Acta Mathematica Sinica, English Series
Dec., 2018, Vol. 34, No. 12, pp. 1892–1906
Published online: January 25, 2018
https://doi.org/10.1007/s10114-018-7096-8
http://www.ActaMath.com

Acta Mathematica Sinica, English Series

An Alternating Direction Method of Multipliers for MCP-penalized Regression with High-dimensional Data

Yue Yong SHI
School of Economics and Management, China University of Geosciences, Wuhan 430074*, P. R. China*
and
Center for Resources and Environmental Economic Research, China University of Geosciences, Wuhan 430074*, P. R. China*
E-mail: *yueyongshi@cug.edu.cn*

Yu Ling JIAO **Yong Xiu CAO**
School of Statistics and Mathematics, Zhongnan University of Economics and Law, Wuhan 430073*, P. R. China*
E-mail: *yljiaostatistics@zuel.edu.cn* *yxcao@zuel.edu.cn*

Yan Yan LIU[1)]
School of Mathematics and Statistics, Wuhan University, Wuhan 430072*, P. R. China*
E-mail: *liuyy@whu.edu.cn*

Abstract The minimax concave penalty (MCP) has been demonstrated theoretically and practically to be effective in nonconvex penalization for variable selection and parameter estimation. In this paper, we develop an efficient alternating direction method of multipliers (ADMM) with continuation algorithm for solving the MCP-penalized least squares problem in high dimensions. Under some mild conditions, we study the convergence properties and the Karush–Kuhn–Tucker (KKT) optimality conditions of the proposed method. A high-dimensional BIC is developed to select the optimal tuning parameters. Simulations and a real data example are presented to illustrate the efficiency and accuracy of the proposed method.

Keywords Alternating direction method of multipliers, coordinate descent, continuation, high-dimensional BIC, minimax concave penalty, penalized least squares

MR(2010) Subject Classification 62J05, 62J07, 62J99

1 Introduction

Consider the linear regression model

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{1.1}$$

Received February 27, 2017, revised June 5, 2017, accepted June 21, 2017
Supported by the National Natural Science Foundation of China (Grant Nos. 11571263, 11501579, 11701571 and 41572315) and the Fundamental Research Funds for the Central Universities, China University of Geosciences (Wuhan) (Grant No. CUGW150809)
1) Corresponding author

where $\mathbf{y} = (y_1, \ldots, y_n)^T \in \mathbb{R}^n$ is a response vector, $\mathbf{X} = (x_{ij})_{n\times d} = (\mathbf{x}_1, \ldots, \mathbf{x}_d) \in \mathbb{R}^{n\times d}$ is a design matrix, $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_d)^T \in \mathbb{R}^d$ is an unknown sparse coefficient vector of interest and $\boldsymbol{\varepsilon} = (\varepsilon_1, \ldots, \varepsilon_n)^T \in \mathbb{R}^n$ is a noise vector of i.i.d. random variables with mean zero and finite variance σ^2. We assume that $\boldsymbol{\beta}$ is sparse in the sense that only a small portion of the components of $\boldsymbol{\beta}$ are nonzero. We focus on the high dimensional case where $d > n$ and our goal is to reconstruct the unknown vector $\boldsymbol{\beta}$. To achieve sparsity, penalization (or regularization) methods have been widely used in the literature (e.g., [4]). Let $\|\cdot\|$ be the Euclidean norm. In this paper, we consider the following so-called MCP-penalized least squares (PLS) problem:

$$\hat{\boldsymbol{\beta}} \triangleq \hat{\boldsymbol{\beta}}(\lambda, \gamma) := \arg\min_{\boldsymbol{\beta}\in\mathbb{R}^d} \left\{ F(\boldsymbol{\beta}) = \frac{1}{2n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \sum_{j=1}^{d} p_{\lambda,\gamma}(\beta_j) \right\}, \tag{1.2}$$

where $p_{\lambda,\gamma}(\beta_j)$ is the MCP (minimax concave penalty) proposed by [34], which is defined as

$$p_{\lambda,\gamma}(t) = \lambda \int_0^{|t|} \left(1 - \frac{x}{\lambda\gamma}\right)_+ dx = \begin{cases} \lambda|t| - \frac{t^2}{2\gamma}, & \text{if } |t| \le \lambda\gamma, \\ \frac{\lambda^2\gamma}{2}, & \text{if } |t| > \lambda\gamma, \end{cases} \tag{1.3}$$

and in consequence its derivative is given by

$$p'_{\lambda,\gamma}(t) = \lambda\left(1 - \frac{|t|}{\lambda\gamma}\right)_+ \operatorname{sgn}(t) = \begin{cases} \lambda \operatorname{sgn}(t) - \frac{t}{\gamma}, & \text{if } |t| \le \lambda\gamma, \\ 0, & \text{if } |t| > \lambda\gamma, \end{cases} \tag{1.4}$$

where λ and γ are two positive tuning (or regularization) parameters with $\gamma > 1$, and $\operatorname{sgn}(t) = -1$, 0, or 1 if $t < 0$, $= 0$, or > 0. In particular, λ is the sparsity tuning parameter obtaining sparse solutions and γ is the shape (or concavity) tuning parameter making MCP a bridge between L_0 ($\gamma \to 1+$) and L_1 ($\gamma \to \infty$), where L_0 and L_1 admit $p_{\lambda,\gamma}(t) = \lambda I(|t| \neq 0)$ and $p_{\lambda,\gamma}(t) = \lambda|t|$ respectively. $\hat{\boldsymbol{\beta}}$, which is dependent on λ and γ, is denoted as a MPLS (MCP-PLS) estimator. Without loss of generality, we assume that $\sum_{i=1}^n x_{ij} = 0$, $n^{-1}\sum_{i=1}^n x_{ij}^2 = 1$ and $\sum_{i=1}^n y_i = 0$. It is noteworthy that the estimator of one-dimensional MPLS

$$\hat{\theta} = \arg\min_{\theta\in\mathbb{R}} \left\{ \frac{1}{2}(z - \theta)^2 + p_{\lambda,\gamma}(\theta) \right\} \tag{1.5}$$

is a thresholding rule which can be given as

$$\hat{\theta} = T_{\lambda,\gamma}(z) = \begin{cases} \dfrac{S_\lambda(z)}{1 - 1/\gamma}, & \text{if } |z| \le \lambda\gamma, \\ z, & \text{if } |z| > \lambda\gamma, \end{cases} \tag{1.6}$$

where $S_\lambda(z) = \operatorname{sgn}(z)(|z| - \lambda)_+$ is the soft-thresholding operator. $T_{\lambda,\gamma}(\cdot)$ is known as the firm thresholding operator introduced in [7].

As pointed out in [10], popular penalties LASSO, SCAD and MCP can give rise to closed-form solutions to one dimensional PLS, and they all belong to the family of quadratic spline penalties with the sparsity and continuity properties, while MCP is the simplest penalty that results in an estimator that is nearly unbiased, sparse and continuous.

The coordinate descent (CD) algorithm [3, 19] is a popular algorithm that has been proposed for solving the resulting nonconvex MPLS problem (1.2) when $d > n$. The alternating direction method of multipliers (ADMM) [2], which is a powerful algorithm that combines the

benefits of dual decomposition and method of multipliers, can solve nonsmooth and nonconvex problems involving high-dimensional data in reasonable computational cost. Furthermore, it is well suited to parallel and distributed sparse optimization. ADMM and its many variants have recently been widely used to solve large-scale problems in compressed sensing, signal and image processing, machine learning and statistics [9, 15, 16, 18, 23, 28–31, 33]. In this paper, we develop an ADMM with continuation algorithm for solving the MPLS problem in high dimensions. Under some mild conditions, we study the convergence properties and corresponding KKT optimality conditions of the MPLS-ADMM algorithm. We adopt a high-dimensional BIC (HBIC) [27] to select the optimal tuning parameter. When coupled with the continuation strategy, the proposed MPLS-ADMM-HBIC procedure is very efficient and accurate.

