SIAM J. Sci. Comput.
Vol. 42, No. 1, pp. A395–A423

AN EFFICIENT PROXIMAL BLOCK COORDINATE HOMOTOPY METHOD FOR LARGE-SCALE SPARSE LEAST SQUARES PROBLEMS[*]

GUOQIANG WANG[†], XINYUAN WEI[†], BO YU[‡], AND LIJUN XU[§]

Abstract. In this paper, an efficient and robust algorithm framework is presented for large-scale sparse least squares problems. This framework decomposes the original sparse least squares problem into a sequence of small-scale l_1-minimization subproblems. Every subproblem is solved by an improved l_1-homotopy method which differs from the original l_1-homotopy method by adopting a warm-start procedure and an ε-precision verification-correction technique. Moreover, based on a carefully designed block coordinate update strategy, the algorithm framework is proved to converge to a $\tilde{\tau}$-"precise" solution in a finite number of steps and the value of the objective function linearly converges. Numerical comparisons between the presented algorithms and a number of state-of-the-art algorithms on real and randomly generated data sets demonstrate the robustness and high performance of the presented algorithms. As an example, the presented algorithm only needs 13 seconds to solve an l_1-minimization problem with tens of millions of samples and features.

Key words. sparse optimization, LASSO, decomposition method, homotopy method, l_{1-2}-minimization, highly coherent matrix

AMS subject classifications. 90C06, 90C26

DOI. 10.1137/19M1243828

1. Introduction.

In this paper, we propose a highly efficient approach for solving the following sparse least squares problem:

$$\min_{x\in\mathbb{R}^n}\left\{\frac{1}{2}\|Ax-b\|^2+\lambda(\|x\|_1-\alpha\|x\|)\right\}, \tag{1.1}$$

where $A \in \mathbb{R}^{m\times n}$ is a real matrix, and $\lambda > 0$ and $\alpha \geq 0$ are given. Throughout this paper, $\|\cdot\|$ denotes the Euclidean norm. When $\alpha = 0$, the problem reduces to the l_1-minimization problem

$$\min_{x\in\mathbb{R}^n}\left\{\frac{1}{2}\|Ax-b\|^2+\lambda\|x\|_1\right\}, \tag{1.2}$$

which is also called LASSO [32]; when $\alpha \neq 0$, the problem is an l_{1-2}-minimization problem [18]. x^* is a Karush–Kuhn–Tucker (KKT) point of (1.1) if it satisfies

[*]Submitted to the journal's Methods and Algorithms for Scientific Computing section February 11, 2019; accepted for publication (in revised form) December 13, 2019; published electronically February 19, 2020.
https://doi.org/10.1137/19M1243828

Funding: This work was supported by the National Natural Science Foundation of China through grants 11571061, 11401075, and 11701065 and by the Fundamental Research Funds for the Central Universities through grant 724 DUT16LK05 and DUT17LK14.

[†]School of Mathematical Sciences, Dalian University of Technology, Dalian, Liaoning 116024, People's Republic of China (wangguojim@mail.dlut.edu.cn, xinyuanwei@mail.dlut.edu.cn).
[‡]School of Mathematical Sciences, Dalian University of Technology, Dalian, Liaoning 116024, People's Republic of China, and Interdisciplinary Research Institute, Dalian University of Technology at Panjin, Panjin, Liaoning 124221, China (yubo@dlut.edu.cn).
[§]School of Science, Dalian Maritime University, Dalian 116026, People's Republic of China (lijun_xu@dlmu.edu.cn).

$$A_j^T(Ax^* - b) - \lambda\alpha\frac{x_j^*}{\|x^*\|} + \lambda \text{sign}(x_j^*) = 0, x_j^* \neq 0, \tag{1.3}$$

$$\left| A_j^T(Ax^* - b) - \lambda\alpha\frac{x_j^*}{\|x^*\|} \right| \leq \lambda, x_j^* = 0, \tag{1.4}$$

where A_j denotes the jth column of A and

$$\text{sign}(x_j) = \begin{cases} 1, & x_j > 0; \\ 0, & x_j = 0; \\ -1, & x_j < 0. \end{cases}$$

(1.3) and (1.4) are the optimality conditions in a complementary form. If for any j with $x_j^* = 0$,

$$\left| A_j^T(Ax^* - b) - \lambda\alpha\frac{x_j^*}{\|x^*\|} \right| < \lambda,$$

then the complementarity conditions strictly hold, moreover, if $|A_j^T(Ax^* - b) - \lambda\alpha\frac{x_j^*}{\|x^*\|}|$ is close to λ, the complementarity is weak.

Over the past decade, the advent of technological breakthroughs in automated data collection has brought about significant challenges in high-dimensional and massive data analysis. To meet these challenges, researchers have developed many new sophisticated statistical tools. Among these studies, the l_1-minimization models have received extensive attention since their wide range of applications: high-dimensional variables selection and features selection [32, 33], compressed sensing [15], signal and image processing [2, 13, 20, 35, 36], etc.

The l_1-minimization problems are closely related to the basis pursuit (BP) problem [13]

$$\min_{x \in \mathbb{R}^n} \|x\|_1 \text{ s.t. } Ax = b, \tag{1.5}$$

which is a loose approximation of the l_0-minimization problem

$$\min_{x \in \mathbb{R}^n} \|x\|_0 \text{ s.t. } Ax = b. \tag{1.6}$$

Candès, Romberg, and Tao [9, 10, 11] have shown that if A satisfies the restricted isometry property (RIP) condition and the solution of the l_0-minimization problem satisfies some reasonable conditions, then it can be found by solving the BP problem or the l_1-minimization problem. However, to verify a deterministic matrix A satisfies the RIP condition is NP-hard. Further, Esser, Lou, and Xin [18] and Yin et al. [39] derived a coherence condition which is similar to the RIP condition. Specifically, a matrix satisfying the RIP conditions tends to have small coherence or to be incoherent; conversely, a highly coherent matrix is unlikely to possess small restricted isometry constant. An advantage of the coherence condition is that it is easy to verify.

Besides the l_1 regularizer, many nonconvex regularizers, interpolated between l_0 and l_1, have been proposed to better approximate the l_0 regularizer, to name a few, $l_p(0 < p < 1)$ regularizers [23], smoothly clipped absolute deviation (SCAD) [19], and l_{1-2} regularizer [18, 39]. Contour plots for these regularizers can be seen in Figure 1.1. l_{1-2} regularizer is nonconvex yet Lipschitz continuous and used for sparse signal recovery and spectroscopic imaging. Esser, Lou, and Xin [18] and Yin et al. [39] showed that, for a highly coherent matrix A, l_{1-2} regularizer outperforms $l_{1/2}$ and l_1 on promoting sparsity.

The wide range of applications of the sparse least squares problems has inspired researchers to develop various methods for solving them, such as the state-of-the-art

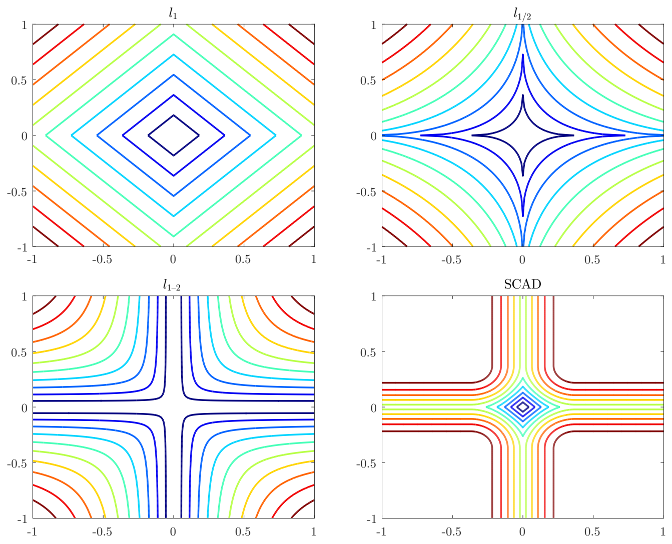

FIG. 1.1. *Contour plots of l_1, $l_p(0 < p < 1)$, l_{1-2} and SCAD regularizer functions.*

first-order iterative methods for l_1-minimization, including ISTA [14], FISTA (also called APG), [2], ADMM [6, 17], which is equivalent to Bergman iteration [25], GPSR [20], SPGL1 [3], SpaRSA [36], FPC_AS [34], NESTA [5], etc., as well as second-order methods, including least angle regression (LARS) [16], l_1-homotopy [1], mfIPM [24], SNF [31], BAS [7], SQA [8], OBA [28], SSNAL [30], etc. Compared with second-order methods, first-order methods often take a smaller amount of calculation at each step and are more efficient to obtain an approximate solution. However, first-order methods often converge slowly when the iterative point is close to one of the solutions. Second-order methods utilize the second-order information to accelerate the convergence, such as the asymptotic superlinear convergence of SSNAL. For nondegenerate problems, second-order methods work quite well and often outperform first-order methods when high accuracy is required.

