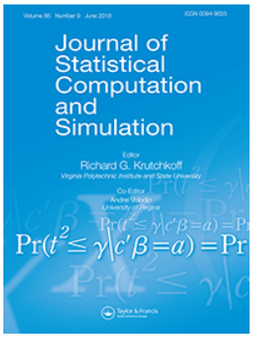

Taylor & Francis
Taylor & Francis Group

Journal of Statistical Computation and Simulation

ISSN: 0094-9655 (Print) 1563-5163 (Online) Journal homepage: http://www.tandfonline.com/loi/gscs20

An ADMM with continuation algorithm for non-convex SICA-penalized regression in high dimensions

Yueyong Shi, Yuanshan Wu, Deyi Xu & Yuling Jiao

To cite this article: Yueyong Shi, Yuanshan Wu, Deyi Xu & Yuling Jiao (2018) An ADMM with continuation algorithm for non-convex SICA-penalized regression in high dimensions, Journal of Statistical Computation and Simulation, 88:9, 1826-1846, DOI: 10.1080/00949655.2018.1448397

To link to this article: https://doi.org/10.1080/00949655.2018.1448397

JOURNAL OF STATISTICAL COMPUTATION AND SIMULATION, 2018
VOL. 88, NO. 9, 1826–1846
https://doi.org/10.1080/00949655.2018.1448397

Taylor & Francis
Taylor & Francis Group

An ADMM with continuation algorithm for non-convex SICA-penalized regression in high dimensions

Yueyong Shi[a,b], Yuanshan Wu[c], Deyi Xu[a] and Yuling Jiao[d]

[a]School of Economics and Management, China University of Geosciences, Wuhan, People's Republic of China; [b]Research Center of Resource and Environmental Economics, China University of Geosciences, Wuhan, People's Republic of China; [c]School of Mathematics and Statistics, Wuhan University, Wuhan, People's Republic of China; [d]School of Statistics and Mathematics, Zhongnan University of Economics and Law, Wuhan, People's Republic of China

ABSTRACT

The smooth integration of counting and absolute deviation (SICA) penalty has been demonstrated theoretically and practically to be effective in non-convex penalization for variable selection. However, solving the non-convex optimization problem associated with the SICA penalty when the number of variables exceeds the sample size remains to be enriched due to the singularity at the origin and the non-convexity of the SICA penalty function. In this paper, we develop an efficient and accurate alternating direction method of multipliers with continuation algorithm for solving the SICA-penalized least squares problem in high dimensions. We establish the convergence property of the proposed algorithm under some mild regularity conditions and study the corresponding Karush–Kuhn–Tucker optimality condition. A high-dimensional Bayesian information criterion is developed to select the optimal tuning parameters. We conduct extensive simulations studies to evaluate the efficiency and accuracy of the proposed algorithm, while its practical usefulness is further illustrated with a high-dimensional microarray study.

ARTICLE HISTORY
Received 17 October 2017
Accepted 1 March 2018

KEYWORDS
ADMM; continuation; coordinate descent; high-dimensional BIC; SICA

2010 MATHEMATICS SUBJECT CLASSIFICATIONS
62F12; 62J05; 62J07

1. Introduction

Learning sparse representations for high-dimensional data is a hot and important issue [1,2]. Consider the linear regression model

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{1}$$

where $\mathbf{y} = (y_1, y_2, \ldots, y_n)^{\mathrm{T}} \in \mathbb{R}^n$ is a response vector, $\mathbf{X} = (x_{ij}) \in \mathbb{R}^{n\times p}$ is a design matrix, $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_p)^{\mathrm{T}} \in \mathbb{R}^p$ is a vector of underlying regression coefficients, and $\boldsymbol{\varepsilon} \in \mathbb{R}^n$ is a vector of random errors. We assume without loss of generality that $\mathbf{y}$ is centred and the columns of $\mathbf{X}$ are centred and $\sqrt{n}$-normalized, that is, $\sum_{i=1}^n y_i = 0$, $\sum_{i=1}^n x_{ij} = 0$ and $n^{-1}\sum_{i=1}^n x_{ij}^2 = 1$. We assume that $\boldsymbol{\beta}$ is sparse in the sense that only a relatively small portion of the components of $\boldsymbol{\beta}$ are non-zero. Let $\mathcal{A} = \{j : \beta_j \neq 0\}$ be the true model and

CONTACT Yuling Jiao yulingjiaomath@whu.edu.cn School of Statistics and Mathematics, Zhongnan University of Economics and Law, Wuhan, Hubei 430073, People's Republic of China

suppose that $T = |\mathcal{A}|$ is the size of the true model, where $|\mathcal{A}|$ denotes the cardinality of $\mathcal{A}$. T is also called the sparsity level of $\boldsymbol{\beta}$. We focus on the high-dimensional case where $p > n$ and our goal is to reconstruct the unknown vector $\boldsymbol{\beta}$.

Without any constraints on $\boldsymbol{\beta}$ there exist infinitely many least squares solutions for (1) since it is a highly undetermined linear system when $p > n$. Some of the solutions usually over-fit the data. Thus, the traditional least squares method is not applicable, and regularized or penalized methods are needed. The penalized method, which optimizes a objective function defined as the sum of certain empirical loss function and a regularizer (penalty term) and can simultaneously accomplish parameter estimation and variable selection by shrinking some regression coefficients to zero, has been widely used in the literature (cf., e.g. [3–6]). Under the sparsity assumption, one can estimate $\boldsymbol{\beta}$ by the L_0 regularization [7], which gives a nice interpretation of best subset selection and reads

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \left\{ \frac{1}{2n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda\|\boldsymbol{\beta}\|_0 \right\}, \tag{2}$$

where $\|\cdot\|$ denotes the standard Euclidean norm, $\lambda > 0$ is a regularization or tuning parameter controlling the sparsity level of the regularized solution, and $\|\boldsymbol{\beta}\|_0$ denotes the number of non-zero components in $\boldsymbol{\beta}$, that is, $\|\boldsymbol{\beta}\|_0 = \sum_{j=1}^{p} I(\beta_j \neq 0)$. However, the computation of (2) is generally NP hard due to the discontinuity of the function $\|\boldsymbol{\beta}\|_0$, hence it is challenging to design a stable and fast algorithm to solve it. In this paper, we consider the following so-called SICA-penalized least squares (PLS) problem:

$$\widehat{\boldsymbol{\beta}} \triangleq \widehat{\boldsymbol{\beta}}(\lambda, a) := \arg\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \left\{ Q(\boldsymbol{\beta}) = \frac{1}{2n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \sum_{j=1}^{p} p_{\lambda,a}(\beta_j) \right\}, \tag{3}$$

where

$$p_{\lambda,a}(\beta_j) = \lambda \frac{(a+1)|\beta_j|}{|\beta_j| + a} \tag{4}$$

is the SICA penalty function proposed by Lv and Fan [7], $\lambda > 0$ is the sparsity tuning parameter that controls the tradeoff between the loss function and the regularizer, offering a very useful way of obtaining sparse solutions, and $a > 0$ is the shape or concavity tuning parameter making SICA a bridge between L_0 ($a \to 0+$) and L_1 ($a \to \infty$), where L_0 and L_1 admit $p_\lambda(\beta_j) = \lambda I(\beta_j \neq 0)$ and $p_\lambda(\beta_j) = \lambda|\beta_j|$, respectively. $\widehat{\boldsymbol{\beta}}$, which is dependent on λ and a, is denoted as an SICA–PLS (SPLS) estimator. Let $\widehat{\mathcal{A}} = \{j : \widehat{\beta}_j \neq 0\}$ denote the estimated model. Figure 1 depicts SICA penalties for a few a's while fixing $\lambda = 1$.

The SICA regularization, which bridges L_0 and L_1 and thus is expected to retain the advantages of both L_0 and L_1 by delivering better variable selection than L_1 (L_0 is interpreted as best subset selection) while yielding a more stable model than L_0 (L_1 is continuous), has been successfully used in several literature. Under linear models, Lv and Fan [8] propose the SICA method for model selection and sparse recovery. They establish conditions under which the SPLS estimator enjoys a so-called weak oracle property, where the dimensionality can grow exponentially with sample size, and apply the local linear approximation (LLA) [9] and the sequentially and iteratively reweighted squares (SIRS) algorithms for model selection and sparse recovery, respectively. Lin and Lv [10] use the

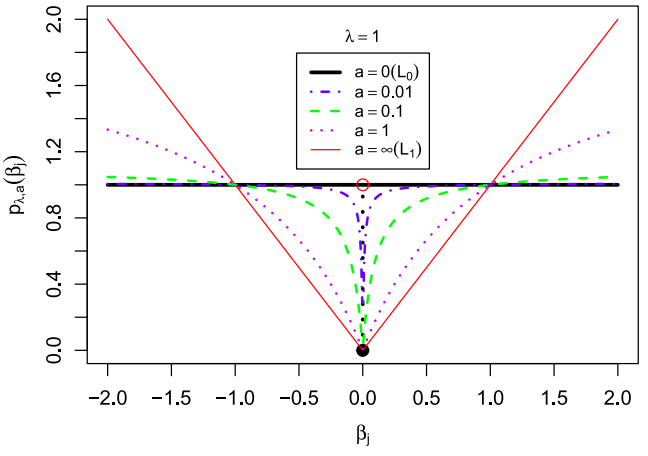

Figure 1. Plot of SICA penalty functions for a few values of a with $\lambda = 1$: $a = 0$ (L_0, thick solid), $a = 0.01$ (dotdash), $a = 0.1$ (dashed), $a = 1$ (dotted) and $a = \infty$ (L_1, thin solid).