The remainder of this paper is organized as follows. In Section 2, we develop the MPLS-ADMM algorithm for linear regression in high dimensions and study its convergence properties and KKT optimality conditions. The computational complexity analysis and the tuning parameter selection procedure are given in Section 3. Simulation studies are also conducted in Section 3 to evaluate the finite sample performance of the proposed algorithm, which is further illustrated with a real data example, comparing with the popular CD algorithm. We conclude the paper in Section 4.

2 MPLS-ADMM Algorithm

2.1 Methodology

We study the ADMM algorithm [2] for solving the MPLS problem (1.2) and discuss its algorithmic implementation details.

We introduce an auxiliary variable $\boldsymbol{\theta} \in \mathbb{R}^d$ and rewrite (1.2) as follows:

$$(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}) := \arg\min_{\boldsymbol{\beta},\boldsymbol{\theta}} \left\{ F(\boldsymbol{\beta}, \boldsymbol{\theta}) = \frac{1}{2n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \sum_{j=1}^{d} p_{\lambda,\gamma}(\theta_j) \right\} \quad \text{s.t. } \boldsymbol{\beta} = \boldsymbol{\theta}. \tag{2.1}$$

The corresponding augmented Lagrangian function of problem (2.1) is

$$\mathcal{L}_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau}) = \frac{1}{2n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \sum_{j=1}^{d} p_{\lambda,\gamma}(\theta_j) + \langle \boldsymbol{\beta} - \boldsymbol{\theta}, \boldsymbol{\tau} \rangle + \frac{\rho}{2}\|\boldsymbol{\beta} - \boldsymbol{\theta}\|^2, \tag{2.2}$$

where $\boldsymbol{\tau} \in \mathbb{R}^d$ is the Lagrangian multiplier, $\langle \cdot, \cdot \rangle$ denotes the inner product operator and $\rho > 0$ is the penalty parameter for the violation of the linear constraint. It is noteworthy that we have $\mathcal{L}_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau}) = F(\boldsymbol{\beta}, \boldsymbol{\theta}) = F(\boldsymbol{\beta})$ when $\boldsymbol{\beta} = \boldsymbol{\theta}$. Each iteration of the ADMM involves alternating minimization of $\mathcal{L}_\rho$ with respect to $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$, followed by an update of $\boldsymbol{\tau}$. In particular, given $(\boldsymbol{\beta}^k, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k)$, the iteration scheme of the ADMM for problem (2.2) can be described as follows:

$$\begin{cases} \boldsymbol{\beta}^{k+1} \in \arg\min_{\boldsymbol{\beta} \in \mathbb{R}^d} \mathcal{L}_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k), & (2.3) \\ \boldsymbol{\theta}^{k+1} \in \arg\min_{\boldsymbol{\theta} \in \mathbb{R}^d} \mathcal{L}_\rho(\boldsymbol{\beta}^{k+1}, \boldsymbol{\theta}, \boldsymbol{\tau}^k), & (2.4) \\ \boldsymbol{\tau}^{k+1} = \boldsymbol{\tau}^k + \rho(\boldsymbol{\beta}^{k+1} - \boldsymbol{\theta}^{k+1}). & (2.5) \end{cases}$$

After some algebra, the $\boldsymbol{\beta}$-subproblem (2.3) can be reformulated as

$$\boldsymbol{\beta}^{k+1} = (n^{-1}\mathbf{X}^T\mathbf{X} + \rho\mathbf{I}_d)^{-1}(n^{-1}\mathbf{X}^T\mathbf{y} + \rho\boldsymbol{\theta}^k - \boldsymbol{\tau}^k), \tag{2.6}$$

where $\mathbf{I}_d$ is the $d \times d$ identity matrix. In practice, it may be expensive to solve the linear system (2.6) directly, especially when $d > n$. Noticing that

$$(n^{-1}\mathbf{X}^T\mathbf{X} + \rho\mathbf{I}_d)^{-1} = (\mathbf{I}_d - \mathbf{X}^T(\rho\mathbf{I}_n + n^{-1}\mathbf{X}\mathbf{X}^T)^{-1}n^{-1}\mathbf{X})\rho^{-1}, \tag{2.7}$$

where $\mathbf{I}_n$ is the $n \times n$ identity matrix and $\rho\mathbf{I}_n + n^{-1}\mathbf{X}\mathbf{X}^T$ is well defined since $d > n$, then (2.6) can be solved effectively by employing Cholesky factorization of $\rho\mathbf{I}_n + n^{-1}\mathbf{X}\mathbf{X}^T$.

Equivalently, the $\boldsymbol{\theta}$-subproblem (2.4) can be written as

$$\boldsymbol{\theta}^{k+1} = \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^d}\left\{\frac{1}{2}\|\boldsymbol{\theta} - \mathbf{z}^k\|^2 + \sum_{j=1}^{d} p_{\lambda/\rho,\gamma}(\theta_j)\right\}, \tag{2.8}$$

where $\mathbf{z}^k = \boldsymbol{\beta}^{k+1} + \frac{\boldsymbol{\tau}^k}{\rho}$. Then it follows that

$$\boldsymbol{\theta}^{k+1} = T_{\lambda/\rho,\gamma}\left(\boldsymbol{\beta}^{k+1} + \frac{\boldsymbol{\tau}^k}{\rho}\right), \tag{2.9}$$

where $T_{\lambda/\rho,\gamma}(\cdot)$ is the element-wise firm thresholding operator defined in (1.6).

Given tuning parameters λ and γ, we propose the MPLS-ADMM implementation in Algorithm 1.

Algorithm 1 MPLS-ADMM

Input: tuning parameters $\lambda > 0,\ \gamma > 1$; constant $\rho > 0$; set $k = 0$; initial values $\boldsymbol{\beta}^0 \in \mathbb{R}^d,\ \boldsymbol{\theta}^0 \in \mathbb{R}^d,\ \boldsymbol{\tau}^0 \in \mathbb{R}^d$; maximum number of iterations M.

1: **while** $k < M$ **do**
2: update $\boldsymbol{\beta}^{k+1}$ using (2.6) and (2.7);
3: **for** $j = 1, 2, \ldots, d$ **do**
4: update θ_j^{k+1} using (2.9);
5: **end for**
6: update $\boldsymbol{\tau}^{k+1}$ using (2.5);
7: check the stopping criterion.
8: **end while**

Output: $(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}})$, the estimate of $(\boldsymbol{\beta}, \boldsymbol{\theta})$ in (2.1).

2.2 Convergence Analysis

We establish that, under some regularity conditions, the proposed MPLS-ADMM algorithm converges to a KKT point of $\mathcal{L}_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau})$, which is a triple $(\boldsymbol{\beta}^*, \boldsymbol{\theta}^*, \boldsymbol{\tau}^*)$ satisfying the following system:

$$\begin{cases} \boldsymbol{\theta}^* = T_{\lambda/\rho,\gamma}\left(\boldsymbol{\beta}^* + \dfrac{\boldsymbol{\tau}^*}{\rho}\right), \\ \dfrac{1}{n}\mathbf{X}^T(\mathbf{X}\boldsymbol{\beta}^* - \mathbf{y}) + \boldsymbol{\tau}^* = 0, \\ \boldsymbol{\beta}^* = \boldsymbol{\theta}^*. \end{cases} \tag{2.10}$$

Theorem 2.1 (Convergence property of MPLS-ADMM) *Let* $\{(\boldsymbol{\beta}^k, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k)\}$ *be a sequence generated by* MPLS-ADMM. *Assume that* $\lim_{k\to\infty}\|\boldsymbol{\tau}^{k+1} - \boldsymbol{\tau}^k\| = 0$ *and* $\{\boldsymbol{\theta}^k\}$ *is bounded, then there exists a subsequence of* $\{(\boldsymbol{\beta}^k, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k)\}$ *such that it converges to a* KKT *point satisfying* (2.10).