Among these methods, LARS is a homotopy-like method for the l_1-minimization problem. It treats (1.7) as a parametric programming problem about λ and tracks its solution path $\bar{x}(\lambda)$ which is piecewise-linear by decreasing λ from $\|A^T y\|_\infty$, where $\|\cdot\|_\infty$ denotes the infinity norm of vector. When the number of nonzero components in $\bar{x}(\lambda)$ is small, LARS is computationally effective even for large-scale problems. Different from LARS which tracks the solution path along the parameter λ until it reaches a given value, the l_1-homotopy method first constructs a parametric programming problem

$$\min_{x\in\mathbb{R}^n} \frac{1}{2}\|Ax-b\|^2 + \lambda\|x\|_1 + tu^T x \tag{1.7}$$

with a given initial point $\tilde{x}$, such that $\tilde{x}$ is the solution of (1.7) at $t = 1$, and (1.7) degenerates to (1.2) at $t = 0$, where u can be constructed as

$$u_j = \begin{cases} -A_j^T(A\tilde{x}-b) - \lambda \text{sign}(\tilde{x}_j), & \tilde{x}_j \neq 0; \\ -A_j^T(A\tilde{x}-b) + \dfrac{\lambda}{2}\text{sign}\left(A_j^T(A\tilde{x}-b)\right), & \tilde{x}_j = 0. \end{cases}$$

By tracking the piecewise-linear solution path $\hat{x}(t)$ of (1.7) from $t = 1$ to $t = 0$ with a fixed λ, l_1-homotopy can obtain the solution of (1.2). Moreover, the solution path $\hat{x}(t), 0 \leq t \leq 1$ satisfies

$$\begin{aligned} \hat{x}_{W(t)}(t) &= \left(A_{W(t)}^T A_{W(t)}\right)^{-1}\left(A_{W(t)}^T b - \lambda \text{sign}(\hat{x}_{W(t)}) - t u_{W(t)}\right), \\ \hat{x}_{W^c(t)}(t) &= 0, \end{aligned} \tag{1.8}$$

which are derived from the KKT systems

$$A_j^T(A\hat{x}(t) - b) + t u_j + \lambda \text{sign}(\hat{x}_j(t)) = 0, j \in W(t), \tag{1.9}$$

$$|A_j^T(A\hat{x}(t) - b) + t u_j| \leq \lambda, j \in W^c(t), \tag{1.10}$$

where $W(t) := \{j : \hat{x}_j(t) \neq 0\}$ denotes the working set, $W^c(t) = \{1, \ldots, n\}/W(t)$, and $A_{W(t)}$ and $A_{W^c(t)}$ denote the columns of A indexed by $W(t)$ and $W^c(t)$, respectively. Moreover, $\text{sign}(\hat{x}_j(t))$ remains constant in the intervals of [0,1].

The details of the l_1-homotopy method can be seen in Algorithm 1.

Algorithm 1. l_1-homotopy method [2].

```
Inputs:   $\hat{x}(1) = \tilde{x}$, $W(1) := \{j : \hat{x}_j(1) \neq 0\}$, $W^c(1) = \{1, \ldots, n\}/W(1)$
Outputs:   $\hat{x}(0)$;
   while $t > 0$ do
      Compute $t^* = \max(t^-, t^+)$, where

      (1.11) $t^- = \max_{j \in W(t)} \left\{ \frac{c_j}{d_j} < t | c = \left(A_{W(t)}^T A_{W(t)}\right)^{-1}\left(A_{W(t)}^T b - \lambda \text{sign}(\hat{x}_{W(t)}(t))\right), d = \left(A_{W(t)}^T A_{W(t)}\right)^{-1} u_{W(t)} \right\}$

      and

      (1.12) $t^+ = \max_{j \in W^c(t)} \left\{ \frac{\lambda \text{sign}(\hat{d}_j) - \hat{c}_j}{\hat{d}_j} < t | \hat{c} = A_{W^c(t)}^T A_{W(t)} c - A_{W^c(t)}^T b, \hat{d} = u_{W^c(t)} - A_{W^c(t)}^T A_{W(t)} d \right\}$

      if $t^* > 0$ then
         $t = t^*$
         if $t^- \geq t^+$ then
            Find $\hat{j} \in W(t)$ such that it satisfies (1.11).
            Removing: $W(t) = W(t) \backslash \hat{j}, W^c(t) = W^c(t) \cup \hat{j}$.
         else
            Find $\bar{j} \in W^c(t)$ such that it satisfies (1.12).
            Adding: $W(t) = W(t) \cup \bar{j}, W^c(t) = W^c(t) \backslash \bar{j}$.
         end if
      else
         $t = 0$, $\hat{x}_{W(t)}(0) = c, \hat{x}_{W^c(t)}(0) = 0$.
      end if
   end while
```

The l_1-homotopy method tracks the solution path which satisfies (1.8) by updating the working set $W(t)$ such that the KKT systems (1.9) and (1.10) always hold. By decreasing t, there may exist $\hat{j} \in W(t)$ such that $\hat{x}_{\hat{j}}(t)$ becomes zero. Since $W(t)$ contains the indices of the nonzero components of $\hat{x}(t)$, $\hat{j}$ should be moved from $W(t)$ to $W^c(t)$. Conversely, for any $\bar{j} \in W^c(t)$, the inequality (1.10) should be satisfied all the time. As t decreases, there may exist some $\bar{j} \in W^c(t)$ such that the two sides of the inequality becomes equal. In such case, if we do not update $W(t)$ and $W^c(t)$, and continue decreasing t, the inequality (1.10) would not hold. This is because the value of $x_{\bar{j}}(t)$ would turn to nonzero at this t. Hence, $\bar{j}$ should be moved from $W^c(t)$ to $W(t)$.

At each step, l_1-homotopy needs to calculate the matrix-vector multiplications $A^T_{W^c(t)}(A_{W(t)}c)$ and $A^T_{W^c(t)}(A_{W(t)}d)$ solve two linear systems of size $|W(t)|$,

$$\begin{aligned} &\left(A^T_{W(t)}A_{W(t)}\right)c = A^T_{W(t)}b - \lambda \text{sign}\left(\hat{x}_{W(t)}(t)\right), \\ &\left(A^T_{W(t)}A_{W(t)}\right)d = u_{W(t)}, \end{aligned} \tag{1.13}$$

where $|W(t)|$ denotes the number of the elements in $W(t)$. Since at each step, $W(t)$ changes one index, the l_1-homotopy method updates the Cholesky factorization [4, 26] of $A^T_{W(t)}A_{W(t)}$ instead of solving the linear systems from scratch.

It is clear that the efficiency of the l_1-homotopy method relies on the value of $|W(t)|$ and the difference between the nonzero components in the start point $\hat{x}(1)$ and that in the target solution $\hat{x}(0)$. If both $|W(t)|$ and the difference are small, the l_1-homotopy method is very computationally efficient.

However, if $|W(t)|$ is large, solving the linear systems takes much computational cost; otherwise, when $|W^c(t)|$ is large, calculating the matrix-vector multiplications $A^T_{W^c(t)}(A_{W(t)}c)$ and $A^T_{W^c(t)}(A_{W(t)}d)$ takes much computational cost, even more than solving the linear systems. Note that, for most $j \in W^c(t)$, the inequality (1.10) would strictly hold in the whole homotopy tracking steps, but the l_1-homotopy method needs to check them. This implies that much computation is “wasted” when $|W^c(t)|$ is very large.

On the other hand, when $W(1)$ is far away from $W(0)$, many iterations are required. Asif and Romberg [1] used the solution of the previous l_1-minimization problem to warm start the current one since there are a sequence of l_1-minimization problems need to be solved. However, the solution of the previous problem is often not a good warm start. Moreover, for a general l_1-minimization problems, this warm start procedure is inapplicable.

Another problem of the l_1-homotopy method is that a large condition number of $A^T_{W(t)}A_{W(t)}$ and the weak complementarity of the KKT systems may result in an incorrect update of the working set $W(t)$.

Different from the LARs and the l_1-homotopy method, Xiao and Zhang [37] proposed an approximate homotopy continuation method called PGH to solve l_1-minimization. This method approximately solves the solution path $\bar{x}(\lambda)$ at a set of λ by APG method.

For nonconvex regularizers, researchers have also proposed efficient algorithms, to name a few, HTP, FHTP [21], and SHTP [22] for l_0-minimization, adaptive LASSO [42] for l_p-minimization ($0 < p < 1$) and iterative half thresholding algorithm [38, 41] for $l_{1/2}$-minimization. In addition, Esser, Lou, and Xin [18] proposed a difference of convex functions (Yin et al. [39]) algorithm (DCA-l_{1-2}) for l_{1-2}-minimization problem.

As described above, when $|W(t)|$ is large, or the matrices $A_{W(t)}^T A_{W(t)}$ of the linear systems in the homotopy tracking steps are ill-conditioned, the performance of the l_1-homotopy method is unsatisfactory. In addition, when $|W^c(t)|$ is large, the matrix-vector multiplications $A_{W^c(t)}^T(A_{W(t)}c)$ and $A_{W^c(t)}^T(A_{W(t)}d)$ take much computational cost. To avoid checking the condition (1.10) for most indices of $W^c(t)$ which would always strictly satisfy (1.10), we present a decomposition l_1-homotopy algorithm framework for large-scale sparse optimization. In particular, based on a block coordinate minimization method, this framework decomposes the large-scale problem into a sequence of small-scale l_1-minimization problems which is added with a proximal term to ensure strong convexity. Simultaneously, based on a carefully designed block coordinate update strategy, we prove that the iterative points converge to a $\tilde{\tau}$-"precise" solution in a finite number of iterations, and simultaneously, we prove that the value of the objective function monotonically decreases and linearly converges before a $\tilde{\tau}$-"precise" solution is obtained. Moreover, an improved l_1-homotopy method is presented for solving the subproblems. The l_1-homotopy method is improved in two respects. First, since the warm-start would significantly improve the performance of the l_1-homotopy method in many cases, an efficient warm-start procedure which is based on FISTA is designed. Second, to correct the incorrect update of the working set caused by the error from solving the linear systems in the homotopy tracking steps, we design an ε-precision verification-correction technique to ensure the robustness of the tracking steps.

Note that the algorithm framework solves small-scale l_1-minimization subproblems, which means that the verification of the KKT systems (1.10) in the homotopy tracking steps is not "wasted" that much as in the homotopy tracking steps for the original problem, since the $|W^c(t)|$ in the subproblems is much smaller than that in the original problem. This is a great advantage of the algorithm framework which makes it capable of solving very-large-scale problems. On the other hand, the size of the linear systems for the subproblems is much smaller. We use the proximal block coordinate l_1-homotopy (PBC_l_1-Hom) method to denote the above algorithm framework with the improved l_1-homotopy method solving the subproblems.

Simultaneously, we present a PBC minimization based difference of convex functions algorithm (PBCDCA) framework for the l_{1-2}-minimization problem. Similar to PBC_l_1-Hom, every subproblem of PBCDCA is solved by the improved l_1-homotopy method. Hence, we use PBCDCA_l_1-Hom to denote the above algorithm. Also, we give an analogous convergence result as PBC_l_1-Hom for PBCDCA_l_1-Hom.