SICA-penalized likelihood approach combining the pseudoscore method for simultaneous variable selection and estimation in the additive hazards model in a high-dimensional setting. They establish the weak oracle property and oracle property under mild, interpretable conditions and present a coordinate descent (CD) algorithm for efficient implementation and rigorously investigate its convergence properties. Shi et al. [11] use the SICA-penalized likelihood method for variable selection and parameter estimation in Cox regression in situations where the number of parameters diverges with the sample size. Under appropriate sparsity conditions, they show the resulting estimator of the regression coefficients possesses the oracle property. They carry out corresponding numerical analysis via the smoothing quasi-Newton algorithm with backtracking linear search strategy. Shi et al. [12] propose a modified Bayesian information criterion tuning parameter selector for SICA-penalized Cox regression models with a diverging number of covariates and prove its model selection consistency under some regularity conditions.

Alternating direction method of multipliers (ADMM) [13], which is a very popular algorithm that combines the benefits of dual decomposition and method of multipliers, is capable of producing high quality solutions at reasonable computational cost for non-smooth and non-convex optimization problems in sparse high-dimensional data settings. 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 [14–17]. In this paper, we develop an ADMM with continuation algorithm for solving the non-convex SPLS problem (3). Our algorithm combines the strengths of four parts: ADMM, the SICA thresholding operator, a continuation strategy and a high-dimensional BIC (HBIC). The main contributions of this paper are twofold. First, we extent the ADMM for solving the non-smooth and non-convex regularized optimization problem with the SICA penalty and establish the theoretical convergence results. Second, we couple ADMM with a continuation strategy on the regularization parameter, i.e. given a decreasing sequence of parameter $\{\lambda_g\}$, we apply ADMM to solve the λ_{g+1}-problem with the initial guess from the λ_g-problem. The idea of continuation is well established for iterative algorithms with the purpose of 'warm starting' and globalizing the convergence, see

similar approach as in [18,19]. We adopt an HBIC to select a suitable tuning parameter during the continuation process.

The remainder of this paper is organized as follows. In Section 2, we develop the SPLS–ADMM algorithm for high-dimensional linear regression and give its algorithmic implementation procedure. Then we establish the convergent property of the proposed algorithm and investigate its Karush–Kuhn–Tucker (KKT) optimality condition. In Section 3, simulation studies and an application are provided to illustrate the finite sample performance of the proposed algorithm. The computational complexity analysis, the tuning parameter selection criterion and the continuation strategy are also presented in Section 3. Finally, we conclude the paper with Section 4.

2. SPLS–ADMM algorithm

In this section, we construct the ADMM to solve the SPLS problem (3). We first show that the resulting subproblem has a closed-form solution. Then we propose the SPLS-ADMM algorithm. Finally, we show the convergence property and the KKT optimality condition of the proposed algorithm.

2.1. Methodology

By introducing an auxiliary variable $\boldsymbol{\theta} \in \mathbb{R}^p$, the SPLS problem (3) can be equivalently transformed into

$$(\widehat{\boldsymbol{\beta}}, \widehat{\boldsymbol{\theta}}) := \arg\min_{\boldsymbol{\beta},\boldsymbol{\theta}} \left\{ Q(\boldsymbol{\beta},\boldsymbol{\theta}) = \frac{1}{2n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \sum_{j=1}^{p} p_{\lambda,a}(\theta_j) \right\} \quad \text{s.t. } \boldsymbol{\beta} = \boldsymbol{\theta}. \tag{5}$$

The corresponding augmented Lagrangian function of problem (5) is

$$L_\rho(\boldsymbol{\beta},\boldsymbol{\theta},\boldsymbol{\tau}) = \frac{1}{2n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \sum_{j=1}^{p} p_{\lambda,a}(\theta_j) + \frac{\rho}{2}\|\boldsymbol{\beta} - \boldsymbol{\theta}\|^2 + \langle \boldsymbol{\beta} - \boldsymbol{\theta}, \boldsymbol{\tau}\rangle, \tag{6}$$

where $\boldsymbol{\tau} \in \mathbb{R}^p$ is the Lagrangian multiplier, and $\rho > 0$ is the penalty parameter for the violation of the linear constraint. Obviously, $L_\rho(\boldsymbol{\beta},\boldsymbol{\theta},\boldsymbol{\tau}) = Q(\boldsymbol{\beta},\boldsymbol{\theta}) = Q(\boldsymbol{\beta})$ when $\boldsymbol{\beta} = \boldsymbol{\theta}$. Each iteration of the ADMM involves alternating minimization of L_ρ with respect to $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$, followed by an update of $\boldsymbol{\tau}$. In particular, for a given $(\boldsymbol{\beta}^k, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k)$, the iteration scheme of the ADMM for problem (6) proceeds as follows:

$$\boldsymbol{\beta}^{k+1} \in \arg\min_{\boldsymbol{\beta}\in\mathbb{R}^p} L_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k), \tag{7}$$

$$\boldsymbol{\theta}^{k+1} \in \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^p} L_\rho(\boldsymbol{\beta}^{k+1}, \boldsymbol{\theta}, \boldsymbol{\tau}^k), \tag{8}$$

$$\boldsymbol{\tau}^{k+1} = \boldsymbol{\tau}^k + \rho(\boldsymbol{\beta}^{k+1} - \boldsymbol{\theta}^{k+1}). \tag{9}$$

After some algebra, the $\boldsymbol{\beta}$-subproblem (7) can be reformulated as

$$\boldsymbol{\beta}^{k+1} = (n^{-1}\mathbf{X}^T\mathbf{X} + \rho\mathbf{I}_p)^{-1}(n^{-1}\mathbf{X}^T\mathbf{y} + \rho\boldsymbol{\theta}^k - \boldsymbol{\tau}^k), \tag{10}$$

where $\mathbf{I}_p$ is the $p \times p$ identity matrix. In practice, it may be expensive to solve the linear system (10) directly, especially when $p > n$. It is noteworthy that

$$(n^{-1}\mathbf{X}^T\mathbf{X} + \rho\mathbf{I}_p)^{-1} = (\mathbf{I}_p - \mathbf{X}^T(\rho\mathbf{I}_n + n^{-1}\mathbf{X}\mathbf{X}^T)^{-1}n^{-1}\mathbf{X})\rho^{-1}, \tag{11}$$

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 $p > n$. As a result, it is easy to carry out the Cholesky factorization [20] of $\rho\mathbf{I}_n + n^{-1}\mathbf{X}\mathbf{X}^T$ for solving Equation (10).

Equivalently, the $\boldsymbol{\theta}$-subproblem (8) can be written as

$$\boldsymbol{\theta}^{k+1} = \underset{\boldsymbol{\theta}\in\mathbb{R}^p}{\arg\min} \left\{ \frac{1}{2}\|\boldsymbol{\theta} - \mathbf{z}^k\|^2 + \sum_{j=1}^{p} p_{\lambda/\rho,a}(\theta_j) \right\}, \tag{12}$$

where $\mathbf{z}^k = \boldsymbol{\beta}^{k+1} + \boldsymbol{\tau}^k/\rho$. Consider the coordinate-wise minimization of (12), namely the one-dimensional SPLS problem

$$\widehat{\theta} = \underset{\theta\in\mathbb{R}}{\arg\min}\{\tfrac{1}{2}(\theta - z)^2 + p_{\mu,a}(\theta)\}, \tag{13}$$

where $\mu = \lambda/\rho$. Lin and Lv [10] provide an analytic form of the univariate SPLS estimator (i.e. the element-wise SICA thresholding operator, TSICA) as the solution of (13). We summarize TSICA in Algorithm 1 and denote TSICA as $\mathbb{T}_{\mu,a}(\cdot)$. Accordingly, the solution of $\boldsymbol{\theta}$-subproblem (8) can be described as

$$\boldsymbol{\theta}^{k+1} = \mathbb{T}_{\mu,a}(\mathbf{z}^k) = \mathbb{T}_{\lambda/\rho,a}\left(\boldsymbol{\beta}^{k+1} + \frac{\boldsymbol{\tau}^k}{\rho}\right). \tag{14}$$

Based on TSICA in Algorithm 1, we propose SPLS–ADMM in Algorithm 2 for solving the SPLS problem with fixed tuning parameters λ and a.

2.2. Convergence analysis

Due to the non-convexity of the problem, it is not easy to prove that the SPLS–ADMM converges to a global minimizer. Motivated by Wen et al. [21], the main result in this subsection establishes that, under some regularity conditions, any limit point of the iteration sequence generated by the SPLS–ADMM is a KKT point of $L_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau})$, which is a triple $(\boldsymbol{\beta}^*, \boldsymbol{\theta}^*, \boldsymbol{\tau}^*)$ satisfying the following system:

$$\begin{cases} \boldsymbol{\theta}^* = \mathbb{T}_{\lambda/\rho,a}\left(\boldsymbol{\beta}^* + \dfrac{\boldsymbol{\tau}^*}{\rho}\right), \\ \dfrac{1}{n}\mathbf{X}^T(\mathbf{X}\boldsymbol{\beta}^* - \mathbf{y}) + \boldsymbol{\tau}^* = 0, \\ \boldsymbol{\beta}^* = \boldsymbol{\theta}^*. \end{cases} \tag{15}$$

Algorithm 1 TSICA

Input: Tuning parameters $\mu > 0$ and $a > 0$; constant $z \in \mathbb{R}$.