Proof Since

$$\lim_{k\to\infty} \|\boldsymbol{\tau}^{k+1} - \boldsymbol{\tau}^k\| = 0 \tag{2.11}$$

and $\rho > 0$, we get from (2.5) that

$$\lim_{k\to\infty} \|\boldsymbol{\beta}^{k+1} - \boldsymbol{\theta}^{k+1}\| = \lim_{k\to\infty} \frac{1}{\rho}\|\boldsymbol{\tau}^{k+1} - \boldsymbol{\tau}^k\| = 0. \tag{2.12}$$

Then $\{\boldsymbol{\beta}^k\}$ is bounded by (2.12) and the boundedness assumption on $\{\boldsymbol{\theta}^k\}$. It follows from (2.6) and the boundedness of $\{\boldsymbol{\theta}^k\}$ and $\{\boldsymbol{\beta}^k\}$ that $\{\boldsymbol{\tau}^k\}$ is also bounded. Since $\{(\boldsymbol{\beta}^k, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k)\}$ is bounded and the augmented Lagrangian function $\mathcal{L}_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau})$ is continuous, we can obtain that $\mathcal{L}_\rho(\boldsymbol{\beta}^k, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k)$ is bounded. It is obvious that $\mathcal{L}_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau})$ is strongly convex with respect to the variable $\boldsymbol{\beta}$, so it holds that for any $\boldsymbol{\beta}$ and $\Delta\boldsymbol{\beta}$,

$$\mathcal{L}_\rho(\boldsymbol{\beta} + \Delta\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau}) - \mathcal{L}_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau}) \geq \langle \nabla_{\boldsymbol{\beta}} \mathcal{L}_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau}), \Delta\boldsymbol{\beta}\rangle + c\|\Delta\boldsymbol{\beta}\|^2, \tag{2.13}$$

where $c > 0$ is a constant. Since $\boldsymbol{\beta}^{k+1}$ minimizes (2.3), we further have

$$\langle \nabla_{\boldsymbol{\beta}} \mathcal{L}_\rho(\boldsymbol{\beta}^{k+1}, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k), (\boldsymbol{\beta}^k - \boldsymbol{\beta}^{k+1})\rangle \geq 0. \tag{2.14}$$

By letting $\Delta\boldsymbol{\beta} = \boldsymbol{\beta}^k - \boldsymbol{\beta}^{k+1}$ in (2.13) and combining with (2.14), we can obtain that

$$\mathcal{L}_\rho(\boldsymbol{\beta}^k, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k) - \mathcal{L}_\rho(\boldsymbol{\beta}^{k+1}, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k) \geq c\|\boldsymbol{\beta}^{k+1} - \boldsymbol{\beta}^k\|^2. \tag{2.15}$$

Moreover, since $\boldsymbol{\theta}^{k+1}$ minimizes (2.4), we have

$$\mathcal{L}_\rho(\boldsymbol{\beta}^{k+1}, \boldsymbol{\theta}^{k+1}, \boldsymbol{\tau}^k) \leq \mathcal{L}_\rho(\boldsymbol{\beta}^{k+1}, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k). \tag{2.16}$$

Thus together with (2.15) and (2.5), we get that

$$\mathcal{L}_\rho(\boldsymbol{\beta}^k, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k) - \mathcal{L}_\rho(\boldsymbol{\beta}^{k+1}, \boldsymbol{\theta}^{k+1}, \boldsymbol{\tau}^{k+1}) + \frac{1}{\rho}\|\boldsymbol{\tau}^{k+1} - \boldsymbol{\tau}^k\|^2 \geq c\|\boldsymbol{\beta}^{k+1} - \boldsymbol{\beta}^k\|^2. \tag{2.17}$$

Denote $G_k^L = \mathcal{L}_\rho(\boldsymbol{\beta}^k, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k)$, $G_k^{\boldsymbol{\tau}} = \frac{1}{\rho}\|\boldsymbol{\tau}^{k+1} - \boldsymbol{\tau}^k\|^2$ and $G_k^{\boldsymbol{\beta}} = c\|\boldsymbol{\beta}^{k+1} - \boldsymbol{\beta}^k\|^2$. Then we rewrite (2.17) as

$$G_k^L - G_{k+1}^L + G_k^{\boldsymbol{\tau}} \geq G_k^{\boldsymbol{\beta}} \geq 0. \tag{2.18}$$

Since G_k^L is bounded, there exists a subsequence k_j such that

$$\lim_{k_j\to\infty} G_{k_j}^L = \lim_{k\to\infty} G_k^L.$$

It follows from (2.18), the nonnegativity of $G_k^{\boldsymbol{\beta}}$ and $G_k^{\boldsymbol{\tau}}$, and the assumption $G_k^{\boldsymbol{\tau}} \to 0$, that

$$\begin{aligned} 0 &\leq \lim_{k_j\to\infty} G_{k_j}^{\boldsymbol{\beta}} \leq \lim_{k_j\to\infty} (G_{k_j}^L - G_{k_j+1}^L + G_{k_j}^{\boldsymbol{\tau}}) \\ &\leq \lim_{k_j\to\infty} (G_{k_j}^L + G_{k_j}^{\boldsymbol{\tau}}) - \lim_{k_j\to\infty} G_{k_j+1}^L \leq 0, \end{aligned} \tag{2.19}$$

which implies

$$\lim_{k_j\to\infty} G_{k_j}^{\boldsymbol{\beta}} = 0, \tag{2.20}$$

i.e.,

$$\lim_{k_j\to\infty} \|\boldsymbol{\beta}^{k_j+1} - \boldsymbol{\beta}^{k_j}\| = 0. \tag{2.21}$$

Together with (2.12), we can also get that

$$\lim_{k_j\to\infty}\|\boldsymbol{\theta}^{k_j+1}-\boldsymbol{\theta}^{k_j}\|=0. \tag{2.22}$$

Then, by the boundedness of $\{\boldsymbol{\beta}^{k_j},\boldsymbol{\theta}^{k_j},\boldsymbol{\tau}^{k_j}\}$, there exists a convergence subsequence, still be denoted by $\{k_j\}$, such that $\{\boldsymbol{\beta}^{k_j},\boldsymbol{\theta}^{k_j},\boldsymbol{\tau}^{k_j}\}$ converges to some point $(\boldsymbol{\beta}^*,\boldsymbol{\theta}^*,\boldsymbol{\tau}^*)$. By the fact

$$\lim_{k_j\to\infty}\boldsymbol{\beta}^{k_j}=\boldsymbol{\beta}^*,\quad \lim_{k_j\to\infty}\boldsymbol{\theta}^{k_j}=\boldsymbol{\theta}^* \tag{2.23}$$

and (2.12), we can obtain that

$$\boldsymbol{\beta}^*=\boldsymbol{\theta}^*. \tag{2.24}$$

After some algebra, (2.6) can be transformed into the following form:

$$\frac{\mathbf{X}^T(\mathbf{X}\boldsymbol{\beta}^{k+1}-\mathbf{y})}{n}=-\rho\boldsymbol{\beta}^{k+1}+\rho\boldsymbol{\theta}^k-\boldsymbol{\tau}^k. \tag{2.25}$$

Taking the limit of both sides of (2.25) on k_j and together with (2.11) and (2.23), it follows that

$$\frac{1}{n}\mathbf{X}^T(\mathbf{X}\boldsymbol{\beta}^*-\mathbf{y})+\boldsymbol{\tau}^*=0. \tag{2.26}$$

Using (2.21) and taking the limit of the both sides of (2.9) on k_j, we get

$$\boldsymbol{\theta}^*=T_{\lambda/\rho,\gamma}\bigg(\boldsymbol{\beta}^*+\frac{\boldsymbol{\tau}^*}{\rho}\bigg). \tag{2.27}$$

Combining (2.27) with (2.24) and (2.26), we obtain that $(\boldsymbol{\beta}^*,\boldsymbol{\theta}^*,\boldsymbol{\tau}^*)$ is a KKT point of $F(\boldsymbol{\beta})$ satisfying (2.10). □

Next, we study the optimality of the KKT point $(\boldsymbol{\beta}^*,\boldsymbol{\theta}^*,\boldsymbol{\tau}^*)$ with $\rho=1$.