In both PBC_l_1-Hom and PBCDCA_l_1-Hom, the setting of some parameters is important to the performance. However, different databases adapt to different parameter setting, and it is difficult to know the best parameter setting in advance. Further, the best parameters setting changes as the number of iterations increases. Thus, we present the parameters adjusting technique for PBC_l_1-Hom and PBCDCA_l_1-Hom. Moreover, we present a shrinking technique to reduce the computational cost in selecting the optimized components at each step and an economical solving strategy for the subproblems to make the algorithms perform better.

The rest of this paper is organized as follows. In section 2, we propose the decomposition framework for l_1-minimization problems and present the convergence analysis. In section 3, we extend the framework to solve (1.1). Based on the original l_1-homotopy method, an improved l_1-homotopy method with an efficient warm-start procedure and an ε-precision verification-correction technique is presented in section 4. An efficient implementation with adaptive parameters updates, an economical solving strategy, and a shrinking technique are presented in section 5.

2. Proximal block coordinate minimization.

As mentioned above, the l_1-homotopy method needs to solve two linear systems (1.13) at each step. When $|W(t)|$ is large and the matrix $A_{W(t)}^T A_{W(t)}$ is not well-conditioned, the efficiency is not so satisfactory. Moreover, note that when the solution is very sparse it means most components are in $W^c(t)$. Hence, most of the computation of the matrix-vector multiplications $A_{W^c(t)}^T(A\hat{x}(t))$ in (1.10) are "wasted." For these reasons, we present an algorithm framework that decomposes the original l_1-minimization problem to a sequence of small-scale l_1-minimization problems.

When the solution is sparse, most components are zero. Hence, we do not need to optimize the objective function on all variables. We can reduce the problem (1.1) to a smaller-scale problem

$$\min_{x\in\mathbb{R}^{|B_*|}} \left\{ \frac{1}{2}\|A_{B_*}x - b\|^2 + \lambda(\|x\|_1 - \alpha\|x\|) \right\}, \tag{2.1}$$

where B_* denotes the index set of the nonzero components of the solution of x^* of (1.1). Note that, when the solution is very sparse, the size of (2.1) if much smaller than that of (1.1). However, it is impossible to know B_* in advance. In this section, we take advantage of the sparsity of the solution to decompose the original large-scale problem into a sequence of small-scale problems.

Given an index set $B \subset \{1, 2, \ldots, n\}$, the objective function of the l_1-minimization problem can be decomposed as

$$\begin{aligned} f(x) &= \frac{1}{2}\|Ax - b\|^2 + \lambda\|x\|_1 \\ &= \frac{1}{2}\left\| \begin{bmatrix} A_B & A_{B^c} \end{bmatrix} \begin{bmatrix} x_B \\ x_{B^c} \end{bmatrix} - b \right\|^2 + \lambda\|x_B\|_1 + \lambda\|x_{B^c}\|_1 \\ &= \frac{1}{2}\|A_B x_B - (b - A_{B^c}x_{B^c})\|^2 + \lambda\|x_B\|_1 + \lambda\|x_{B^c}\|_1, \end{aligned} \tag{2.2}$$

where $B^c = \{1, 2, \ldots, n\}\backslash B$, and A_B and A_{B^c} denote the columns of A indexed by B and B^c, respectively. Hence, at each step, we fix the components x_{B^c} and optimize the problem on the other components x_B, that is, we solve the small-scale l_1-minimization problem

$$\min_y \left\{ \frac{1}{2}\left\|A_B y - \left(b - A_{B^c}x_{B^c}^k\right)\right\|^2 + \lambda\|y\|_1 \right\} \tag{2.3}$$

at the kth iteration. Although the size of B can be arbitrary, a large $|B|$ means that solving the subproblems may require solving large-scale linear systems, such as the l_1-homotopy method. It is clear that (2.3) is a block coordinate minimization algorithm, which ensures that the value of the objective function monotonically decreases. Further, to ensure that the subproblems are strictly convex, we add proximal terms to the objective function of the subproblems. That is, we solve a sequence of the PBC minimization subproblems

$$\min_y \left\{ \frac{1}{2} \left\| A_B y - \left(b - A_{B^c} x_{B^c}^k\right) \right\|^2 + \lambda \|y\|_1 + \frac{\delta}{2} \left\| y - x_B^k \right\|^2 \right\}, \tag{2.4}$$

where $\delta > 0$ is the proximal parameter.

Although B can be arbitrarily selected, we hope the value of the objective function would decrease more at each step. Moreover, to ensure convergence of the algorithm, we carefully design an update strategy for B.

For a given tolerance $\tau > 0$ and a size $q > 0$, we select

$$\begin{aligned} \hat{B}_1 &= \left\{ j : |A_j^T(Ax - b) + \lambda \mathrm{sign}(x_j)| > \tau, x_j \neq 0 \right\}, \\ \hat{B}_2 &= \left\{ j : |A_j^T(Ax - b)| > \lambda + \tau, x_j = 0 \right\}. \end{aligned} \tag{2.5}$$

Choose subsets $B_1 \subset \hat{B}_1$ and $B_2 \subset \hat{B}_2$, respectively, such that $|B_1| \leq \frac{q}{2}$, $|B_2| \leq \frac{q}{2}$,

$$\begin{aligned} &\min\nolimits_{j \in B_1} \max\left(|A_j^T(Ax - b)| - \lambda, |x_j|\right) \\ &\quad \geq \max\nolimits_{j \in \hat{B}_1 \setminus B_1} \max\left(|A_j^T(Ax - b)| - \lambda, |x_j|\right) \end{aligned} \tag{2.6}$$

and

$$\begin{aligned} &\min\nolimits_{j \in B_2} \left| A_j^T(Ax - b) \right| \\ &\quad \geq \max\nolimits_{j \in \hat{B}_2 \setminus W_2} \left| A_j^T(Ax - b) \right|. \end{aligned} \tag{2.7}$$

Let $B = B_1 \cup B_2$. Obviously, B contains no more than q components which violate the KKT conditions most. Moreover, B contains the nonzero components which are further away from zero. This condition is important to the following proof of the convergence of the algorithm.

Hence, we have the following PBC minimization algorithm for l_1-minimization problems.

Note that once the nonzero components of the solution of (1.2) are contained in the set B, PBC obtains the solution of (1.2) in one step. This means that PBC just needs to solve several small-scale l_1-minimization problems to obtain a highly accurate solution of (1.2).

2.1. Convergence analysis. In this part, we present the convergence results for the PBC algorithm. Before that, we introduce a lemma.

LEMMA 2.1. *Assume that g^* is a minimum of the following l_1-minimization problem:*

$$\min_g \in \mathbb{R}^q \ \phi(g) := \frac{1}{2} g^T Q g + r^T g + \lambda \|g\|_1, \tag{2.8}$$

where $Q \in \mathbb{R}^{n \times n}$ is symmetric and positive semidefinite. Then for any $g \in \mathbb{R}^q$,

$$\phi(g) - \phi(g^*) \geq \frac{\lambda_{\min}(Q)}{2} \|g - g^*\|^2.$$

Proof. According to the expression of $\phi(g)$, we have

$$\begin{aligned}\phi(g)-\phi(g^*) &= \frac{1}{2}g^TQg + r^Tg + \lambda\|g\|_1 - \left(\frac{1}{2}{g^*}^TQg^* + r^Tg^* + \lambda\|g^*\|_1\right)\\ &= \frac{1}{2}(g^*+g-g^*)^TQ(g^*+g-g^*) + r^T(g^*+g-g^*) + \lambda\|g\|_1\\ &\quad -\left(\frac{1}{2}{g^*}^TQg^* + r^Tg^* + \lambda\|g^*\|_1\right)\\ &= (g-g^*)^T(Qg^*+r) + \lambda\|g\|_1 - \lambda\|g^*\|_1 + \frac{1}{2}(g-g^*)^TQ(g-g^*)\\ &= (g-g^*)^T(Qg^*+r+\lambda\mathrm{sign}(g^*)) + \lambda\|g\|_1 - \lambda\mathrm{sign}(g^*)^Tg\\ &\quad + \frac{1}{2}(g-g^*)^TQ(g-g^*)\\ &\geq \frac{1}{2}(g-g^*)^TQ(g-g^*).\end{aligned} \tag{2.9}$$

The inequality is derived from the KKT conditions

$$\begin{aligned} &Q_j^Tg^* + r_j + \lambda\mathrm{sign}(g_j^*) = 0, \quad g_j^* \neq 0,\\ &|Q_j^Tg^* + r_j| \leq \lambda, \quad g_j^* = 0,\end{aligned} \tag{2.10}$$

and

$$\lambda\|g\|_1 \geq \lambda\mathrm{sign}(g^*)^Tg. \qquad \square$$

THEOREM 2.2. *Let $\{x^k\}$ denote the sequence obtained by Algorithm 2. If $B \neq \emptyset$, then there exists $q^{-\frac{1}{2}} \leq \omega \leq 1$ such that*

$$\left\|x_B^k - x_B^{k+1}\right\| \geq \frac{\omega\tau q^{\frac{1}{2}}}{\tilde{L}+\delta} \tag{2.11}$$

or

(2.12)

$$B_2=\emptyset,\ \|x_B^k\| \leq \|x_B^k - x_B^{k+1}\| < \frac{\tau}{\tilde{L}+\delta} \text{ and } |A_j^T(Ax^k-b)| < \lambda+\tau, \text{ for } j \in B_1,$$

where $\tilde{L} = \max_{k=1,2,3,\ldots}\ \{L_{B^k} := \|A_{B^k}^TA_{B^k}\|\}$ and B^k denotes the set B in the kth iteration of the PBC algorithm.