Compute

$$c_2 = 2a - |z|,\ c_1 = a^2 - 2a|z|,\ c_0 = \mu a(a+1) - a^2|z|,$$
$$Q = (c_2^2 - 3c_1)/9,\ R = (2c_2^3 - 9c_1c_2 + 27c_0)/54,$$
$$\alpha = \arccos(R/\sqrt{Q^3}),\ t_1 = -2\sqrt{Q}\cos\left(\frac{\alpha - 2\pi}{3}\right) - \frac{c_2}{3},$$
$$t_2 = -2\sqrt{Q}\cos\left(\frac{\alpha + 2\pi}{3}\right) - \frac{c_2}{3}.$$

if $Q^3 \le R^2$ **then**
  $\widehat{\theta} = 0$;
else
  if $t_1 > 0$ && $t_2/2 + \mu(a+1)/(a+t_2) < z$ **then**
    $\widehat{\theta} = \mathrm{sgn}(z)t_2$;
  else if $t_1 < 0$ && $t_2 > 0$ **then**
    $\widehat{\theta} = \mathrm{sgn}(z)t_2$;
  else
    $\widehat{\theta} = 0$.
  end if
end if

Output: $\widehat{\theta}$, the estimate of θ in Equation (13).

Algorithm 2 SPLS-ADMM

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

while $k < M$ **do**
  Update $\boldsymbol{\beta}^{k+1}$ using Equations (10) and (11);
  for $j = 1, 2, \ldots, p$, **do**
    Update $\theta_j^{k+1} = \mathbb{T}_{\lambda/\rho,a}(z_j^k)$ using Algorithm 1, where z_j^k is the jth element of $\mathbf{z}^k$ in Equation (12);
  end for
  Update $\boldsymbol{\tau}^{k+1}$ using Equation (9);
  Check the stopping criterion.
end while

Output: $(\widehat{\boldsymbol{\beta}}, \widehat{\boldsymbol{\theta}})$, the estimate of $(\boldsymbol{\beta}, \boldsymbol{\theta})$ in Equation (5).

Theorem 2.1 (Convergence property of SPLS–ADMM): *Let* $\{(\boldsymbol{\beta}^k, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k)\}$ *be a sequence generated by SPLS–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 Equation (15).*

Proof: Since

$$\lim_{k\to\infty} \|\boldsymbol{\tau}^{k+1} - \boldsymbol{\tau}^k\| = 0 \tag{16}$$

and $\rho > 0$, we get from Equation (9) that

$$\lim_{k\to\infty} \|\boldsymbol{\beta}^{k+1} - \boldsymbol{\theta}^{k+1}\| = \lim_{k\to\infty} \frac{1}{\rho}\|\boldsymbol{\tau}^{k+1} - \boldsymbol{\tau}^k\| = 0. \tag{17}$$

Then $\{\boldsymbol{\beta}^k\}$ is bounded by Equation (17) and the boundedness assumption on $\{\boldsymbol{\theta}^k\}$. It follows from Equation (10) 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 $L_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau})$ is continuous, we can obtain that $L_\rho(\boldsymbol{\beta}^k, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k)$ is bounded. It is obvious that $L_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau})$ is strongly convex with respect to the variable $\boldsymbol{\beta}$, so it holds that for any $\boldsymbol{\beta}$ and $\triangle\boldsymbol{\beta}$,

$$L_\rho(\boldsymbol{\beta} + \triangle\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau}) - L_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau}) \geq \langle \nabla_{\boldsymbol{\beta}} L_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau}), \triangle\boldsymbol{\beta}\rangle + c\|\triangle\boldsymbol{\beta}\|^2, \tag{18}$$

where $c > 0$ is a constant. Since $\boldsymbol{\beta}^{k+1}$ minimizes Equation (7), we further have

$$\langle \nabla_{\boldsymbol{\beta}} L_\rho(\boldsymbol{\beta}^{k+1}, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k), (\boldsymbol{\beta}^k - \boldsymbol{\beta}^{k+1})\rangle \geq 0. \tag{19}$$

By letting $\triangle\boldsymbol{\beta} = \boldsymbol{\beta}^k - \boldsymbol{\beta}^{k+1}$ in Equation (18) and combining with (19), we can obtain that

$$L_\rho(\boldsymbol{\beta}^k, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k) - L_\rho(\boldsymbol{\beta}^{k+1}, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k) \geq c\|\boldsymbol{\beta}^{k+1} - \boldsymbol{\beta}^k\|^2. \tag{20}$$

Moreover, since $\boldsymbol{\theta}^{k+1}$ minimizes Equation (8), we have

$$L_\rho(\boldsymbol{\beta}^{k+1}, \boldsymbol{\theta}^{k+1}, \boldsymbol{\tau}^k) \leq L_\rho(\boldsymbol{\beta}^{k+1}, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k). \tag{21}$$

Thus together with Equations (20) and (9), we get that

$$L_\rho(\boldsymbol{\beta}^k, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k) - L_\rho(\boldsymbol{\beta}^{k+1}, \boldsymbol{\theta}^{k+1}, \boldsymbol{\tau}^{k+1}) + \frac{1}{\rho}\|\boldsymbol{\tau}^{k+1} - \boldsymbol{\tau}^k\|^2 \geq c\|\boldsymbol{\beta}^{k+1} - \boldsymbol{\beta}^k\|^2. \tag{22}$$

Denote $V_k^L = L_\rho(\boldsymbol{\beta}^k, \boldsymbol{\theta}^k, \boldsymbol{\tau}^k)$, $V_k^{\boldsymbol{\tau}} = (1/\rho)\|\boldsymbol{\tau}^{k+1} - \boldsymbol{\tau}^k\|^2$ and $V_k^{\boldsymbol{\beta}} = c\|\boldsymbol{\beta}^{k+1} - \boldsymbol{\beta}^k\|^2$. Then we rewrite Equation (22) as

$$V_k^L - V_{k+1}^L + V_k^{\boldsymbol{\tau}} \geq V_k^{\boldsymbol{\beta}} \geq 0. \tag{23}$$

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

$$\lim_{k_j\to\infty} V_{k_j}^L = \underline{\lim}_{k\to\infty} V_k^L.$$

It follows from Equation (23), the non-negativity of $V_k^{\boldsymbol{\beta}}$ and $V_k^{\boldsymbol{\tau}}$, and the assumption $\lim_{k\to\infty} V_k^{\boldsymbol{\tau}} = 0$, that

$$\begin{aligned} 0 &\leq \underline{\lim}_{k_j\to\infty} V_{k_j}^{\boldsymbol{\beta}} \leq \underline{\lim}_{k_j\to\infty}(V_{k_j}^L - V_{k_j+1}^L + V_{k_j}^{\boldsymbol{\tau}}) \leq \underline{\lim}_{k_j\to\infty}(V_{k_j}^L + V_{k_j}^{\boldsymbol{\tau}}) \\ &\quad - \underline{\lim}_{k_j\to\infty}(V_{k_j+1}^L) \\ &= \lim_{k_j\to\infty}(V_{k_j}^L + V_{k_j}^{\boldsymbol{\tau}}) - \underline{\lim}_{k_j\to\infty} V_{k_j+1}^L \leq 0, \end{aligned} \tag{24}$$

which implies

$$\underline{\lim}_{k_j\to\infty} V_{k_j}^{\boldsymbol{\beta}} = 0, \tag{25}$$

that is,

$$\underline{\lim}_{k_j\to\infty} \|\boldsymbol{\beta}^{k_j+1} - \boldsymbol{\beta}^{k_j}\| = 0. \tag{26}$$

Together with Equation (17), we can also get that

$$\underline{\lim}_{k_j\to\infty} \|\boldsymbol{\theta}^{k_j+1} - \boldsymbol{\theta}^{k_j}\| = 0. \tag{27}$$

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

$$\lim_{k_j\to\infty} \boldsymbol{\beta}^{k_j} = \boldsymbol{\beta}^*, \lim_{k_j\to\infty} \boldsymbol{\theta}^{k_j} = \boldsymbol{\theta}^* \tag{28}$$

and Equation (17), we can obtain that

$$\boldsymbol{\beta}^* = \boldsymbol{\theta}^*. \tag{29}$$

After some algebra, Equation (10) can be transformed into the following form:

$$\frac{\mathbf{X}^T(\mathbf{X}\boldsymbol{\beta}^{k+1} - \mathbf{y})}{n} = -\rho\boldsymbol{\beta}^{k+1} + \rho\boldsymbol{\theta}^k - \boldsymbol{\tau}^k. \tag{30}$$

Taking the limit of both sides of Equation (30) on k_j and together with Equations (16) and (28), it follows that

$$\frac{1}{n}\mathbf{X}^T(\mathbf{X}\boldsymbol{\beta}^* - \mathbf{y}) + \boldsymbol{\tau}^* = 0. \tag{31}$$

Using the condition (26) and taking the limit of the both sides of Equation (14) on k_j, we get

$$\boldsymbol{\theta}^* = \mathbb{T}_{\lambda/\rho,a}\left(\boldsymbol{\beta}^* + \frac{\boldsymbol{\tau}^*}{\rho}\right). \tag{32}$$

Combining Equation (32) with Equations (29) and (31), we obtain that $(\boldsymbol{\beta}^*, \boldsymbol{\theta}^*, \boldsymbol{\tau}^*)$ is a KKT point of $Q(\boldsymbol{\beta})$ satisfying Equation (15). ■

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 SICA-penalized least square (3), then there exist $\boldsymbol{\theta}^*$ and $\boldsymbol{\tau}^*$ such that Equation (15) holds with $\rho = 1$. Conversely, if $(\boldsymbol{\beta}^*, \boldsymbol{\theta}^*, \boldsymbol{\tau}^*)$ satisfies Equation (15) with $\rho = 1$, then $\boldsymbol{\beta}^*$ is a coordinate-wise minimizer and a stationary point of (3).*