Theorem 2.2 (KKT optimality condition) *Let $\boldsymbol{\beta}^*$ be a global minimizer of* MPLS (1.2)*, then there exist $\boldsymbol{\theta}^*$ and $\boldsymbol{\tau}^*$ such that* (2.10) *holds with $\rho=1$. Conversely, if $(\boldsymbol{\beta}^*,\boldsymbol{\theta}^*,\boldsymbol{\tau}^*)$ satisfies* (2.10) *with $\rho=1$, then $\boldsymbol{\beta}^*$ is a coordinate-wize minimizer and a stationary point of* (1.2)*.*

Proof Suppose $\boldsymbol{\beta}^*=(\beta_1^*,\dots,\beta_d^*)^T$ is a minimizer of $F(\boldsymbol{\beta})$ in (1.2). Then

$$\begin{aligned}
&\beta_j^*\in\arg\min_{t\in\mathbb{R}}F(\beta_1^*,\dots,\beta_{j-1}^*,t,\beta_{j+1}^*,\dots,\beta_d^*)\\
&\quad\Rightarrow\ \beta_j^*\in\arg\min_{t\in\mathbb{R}}\frac{1}{2n}\|\mathbf{X}\boldsymbol{\beta}^*-\mathbf{y}+(t-\beta_j^*)\mathbf{X}_j)\|_2^2+p_{\lambda,\gamma}(t)\\
&\quad\Rightarrow\ \beta_j^*\in\arg\min_{t\in\mathbb{R}}\frac{1}{2}(t-\beta_j^*)^2+\tfrac{1}{n}(t-\beta_j^*)\mathbf{X}_j^T(\mathbf{X}\boldsymbol{\beta}^*-\mathbf{y})+p_{\lambda,\gamma}(t)\\
&\quad\Rightarrow\ \beta_j^*\in\arg\min_{t\in\mathbb{R}}\frac{1}{2}\bigg(t-\beta_j^*-\frac{\mathbf{X}_j^T(\mathbf{y}-\mathbf{X}\boldsymbol{\beta}^*)}{n}\bigg)^2+p_{\lambda,\gamma}(t),
\end{aligned} \tag{2.28}$$

where $\mathbf{X}_j$ is the jth column of $\mathbf{X}$. Let

$$\boldsymbol{\tau}^*=\mathbf{X}^T(\mathbf{y}-\mathbf{X}\boldsymbol{\beta}^*)/n. \tag{2.29}$$

By the definition of the thresholding operator in (1.6), we have

$$\beta_j^*=T_{\lambda,\gamma}(\beta_j^*+\tau_j^*),\quad j=1,2,\dots,d. \tag{2.30}$$

Let $\boldsymbol{\theta}^* = \boldsymbol{\beta}^*$. From equations (2.28), (2.29) and (2.30), it follows that $(\boldsymbol{\beta}^*, \boldsymbol{\theta}^*, \boldsymbol{\tau}^*)$ satisfies (2.10) with $\rho = 1$. Conversely, if $(\boldsymbol{\beta}^*, \boldsymbol{\theta}^*, \boldsymbol{\tau}^*)$ satisfies (2.10) with $\rho = 1$, then we have

$$\boldsymbol{\beta}^* = T_{\lambda,\gamma}(\boldsymbol{\beta}^* + \mathbf{X}^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}^*)/n), \tag{2.31}$$

i.e., (2.29)–(2.30) hold, which implies $\beta_j^* \in \arg\min_{t\in\mathbb{R}} F(\beta_1^*, \ldots, \beta_{j-1}^*, t, \beta_{j+1}^*, \ldots, \beta_d^*)$, i.e., $\boldsymbol{\beta}^*$ is a coordinate wise minimizer of $F(\boldsymbol{\beta})$. Furthermore, by Lemma 3.1 of [24], we get the coordinate-wise minimizer $\boldsymbol{\beta}^*$ is a stationary point of $F(\boldsymbol{\beta})$ in the sense that

$$\liminf_{t\to 0^+} \frac{F(\boldsymbol{\beta}^* + t\mathbf{w}) - F(\boldsymbol{\beta}^*)}{t} \geq 0, \quad \forall\, \mathbf{w} \in \mathbb{R}^d. \tag{2.32}$$

Remark 2.3 By the above theorems we know that numerically, setting $\rho = 1$ in the augmented Lagrangian $\mathcal{L}_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau})$ is a good choice, since the MPLS-ADMM algorithm will converge to a coordinate minimizer and a stationary point of $F(\boldsymbol{\beta})$.

3 Numerical Studies

We conduct numerical studies to evaluate the performance of the MPLS-ADMM estimator. First, we introduce the coordinate descent (CD) algorithm implemented in [3] for solving MPLS and use it as a comparison with our method. Then, we discuss the tuning parameter selection issues. Finally, we investigate the numerical performance of the estimator through simulation studies and a real data analysis. All codes, available from the authors, are written in Matlab and all experiments are performed in MATLAB R2010b on a quad-core laptop with an Intel Core i5 CPU (2.60 GHz) and 8 GB RAM running Windows 8.1 (64 bit).

3.1 Comparison with A Coordinate Descent Algorithm

The CD algorithm and its variants are simple, intuitionistic and fast algorithms that are widely used in nonconvex MCP regularized optimization problems (e.g., [3, 19, 20]). We use the MPLS-CD method as a comparison with our MPLS-ADMM procedure. We adopt the CD iteration scheme described in Section 2.1 in [3] and summarize it in Algorithm 2.

Algorithm 2 MPLS-CD

Input: tuning parameters $\lambda > 0$, $\gamma > 1$; set $k = 0$; initial guess $\boldsymbol{\beta}^0 \in \mathbb{R}^d$; initial residual value $\mathbf{r}^0 = \mathbf{y} - \mathbf{X}\boldsymbol{\beta}^0$; the maximum number of iterations M.

1: **while** $k < M$ **do**
2: **for** $j = 1, 2, \ldots, d$ **do**
3: calculate $z_j^k = n^{-1}\mathbf{X}_j^T\mathbf{r}_{-j}^k = n^{-1}\mathbf{X}_j^T\mathbf{r}^k + \beta_j^k$, where $\mathbf{X}_j$ is the jth column of $\mathbf{X}$, $\mathbf{r}_{-j}^k = \mathbf{y} - \mathbf{X}_{-j}\boldsymbol{\beta}_{-j}^k$, " $-j$" is introduced to refer to the portion that remains after the jth column or element is removed and $\mathbf{r}^k = \mathbf{y} - \mathbf{X}\boldsymbol{\beta}^k$ is the current residual value;
4: update $\beta_j^{k+1} \leftarrow T_{\lambda,\gamma}(z_j^k)$ using (1.6);
5: update $\mathbf{r}^{k+1} \leftarrow \mathbf{r}^k - (\beta_j^{k+1} - \beta_j^k)\mathbf{x}_j$.
6: **end for**
7: check the stopping criterion.
8: **end while**

Output: $\hat{\boldsymbol{\beta}}$, the estimate of $\boldsymbol{\beta}$ in (1.2).

3.2 Computational Complexity

We look at the number of floating point operations line by line in Algorithm 1. Clearly it takes $O(d)$ flops to finish thresholding steps in line 3–5. In line 6, the addition and subtraction of d-vectors require $O(d)$ flops. The most time consuming step of Algorithm 1 is line 2, where we need to solve a $d \times d$ linear equation taking $O(d^3)$ flops. However, the cost can be reduced to $O(nd)$ flops since the Cholekey factorization of $\rho\mathbf{I}_n + n^{-1}\mathbf{X}\mathbf{X}^T$ in (2.7) can be precomputed and stored. Then the inverse of $\rho\mathbf{I}_n + n^{-1}\mathbf{X}\mathbf{X}^T$ takes $O(n^2)$ flops by backward substitution (e.g., [8]). So the overall cost per iteration of Algorithm 1 is $O(nd)$. On the other hand, it can be easily verified that Algorithm 2 also costs $O(nd)$ flops in each iteration.

