practical and powerful approach to multiple testing. J Royal Stat Soc Ser B 57(1):289–300
4. Boyd S, Parikh N, Chu E, Peleato B, Eckstein J (2010) Distributed optimization and statistical learning via the alternating direction method of multipliers. Found Trends Mach Learn 3(1):1–122
5. Cagle PT, Allen TC, Olsen RJ (2013) Lung cancer biomarkers: present status and future developments. Arch Pathol Labor Med 137(9):1191–1198
6. Efron B, Hastie T, Johnstone I, Tibshirani R (2004) Least angle regression. Ann Stat 32(2):407–499
7. El-Telbany A, Ma PC (2012) Cancer genes in lung cancer: racial disparities: are there any? Genes Cancer 3:467–480
8. Friedman J, Hastie T, Hofling H, Tibshirani R (2007) Pathwise coordinate optimization. Ann Appl Stat 1(2):307–332
9. Gemmeke JF, Hamme HV, Cranen B, Boves L (2010) Compressive sensing for missing data imputation in noise robust speech recognitions. J Sel Topics Signal Process 4(2):272–287
10. Huang DW, Sherman BT, Lempicki RA (2009) Systematic and integrative analysis of large gene lists using david bioinformatics resources. Nat Protoc 4(1):44–57
11. Hunter DR, Lange K (2000) Quantile regression via an mm algorithm. Journal of Computational and Graphical Statistics pp 60–77
12. Hunter DR, Li R (2005) Variable selection using mm algorithms. Ann Stat 33(4):1617–1642
13. Irizarry RA, Bolstad BM, Collin F, Cope LM, Hobbs B, Speed TP (2003) Summaries of affymetrix genechip probe level data. Nucleic Acids Res 31(4):e15
14. Jemal A, Siegel R, Xu J, Ward E (2010) Cancer statistics. Cancer J Clin 60(5):277–300
15. Kati C, Alacam H, Duran L, Guzel A, Akdemir HU, Sisman B, Sahin C, Yavuz Y, Altintas N, Murat N, Okuyucu A (2014) The effectiveness of the serum surfactant protein d (sp-d) level to indicate lung injury in pulmonary embolism. Clin Lab 60(9):1457–1464
16. Parikh N, Boyd S (2013) Proximal algorithms. Found Trends Optim 1(3):123–231
17. Peng J, Wang P, Zhou N, Zhu J (2012) Partial correlation estimation by joint sparse regression models. Journal of the American Statistical Association
18. Pounds S, Morris SW (2003) Estimating the occurrence of false positives and false negatives in microarray studies by approximating and partitioning the empirical distribution of p-values. Bioinformatics 19(10):1236–1242
19. Schwarz G (1978) Estimating the dimension of a model. Ann Stat 6(2):461–464
20. Shedden K, Taylor JM, Enkemann SA, Tsao MS, Yeatman TJ, Gerald WL, Eschrich S, Jurisica I, Giordano TJ, Misek DE, Chang AC, Zhu CQ, Strumpf D, Hanash S, Shepherd FA, Ding K, Seymour L, Naoki K, Pennell N, Weir B, Verhaak R, Ladd-Acosta C, Golub T, Gruidl M, Sharma A, Szoke J, Zakowski M, Rusch V, Kris M, Viale A, Motoi N, Travis W, Conley B, Seshan VE, Meyerson M, Kuick R, Dobbin KK, Lively T, Jacobson JW, Beer DG (2008) Gene expression-based survival prediction in lung adenocarcinoma: a multi-site, blinded validation study. Nat Med 14(8):822–827
21. Shewchuk JR (1994) An introduction to the conjugate gradient method without the agonizing pain. Carnegie Mellon University, Pittsburgh, PA
22. Tang H, Xiao G, Behrens C, Schiller J, Allen J, Chow CW, Suraokar M, Corvalan A, Mao J, White MA, Wistuba II, Minna JD, Xie Y (2013) A 12-gene set predicts survival benefits from adjuvant chemotherapy in non-small cell lung cancer patients. Clin Cancer Res 19(6):1577–1586
23. Tibshirani R (1996) Regression shrinkage and selection via the lasso. J Royal Stat Soc Ser B 58:267–288
24. Tibshirani R, Bien J, Friedman J, Hastie T, Simon N, Taylor J, Tibshirani RJ (2012) Strong rules for discarding predictors in lasso-type problems. J Royal Stat Soc Ser B (Stat Methodol) 74(2):245–266

25. Woenckhaus M, Klein-Hitpass L, Grepmeier U, Merk J, Pfeifer M, Wild P, Bettstetter M, Wuensch P, Blaszyk H, Hartmann A, et al. (2006) Smoking and cancer-related gene expression in bronchial epithelium and non-small-cell lung cancers. J Pathol 210(2):192–204
26. Wright J, Yang AY, Ganesh A, Sastry S, Ma Y (2009) Robust face recognition via sparse representation. IEEE Trans Pattern Anal Mach Intell 31(2):210–227
27. Wu TT, Lange K (2008) Coordinate descent algorithms for lasso penalizaed regression. Ann Appl Stat 2(1):224–244
28. Yang AY, Zhou Z, Ganesh A, Shankar SS, Ma Y (2013) Fast l1-minimization algorithms for robust face recognition. IEEE Trans Image Process 22:8
29. Yu D, Son W, Lim J, Xiao G (2015) Statistical completion of partially identified graph with application to estimation of gene regulatory network. Biostatistics 16(4):670–685