Algorithm 2. PBC algorithm.

```
Inputs:    $x^0$, $\tau$ (tolerance of the KKT residual), $q$ (the size of $B$), $tol$;
Outputs:   $x^{k+1}$;
    while $\|x^k - x^{k+1}\| > tol * (\|x^k\| + 1e^{-3})$ do
        Select $B_1$ and $B_2$ satisfying (2.5)–(2.7) and let $B = B_1 \cup B_2$
        Solve the $l_1$-minimization subproblem
            $x_B^{k+1} = \arg\min_y \frac{1}{2}\left\|A_By - (b - A_{B^c}x_{B^c}^k)\right\|^2 + \lambda\|y\|_1 + \frac{\delta}{2}\left\|y - x_B^k\right\|^2$
    end while
```

Proof. We have

$$\left|A_j^T\left(Ax^k - b\right) + \lambda \operatorname{sign}\left(x_j^k\right)\right| > \tau \text{ for } j \in B_1$$

and

$$|A_j^T\left(Ax^k - b\right)| > \lambda + \tau \text{ for } j \in B_2 \tag{2.13}$$

from (2.5). Moreover, since x_B^{k+1} is the solution of (2.4), we have

$$\left|A_j^T\left(A_B x_B^{k+1} + A_{B^c} x_{B^c}^k - b\right) + \delta\left(x_j^{k+1} - x_j^k\right)\right| \le \lambda, j \in B, \tag{2.14}$$

from the KKT conditions (1.3) and (1.4).

If $B_2 \neq \emptyset$, combine (2.13) and (2.14), and we deduce that

$$\begin{aligned}\sqrt{|B_2|}\tau &\le \left\|A_{B_2}^T\left(Ax^k - b\right) - A_{B_2}^T\left(A_B x_B^{k+1} + A_{B^c} x_{B^c}^k - b\right) - \delta\left(x_{B_2}^{k+1} - x_{B_2}^k\right)\right\| \\ &= \left\|A_{B_2}^T A_B\left(x_B^k - x_B^{k+1}\right) + \delta\left(x_{B_2}^k - x_{B_2}^{k+1}\right)\right\| \\ &\le \left(\left\|A_{B_2}^T A_B\right\| + \delta\right)\left\|x_{B_2}^k - x_{B_2}^{k+1}\right\| \\ &\le (\tilde{L} + \delta)\left\|x_{B_2}^k - x_{B_2}^{k+1}\right\|.\end{aligned} \tag{2.15}$$

If $B_2 = \emptyset$, since $B = B_1 \cup B_2 \neq \emptyset$, we have $B_1 \neq \emptyset$. Moreover, let $\tilde{B}_1 = \{j : j \in B_1 \text{ and } \operatorname{sign}(x_j^k) = \operatorname{sign}(x_j^{k+1})\}$. When $\tilde{B}_1 \neq \emptyset$, since x_B^{k+1} is the solution of (2.4) which satisfies the KKT conditions (1.3) and (1.4), and for any $j \in B_1$, it satisfies (2.5), we have

$$\begin{aligned}\sqrt{|\tilde{B}_1|}\tau &\le \left\|A_{\tilde{B}_1}^T\left(Ax^k - b\right) + \lambda \operatorname{sign}\left(x_{\tilde{B}_1}^k\right) - \left(A_{\tilde{B}_1}^T\left(A_B x_B^{k+1} + A_{B^c} x_{B^c}^k - b\right)\right.\right. \\ &\quad \left.\left. + \delta\left(x_{\tilde{B}_1}^{k+1} - x_{\tilde{B}_1}^k\right) + \lambda \operatorname{sign}\left(x_{\tilde{B}_1}^{k+1}\right)\right)\right\| \\ &= \left\|A_{\tilde{B}_1}^T\left(A_B\left(x_B^k - x_B^{k+1}\right)\right) + \delta\left(x_{\tilde{B}_1}^k - x_{\tilde{B}_1}^{k+1}\right)\right\| \\ &\le \left(\left\|A_{\tilde{B}_1}^T A_B\right\| + \delta\right)\left\|x_{\tilde{B}_1}^k - x_{\tilde{B}_1}^{k+1}\right\| \\ &\le (\tilde{L} + \delta)\left\|x_{\tilde{B}_1}^k - x_{\tilde{B}_1}^{k+1}\right\|.\end{aligned} \tag{2.16}$$

From (2.15) and (2.16), we have that if $B_2 \cup \tilde{B}_1 \neq \emptyset$, then (2.11) holds with

$$\omega = q^{-\frac{1}{2}}\sqrt{|B_2 \cup \tilde{B}_1|}. \tag{2.17}$$

Otherwise, if $B_2 \cup \tilde{B}_1 = \emptyset$, then $B_1 = B$ and for all $j \in B$, $\operatorname{sign}(x_j^k) \neq \operatorname{sign}(x_j^{k+1})$, hence

$$\left\|x_B^k - x_B^{k+1}\right\| \ge \left\|x_B^k\right\|. \tag{2.18}$$

Moreover, if $\|x_B^k - x_B^{k+1}\| \ge \frac{\tau}{\tilde{L}+\delta}$, we obtain (2.11) with $\omega = q^{-\frac{1}{2}}$; if $\|x_B^k - x_B^{k+1}\| < \frac{\tau}{\tilde{L}+\delta}$, then

$$
\begin{aligned}
\left|A_j^T\left(Ax^k-b\right)\right| &= \left|A_j^T\left(A_Bx_B^{k+1}+A_{B^c}x_{B^c}^k-b\right)-A_j^TA_B\left(x_B^{k+1}-x_B^k\right)\right| \\
&\le \left|A_j^T\left(A_Bx_B^{k+1}+A_{B^c}x_{B^c}^k-b\right)+\delta\left(x_j^{k+1}-x_j^k\right)\right| \\
&\quad +\left|A_j^TA_B\left(x_B^k-x_B^{k+1}\right)+\delta\left(x_j^k-x_j^{k+1}\right)\right| \\
&\le \lambda+\left\|A^TjA_B\right\|\left\|x_B^k-x_B^{k+1}\right\|+\delta\left\|x_B^k-x_B^{k+1}\right\| \\
&< \lambda+\tau,
\end{aligned}
\tag{2.19}
$$

where the last inequality is derived from (2.14). ∎

Before the next theorem, we define x as a τ-"precise" solution of (1.2) if it satisfies

$$
\begin{aligned}
&\left|A_j^T(Ax-b)+\lambda\mathrm{sign}(x_j)\right|\le\tau && \text{for } |x_j|\ge\tau,\\
&\left|A_j^T(Ax-b)\right|\le\lambda+\tau && \text{for } |x_j|<\tau.
\end{aligned}
\tag{2.20}
$$

Note that at the early iterations of the PBC algorithm, it is easy to find $\frac{q}{2}$ indices that satisfy the second expression of (2.5), i.e., B_2 would have $\frac{q}{2}$ elements. On the other hand, since we chose B_1 according to (2.5) and (2.6), B_1 contains the nonzero components which violate the KKT conditions most and are further away from zero. Hence, at the early iterations of PBC algorithm, $|B_2\cup\tilde{B}_1|$ would be close to q, that is, ω would be close to 1. When $|B_2\cup\tilde{B}_1|$ is small, it means that only a small number of the components of x^k would violate the KKT conditions.

Next, we show that the PBC algorithm converges to a $\tilde{\tau}$-"precise" solution in finite number of iterations, where $\tilde{\tau}=\max(\tau,\frac{\tau}{\tilde{L}+\delta})$.

THEOREM 2.3. *Let $\{x^k\}$ denote the sequence obtained by Algorithm 2; then we have the following:*

(i) $\|x^k-x^{k+1}\|\to 0$.

(ii) *For any $k\ge N=\frac{2f(x^0)(\tilde{L}+\delta)^2}{\delta\omega^2\tau^2q}$, x^k is a $\tilde{\tau}$-"precise" solution of* (1.2).

(iii) *Before x^k converges to a $\tilde{\tau}$-"precise" solution, the value of the objective function decrease with a linear convergence as*

$$
f\left(x^{k+1}\right)-f(x^*)\le\left(1-\frac{\delta\omega^2\tau^2q}{2(\tilde{L}+\delta)^2f(x^0)}\right)\left(f\left(x^k\right)-f(x^*)\right).
\tag{2.21}
$$

Proof. From Theorem 2.2, we deduce that

$$
\begin{aligned}
\infty>f\left(x^0\right)-f\left(x^{k+1}\right) &= \sum_{i=0}^{k}\left(f\left(x^i\right)-f\left(x^{i+1}\right)\right)\\
&\ge \sum_{i=0}^{k}\frac{1}{2}\left(\lambda_{\min}\left(A_B^TA_B\right)+\delta\right)\left\|x_B^i-x_B^{i+1}\right\|^2\\
&\ge \sum_{i=0}^{k}\frac{1}{2}\left(\lambda_{\min}\left(A^TA\right)+\delta\right)\left\|x^i-x^{i+1}\right\|^2,
\end{aligned}
\tag{2.22}
$$

which implies $\|x^k-x^{k+1}\|\to 0$.

For item (ii), when $B=\emptyset$, it is clear that x^k is a τ-"precise" solution.

When $B_2\ne\emptyset$ or there exists $j\in B_1$ such that $\mathrm{sign}(x_j^k)=\mathrm{sign}(x_j^{k+1})$, we have $\|x_B^k-x_B^{k+1}\|\ge\frac{\omega\tau q^{\frac{1}{2}}}{\tilde{L}+\delta}$, then

$$\|f(x^k) - f(x^{k+1})\| \geq \frac{(\lambda_{\min}(A_B^T A_B) + \delta)\omega^2\tau^2 q}{2(\tilde{L}+\delta)^2} \geq \frac{\delta\omega^2\tau^2 q}{2(\tilde{L}+\delta)^2} \tag{2.23}$$

from (2.22). Since $f(x) \geq 0$ for any $x \in R^n$, let

$$N = \frac{2f(x^0)(\tilde{L}+\delta)^2}{\delta\omega^2\tau^2 q}; \tag{2.24}$$

hence, for any $k > N$, (2.12) holds, and then we have $\hat{B}_2 = \emptyset$ from (2.7), that is,

$$|A_j^T(Ax^k - b)| \leq \lambda + \tau \text{ for any } x_j^k = 0.$$

Moreover, for all $j \in B_1$, (2.12) holds. Furthermore, since we choose B_1 such that it satisfies (2.6), we have

$$\max_{j\in\hat{B}_1}\{\max(|x_j^k|, |A_j^T(Ax^k - b)| - \lambda)\} = \max_{j\in B_1}\{\max(|x_j^k|, |A_j^T(Ax^k - b)| - \lambda)\} \leq \tilde{\tau},$$

which implies that x^k is a $\tilde{\tau}$-"precise" solution.