3.3 Tuning Parameter Selection

The MPLS procedure depends on the selection of tuning parameters λ and γ. In practice, one may tune over a fine two-dimensional grid comprised of different values of λ and γ when implementing the MPLS method. In order to improve computing efficiency, we only consider $\gamma = 1.3$ and $\gamma = 3$, which is borrowed from [32], and concentrate on tuning λ.

To choose a proper tuning parameter, one may employ the Bayesian information criterion (BIC) procedure in different dimensional scenarios (i.e., fixed $d < n$, $n > d = d_n \to \infty$ or $d = d_n > n$), which is a data driven method and widely used in statistics due to its model selection consistency. See [25–27] and references therein.

In our tuning parameter selection implementation, we adopt a high-dimensional BIC (HBIC) with $d > n$ proposed by [27] to select the optimal tuning parameter $\hat{\lambda}$, which is defined as

$$\hat{\lambda} = \arg\min_{\lambda\in\Lambda}\left\{\mathrm{HBIC}(\lambda) = \log(\|\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}(\lambda)\|^2/n) + \frac{C_n\log(d)}{n}|M(\lambda)|\right\}, \tag{3.1}$$

where Λ is a subset of $(0, +\infty)$, $M(\lambda) = \{j :\ \hat{\beta}_j(\lambda) \neq 0\}$ and $|M(\lambda)|$ denotes the cardinality of $M(\lambda)$, and $C_n = \log(\log n)$.

Proposition 3.1 *There exists a* $\lambda_{\max}$ *such that* $\hat{\boldsymbol{\beta}} = 0$ *whenever* $\lambda \geq \lambda_{\max}$ *in the* MPLS *problem* (1.2)*, where*

$$\lambda_{\max} = \|\mathbf{X}^T\mathbf{y}/n\|_\infty. \tag{3.2}$$

Proof For any penalty function $\rho(\cdot)$, first we define

$$\begin{cases} g(t) = \begin{cases} \frac{t}{2} + \frac{\rho(t)}{t}, & t \neq 0, \\ \liminf_{t\to 0^+} g(t), & t = 0, \end{cases} \\ t^* = \arg\min_{t\geq 0} g(t), \\ T^* = \inf\limits_{t>0} g(t) = \lim_{t\to t^*} g(t), \end{cases}$$

where $\frac{0}{0} = 0$. For MCP, it easily follows that

$$\begin{cases} t^* = 0, \\ T^* = \lambda. \end{cases} \tag{3.3}$$

Next we introduce the thresholding operator S^ρ defined in univariate setting by

$$S^\rho(z) = \arg\min_{\beta\in\mathbb{R}}[(z-\beta)^2/2 + \rho(\beta)], \tag{3.4}$$

which can be set-valued. Lemma 3.2 in [14] tells us if $\beta^* \in S^\rho(z)$, then $T^* > |z|$, which further implies $\beta^* = 0$. According to Lemma 3.3 in [14], an element $\boldsymbol{\beta}^* \in \mathbb{R}^d$ is a coordinate-wise minimizer to problem (1.2) if and only if $\beta_j^* \in S^\rho(\beta_j^* + d_j^*),\ j = 1, 2, \ldots, d$, where d_j^* is the jth element of the dual variable $\mathbf{d}^* = \mathbf{X}^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}^*)/n$. One can easily check that $\boldsymbol{\beta}^* = \mathbf{0}$ satisfies the above inclusion provided $T^* > \|\mathbf{X}^T\mathbf{y}/n\|_\infty$. Hence, we complete the proof for Proposition 3.1 by (3.3). □

Algorithm 1 and Algorithm 2 are designed for solving MPLS problem with a fixed tuning parameter λ. In order to obtain accurate solutions, with $\lambda_{\max}$ and HBIC, we apply the data-driven continuation strategy during the tuning parameter selection process. The idea of continuation is well established for iterative algorithms with the purpose of "warm starting" and globalizing the convergence. To be precise, given a decreasing sequence of parameter $\{\lambda_s\}_s$, we run MPLS-ADMM or MPLS-CD to solve the λ_{s+1}-problem initialized with the solution of λ_s-problem, then we pickup the optimal tuning parameter and the corresponding solution via HBIC. We summarize the above ideas in Algorithm 3. G in Algorithm 3 usually takes 100 or 200. Due to the continuation strategy, one can set $M \leq 5$ in Algorithm 1 or Algorithm 2 to get an approximate solution with high accuracy. See [5, 13, 14] for more details.

Algorithm 3 Continuation

Input: $\lambda_0 = \lambda_{\max}$; $\boldsymbol{\beta}(\lambda_0) = \mathbf{0}$, $\boldsymbol{\theta}(\lambda_0) = \mathbf{0}$; $\mu \in (0, 1)$; the number of grid points G.

1: **for** $s = 1, 2, 3, \cdots, G$ **do**
2: set $\lambda_s = \lambda_0\mu^s$ and $(\boldsymbol{\beta}^0, \boldsymbol{\theta}^0) = (\boldsymbol{\beta}(\lambda_{s-1}), \boldsymbol{\theta}(\lambda_{s-1}))$.
3: find $\boldsymbol{\beta}(\lambda_{s-1})$ and $\boldsymbol{\theta}(\lambda_{s-1}))$ by Algorithm 1 or Algorithm 2.
4: compute the HBIC value.
5: **end for**

Output: select $\hat{\lambda}$ by (3.1).

3.4 Simulation

We generate synthetic data from (1.1). The rows of the $n \times d$ matrix $\mathbf{X}$ are sampled as i.i.d.copies from $N(\mathbf{0}, \boldsymbol{\Sigma})$ with $\boldsymbol{\Sigma} = (r^{|i-j|})_{i,j=1,2,\ldots,d}$, where r is the correlation coefficient of $\mathbf{X}$. The noise vector $\boldsymbol{\epsilon}$ is generated independently from $N(\mathbf{0}, \sigma^2\mathbf{I}_n)$, where σ is the noise level. The underlying regression coefficient vector $\boldsymbol{\beta}$ is a random sparse vector chosen as s-sparse with a dynamic range (DR) defined by

$$\mathrm{DR} := \frac{\max\{|\beta_j| : \beta_j \neq 0\}}{\min\{|\beta_j| : \beta_j \neq 0\}} = 10^\alpha, \tag{3.5}$$

where $\alpha = 1$ in our simulations. Following [1], each nonzero entry of $\boldsymbol{\beta}$ is generated as follows:

$$\beta_j = \eta_{1j} 10^{\alpha\eta_{2j}}, \tag{3.6}$$

where $\eta_{1j} = \pm 1$ with probability $\frac{1}{2}$, η_{2j} is uniformly distributed in $[0, 1]$ and $j \in \mathcal{A} = \{j : \beta_j \neq 0, j = 1, 2, \ldots, d\}$. Then $\boldsymbol{\beta}_{\mathcal{A}^c}$ is a zero vector of length $d - s$, where $\mathcal{A}^c$ is the complement of $\mathcal{A}$ in $\{1, 2, \ldots, d\}$.

We choose three levels of correlation $r = 0.2, 0.5$ and 0.7, which correspond to the low, moderate and high correlation level. Two noise levels $\sigma = 0.1$ and 0.3 are considered. We set

$(n, d) = (100, 300)$ and $(100, 600)$ and fix the sparsity level $s = 3$. For each configuration, we randomly generate 100 copies and then compare MPLS-ADMM and MPLS-CD algorithms. For MPLS-ADMM, we take $\rho = 1$ according to Theorem 2.2. The default initial values chosen for MPLS-ADMM and MPLS-CD are $\boldsymbol{\beta}^0 = \boldsymbol{\theta}^0 = \boldsymbol{\tau}^0 = \mathbf{0} \in \mathbb{R}^d$ and $\boldsymbol{\beta}^0 = \mathbf{0} \in \mathbb{R}^d$, respectively. The convergence criterion is $\|\boldsymbol{\beta}^{k+1} - \boldsymbol{\beta}^k\| \leq \delta$ with $\delta = 10^{-4}$.