Proof: Suppose $\boldsymbol{\beta}^* = (\beta_1^*, \ldots, \beta_p^*)^{\mathrm{T}}$ is a minimizer of $Q(\boldsymbol{\beta})$ in Equation (3). Then

$$\begin{aligned}
& \beta_j^* \in \arg\min_{t\in\mathbb{R}} Q(\beta_1^*, \ldots, \beta_{j-1}^*, t, \beta_{j+1}^*, \ldots, \beta_p^*) \\
\Rightarrow \quad & \beta_j^* \in \arg\min_{t\in\mathbb{R}} \frac{1}{2n}\|\mathbf{X}\boldsymbol{\beta}^* - \mathbf{y} + (t-\beta_j^*)\mathbf{X}_j)\|_2^2 + p_{\lambda,a}(t) \\
\Rightarrow \quad & \beta_j^* \in \arg\min_{t\in\mathbb{R}} \frac{1}{2}(t-\beta_j^*)^2 + \frac{1}{n}(t-\beta_j^*)\mathbf{X}_j^T(\mathbf{X}\boldsymbol{\beta}^* - \mathbf{y}) + p_{\lambda,a}(t) \\
\Rightarrow \quad & \beta_j^* \in \arg\min_{t\in\mathbb{R}} \frac{1}{2}(t-\beta_j^* - \mathbf{X}_j^T(\mathbf{y}-\mathbf{X}\boldsymbol{\beta}^*)/n)^2 + p_{\lambda,a}(t), \qquad (33)
\end{aligned}$$

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

$$\boldsymbol{\tau}^* = \mathbf{X}^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}^*)/n. \qquad (34)$$

By the definition of the thresholding operator in Equation (13), we have

$$\beta_j^* = \mathbb{T}_{\lambda,a}(\beta_j^* + \tau_j^*), \quad j = 1, 2, \ldots, p. \qquad (35)$$

Let $\boldsymbol{\theta}^* = \boldsymbol{\beta}^*$. From Equations (33) –(35), it follows that $(\boldsymbol{\beta}^*, \boldsymbol{\theta}^*, \boldsymbol{\tau}^*)$ satisfies Equation (15) with $\rho = 1$. Conversely, if $(\boldsymbol{\beta}^*, \boldsymbol{\theta}^*, \boldsymbol{\tau}^*)$ satisfies Equation (15) with $\rho = 1$, then we have

$$\boldsymbol{\beta}^* = \mathbb{T}_{\lambda,a}(\boldsymbol{\beta}^* + \mathbf{X}^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}^*)/n), \qquad (36)$$

that is, Equations (34) –(35) hold, which implies $\beta_j^* \in \arg\min_{t\in\mathbb{R}} Q(\beta_1^*, \ldots, \beta_{j-1}^*, t, \beta_{j+1}^*, \ldots, \beta_p^*)$, that is, $\boldsymbol{\beta}^*$ is a coordinate wise minimizer of $Q(\boldsymbol{\beta})$. Furthermore, by Lemma 3.1 of [22], we get the coordinate-wise minimizer $\boldsymbol{\beta}^*$ is a stationary point of $Q(\boldsymbol{\beta})$ in the sense that

$$\liminf_{t\to 0^+} \frac{Q(\boldsymbol{\beta}^* + t\mathbf{d}) - Q(\boldsymbol{\beta}^*)}{t} \geq 0, \quad \forall \mathbf{d} \in \mathbb{R}^p. \qquad (37)$$

■

Remark 2.1: By the above theorems we know that numerically, setting $\rho = 1$ in the augmented Lagrangian $L_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\tau})$ is a good choice, since the SPLS–ADMM algorithm will converge to a coordinate minimizer and stationary point of $Q(\boldsymbol{\beta})$.

3. Numerical studies

In this section, we conduct numerical studies to assess the performance of SPLS–ADMM. First, we introduce a CD algorithm for solving SPLS and use it as a comparison with our method. Then, we discuss the tuning parameter selection issues. Finally, we investigate the numerical performance of SPLS–ADMM on both simulated and real data sets. All codes 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).

Algorithm 3 SPLS–CD

Input: Tuning parameters $\lambda > 0$ and $a > 0$; set $k = 0$; initial guess $\boldsymbol{\beta}^0 \in \mathbb{R}^p$; 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** $j = 1, 2, \ldots, p$, **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 $\beta_j^{k+1} \leftarrow \mathbb{T}_{\lambda,a}(z_j^k)$ using Algorithm 1;
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: $\widehat{\boldsymbol{\beta}}$, the estimate of $\boldsymbol{\beta}$ in Equation (3).

3.1. Comparison with a CD algorithm

The CD algorithm and its variants are simple, intuitionistic and fast algorithms that are widely used in non-convex regularized optimization problems (cf., e.g. [23–25]). Breheny and Huang [23] propose a CD algorithm for fitting non-convex SCAD and MCP models. We use their CD algorithm and combine it with TSICA in Algorithm 1 for solving the SPLS problem (3), which is named as SPLS–CD. We summarize SPLS–CD in Algorithm 1 and use it as a comparison with SPLS–ADMM.

3.2. Computational complexity

We look at the number of floating point operations line by line in Algorithm 1 (SPLS–ADMM). Clearly it takes $O(p)$ flops to finish thresholding steps in lines 3–5. In line 6, the addition and subtraction of p-vectors require $O(p)$ flops. The most time consuming step of Algorithm 1 is line 2, where we need to solve a $p \times p$ linear equation taking $O(p^3)$ flops. However, the cost can be reduced to $O(np)$ flops since the Cholekey factorization of $\rho\mathbf{I}_n + n^{-1}\mathbf{X}\mathbf{X}^T$ in Equation (11) 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 (cf., e.g. [26]). Noticing that $p > n$, it follows that the overall cost per iteration of Algorithm 1 is $O(np)$. On the other hand, it can be easily verified that Algorithm 1 (SPLS–CD) also costs $O(np)$ flops in each iteration. Thus, these two algorithms have the same time complexity.

3.3. Tuning parameter selection

Tuning parameter selection is an important issue in PLS procedures. To choose a proper tuning parameter, one may employ the Bayesian information criterion (BIC) procedure in different dimensional scenarios (i.e. fixed $p < n$, $n > p = p_n \to \infty$ or $p = p_n \gg n$), which

is a data-driven method and widely used in statistics due to its model selection consistency. See [27–29] and references therein. To obtain an SPLS solution path, we implement SPLS–ADMM or SPLS–CD for a range of values of (λ, a). For each a, we adopt an HBIC proposed by Wang et al. [29] to select the optimal tuning parameter $\widehat{\lambda}$, which is defined as

$$\widehat{\lambda} = \arg\min_{\lambda \in \Lambda} \left\{ \mathrm{HBIC}(\lambda) = \log(\|\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}}(\lambda)\|^2/n) + \frac{C_n \log(p)}{n} |M(\lambda)| \right\}, \tag{38}$$

where Λ is a subset of $(0, +\infty)$, $M(\lambda) = \{j : \widehat{\beta}_j(\lambda) \neq 0\}$ 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* $\widehat{\boldsymbol{\beta}} = 0$ *whenever* $\lambda \geq \lambda_{\max}$ *in SPLS procedure (3), where*

$$\lambda_{\max} = \frac{(\|\mathbf{X}^T\mathbf{y}/n\|_\infty + a/2)^2}{2(a+1)}. \tag{39}$$

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

$$g(t) = \begin{cases} \dfrac{t}{2} + \dfrac{\rho(t)}{t}, & t \neq 0, \\ \liminf\limits_{t \to 0^+} g(t), & t = 0. \end{cases} \quad t^* = \arg\min_{t \geq 0} g(t), \quad T^* = \inf_{t>0} g(t) = \lim_{t \to t^*} g(t), \tag{40}$$

where $0/0 = 0$. For the SICA penalty, it easily follows that

$$t^* = \sqrt{2\lambda(a+1)} - a, \quad T^* = \sqrt{2\lambda(a+1)} - \frac{a}{2}. \tag{41}$$

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

$$S^\rho(z) = \arg\min_{\beta \in \mathbb{R}} [(z-\beta)^2/2 + \rho(\beta)], \tag{42}$$

which can be set-valued. Lemma 3.2 in [30] tells us if $\beta^* \in S^\rho(z)$, then $T^* > |z|$ implies $\beta^* = 0$. After some algebra, the dual variable of SPLS (3) is given by $\mathbf{d}^* = \mathbf{X}^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}^*)/n = \{d_j^* : j = 1, 2, \ldots, p\}$. Then according to Lemma 3.3 in [30], an element $\boldsymbol{\beta}^* \in \mathbb{R}^p$ is a coordinate-wise minimizer to problem (3) if and only if $\beta_j^* \in S^\rho(\beta_j^* + d_j^*)$, $j = 1, 2, \ldots, p$. One can easily check that $\boldsymbol{\beta}^* = \mathbf{0}$ satisfies the above inclusion provided $T^* > \|\mathbf{X}^T\mathbf{y}/n\|_\infty$. Thus, we complete the proof for Proposition 3.1 by (41). ■

3.4. Continuation strategy

We couple SPLS–ADMM and SPLS–CD with the continuation strategy for efficient computational implementation. To be precise, one needs a starting value λ_0 for the parameter λ and a decreasing factor $\mu \in (0, 1)$ to obtain a decreasing sequence $\{\lambda_g\}$, where $\lambda_g = \lambda_0 \mu^g$, and then run Algorithm 1 or Algorithm 1 to solve the λ_{g+1}-problem initialized with the solution of λ_g-problem. In practice, we use $\lambda_0 = \lambda_{\max}$ in Equation (39) and set $\lambda_{\min} = 1e-10\lambda_{\max}$, and then divide the interval $[\lambda_{\min}, \lambda_{\max}]$ into G (the number of grid points) equally distributed subintervals in the logarithmic scale. Numerically, μ is determined by

Algorithm 4 Continuation for SPLS–ADMM and SPLS–CD

Input: $\lambda_0 = \lambda_{\max} \geq \frac{(\|\mathbf{X}^T\mathbf{y}/n\|_\infty + a/2)^2}{2(a+1)}$; $\boldsymbol{\beta}(\lambda_0) = \mathbf{0}, \boldsymbol{\theta}(\lambda_0) = \mathbf{0}$; $\mu \in (0,1)$.