Moreover

$$\begin{aligned} f(x^{k+1}) - f(x^*) &\leq f(x^k) - f(x^*) - \frac{\delta\omega^2\tau^2 q}{2(\tilde{L}+\delta)^2} \\ &\leq \left(1 - \frac{\delta\omega^2\tau^2 q}{2(\tilde{L}+\delta)^2(f(x^k) - f(x^*))}\right)(f(x^k) - f(x^*)) \\ &\leq \left(1 - \frac{\delta\omega^2\tau^2 q}{2(\tilde{L}+\delta)^2 f(x^0)}\right)(f(x^k) - f(x^*)). \end{aligned} \tag{2.25}$$

□

From the convergence analysis, we see that the carefully designed update strategy (2.6) for B is important to the convergence. In fact, at the early iterations of the PBC algorithm, $|A_j^T(Ax-b)| - \lambda$ is often larger than $|x_j|$. From the proof of Theorems 2.2 and 2.3, a larger $|A_j^T(Ax^k-b)| - \lambda$ can make the objective function more. At the latter iterations, $|A_j^T(Ax-b)| - \lambda$ becomes small, and in such case, the weak complementarity of KKT conditions may result in that the value of the nonzero components of x^* is close to zero. Since $|A_j^T(Ax-b) + \lambda\text{sign}(x)|$ is not continuous at 0, the value of $|A_j^T(Ax-b) + \lambda\text{sign}(x)|$ would change a lot with a small perturbation. Hence, it is not suitable to choose j that has maximum $|A_j^T(Ax-b) + \lambda\text{sign}(x)|$, especially for the complementarity condition.

3. PBCDCA minimization.

In this section, we present an analogous algorithm framework to PBC, the PBCDCA minimization algorithm for the l_{1-2}-minimization problem (1.1).

The DCA-l_{1-2} method solves the l_{1-2}-minimization problem by iteratively solving the following DCA l_1-minimization subproblems:

$$\begin{cases} x^{k+1} = \arg\min_{x\in\mathbb{R}^n} \frac{1}{2}\|Ax-b\|^2 + \lambda\|x\|_1 & \text{if } x^k = 0; \\ x^{k+1} = \arg\min_{x\in\mathbb{R}^n} \frac{1}{2}\|Ax-b\|^2 + \lambda\|x\|_1 - \left\langle x, \lambda\alpha\frac{x^k}{\|x^k\|}\right\rangle & \text{otherwise.} \end{cases} \tag{3.1}$$

Different from (3.1), we decompose the l_{1-2}-minimization problem into a sequence of small-scale DCA l_1-minimization subproblems as

$$\min_y \frac{1}{2}\left\|A_B y - \left(b - A_{B^c} x^k_{B^c}\right)\right\|^2 - \left\langle y, \lambda\alpha \frac{x^k_B}{\|x^k\|}\right\rangle + \lambda\|y\|_1 + \frac{\delta}{2}\left\|y - x^k_B\right\|^2, \tag{3.2}$$

where $B = B_1 \cup B_2$. B_1 and B_2 are selected based on the following procedures.

For a given tolerance $\tau > 0$ and a size $q > 0$, when $\|x\| \neq 0$, similar to (2.5) and (2.6), we select

$$\begin{aligned} \hat{B}_1 &= \left\{ j : \left| A_j^T(Ax-b) - \lambda\alpha\frac{x_j}{\|x\|} + \lambda \text{sign}(x_j)\right| > \tau, x_j \neq 0 \right\}, \\ \hat{B}_2 &= \left\{ j : \left| A_j^T(Ax-b) - \lambda\alpha\frac{x_j}{\|x\|}\right| > \lambda + \tau, x_j = 0 \right\}. \end{aligned} \tag{3.3}$$

Choose subsets $B_1 \subset \hat{B}_1$ and $B_2 \subset \hat{B}_2$, respectively, such that $|B_1| \leq \frac{q}{2}$, $|B_2| \leq \frac{q}{2}$,

$$\begin{aligned} &\min_{j\in B_1} \max\left(\left| A_j^T(Ax-b) - \lambda\alpha\frac{x_j}{\|x\|}\right| - \lambda, |x_j| \right) \\ &\quad \geq \max_{j\in \hat{B}_1\setminus B_1} \max\left(\left| A_j^T(Ax-b) - \lambda\alpha\frac{x_j}{\|x\|}\right| - \lambda, |x_j| \right) \end{aligned} \tag{3.4}$$

and

$$\begin{aligned} &\min_{j\in B_2} \left| A_j^T(Ax-b) - \lambda\alpha\frac{x_j}{\|x\|}\right| \\ &\quad \geq \max_{j\in \hat{B}_2\setminus B_2} \left| A_j^T(Ax-b) - \lambda\alpha\frac{x_j}{\|x\|}\right|. \end{aligned} \tag{3.5}$$

When $\|x\| = 0$, the selection procedure of B_1 and B_2 is the same as (2.5) and (2.6).

Algorithm 3. PBCDCA algorithm.

```
Inputs:   $x^0$, $\tau$ (tolerance of the KKT residual), $q$ (the size of $B$), $tol$;
Outputs:  $x^{k+1}$;
  while $\|x^k - x^{k+1}\| > tol * (\|x^k\| + 1e^{-3})$ do
    if $\|x^k\| = 0$ then
      Select $B_1$ and $B_2$ satisfying (2.5)–(2.7) and let $B = B_1 \cup B_2$
      Solve the $l_1$-minimization subproblem
      (3.6)  $x^{k+1}_B = \arg\min_y \frac{1}{2}\|A_B y - (b - A_{B^c} x^k_{B^c})\|^2 + \lambda\|y\|_1 + \frac{\delta}{2}\|y - x^k_B\|^2$
    else
      Select $B_1$ and $B_2$ satisfying (3.3)–(3.5) and let $B = B_1 \cup B_2$
      Solve the DCA $l_1$-minimization subproblem
      (3.7)  $x^{k+1}_B = \arg\min_y \frac{1}{2}\|A_B y - (b - A_{B^c} x^k_{B^c})\|^2 - \langle y, \lambda\alpha \frac{x^k_B}{\|x^k\|}\rangle + \lambda\|y\|_1$
                 $+ \frac{\delta}{2}\|y - x^k_B\|^2$
    end if
  end while
```

Let $\hat{f}_\alpha(x) = \frac{1}{2}\|Ax - b\|^2 + \lambda(\|x\|_1 - \alpha\|x\|)$. Similar to (2.20), we define x as a τ-"precise" KKT point of (1.1) if it satisfies

$$\begin{aligned} &\left|A_j^T(Ax-b) - \lambda\alpha\frac{x_j}{\|x\|} + \lambda \mathrm{sign}(x_j)\right| \le \tau \qquad \text{for } |x_j| \ge \tau,\\ &\left|A_j^T(Ax-b) - \lambda\alpha\frac{x_j}{\|x\|}\right| \le \lambda + \tau \qquad \text{for } |x_j| < \tau. \end{aligned} \tag{3.8}$$

Theorem 3.1. *Let $\{x^k\}$ denote the sequence obtained by Algorithm 3; then we have the following:*

(i) *$\hat{f}(x^k) - \hat{f}(x^{k+1}) \ge 0$ and $\|x^k - x^{k+1}\| \to 0$.*

(ii) *For any $k \ge N = \frac{2\hat{f}(x^0)(\tilde{L}+\delta)^2}{\delta\omega^2\tau^2 q}$, x^k is a $\tilde{\tau}$-"precise" KKT point of (1.1), where $\tilde{\tau} = \max(\tau, \frac{\tau}{\tilde{L}+\delta})$, $\tilde{L} = \max_{k=1,2,3,\dots}\ \{L_{B^k} := \|A_{B^k}^T A_{B^k}\|\}$, and B^k denotes the set B in the kth iteration of PBCDCA algorithm.*

(iii) *Before x^k converges to a $\tilde{\tau}$-"precise" solution, the value of the objective function decrease with a linear convergence as*

$$\hat{f}(x^{k+1}) - \hat{f}(x^*) \le \left(1 - \frac{\delta\omega^2\tau^2 q}{2(\tilde{L}+\delta)^2\hat{f}(x^0)}\right)\left(\hat{f}(x^k) - \hat{f}(x^*)\right). \tag{3.9}$$

Proof. Since $g(y) = \lambda\alpha\sqrt{\|x_{B^c}^k\|^2 + \|y\|^2}$ is convex, for any $y \in \mathbb{R}^q$,

$$\begin{aligned} \lambda\alpha\|x^{k+1}\| = g(x_B^{k+1}) &\ge g(x_B^k) + \langle \nabla g(x_B^k), x_B^{k+1} - x_B^k\rangle\\ &= \lambda\alpha\|x^k\| + \left\langle \lambda\alpha\frac{x_B^k}{\|x^k\|}, x_B^{k+1} - x_B^k\right\rangle\\ &= \left\langle \lambda\alpha\frac{x_B^k}{\|x^k\|}, x_B^{k+1}\right\rangle + \left\langle x_{B^c}^k, \lambda\alpha\frac{x_{B^c}^k}{\|x^k\|}\right\rangle. \end{aligned} \tag{3.10}$$

Thus,

$$\begin{aligned} \hat{f}_\alpha(x^k) &= \frac{1}{2}\left\|Ax^k - b\right\|^2 + \lambda\left(\left\|x^k\right\|_1 - \alpha\left\|x^k\right\|\right)\\ &\ge \frac{1}{2}\left\|A_B x_B^{k+1} - (b - A_{B^c}x_{B^c}^k)\right\|^2 + \lambda\left\|x_B^{k+1}\right\|_1 + \lambda\left\|x_{B^c}^k\right\|_1\\ &\quad - \left\langle x_B^{k+1}, \lambda\alpha\frac{x_B^k}{\|x^k\|}\right\rangle - \left\langle x_{B^c}^k, \lambda\alpha\frac{x_{B^c}^k}{\|x^k\|}\right\rangle + \frac{\delta}{2}\left\|x_B^{k+1} - x_B^k\right\|^2\\ &= \hat{f}_\alpha(x^{k+1}) + \frac{\delta}{2}\left\|x_B^{k+1} - x_B^k\right\|^2\\ &\ge \hat{f}_\alpha(x^{k+1}). \end{aligned} \tag{3.11}$$

Moreover, since $\hat{f}_\alpha(x^k) \ge \hat{f}_\alpha(x^{k+1}) + \frac{\delta}{2}\|x_B^{k+1} - x_B^k\|^2$ and $\hat{f}_\alpha(x)$ is bounded below, we have $\|x^k - x^{k+1}\| \to 0$.