To further illustrate the efficiency and accuracy of the proposed MPLS-ADMM algorithm, based on $N = 100$ replications, we compare it with the MPLS-CD method in terms of the average CPU time (Time, in seconds), the average ℓ_2 relative error $N^{-1}\sum_{m=1}^{N}(\|\hat{\boldsymbol{\beta}}^{(m)} - \boldsymbol{\beta}\|_2/\|\boldsymbol{\beta}\|_2)$ (ℓ_2 RE) and the average ℓ_∞ absolute error $N^{-1}\sum_{m=1}^{N}\|\hat{\boldsymbol{\beta}}^{(m)} - \boldsymbol{\beta}\|_\infty$ (ℓ_∞ AE). Let $\mathcal{A} = \{j : \beta_j \neq 0, j = 1, 2, \ldots, d\}$ be the true model and $\widehat{\mathcal{A}} = \{j : \hat{\beta}_j \neq 0, j = 1, 2, \ldots, d\}$ be the estimated model. The numerical results also include the average estimated model size $N^{-1}\sum_{m=1}^{N}|\widehat{\mathcal{A}}^{(m)}|$ (MS) and the proportion of correct models $N^{-1}\sum_{m=1}^{N} I\{\widehat{\mathcal{A}}^{(m)} = \mathcal{A}\}$ (CM, in percentage terms). We use the HBIC to select λ. Simulation results are summarized in Table 1 and Table 2, which correspond to $\gamma = 1.3$ and $\gamma = 3$, respectively.

d	r	σ	Method	Time(Sec.)	ℓ_2 RE	ℓ_∞ AE	MS	CM(%)
300	0.2	0.1	MPLS-ADMM	0.4578	0.0033	0.0223	7.10	67
			MPLS-CD	0.7896	0.0061	0.0292	16.57	42
		0.3	MPLS-ADMM	0.3908	0.0094	0.0732	5.31	60
			MPLS-CD	0.6763	0.0197	0.0946	17.61	33
	0.5	0.1	MPLS-ADMM	0.4517	0.0025	0.0230	3.52	61
			MPLS-CD	0.7804	0.0050	0.0296	10.94	35
		0.3	MPLS-ADMM	0.3907	0.0085	0.0712	4.36	59
			MPLS-CD	0.6717	0.0130	0.0854	7.70	38
	0.7	0.1	MPLS-ADMM	0.4497	0.0026	0.0228	3.59	63
			MPLS-CD	0.7693	0.0032	0.0246	4.72	50
		0.3	MPLS-ADMM	0.3869	0.0076	0.0699	3.54	59
			MPLS-CD	0.6619	0.0089	0.0724	4.23	51
600	0.2	0.1	MPLS-ADMM	0.8917	0.0023	0.0214	3.39	66
			MPLS-CD	1.6378	0.0120	0.0478	45.95	1
		0.3	MPLS-ADMM	0.7691	0.0075	0.0649	4.39	63
			MPLS-CD	1.4220	0.0356	0.1398	47.33	1
	0.5	0.1	MPLS-ADMM	0.8865	0.0022	0.0211	3.39	67
			MPLS-CD	1.6292	0.0111	0.0446	41.04	7
		0.3	MPLS-ADMM	0.7757	0.0064	0.0606	3.39	64
			MPLS-CD	1.4206	0.0324	0.1319	40.57	9
	0.7	0.1	MPLS-ADMM	0.8776	0.0021	0.0196	3.38	66
			MPLS-CD	1.6003	0.0069	0.0347	20.65	32
		0.3	MPLS-ADMM	0.7599	0.0069	0.0630	3.28	75
			MPLS-CD	1.3900	0.0199	0.1007	18.42	37

Table 1 Simulation results for variable selection with $\gamma = 1.3$

For $\gamma = 1.3$, we see from Table 1 that the MPLS-ADMM outperforms the MPLS-CD in terms of both CPU time and solution quality. For example, when $(n, d, r, \sigma) = (100, 300, 0.2, 0.1)$,

MPLS-ADMM is about 2 times faster than MPLS-CD, with smaller ℓ_2 RE and ℓ_∞ AE, and better MS and CM. For $\gamma = 3$, we see from Table 2 that MPLS-ADMM still has better timing performance than MPLS-CD. In fact, CPU times of two methods are fairly robust to the choice of γ, and they both grow linearly with dimension d. Both methods can provide more accurate solutions when γ increases from 1.3 to 3, and MPLS-CD is slightly better than MPLS-ADMM in terms of solution quality criteria in most cases for $\gamma = 3$. Overall, MPLS-CD is more sensitive than MPLS-ADMM for large d and small γ, while MPLS-ADMM has better speed performance and numerically more stable than MPLS-CD. It is fair to say that the proposed MPLS-ADMM procedure yields solutions that are comparable with that by the popular MPLS-CD method in most cases, but with less computing time.

d	r	σ	Method	Time(Sec.)	ℓ_2 RE	ℓ_∞ AE	MS	CM(%)
300	0.2	0.1	MPLS-ADMM	0.4731	0.0017	0.0154	3.20	88
			MPLS-CD	0.8104	0.0018	0.0150	3.85	97
		0.3	MPLS-ADMM	0.3943	0.0054	0.0481	3.23	85
			MPLS-CD	0.6679	0.0051	0.0435	3.50	95
	0.5	0.1	MPLS-ADMM	0.4550	0.0018	0.0161	3.27	85
			MPLS-CD	0.7837	0.0019	0.0152	4.78	93
		0.3	MPLS-ADMM	0.3932	0.0056	0.0489	3.29	83
			MPLS-CD	0.6741	0.0046	0.0412	3.06	95
	0.7	0.1	MPLS-ADMM	0.4589	0.0019	0.0167	3.28	83
			MPLS-CD	0.7832	0.0015	0.0140	3.07	95
		0.3	MPLS-ADMM	0.3924	0.0054	0.0498	3.22	88
			MPLS-CD	0.6680	0.0047	0.0431	3.04	97
600	0.2	0.1	MPLS-ADMM	0.8755	0.0020	0.0180	3.25	87
			MPLS-CD	1.5817	0.0030	0.0193	8.80	83
		0.3	MPLS-ADMM	0.7688	0.0057	0.0530	3.22	83
			MPLS-CD	1.3913	0.0087	0.0542	8.99	84
	0.5	0.1	MPLS-ADMM	0.8872	0.0017	0.0148	3.17	86
			MPLS-CD	1.6179	0.0020	0.0143	6.17	89
		0.3	MPLS-ADMM	0.7664	0.0057	0.0516	3.28	82
			MPLS-CD	1.3936	0.0077	0.0511	7.75	86
	0.7	0.1	MPLS-ADMM	0.8751	0.0017	0.0158	3.21	87
			MPLS-CD	1.6030	0.0019	0.0145	4.73	94
		0.3	MPLS-ADMM	0.7524	0.0061	0.0548	3.18	85
			MPLS-CD	1.3696	0.0068	0.0533	5.14	94

Table 2 Simulation results for variable selection with $\gamma = 3$

By [34], the concavity tuning parameter γ is free to vary from 1+ to ∞. When γ is close to 1+, the MCP penalty approximates the L_0 penalty. As γ gets larger, MCP behaves more like LASSO. Figure 1 and Figure 2 show the influence of γ on the performance of MPLS-ADMM and MPLS-CD in terms of the CPU time and solution quality, respectively. Data are generated from the model with $(n = 100, d = 300, r = 0.5, \sigma = 0.1, s = 3, \rho = 1, \gamma = 1.1 : 0.2 : 3)$ and results are averaged over 100 independent runs.