1: **for** $g = 1, 2, 3, \cdots G$ **do**
2: Set $\lambda_g = \lambda_0 \mu^g$ and $(\boldsymbol{\beta}^0, \boldsymbol{\theta}^0) = (\boldsymbol{\beta}(\lambda_{g-1}), \boldsymbol{\theta}(\lambda_{g-1}))$.
3: Find $(\boldsymbol{\beta}(\lambda_g), \boldsymbol{\theta}(\lambda_g))$ and $\boldsymbol{\beta}(\lambda_g)$ by Algorithm 2 and Algorithm 3, respectively.
4: Compute the HBIC values.
5: **end for**

Output: Select $\widehat{\lambda}$ by Equation (38).

G. Clearly, a large G value implies a large decreasing factor μ. Implementing Algorithm 1 or Algorithm 1 for each value of a and the sequence $\lambda_{\max} = \lambda_0 > \lambda_1 > \cdots > \lambda_G = \lambda_{\min}$ to be considered gives the entire SPLS solution path. Then we select the optimal λ from the candidate set $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_G\}$ using HBIC (38). Summarizing the idea leads to Algorithm 1. We refer the reader to [18,19,30] for more detailed discussions.

Now we study the influence of the parameters G and M in the SPLS–ADMM with continuation algorithm on the exact support recovery probability (Probability), that is, the percentage of the estimated model $\widehat{\mathcal{A}}$ agrees with the true model $\mathcal{A}$, and the CPU time (Time, in seconds), respectively. To this end, we fix $a = 0.01$ and consider the design matrix $\mathbf{X}$ to be a 200×400 random Gaussian matrix, whose rows are drawn independently from $N(0, \Sigma)$ with $\Sigma = (r^{|j-k|})$, $1 \leq j, k \leq p$, where $r = 0.1$. The noise vector $\boldsymbol{\varepsilon} \sim N(0, \sigma^2 \mathbf{I}_n)$ with the noise level $\sigma = 0.01$. The underling regression coefficient vector $\boldsymbol{\beta} \in \mathbb{R}^p$ is generated following [31], which will be specified in the next section (see Equations (44)–(45)). Then the observation vector $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$. For different parameter tuples (G, M), we combine them with 10 different sparsity levels with $T = 2{:}2{:}20$, that is, $T \in \{2, 4, , \ldots, 20\}$. All the results are computed based on 100 independent realizations of the problem setup. The numerical results are summarized in Figure 2, which consider the following two settings: (a) $G = 100$, and varying $M \in \{1, 2, 3\}$; (b) $M = 1$, and varying $G \in \{40, 60, 80, 100\}$.

It is observed from Figure 2 that the influence of the parameter M is very mild on the exact support recovery probability. By increasing M, at the expense of more computing time, one can hardly improve the reconstruction accuracy. We can also find in Figure 2 that Larger G values make the algorithm have better exact support recovery probability, but the enhancement decreases as G increases. Unsurprisingly, a relatively small value of G, for example, $G = 40$, can degrade the accuracy of support recovery, due to insufficient resolution of the solution path. Thus, it is reasonable to choose $(G, M) = (100, 1)$ for the SPLS–ADMM with continuation algorithm through synthetical consideration in terms of efficiency and accuracy. For the sake of fairness, we also couple SPLS–CD with the continuation strategy with the same choice of (G, M) for SPLS–ADMM. In practice, in order to acquire sufficient resolution of the solution path, one can set G larger than 100, especially in the case of unknown (r, σ, T) in certain given data set. For instance, we use $G = 200$ for the real data set in Section 3.6.

For λ_g with $g \in \{1, 2, \ldots, G\}$, we denote that m_{λ_g} is the number of iterations at the grid point λ_g (clearly $1 \leq m_{\lambda_g} \leq M$). By Equation (38), we have $\widehat{\lambda} = \lambda_{\widehat{g}}$, where

$$\widehat{g} = \underset{g \in \{1,2,\ldots,G\}}{\arg\min} \{\mathrm{HBIC}(\lambda_g)\}.$$

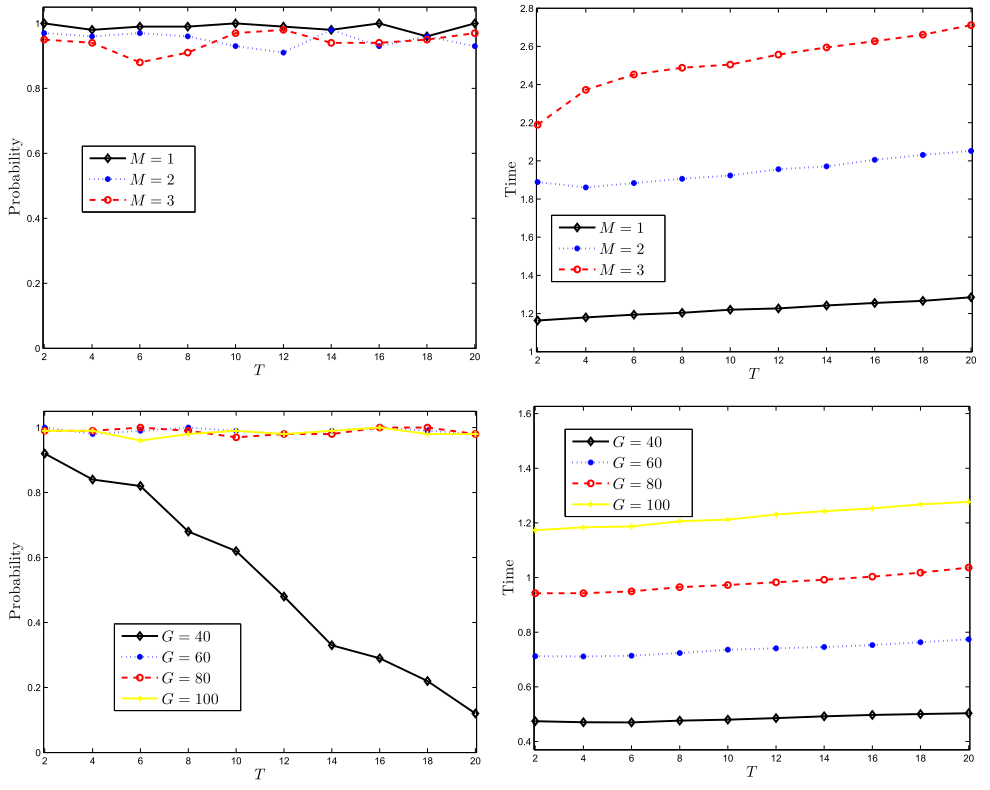

Figure 2. Influence of M on the exact support recovery probability (top left panel), influence of M on the CPU time (top right panel), influence of G on the exact support recovery probability (bottom left panel) and influence of G on the CPU time (bottom right panel).

According to the continuation strategy, i.e. solving the λ_{g+1}-problem initialized with the solution of λ_g-problem, the total number of iterations can be denoted as

$$M_{\widehat{g}} = \sum_{g=1}^{\widehat{g}} m_{\lambda_g}. \tag{43}$$

3.5. Simulation

3.5.1. Implementation setting

We generate synthetic data from Equation (1). The rows of the $n \times p$ matrix $\mathbf{X}$ are sampled as i.i.d. copies from $N(\mathbf{0}, \Sigma)$ with $\Sigma = (r^{|j-k|})$, $1 \leq j, k \leq p$, where r is the correlation coefficient of $\mathbf{X}$. We chose three levels of correlation $r = 0.3, 0.5$ and 0.7, which correspond to the weak, moderate and strong correlation. The noise vector $\boldsymbol{\varepsilon}$ is generated independently from $N(\mathbf{0}, \sigma^2 \mathbf{I}_n)$, where σ is the noise level. We consider two levels of noise: $\sigma = 1$ (higher level) and $\sigma = 0.1$ (lower level). The underlying regression coefficient vector $\boldsymbol{\beta}$ is a random sparse vector chosen as T-sparse with a dynamic range (DR) defined by

$$\mathrm{DR} := \frac{\max\{|\beta_j| : \beta_j \neq 0\}}{\min\{|\beta_j| : \beta_j \neq 0\}} = 10. \tag{44}$$

Table 1. Simulation results for model selection with $a = 0.01$, $n = 200$ and $T = 5$ based on 100 replications.