For item (ii), the proof is almost the same as Theorem 2.3(ii). When $B = \emptyset$, x^k is a τ-"precise" KKT point.

When $B_2 \neq \emptyset$ or there exists $j \in B_1$ such that $\mathrm{sign}(x_j^k) = \mathrm{sign}(x_j^{k+1})$, we have $\|x_B^k - x_B^{k+1}\| \geq \frac{\omega \tau q^{\frac{1}{2}}}{\tilde{L}+\delta}$; then

$$\|\hat{f}_\alpha(x^k) - \hat{f}_\alpha(x^{k+1})\| \geq \frac{(\lambda_{\min}(A_B^T A_B) + \delta)\omega^2\tau^2 q}{2(\tilde{L}+\delta)^2} \geq \frac{\delta\omega^2\tau^2 q}{2(\tilde{L}+\delta)^2}. \tag{3.12}$$

Since $\hat{f}(x) \geq 0$ for any $x \in R^n$, let

$$N = \frac{2\hat{f}(x^0)(\tilde{L}+\delta)^2}{\delta\omega^2\tau^2 q}; \tag{3.13}$$

hence, for any $k > N$, $\hat{B}_2 = \emptyset$, that is,

$$\left|A_j^T(Ax^k - b)\right| \leq \lambda + \tau \text{ for any } x_j^k = 0.$$

Moreover, similar to (2.12), we have

$$\begin{gathered} \|x_{B_1}^k\| \leq \|x_{B_1}^k - x_{B_1}^{k+1}\| < \frac{\tau}{\tilde{L}+\delta}, \\ \left| A_j^T(Ax^k - b) - \lambda\alpha \frac{x_j^k}{\|x^k\|} \right| < \lambda + \tau \text{ for } j \in B_1. \end{gathered} \tag{3.14}$$

Furthermore, since we choose B_1 such that it satisfies (3.5), we have

$$\begin{aligned} &\max_{j \in \hat{B}_1} \left\{ \max\left(|x_j^k|, \left| A_j^T(Ax^k - b) - \lambda\alpha\frac{x_j^k}{\|x^k\|} \right| - \lambda \right) \right\} \\ &\quad = \max_{j \in B_1} \left\{ \max\left(|x_j^k|, \left| A_j^T(Ax^k - b) - \lambda\alpha\frac{x_j^k}{\|x^k\|} \right| - \lambda \right) \right\} \leq \tilde{\tau}, \end{aligned}$$

which implies that x^k is a $\tilde{\tau}$-"precise" KKT point.

Moreover

$$\begin{aligned} \hat{f}(x^{k+1}) - \hat{f}(x^*) &\leq \hat{f}(x^k) - \hat{f}(x^*) - \frac{\delta\omega^2\tau^2 q}{2(\tilde{L}+\delta)^2} \\ &\leq \left(1 - \frac{\delta\omega^2\tau^2 q}{2(\tilde{L}+\delta)^2(\hat{f}(x^k) - \hat{f}(x^*))}\right)(\hat{f}(x^k) - \hat{f}(x^*)) \\ &\leq \left(1 - \frac{\delta\omega^2\tau^2 q}{2(\tilde{L}+\delta)^2\hat{f}(x^0)}\right)(\hat{f}(x^k) - \hat{f}(x^*)). \end{aligned} \tag{3.15}$$

□

4. Improved l_1-homotopy method.

In both Algorithms 2 and 3, a sequence of small-scale l_1-minimization subproblems needs to be solved. Benefiting from the proximal terms, these subproblems are strictly convex. For the sake of convenience, we address the subproblems in a uniform formula as

$$\min_{y \in \mathbb{R}^q} \left\{ \frac{1}{2} y^T H y + s^T y + \lambda \|y\|_1 \right\}, \tag{4.1}$$

where $H = A_B^T A_B + \delta I_q$ is positive definite. Here, I_q denotes the identity matrix of dimension $q = |B|$. Obviously, $s = A_B^T(Ax^k - b)$ in Algorithm 2 and $s = A_B^T(Ax^k - b) - \lambda\alpha \frac{x_B^k}{\|x^k\|}$ in Algorithm 3, respectively.

The l_1-homotopy method solves (4.1) by constructing a parametric programming problem

$$\min_y \left\{ \frac{1}{2} y^T H y + s^T y + \lambda \|y\|_1 + t u^T y \right\} \tag{4.2}$$

and tracking its piecewise-linear solution path $y^*(t)$ which satisfies

$$\begin{aligned} y^*_{W(t)}(t) &= H^{-1}_{W(t)W(t)}(-s_{W(t)} - t u_{W(t)} - \lambda \text{sign}(y^*_{W(t)}(t))), \\ y^*_{W^c(t)}(t) &= 0 \end{aligned} \tag{4.3}$$

from $t = 1$ to $t = 0$ to obtain the solution of (4.1), where $H_{W(t)W(t)}$ denotes a submatrix of H defined by taking rows and columns indexed by $W(t)$. However, when we follow Asif and Romberg [1] using the prior knowledge x_B^k and constructing u such that x_B^k is the solution of (4.2) at $t = 1$, the performance is unsatisfactory because x_B^k is often not a good approximation to x_B^{k+1}, especially in the initial few iterations for both Algorithms 2 and 3. Thus, a warm-start procedure for a better approximation of x_B^{k+1} is necessary. Moreover, a trade-off between the computational cost of the warm-start and the homotopy tracking steps should be taken into consideration. Also, for a given approximate solution, how to select the working set before the homotopy tracking steps is also important to the performance of the l_1-homotopy method.

Although the proximal terms make the KKT systems better-conditioned, it is difficult to completely avoid the incorrect update of the working set in the homotopy tracking steps. To deal with this problem, we present an ε-precision verification-correction technique.

4.1. Warm start. Among the first-order methods for l_1-minimization problems, FISTA [2] is famous for its easy implementation and a fast convergence rate $O(\frac{L_H}{k^2})$, where $L_H \geq \lambda_{\max}(H)$. For a given initial point $y^0 \in \mathbb{R}^q$, FISTA iterates as follows:

```
Step 0 :   Take $v^1 = y^0, \theta_1 = 1$.
Step 1 :   For $k = 1, 2 \ldots,$ do
               $y^k = P_{L_H}(v^k)$,
               $\theta_{k+1} = \frac{1 + \sqrt{1 + 4\theta_k^2}}{2}$,
               $v^{k+1} = y^k + \frac{\theta_k - 1}{\theta_{k+1}} y^{k-1}$,
```

where

$$P_{L_H}(v^k) := \arg\min_y \left\{ \langle y - v^k, Hv^k + s \rangle + \frac{L_H}{2} \|y - v^k\|^2 + \lambda \|y\|_1 \right\}$$

is the shrinkage-thresholding operator.

The main computational cost of FISTA comprises the matrix-vector multiplications Hv^k. In addition, the value of L_H is important to the performance of FISTA.

When the size of H and L_H is small, FISTA is very efficient to obtain an approximate solution. However, when the size of H is fairly large, calculating the matrix-vector multiplication is computationally expensive. What's worse, a large L_H means many more iterations though an adaptive L_H technique [2] is applied. Fortunately, based on the decomposition framework, the size of the subproblems as well as L_H in Algorithms 2 and 3 is small. For these reasons, we adapt FISTA to do warm start for the l_1-homotopy method. Further, since FISTA converges with a rate of $O(\frac{L_H}{k^2})$, which means that it makes small progress when k is large, we terminate FISTA when the following criteria are satisfied:

$$\begin{aligned}&\pi_{\varepsilon_1}(y^k) = \pi_{\varepsilon_1}(y^{k-i}) \text{ for } i = 1, \dots, S_{\max},\\ &\frac{\|y^k - y^{k-1}\|}{\|y^k\|} < \varepsilon_1,\end{aligned} \tag{4.4}$$

where $\pi_{\varepsilon_1}(y) = |\{j | l_j + \|y\|\varepsilon_1 < |y_j|\}|$ and $\varepsilon_1 > 0$ is a given tolerance.

Let $\hat{y}$ denote the approximate solution obtained by FISTA. Since $\hat{y}$ is an approximation of $\tilde{y}$ which denotes the solution of (4.1), if $\hat{y}_j$ is close to zero, $\tilde{y}_j$ is likely to be 0; if $|\hat{y}_j|$ is far from zero, $\tilde{y}_j$ is likely to be nonzero. Moreover, from the KKT conditions (1.3), we have that $\tilde{y}_j$ is likely to be nonzero if $|H_j^T\hat{y} + s_j)|$ is much larger than λ.

For the above reasons, let

$$\bar{y} = \begin{cases}\hat{y}_j, & |\hat{y}_j| \geq \eta\|\hat{y}\|;\\ -\eta\|\hat{y}\| * \text{sign}(H_j^T\hat{y} + s_j), & |\hat{y}_j| < \eta\|\hat{y}\| \text{ and } |H_j^T\hat{y} + s_j)| \geq \lambda + \eta;\\ 0 & \text{else,}\end{cases} \tag{4.5}$$

where $\eta > 0$ is given, H_j denotes the jth column of H.