3.5 Application

We analyze the eyedata set which is publicly available in R package **flare** [17] to illustrate the application of the MPLS-ADMM in high-dimensional settings. This data set is a gene expression data from the microarray experiments of mammalian eye tissue samples of [21] and is detailedly described and applied by many papers (e.g., [10, 12]) that want to find the gene probes that are most related to TRIM32 in sparse high-dimensional regression models. The response variable $\mathbf{y}$ is a numeric vector of length 120 giving expression level of gene TRIM32 which causes Bardet-Biedl syndrome (BBS). The design matrix $\mathbf{X}$ is a 120×200 matrix which represents the data of 120 rats with 200 gene probes.

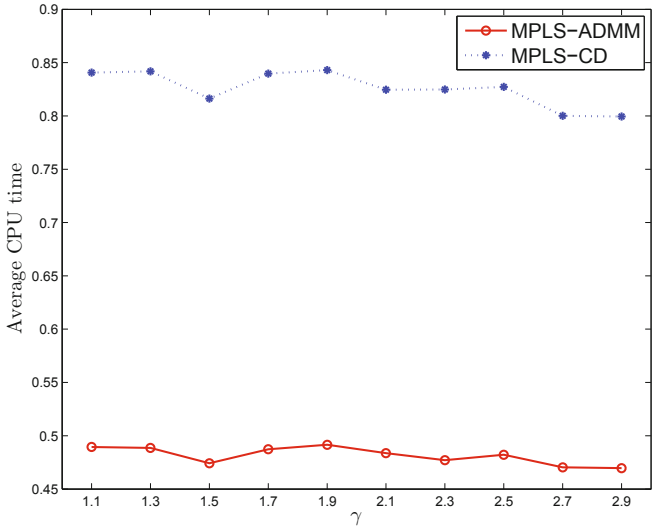
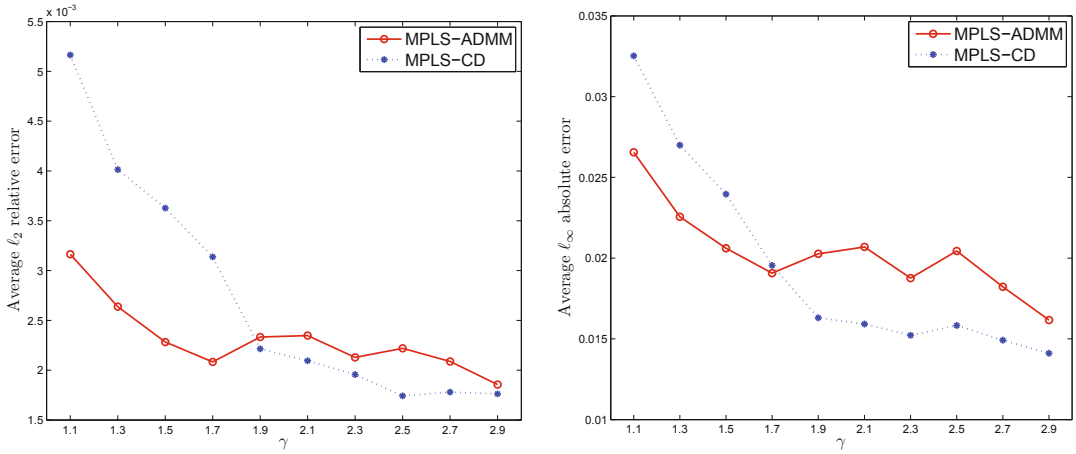


Figure 1 The CPU time of the algorithms along with the varying concavity tuning parameter γ



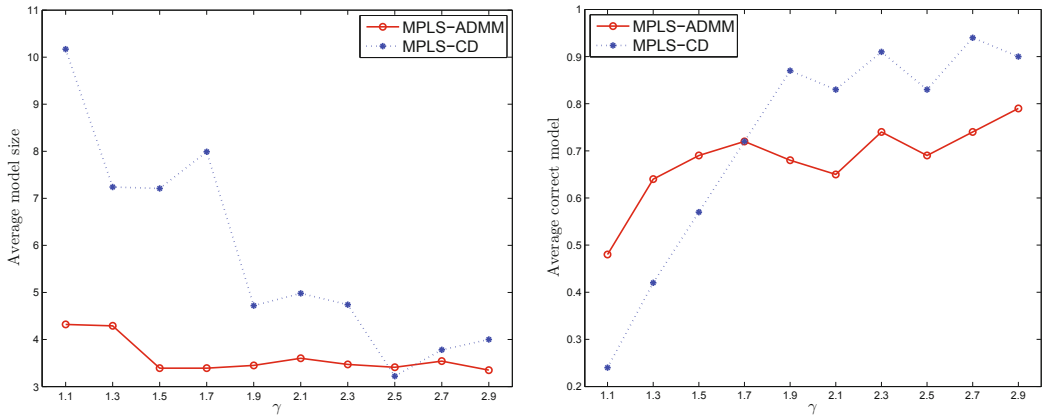


Figure 2 The solution quality of the algorithms along with the varying concavity tuning parameter γ

					$\gamma = 3$		$\gamma = 1.3$	
No.	Term	Probe	flare	glmnet	MPLS-ADMM	MPLS-CD	MPLS-ADMM	MPLS-CD
	Intercept		7.6975	7.7133	7.5258	5.6636	8.0254	5.7815
1	β_{11}	6222	0.0130	0.0140	0	0	0	0
2	β_{16}	7069	0	0	−0.0015	0	0	0
3	β_{28}	10196	0	0	−0.0501	0	0	0
4	β_{42}	12085	0.0151	0.0140	0	0	0	0
5	β_{54}	14949	0.0147	0.0160	0	0	0	0
6	β_{55}	15224	0	0	0.0839	0	0	0
7	β_{59}	15752	0	0	−0.0037	0	0	0
8	β_{62}	15863	−0.0381	−0.0387	0	0	−0.0598	0
9	β_{63}	15940	0	0	0.0470	0	0.0269	0
10	β_{71}	16984	0	0	−0.0593	0	0	0
11	β_{76}	17599	0	0	−0.0635	0	−0.0638	0
12	β_{87}	21092	−0.0933	−0.0932	−0.1293	0	−0.1200	0
13	β_{90}	21550	−0.0189	−0.0183	0	0	0	0
14	β_{102}	22140	0	−0.0027	0	0	0	0
15	β_{127}	23804	−0.0074	−0.0078	0	0	0	0
16	β_{134}	24245	0.0169	0.0161	0	0	0	0
17	β_{136}	24353	−0.0260	−0.0275	0	0	0	0
18	β_{140}	24565	0.0144	0.0184	0	0	0	0
19	β_{146}	24892	0.0072	0.0079	0	0	0	0
20	β_{153}	25141	0.1472	0.1452	0.2376	0.3192	0.2272	0.3022
21	β_{155}	25367	0.0093	0.0092	0	0	0	0
22	β_{174}	27354	0	0	−0.0219	0	0	0
23	β_{180}	28680	0.0684	0.0687	0.1769	0.1903	0.1419	0.1962
24	β_{185}	28967	−0.0829	−0.0845	−0.2848	−0.3033	−0.2173	−0.3118
25	β_{187}	29041	−0.0369	−0.0384	0	0	0	0
26	β_{188}	29045	−0.0068	−0.0073	0	0	0	0
27	β_{200}	30141	−0.0451	−0.0467	−0.0296	0	−0.1222	0
	Time		0.34	0.28	0.19	0.29	0.20	0.27
	PMSE		0.0048	0.0048	0.0042	0.0056	0.0043	0.0055

Table 3 Analysis of the eyedata set. Estimated coefficients of different methods are provided. The zero entries correspond to variables omitted

Since the exact solution for the eyedata set is unknown, we consider two gold standards for comparison purposes: **flare** [17] (the SQRT Lasso with $\lambda = \sqrt{\log(p)/n}$) and **glmnet** [6] (10-fold `cv.glmnet` with the `lambda.1se` rule and `set.seed=0`). We apply the MPLS-ADMM-HBIC and MPLS-CD-HBIC with continuation procedures to the eyedata set. Gene probe information, corresponding nonzero estimates, CPU times (Time) and predictive mean squared errors (PMSE) calculated by $n^{-1}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$ are provided in Table 3. From Table 3, accurate to 4 decimal places, **flare**, **glmnet**, MPLS-ADMM ($\gamma = 3$), MPLS-CD ($\gamma = 3$), MPLS-ADMM ($\gamma = 1.3$) and MPLS-CD ($\gamma = 1.3$), identify 18, 19, 13, 3, 8 and 3 probes, respectively. The sex sets of identified probes have 3 in common. Although the magnitudes of estimates are not equal, they have the same signs, which suggests similar biological conclusions. Notably, the proposed MPLS-ADMM-HBIC with continuation method has smaller PMSE while spending less computing time compared with other competitors.