p	r	σ	Method	Time	Iter	RE	AE	MS	CM
400	0.3	0.1	SPLS–ADMM	0.9864	50.65	0.0015	0.0144	5.39	65%
			SPLS–CD	1.2777	50.91	0.0016	0.0150	5.60	60%
		1	SPLS–ADMM	0.8541	32.47	0.0160	0.1569	5.44	65%
			SPLS–CD	1.1059	32.99	0.0173	0.1654	5.64	57%
	0.5	0.1	SPLS–ADMM	0.9961	50.87	0.0014	0.0137	5.37	71%
			SPLS–CD	1.2899	51.13	0.0015	0.0141	5.53	64%
		1	SPLS–ADMM	0.8602	32.55	0.0178	0.1740	5.60	55%
			SPLS–CD	1.1073	33.34	0.0199	0.1821	5.92	49%
	0.7	0.1	SPLS–ADMM	1.0025	50.66	0.0015	0.0138	5.26	76%
			SPLS–CD	1.2983	51.15	0.0015	0.0145	5.37	70%
		1	SPLS–ADMM	0.8666	32.62	0.0155	0.1462	5.27	75%
			SPLS–CD	1.1102	33.24	0.0169	0.1561	5.47	68%
800	0.3	0.1	SPLS–ADMM	1.9017	49.63	0.0016	0.0149	5.32	72%
			SPLS–CD	2.5398	50.70	0.0018	0.0156	5.62	65%
		1	SPLS–ADMM	1.6265	31.42	0.0193	0.1815	5.48	61%
			SPLS–CD	2.2422	32.71	0.0213	0.1961	5.93	45%
	0.5	0.1	SPLS–ADMM	1.8969	49.87	0.0016	0.0149	5.31	73%
			SPLS–CD	2.5352	50.54	0.0018	0.0161	5.70	62%
		1	SPLS–ADMM	1.6197	31.73	0.0181	0.1711	5.45	60%
			SPLS–CD	2.2274	32.72	0.0195	0.1761	5.77	54%
	0.7	0.1	SPLS–ADMM	1.9070	49.73	0.0015	0.0143	5.23	79%
			SPLS–CD	2.5382	50.73	0.0017	0.0156	5.59	62%
		1	SPLS–ADMM	1.6179	31.86	0.0189	0.1795	5.33	73%
			SPLS–CD	2.2002	32.80	0.0201	0.1854	5.87	50%

Following Becker et al. [31], each non-zero entry of $\boldsymbol{\beta}$ is generated as follows:

$$\beta_j = \eta_{1j} 10^{\eta_{2j}}, \tag{45}$$

where $\eta_{1j} = \pm 1$ with probability $\frac{1}{2}$, η_{2j} is uniformly distributed in $[0, 1]$ and $j \in \mathcal{A} = \{j : \beta_j \neq 0\}$. We set $(n, p) = (200, 400)$ and $(200, 800)$ and fix the sparsity level $T = 5$. The default initial values chosen for SPLS–ADMM and SPLS–CD are $\boldsymbol{\beta}^0 = \boldsymbol{\theta}^0 = \boldsymbol{\tau}^0 = \mathbf{0} \in \mathbb{R}^p$ and $\boldsymbol{\beta}^0 = \mathbf{0} \in \mathbb{R}^p$, respectively. For SPLS–ADMM, we specify $\rho = 1$ according to Theorem 2.2. The convergence criterion is $\|\boldsymbol{\beta}^{k+1} - \boldsymbol{\beta}^k\| \leq \delta$ with $\delta = 10^{-4}$. We use the same stopping criterion for SPLS–CD. The number of simulations is $N = 100$.

3.5.2. Efficiency and accuracy

To further illustrate the efficiency and accuracy of the proposed SPLS–ADMM with continuation algorithm on model selection issues, based on N replications, we compare it with SPLS–CD in terms of the average CPU time (Time, in seconds), the average number of iterations (Iter) $N^{-1}\sum_{s=1}^{N} M_{\widehat{g}}^{(s)}$ ($M_{\widehat{g}}$ is given in Equation (43)), the average ℓ_2 relative error (RE) $N^{-1}\sum_{s=1}^{N}(\|\widehat{\boldsymbol{\beta}}^{(s)} - \boldsymbol{\beta}\|_2/\|\boldsymbol{\beta}\|_2)$, the average ℓ_∞ absolute error (AE) $N^{-1}\sum_{s=1}^{N}\|\widehat{\boldsymbol{\beta}}^{(s)} - \boldsymbol{\beta}\|_\infty$, the estimated average model size (MS) $N^{-1}\sum_{s=1}^{N}|\widehat{\mathcal{A}}^{(s)}|$ and the proportion of correct models (CM) $N^{-1}\sum_{s=1}^{N} I\{\widehat{\mathcal{A}}^{(s)} = \mathcal{A}\}$ (in percentage terms). Simulation results with $a = 0.01$ for model selection with different parameter tuples (p, r, σ) are summarized in Table 1.

For each (p, r, σ) combination, we see from Table 1 that SPLS–ADMM has better speed performance than SPLS–CD by Time and Iter. According to RE and AE, it is clear that SPLS–ADMM is numerically more accurate than SPLS–CD. It can also be seen from Table 1 that with respect to MS and CM, although two solvers tend to overestimate the true model, SPLS–ADMM seems to select a smaller model and the correct model more frequently than SPLS–CD. For fixed values of r and σ, the timing of SPLS–ADMM increases linearly with p. Unsurprisingly, larger σ will degrade the accuracy of SPLS–ADMM. With p and r fixed, the CPU time of SPLS–ADMM slightly decreases as σ increases. Given p and σ, the CPU time of SPLS–ADMM is robust with respect to r. Similar phenomena also hold for SPLS–CD. Overall, as shown in Table 1, SPLS–ADMM does a better job than SPLS–CD in terms of both efficiency and accuracy.

We fix the concavity parameter $a = 0.01$ for SICA following [32], roughly in accord with the recommendation SELO [32] with $\tau = 0.01$ suggested therein, since both the SICA and SELO penalties closely resemble the L_0 penalty, and converge to the L_0 penalty as the concavity parameter goes to 0. However, by reason that $a \in (0, \infty)$, it is important to study the sensitivity of the proposed algorithm with respect to the variation of a. In next subsection, we conduct the sensitivity analysis for a and other model parameters.

3.5.3. Influence of model parameters

We now consider the effects of each of the model parameters (a, n, p, r, σ, T) on the performance of SPLS–ADMM and SPLS–CD more closely in terms of the exact support recovery probability (Probability) and the CPU time (Time, in seconds), and corresponding numerical results averaged over 100 independent runs are given in Figures 3 and 4, respectively. The parameters for solvers are set as follows.

Influence of the concavity parameter a. The top left panels of Figures 3 and 4 show the results on Probability and Time, respectively. Data are generated from the model with $(a \in \{0.1, 0.2, 0.5, 1, 2\}, n = 200, p = 400, r = 0.1, \sigma = 0.1, T = 5)$.

Influence of the sample size n. The top right panels of Figures 3 and 4 show the results on Probability and Time, respectively. Data are generated from the model with $(a = 0.01, n \in \{100, 120, 140, 160, 180, 200\}, p = 400, r = 0.1, \sigma = 0.1, T = 5)$.

Influence of the dimension p. The middle left panels of Figures 3 and 4 show the results on Probability and Time, respectively. Data are generated from the model with $(a = 0.01, n = 200, p \in \{400, 600, 800, 1000\}, r = 0.1, \sigma = 0.1, T = 5)$.

Influence of the correlation level r. The middle right panels of Figures 3 and 4 show the results on Probability and Time, respectively. Data are generated from the model with $(a = 0.01, n = 200, p = 400, r \in \{0.1, 0.3, 0.5, 0.7, 0.9\}, \sigma = 0.1, T = 5)$.

Influence of the noise level σ. The bottom left panels of Figures 3 and 4 show the results on Probability and Time, respectively. Data are generated from the model with $(a = 0.01, n = 200, p = 400, r = 0.1, \sigma \in \{0.2, 0.4, 0.8, 1.2, 1.6\}, T = 5)$.

Influence of the sparsity level T. The bottom right panels of Figures 3 and 4 show the results on Probability and Time, respectively. Data are generated from the model with $(a = 0.01, n = 200, p = 400, r = 0.1, \sigma = 0.1, T \in \{5, 10, 15, 20, 25\})$.

In summary, numerical results shown in Figures 3 and 4 demonstrate that two solvers seem to have similar variation tendency with considered model parameters, while SPLS–ADMM is generally more accurate and more efficient than SPLS–CD.

3.6. Application

We analyse a rat eye expression data called eyedata which is publicly available in R package **flare** [33] to illustrate the application of the SPLS–ADMM with continuation algorithm in high-dimensional settings. This data is a gene expression data from the microarray experiments of mammalian eye tissue samples of [34] and is detailedly described and applied by many papers (cf., e.g. [35,36]) that want to find the gene probes that are most