Moreover, let

$$u = \begin{cases}-H_j^T\bar{y} - s_j - \lambda\text{sign}(\bar{y}_j), & \bar{y}_j \neq 0,\\ -H_j^T\bar{y} - s_j + \zeta_j, & \bar{y}_j = 0,\end{cases} \tag{4.6}$$

where

$$\zeta_j = \frac{\lambda\big(H_j^T\bar{y} + s_j\big)}{2\big(\max_j \big|H_j^T\bar{y} + s_j\big| + 1\big)}.$$

Then we obtain a parametric programming (4.2). By tracking its piecewise-linear solution path (4.3) from $t = 1$ to $t = 0$, we can obtain the solution of the subproblem (4.1).

However, when $H_{W(t)W(t)}$ is not well-conditioned, the error from solving the linear systems may result in an incorrect update of $W(t)$ and $W^c(t)$, which may lead to breaking down of the l_1-homotopy method. Although the proximal terms make $H_{W(t)W(t)}$ better-conditioned, the incorrect update may occur. To address this issue, we present an ε-precision verification-correction technique to correct the incorrect update.

4.2. ε-precision verification-correction. Due to the error from the solving of the linear systems, it is difficult to ensure that the working set would always be updated correctly, especially when the linear systems are ill-conditioned or the complementary condition for (4.2) is not strictly satisfied. Fortunately, it is not essential to ensure that the KKT systems strictly hold. We can relax the KKT conditions as

$$\begin{aligned} &\left|H_j y^*(t) + s_j + tu_j + \lambda \mathrm{sign}\big(y_j^*(t)\big)\right| \le \varepsilon, && j \in W(t), \\ &\left|H_j y^*(t) + s_j + tu_j\right| \le \lambda + \varepsilon, && j \in W^c(t), \end{aligned} \tag{4.7}$$

in the homotopy tracking steps. Obviously, the ε-precision KKT conditions can tolerate the error from solving the linear systems. Hence, the homotopy tracking steps would not break down though some incorrect update of the working set may occur.

To make the solution path $y^*(t)$ satisfy (4.7) in the homotopy tracking steps, we present the following ε-precision verification-correction procedure:

```
Step 0 :   Let $J_1 = \{j \mid j \in W(t)$ such that $|H_j y^*(t) + s_j + tu_j + \lambda \mathrm{sign}(y_j^*(t))| > \varepsilon\}$.
Step 1 :   Let $J_2 = \{j \mid j \in W^c(t)$ such that $|H_j y^*(t) + s_j + tu_j| > \lambda + \varepsilon\}$.
Step 2 :   **If** $J_1 \cup J_2 = \emptyset$
               **break**.
Step 3 :   Let $\bar{j} = \max_{j \in J_1 \cup J_2} j$.
Step 4 :   **If** $\bar{j} \in J_1$
               let $W(t) = W(t) \backslash \bar{j}, W^c(t) = W^c(t) \cup \bar{j}$;
           **elseif** $\bar{j} \in J_2$
               let $W(t) = W(t) \cup \bar{j}, W^c(t) = W^c(t) \backslash \bar{j}$;
           recompute $y^*(t)$ like (4.3).
Step 5 :   **Go to Step 0**.
```

The above correction procedure is a single principal pivoting algorithm [29] which converges to the solution in a finite number of steps [27]. The relaxation of the KKT makes the l_1-homotopy tolerate the errors caused by ill-conditions and the weak complementarity of the KKT systems.

5. An efficient implementation.

In the implementation of both the PBC_l_1-Hom and PBCDCA_l_1-Hom algorithms, some important details are worth discussing. For the sake of brevity, we discuss only the implementation of PBC_l_1-Hom, for that of the PBCDCA_l_1-Hom algorithm is the same.

• Adaptive parameters. In the PBC_l_1-Hom algorithm, the parameter ε_1 in (4.4) and η in (4.5) are important to the performance of the improved l_1-homotopy method. We present the following strategies to update the value of ε_1 and η adaptively.

The parameter ε_1 trades off the computational work of FISTA and that of the l_1-homotopy tracking steps. If ε_1 is too small, a considerable amount of calculation would be spent on FISTA, and it is difficult to make much progress when the iterative point is close to one of the solutions. However, if ε_1 is too big, the warm-start procedure has little improvement in decreasing the number of steps of the l_1-homotopy method. Fortunately, during the outer iterations of PBC_l_1-Hom, we can monitor the time of FISTA and the l_1-homotopy method at each iteration. Based on the feedback, we adjust the value of ε_1 as the following procedures.

Let t_f^k denote the computation time of FISTA at the kth iteration of the PBC algorithm, and let t_h^k denote the computation time of the homotopy tracking steps. Moreover, let ε_1^k denote the value of ε_1 at the kth iteration:

(i) if $t_h^{k+\kappa} > 3t_f^k$, for $\kappa = 1, 2, \ldots, \varkappa$, set $\varepsilon_1^{k+\varkappa+1} = (1-\rho)\varepsilon_1^k$;

(ii) if $t_f^{k+\kappa} > 3t_h^k$, for $\kappa = 1, 2, \ldots, \varkappa$, set $\varepsilon_1^{k+\varkappa+1} = (1+\rho)\varepsilon_1^k$,

where κ is a positive integer and $0 < \rho < 1$.

The parameter η in (4.5) affects the performance of PBC by affecting the number of the adding and removing steps in Algorithm 1. A proper η should balance the number of the adding and removing steps. Similar to the adaptive update of ε_1, we update η as follows.

Let Λ_a^k, Λ_r^k denote the number of the adding and removing steps, respectively. η^k denotes the value of η at kth iteration; then

(a) if $\Lambda_a^{k+\kappa} > 2\Lambda_r^k$, for $\kappa = 1, 2, \ldots, \varkappa$, set $\eta^{k+\varkappa+1} = (1-\rho)\eta^k$;

(b) if $\Lambda_r^{k+\kappa} > 2\Lambda_a^k$, for $\kappa = 1, 2, \ldots, \varkappa$, set $\eta^{k+\varkappa+1} = (1+\rho)\eta^k$.

In our algorithms, the size of B denoted by q is not a constant. At the beginning of our algorithms, we set it to a small number; when the elements of $\hat{B}_1$ and $\hat{B}_2$ tend to be stable, we increase the value of q till it reaches the number of the nonzero components in x^k or a given maximum number.

• Economical solving strategy for the subproblems. The convergence results of the PBC algorithm are established on the assumption that a sequence of subproblems is solved exactly. In practice, the initial few subproblems do not need high-precision solutions. The low-precision solutions can also make much progress on the decline of the value of the objective function. Hence, we apply FISTA to obtain low precision for the initial few iterations, since FISTA is very efficient to obtain an approximate solution. However, FISTA is difficult to use to obtain high-precision solutions for the subproblems. Hence, when the value of the objective function decreases small at one iteration, we use the improved l_1-homotopy method to solve the remaining subproblems.

• Shrinking. In the PBC algorithm, to select B, we need to compute $A^T * (A * x^k - b)$. $A * x$ can be fast updated as

$$Ax^k = Ax^{k-1} + A_B\left(x_B^k - x_B^{k-1}\right).$$

However, calculating the whole components of $A^T * (Ax^k - b)$ takes much computational cost when the size of A is large. Fortunately, note that most j with $x_j^k = 0$ would no longer satisfy the second formula of (2.5) when x^k is close to the solution which is generally sparse. Hence, we can selectively compute a part of components of $A^T * (Ax^k - b)$.

Without loss of generality, at the kth iteration of PBC algorithm, we compute the whole components of $A^T * (Ax^k - b)$, and let

$$\mathcal{S} = \left\{j | \mathrm{abs}\left(A_j^T * \left(Ax^k - b\right)\right) > \lambda, x_j^k = 0\right\}. \tag{5.1}$$

Then at the $(k+i)$th iteration, $i = 1, \ldots, a$, we only compute $A_j^T * (Ax^{k+i} - b)$ for $j \in \mathcal{S} \cup \{j | x_j^{k+i} \neq 0\}$. Since the solution is generally sparse and $\mathcal{S}$ would be smaller as x^k tends closer to the solution, $|\mathcal{S} \cup \{j | x_j^{k+i} \neq 0\}|$ would be much smaller than n. Moreover, we compute the whole $A^T * (Ax^{k+a+1} - b)$ at the $(k+a+1)$th iteration again to ensure that all components would satisfy the KKT conditions at the end.

• Termination criteria. We give two alternative criteria for the PBC algorithm. The first one is that if x^k satisfies the τ-"precise" KKT condition (when $B = \emptyset$), we terminate the algorithm.

The second one is based on the convergence analysis for PBC. From Theorem 2.3, we have $\|x^k - x^{k+1}\| < \frac{\tau}{\bar{L}+\delta}$ only if x^k is the $\tilde{\tau}$-"precise" solution. So we terminate PBC algorithm when $\|x^k - x^{k+1}\|$ is smaller than a given tolerance.

6. Numerical experiments.

In this section, we shall evaluate the performance of PBC_l_1-Hom and PBCDCA_l_1-Hom for large-scale sparse least squares problems.

Table 6.1
A subset of the LIBSVM data sets, "-sub" denotes a part of the samples were randomly selected from the corresponding database. "density" denotes the ratio of the nonzero elements of A; $\lambda_{\max}(A^TA)$ denotes the maximum eigenvalue of A^TA.

Problem	n	m	$\lambda_{\max}(A^TA)$	Density
avazu-app	14,596,137	1,000,000	3.11E+06	1.50E-05
covtype	581,012	54	1.17E+13	0.221
epsilon.t	100,000	2000	3.50E+04	1.000
HIGGS	11,000,000	28	1.95E+08	0.921
kdda	8,407,552	20,216,830	7.68E+03	1.82E-06
kddb-sub	100,000	29,890,095	7.67E+04	1.00E-06
kddb	19,264,097	29,890,095	1.32E+07	1.00E-06
news20	19,996	1,355,191	1.17E+03	3.36E-04
rcv1_test-sub	10,000	47,236	2.60E+02	0.002
rcv1_test	677,399	47,236	1.65E+04	0.002
SUSY	5,000,000	18	6.55E+07	0.982
url	2,396,130	3,231,961	1.64E+08	3.57E-05
webspam	350,000	16,609,143	2.99E+05	2.23E-04

The numerical experiments were performed on a 3.40 GHz Intel Core i7-6700 PC with 32.0 GB of RAM under the MATLAB (R2015b) environment.