4 Concluding Remarks

We focus on the MPLS-ADMM-HBIC with continuation method in the context of linear regression models. This method can be applied in a similar way to other models, such as the generalized linear and Cox models, via a quadratic approximation to the loss function based on the two term Taylor series expansion of the log likelihood (or partial likelihood) (cf., e.g., [3, 22]). As pointed in [2], ADMM can be implemented as a distributed algorithm of practical use. Thus, it would be interesting to extend the results for distributed computing, which is beyond the scope of this paper and will be an interesting topic for future research.

Acknowledgements The authors sincerely thank the associate editor and the referees for their valuable comments and suggestions that have led to significant improvement of this article.

References

[1] Becker, S., Bobin, J., Candès, E. J.: NESTA: A fast and accurate first-order method for sparse recovery. *SIAM J. Imaging Sci.*, **4**, 1–39 (2011)

[2] Boyd, S., Parikh, N., Chu, E., et al.: Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, **3**, 1–122 (2011)

[3] Breheny, P., Huang, J.: Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. *Ann. Appl. Stat.*, **5**, 232–253 (2011)

[4] Bühlmann, P., Van De Geer, S.: Statistics for High-dimensional Data: Methods, Theory and Applications, Springer-Verlag, Berlin, 2011

[5] Fan, Q., Jiao, Y., Lu, X.: A primal dual active set algorithm with continuation for compressed sensing. *IEEE Trans. Signal Process.*, **62**, 6276–6285 (2014)

[6] Friedman, J., Hastie, T., Tibshirani, R.: Regularization paths for generalized linear models via coordinate descent. *J. Stat. Softw.*, **33**, 1–22 (2010)

[7] Gao, H. Y., Bruce, A. G.: WaveShrink with firm shrinkage. *Statist. Sinica*, **7**, 855–874 (1997)

[8] Golub, G. H., Van Loan, C. F.: Matrix Computations (4th Edition), John Hopkins University Press, Baltimore, 2013

[9] Hong, M., Luo, Z. Q., Razaviyayn, M.: Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. *SIAM J. Optim.*, **26**, 337–364 (2016)

[10] Huang, J., Breheny, P., Lee, S., et al.: The Mnet method for variable selection. *Statist. Sinica*, **26**, 903–923 (2016)

[11] Huang, J., Jiao, Y., Liu, Y., et al.: A constructive approach to sparse linear regression in high-dimensions, arXiv preprint arXiv:1701.05128v1, 2017

[12] Huang, J., Ma, S., Zhang, C. H.: Adaptive Lasso for sparse high-dimensional regression models. *Statist. Sinica*, **18**, 1603–1618 (2008)

[13] Jiao, Y., Jin, B., Lu, X.: A primal dual active set with continuation algorithm for the ℓ^0-regularized optimization problem. *Appl. Comput. Harmon. Anal.*, **39**, 400–426 (2015)

[14] Jiao, Y., Jin, B., Lu, X., et al.: A primal dual active set algorithm for a class of nonconvex sparsity optimization, arXiv preprint arXiv:1310.1147v3, 2016

[15] Jiao, Y., Jin, Q., Lu, X., et al.: Alternating direction method of multipliers for linear inverse problems. *SIAM J. Numer. Anal.*, **54**, 2114–2137 (2016)

[16] Jin, Z. F., Wan, Z., Jiao, Y., et al.: An alternating direction method with continuation for nonconvex low rank minimization. *J. Sci. Comput.*, **66**, 849–869 (2015)

[17] Li, X., Zhao, T., Yuan, X., et al.: The flare package for high dimensional linear regression and precision matrix estimation in R. *J. Mach. Learn. Res.*, **16**, 553–557 (2015)

[18] Lu, Z., Pong, T. K., Zhang, Y.: An alternating direction method for finding Dantzig selectors. *Comput. Statist. Data Anal.*, **56**, 4037–4046 (2012)

[19] Mazumder, R., Friedman, J. H., Hastie, T.: Sparsenet: Coordinate descent with nonconvex penalties. *J. Amer. Statist. Assoc.*, **106**, 1125–1138 (2011)

[20] Peng, B., Wang, L.: An iterative coordinate descent algorithm for high-dimensional nonconvex penalized quantile regression. *J. Comput. Graph. Statist.*, **24**, 676–694 (2015)

[21] Scheetz, T., Kim, K., Swiderski, R., et al.: Regulation of gene expression in the mammalian eye and its relevance to eye disease. *Proc. Natl. Acad. Sci. USA*, **103**, 14429–14434 (2006)

[22] Simon, N., Friedman, J., Hastie, T., et al.: Regularization paths for Cox's proportional hazards model via coordinate descent. *J. Stat. Softw.*, **39**, 1–13 (2011)

[23] Song, C., Yoon, S., Pavlovic, V.: Fast ADMM algorithm for distributed optimization with adaptive penalty. Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence and the Twenty-Eighth Innovative Applications of Artificial Intelligence Conference, 2016

[24] Tseng, P.: Convergence of a block coordinate descent method for nondifferentiable minimization. *J. Optim. Theory Appl.*, **109**, 475–494 (2001)

[25] Wang, H., Li, B., Leng, C.: Shrinkage tuning parameter selection with a diverging number of parameters. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, **71**, 671–683 (2009)

[26] Wang, H., Li, R., Tsai, C. L.: Tuning parameter selectors for the smoothly clipped absolute deviation method. *Biometrika*, **94**, 553–568 (2007)

[27] Wang, L., Kim, Y., Li, R.: Calibrating nonconvex penalized regression in ultra-high dimension. *Ann. Statist.*, **41**, 2505–2536 (2013)

[28] Wang, Y., Yin, W., Zeng, J.: Global convergence of ADMM in nonconvex nonsmooth optimization. arXiv preprint, arXiv:1511.06324v5, 2017

[29] Xu, Z., Figueiredo, M. A. T., Goldstein, T.: Adaptive ADMM with spectral penalty parameter selection. arXiv preprint, arXiv:1605.07246v5, 2017

[30] Yang, J., Zhang, Y.: Alternating direction algorithms for ℓ_1-problems in compressive sensing. *SIAM J. Sci. Comput.*, **33**, 250–278 (2011)

[31] Yu, L., Lin, N., Wang, L.: A parallel algorithm for large-scale nonconvex penalized quantile regression. *J. Comput. Graph. Statist.*, DOI:10.1080/10618600.2017.1328366, 2017 (just-accepted)

[32] Yu, Y., Feng, Y.: APPLE: Approximate path for penalized likelihood estimators. *Stat. Comput.*, **24**, 803–819 (2014)

[33] Yuan, X.: Alternating direction method for covariance selection models. *J. Sci. Comput.*, **51**, 261–273 (2012)

[34] Zhang, C. H.: Nearly unbiased variable selection under minimax concave penalty. *Ann. Statist.*, **38**, 894–942 (2010)