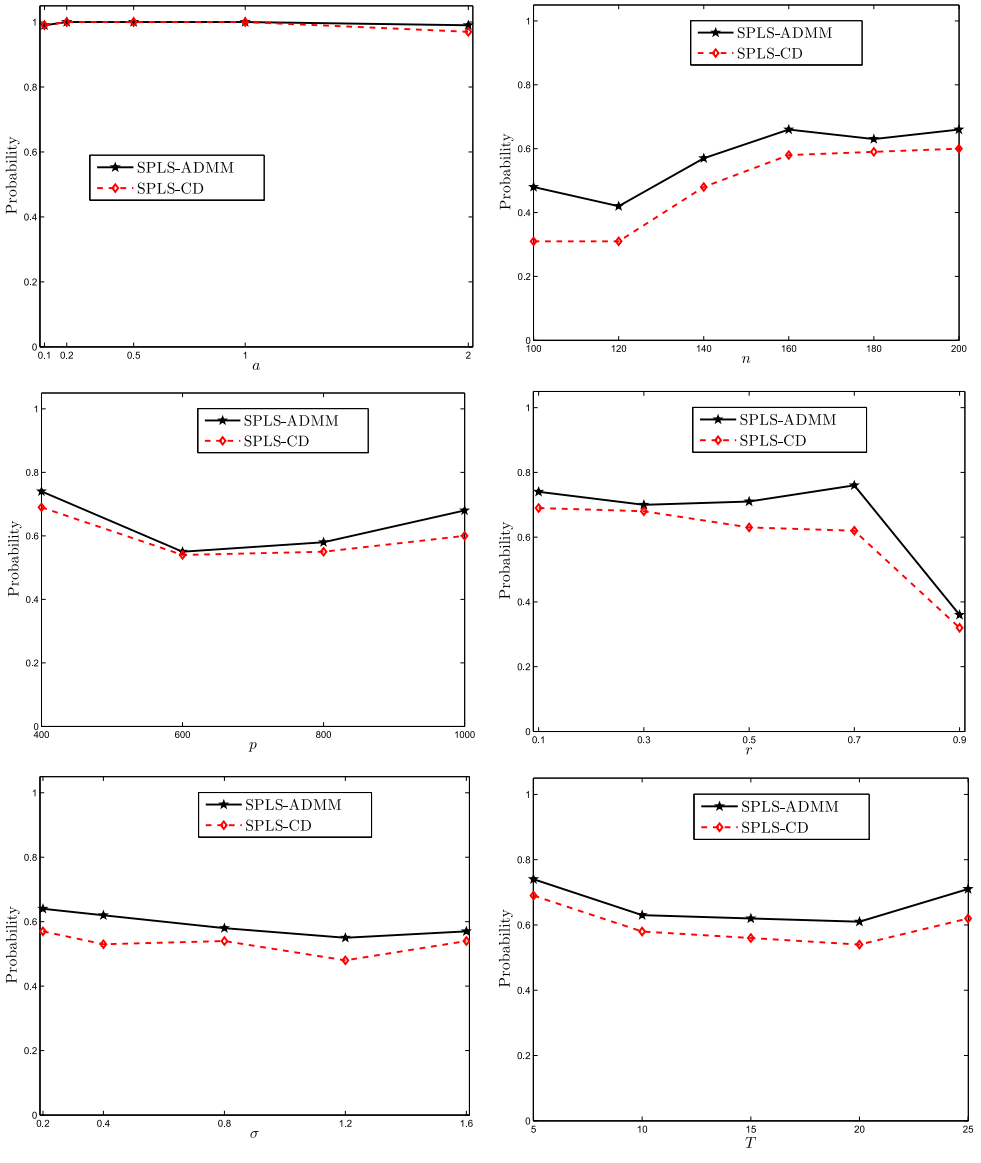

Figure 3. Numerical results of the influence of the concavity parameter a (top left panel), the sample size n (top right panel), the dimension p (middle left panel), the correlation level r (middle right panel), the noise level σ (bottom left panel) and the sparsity level T (bottom right panel) on the exact support recovery probability of two solvers.

related to TRIM32 in sparse high-dimensional regression models. The response variable **y** is a numeric vector of length 120 giving expression level of gene TRIM32 which causes Bardet–Biedl syndrome. The design matrix **X** is a 120×200 matrix which represents the data of 120 rats with 200 gene probes.

Since the exact solution for the eyedata set is unknown, we consider three gold standards (i.e. benchmarks) for comparison purposes: **flare**[33] (the SQRT LASSO with $\lambda = \sqrt{\log(p)/n}$), **glmnet** [37] (10-fold cv.glmnet with the lambda.1se rule and set.seed = 0)

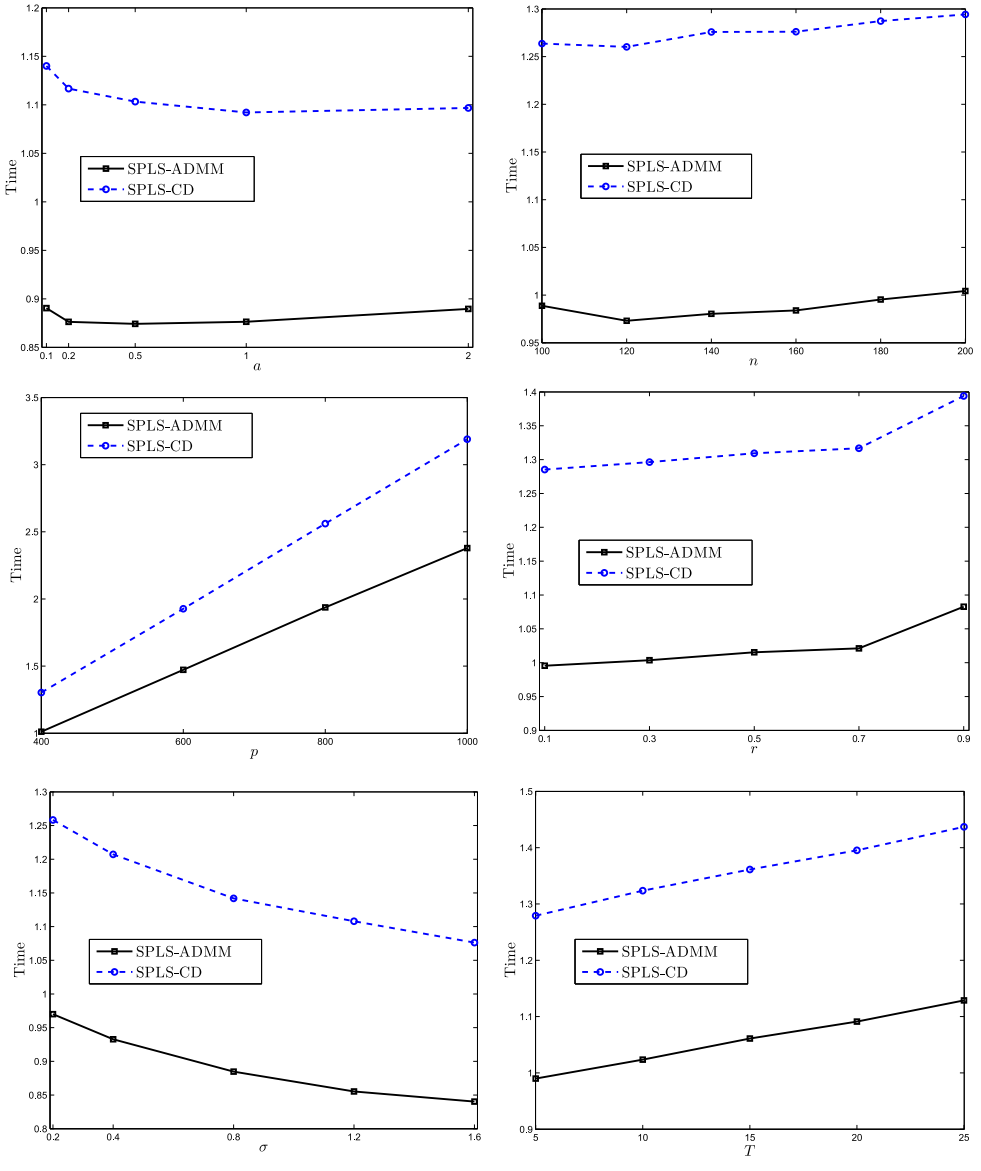

Figure 4. Numerical results of the influence of the concavity parameter a (top left panel), the sample size n (top right panel), the dimension p (middle left panel), the correlation level r (middle right panel), the noise level σ (bottom left panel) and the sparsity level T (bottom right panel) on the CPU time of two solvers.

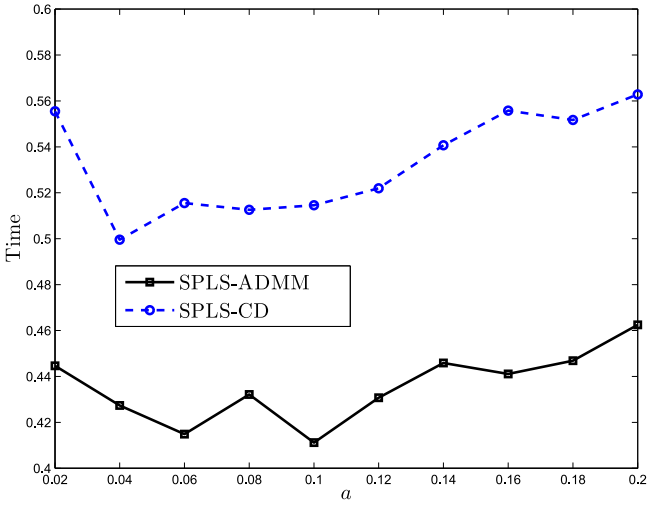

Figure 5. The CPU time of SPLS-ADMM and SPLS-CD for the eyedata set with $(G, M, \rho) = (200, 1, 1)$ and $a = 0.02{:}0.02{:}0.2$.

Table 2. Analysis of the eyedata set.

No.	Term	Probe	**flare**	**glmnet**	**ncvreg**	SPLS–ADMM	SPLS–CD
	Intercept		7.6975	7.7133	6.3610	7.3062	8.2531
1	β_{11}	6222	0.0130	0.0140	0	0.0181	0
2	β_{42}	12085	0.0151	0.0140	0	0	0
3	β_{54}	14949	0.0147	0.0160	0	0.0223	0
4	β_{55}	15224	0	0	0	0	0.1919
5	β_{62}	15863	−0.0381	−0.0387	0	−0.0230	0
6	β_{87}	21092	−0.0933	−0.0932	−0.0642	−0.1847	−0.2430
7	β_{90}	21550	−0.0189	−0.0183	0	0	0
8	β_{99}	22029	0	0	0	0.0316	0
9	β_{102}	22140	0	−0.0027	0	0	0
10	β_{127}	23804	−0.0074	−0.0078	0	0	0
11	β_{134}	24245	0.0169	0.0161	0	0	0
12	β_{136}	24353	−0.0260	−0.0275	0	−0.0666	0
13	β_{140}	24565	0.0144	0.0184	0	0	0
14	β_{146}	24892	0.0072	0.0079	0	0	0
15	β_{153}	25141	0.1472	0.1452	0.2702	0.2106	0
16	β_{155}	25367	0.0093	0.0092	0	0	0
17	β_{162}	25909	0	0	0	0.0280	0
18	β_{172}	27179	0	0	0	0.0403	0
19	β_{180}	28680	0.0684	0.0687	0.1836	0.0298	0
20	β_{185}	28967	−0.0829	−0.0845	−0.2655	−0.0908	0
21	β_{187}	29041	−0.0369	−0.0384	0	0	0
22	β_{188}	29045	−0.0068	−0.0073	0	0	0
23	β_{200}	30141	−0.0451	−0.0467	−0.0239	−0.0566	0
	Time		–	–	–	0.4520	0.5609
	PMSE		0.0048	0.0048	0.0051	0.0049	0.0075

Notes: Estimated coefficients of different methods are provided. The zero entries correspond to variables omitted.

and **ncvreg** [23] (10-fold cv.ncvreg with seed = 0). Coupled with the continuation strategy, we apply the SPLS–ADMM and SPLS–CD algorithms to the eyedata set. For both approaches, tuning parameters are selected using HBIC (38). By similar arguments in simulations, we fix $(\rho, M) = (1, 1)$. Since we do not know the true levels of (r, σ, T) in the

eyedata set, as discussed in Section 3.4, we may need a larger G. Then we specify $G = 200$ for the analysis of real data. To decide a, we consider the parameter settings with varying $a = 0.02{:}0.02{:}0.2$ given $(G, M, \rho) = (200, 1, 1)$. Displayed in Figure 5 is corresponding CPU time plot produced by SPLS–ADMM and SPLS–CD. It is observed from Figure 5 that SPLS–ADMM is faster than SPLS–CD for every a considered, while SPLS–CD has the best timing performance at $a = 0.04$. Thus, we fix $a = 0.04$ for this eyedata set and then compare the solution accuracy of two solvers. Gene probe information, corresponding non-zero estimates, the CPU time and the predictive mean squared errors (PMSE) calculated by $n^{-1}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$ are provided in Table 2. We can see from Table 2 that SPLS–ADMM is faster than SPLS–CD in terms of Time, while the PMSE by SPLS–ADMM is smaller than the counterpart by SPLS–CD, which demonstrates that SPLS–ADMM performs better than SPLS–CD in terms of both efficiency and accuracy. Notably, the accuracy of SPLS–ADMM, in terms of PMSE, is quite comparable with the three benchmarks considered.

4. Concluding remarks

We have developed an ADMM with continuation algorithm for solving high-dimensional non-convex SICA–PLS problems. We rigorously investigate the convergence property and the KKT optimality condition of the proposed algorithm. When coupled with the continuation strategy and an HBIC tuning parameter selector, our proposed procedure is very efficient and accurate.

We focus on our 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 two term Taylor series expansions of the log likelihoods (see [23,38]). Furthermore, we would like to consider the extensions of our method to the high-dimensional structured sparsity SPLS model [39] in future. As pointed in Boyd et al. [13], ADMM can be implemented as a distributed algorithm of practical use [40]. 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.

Our main theoretical results hold under some conditions in Theorem 2.1, where we assumed $\{\boldsymbol{\theta}^k\}$ is bounded therein, which is important to obtain the convergence result of using ADMM on non-convex problems. It is open whether Theorem 2.1 still holds without the requirement on boundedness of $\{\boldsymbol{\theta}^k\}$, which we also leave for future research.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was supported in part by the National Natural Science Foundation of China [Grant Nos. 11501579, 11671311, 11701571, 41572315] and the Fundamental Research Funds for the Central Universities [Grant No. CUGW150809].

References

[1] Bühlmann P, Van De Geer S. Statistics for high-dimensional data: methods, theory and applications. Heidelberg: Springer; 2011.

[2] Cai T, Shen X. High-dimensional data analysis. Beijing: Higher Education Press/Singapore: World Scientific; 2011.
[3] Tibshirani R. Regression shrinkage and selection via the lasso. J R Stat Soc Ser B (Stat Methodol). 1996;58:267–288.
[4] Fan J, Li R. Variable selection via nonconcave penalized likelihood and its oracle properties. JAmer Statist Assoc. 2001;96:1348–1360.
[5] Zou H. The adaptive lasso and its oracle properties. J Amer Statist Assoc. 2006;101:1418–1429.
[6] Zhang CH. Nearly unbiased variable selection under minimax concave penalty. Ann Statist. 2010;38:894–942.
[7] Fan J, Lv J. A selective overview of variable selection in high dimensional feature space. Statist Sinica. 2010;20:101–148.
[8] Lv J, Fan Y. A unified approach to model selection and sparse recovery using regularized least squares. Ann Statist. 2009;37:3498–3528.
[9] Zou H, Li R. One-step sparse estimates in nonconcave penalized likelihood models. Ann Statist. 2008;36:1509–1533.
[10] Lin W, Lv J. High-dimensional sparse additive hazards regression. J Amer Statist Assoc. 2013;108:247–264.
[11] Shi YY, Cao YX, Jiao YL, et al. SICA for Cox's proportional hazards model with a diverging number of parameters. Acta Math Appl Sin Engl Ser. 2014;30:887–902.
[12] Shi YY, Jiao YL, Yan L, et al. A modified BIC tuning parameter selector for SICA-penalized Cox regression models with diverging dimensionality. J Math. 2017;37:723–730.
[13] Boyd S, Parikh N, Chu E. Distributed optimization and statistical learning via the alternating direction method of multipliers. Found Trends Mach Learn. 2011;3:1–122.
[14] Yang J, Zhang Y. Alternating direction algorithms for ℓ_1-problems in compressive sensing. SIAM J Sci Comput. 2011;33:250–278.
[15] Yuan X. Alternating direction method for covariance selection models. J Sci Comput. 2012;51:261–273.
[16] Lu Z, Pong TK, Zhang Y. An alternating direction method for finding Dantzig selectors. Comput Statist Data Anal. 2012;56:4037–4046.
[17] Jin ZF, Wan Z, Jiao Y, et al. An alternating direction method with continuation for nonconvex low rank minimization. J Sci Comput. 2016;66:849–869.
[18] Fan Q, Jiao Y, Lu X. A primal dual active set algorithm with continuation for compressed sensing. IEEE Trans Signal Process. 2014;62:6276–6285.
[19] 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. 2015;39:400–426.
[20] Pham DS, Venkatesh S. Efficient algorithms for robust recovery of images from compressed data. IEEE Trans Image Process. 2013;22:4724–4737.
[21] Wen Z, Yang C, Liu X, et al. Alternating direction methods for classical and ptychographic phase retrieval. Inverse Probl. 2012;28:115010.
[22] Tseng P. Convergence of a block coordinate descent method for nondifferentiable minimization. J Optim Theory Appl. 2001;109:475–494.
[23] Breheny P, Huang J. Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. Ann Appl Stat. 2011;5:232–253.
[24] Mazumder R, Friedman JH, Hastie T. *Sparsenet*: coordinate descent with nonconvex penalties. J Amer Statist Assoc. 2011;106:1125–1138.
[25] Peng B, Wang L. An iterative coordinate descent algorithm for high-dimensional nonconvex penalized quantile regression. J Comput Graph Statist. 2015;24:676–694.
[26] Golub GH, Van Loan CF. Matrix computations. 4th ed. Baltimore: John Hopkins University Press; 2013.
[27] Wang H, Li R, Tsai CL. Tuning parameter selectors for the smoothly clipped absolute deviation method. Biometrika. 2007;94:553–568.
[28] Wang H, Li B, Leng C. Shrinkage tuning parameter selection with a diverging number of parameters. J R Stat Soc Ser B (Stat Methodol). 2009;71:671–683.

[29] Wang L, Kim Y, Li R. Calibrating nonconvex penalized regression in ultra-high dimension. Ann Statist. 2013;41:2505–2536.
[30] Jiao Y, Jin B, Lu X, et al. A primal dual active set algorithm for a class of nonconvex sparsity optimization; 2016. Preprint arXiv:1310.1147v3.
[31] Becker S, Bobin J, Candès EJ. NESTA: a fast and accurate first-order method for sparse recovery. SIAM J Imaging Sci. 2011;4:1–39.
[32] Dicker L, Huang B, Lin X. Variable selection and estimation with the seamless-L_0 penalty. Statist Sinica. 2013;23:929–962.
[33] 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. 2015;16:553–557.
[34] 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. 2006;103:14429–14434.
[35] Huang J, Ma S, Zhang CH. Adaptive Lasso for sparse high-dimensional regression models. Statist Sinica. 2008;18:1603–1618.
[36] Huang J, Breheny P, Ma S, et al. The Mnet method for variable selection. Statist Sinica. 2016;26:903–923.
[37] Friedman J, Hastie T, Tibshirani R. Regularization paths for generalized linear models via coordinate descent. J Stat Softw. 2010;33:1–22.
[38] Simon N, Friedman J, Hastie T, et al. Regularization paths for Cox's proportional hazards model via coordinate descent. J Stat Softw. 2011;39:1–13.
[39] Breheny P. The group exponential lasso for bi-level variable selection. Biometrics. 2015;71:731–740.
[40] Yu L, Lin N, Wang L. A parallel algorithm for large-scale nonconvex penalized quantile regression. J Comput Graph Statist. 2017;26:935–939.