We conducted the numerical experiments in two parts. First, we tested PBC_l_1-Hom on solving large-scale l_1-minimization problems and the databases were drawn from the LIBSVM data sets [12] as shown in Table 6.1. The dimensions of the chosen databases were up to 19,264,097×29,890,095. Second, we tested PBCDCA_l_1-Hom on solving l_{1-2}-minimization. The data sets were randomly generated as in Yin et al. [39], which comprises incoherent and highly coherent matrices.

We use the KKT residual estimation $\|\psi\|$ to measure the precision of an approximate solution of the l_1-minimization ($\alpha = 0$) and l_{1-2}-minimization problems, where

$$\psi_j = \begin{cases} A_j^T(Ax-b) - \lambda\alpha\dfrac{x_j}{\|x\|} + \lambda\mathrm{sign}(x_j) & \text{for } x_j \neq 0, \\ \max\left(0, \left|A_j^T(Ax-b) - \lambda\alpha\dfrac{x_j}{\|x\|}\right| - \lambda\right) & \text{for } x_j = 0. \end{cases} \tag{6.1}$$

Moreover, we set the regularized parameter λ as

$$\lambda = \lambda_c\|A^Tb\|_\infty,$$

where $0 < \lambda_c < 1$. We use the estimation

$$\mathrm{nnz}(x) := \min\left\{k \mid \sum_{i=1}^{k}|\hat{x}_i| > 0.999\|x\|_1\right\}$$

to denote the number of the nonzero components in x, where $\hat{x}$ is obtained by sorting x such that $|\hat{x}_1| \geq |\hat{x}_2| \geq \cdots \geq |\hat{x}_n|$.

6.1. l_1-minimization problem. We note that the relative performance of most of the existing algorithms mentioned in the introduction has recently been well documented in the recent papers [1, 2, 30, 36, 39]. We compared the PBC_l_1-Hom algorithm with several of these state-of-the-art algorithms for l_1-minimization problems.

- FISTA [2] is a classical method for l_1-minimization which is easy to implement and is fast to obtain an approximate solution. We terminated FISTA if $\frac{\|x^k - x^{k+1}\|}{\|x^k\|}$ is smaller than a given tolerance or it reaches the maximum iterations ($\mathbf{maxiter} = \min(\max(1000, 2*n), 10^4)$).
- SSNAL [30], which is efficient for large-scale l_1-minimization problems. SSNAL is shown to be very computationally efficient for large-scale l_1-minimization problems. We terminated SSNAL if the "**stoptol**" (see the software package[1]) is smaller than a given tolerance.
- The original l_1-homotopy method [1] solves the original l_1-minimization problem (1.2) without the decomposition technique presented in this paper. To show PBC_l_1-Hom is more efficient than the original l_1-homotopy method, we chose l_1-homotopy as one of the compared algorithms. Here, the original l_1-homotopy method is not improved with the warm-start procedure and the ε-precision verification-correction technique. The number of the maximum iterations was set to $\mathbf{maxiter} = 50m$.
- SpaRSA [36] is an efficient algorithm proposed for least squares problems with separable regularizer. This method is shown to be more computationally efficient than GPSR [20] and FPC_AS [34] in [36].

The LIBSVM data sets are collected from real-world problems which are often used to test solvers for regression and classification problems [40]. For each database, we chose $(n-1)$ samples as the matrix A and the rest sample as the vector b.

The numerical results are reported in Table 6.2. PBC_l_1-Hom is shown to be much faster than the other methods for most of the tested instances. Moreover, the results demonstrate that PBC_l_1-Hom needs less memory than SSNAL and is more robust than l_1-homotopy method. PBC_l_1-Hom is shown to be highly efficient to obtain a high-precision solution when the number of the nonzero components is small, even for very-large-scale problems. Specially, PBC_l_1-Hom takes 13 seconds to solve the instance **kddb** with 19,264,097 samples and 29,890,095 features ($\lambda_c = 1e^{-1}$) and only 1 second for the instance **url** with 2,396,130 samples and 3,231,961 features ($\lambda_c = 1e^{-2}$).

The results of **covtype** show that PBC_l_1-Hom can handle the problem of which $\lambda_{\max}(A^TA)$ is large, while SpaRSA and FISTA are not capable of solving such problems. Though $\lambda_{\max}(A^TA)$ is large, which means the complementarity is weak, PBC_l_1-Hom can obtain a high-precision solution, while the l_1-homotopy method can obtain only a low-precision solution. This is because when $\lambda_{\max}(A^TA)$ is large, the complementarity is often weak, which may lead to an incorrect update of the working set $W(t)$ in the homotopy tracking steps. Since PBC_l_1-Hom solves the small-scale subproblem, $\lambda_{\max}(A_B^TA_B)$ is much smaller than $\lambda_{\max}(A^TA)$, which means that PBC_l_1-Hom can update the working more accurately and obtain a higher-precision solution.

Figure 6.1 compares the running time versus the value error of the objective function of PBC_l_1-Hom and the other algorithms. We conducted the experiments on four large-scale databases: **epsilon.t**, **HIGGS**, **kdda**, and **webspam**. From Figure 6.1, we can see that PBC_l_1-Hom takes much less time than the other algorithms and obtains a higher-precision solution, while FISTA and SpaRSA are not suitable for these databases.

At each step, PBC_l_1-Hom needs to calculate the matrix-vector multiplication $A^T * (A * x_B)$ in (2.5) which takes $O(mn)$ flops and solve the subproblems which take

[1]http://www.math.nus.edu.sg/mattohkc/LassoNAL.html.

Table 6.2

The performance of PBC_l_1-Hom, SSNAL, l_1-homotopy, SpaRSA, and FISTA for solving the l_1-minimization problems. In the table, "a" = PBC_l_1-Hom, "b" = SSNAL, "c" = l_1-homotopy, "d" = SpaRSA and "e" = FISTA, respectively. "nnz" denotes the number of the nonzero components in the solution obtained by PBC_l_1-Hom. "Error" indicates the algorithm breaks down due to some internal errors. "OM" denotes out of memory. "OT" denotes the computation time more than 10 hours. The computation time is in the format of "hours/minutes/seconds." The best results are indicated by boldface.

Problem n;m	λ_c	nnz		a	b	c	d	e
avazu-app	10^{-4}	190	time	**1m11s**	46m0s	Error	3h58m53s	4h05m0s
14,596,137;1,000,000			$\\|\psi\\|$	1.9E-08	1.0E-05	Error	2.8E+02	1.2E-02
	10^{-3}	284	time	**10s**	15m28s	23m20s	3h07m14s	22m0s
			$\\|\psi\\|$	9.2E-08	2.8E-07	3.4E-14	4.1E+02	2.7E-02
covtype	10^{-4}	10	time	**1s**	1m06s	21s	10m33s	8m22s
581,012;54			$\\|\psi\\|$	1.9E-08	9.4E-03	1.3E-03	2.3E+12	1.3E+05
	10^{-3}	9	time	**1s**	1m55s	17s	8m08s	6m56s
			$\\|\psi\\|$	1.1E-08	9.5E-03	1.8E-03	3.1E+12	2.9E+05
epsilon.t	10^{-2}	1685	time	54s	**25s**	6m33s	39m0s	1h47m10s
100,000;2,000			$\\|\psi\\|$	2.1E-05	2.8E-05	4.2E-14	3.1E+02	2.7E-05
	10^{-1}	808	time	**3s**	16s	1m25s	24m0s	1h08m0s
			$\\|\psi\\|$	1.2E-11	1.9E-06	4.1E-15	3.9E+02	7.2E-05
HIGGS	10^{-4}	27	time	**14s**	9m58s	4m03s	OT	9h50m0s
11,000,000;28			$\\|\psi\\|$	4.6E-10	7.3E-07	3.3E-12	OT	2.6E-01
	10^{-3}	27	time	**13s**	5m23s	3m58s	OT	9h18m20s
			$\\|\psi\\|$	5.7E-10	1.1E-05	1.1E-12	OT	6.1E-01
kdda	10^{-3}	4448	time	**8m29s**	21m00s	3h38m20s	OT	OT
8,407,552;20,216,830			$\\|\psi\\|$	4.9E-06	5.1E-06	1.0E-13	OT	OT
	10^{-2}	834	time	**17s**	10m00s	34m00s	OT	OT
			$\\|\psi\\|$	2.1E-13	2.5E-05	1.1E-13	OT	OT
kddb-sub	10^{-3}	14983	time	**9m17s**	39m44s	Error	2h20m33s	25m20s
100,000;29,890,095			$\\|\psi\\|$	4.2E-04	4.7E-04	Error	9.6E-01	2.2E-02
	10^{-2}	241	time	**4s**	10m13s	Error	7m17s	15m32s
			$\\|\psi\\|$	1.4E-10	1.2E-05	Error	1.5E-02	2.0E-03
kddb	10^{-2}	14546	time	**14m23s**	OM	Error	OT	OT
19,264,097;29,890,095			$\\|\psi\\|$	4.5E-04	OM	Error	OT	OT
	10^{-1}	340	time	**13s**	OM	Error	OT	OT
			$\\|\psi\\|$	1.0E-12	OM	Error	OT	OT
news20	10^{-3}	17304	time	2m44s	**1m01**s	OT	23m20s	57m40s
19,996;1,355,191			$\\|\psi\\|$	7.2E-04	5.5E-04	OT	1.8E-04	5.7E-05
	10^{-2}	9338	time	33s	**18s**	3h33m32s	6m49s	33m40s
			$\\|\psi\\|$	6.1E-04	6.1E-04	5.0E-14	4.9E-04	6.7E-05
rcv1_test-sub	10^{-2}	3652	time	16s	**6s**	12m11s	10s	1m04s
100,000;47,236			$\\|\psi\\|$	1.6E-04	1.9E-04	3.5E-14	2.7E-04	2.9E-05
	10^{-1}	242	time	**0.4s**	1s	1s	1s	23s
			$\\|\psi\\|$	6.3E-09	1.4E-06	6.6E-14	1.6E-03	3.6E-05
rcv1_test	10^{-2}	4197	time	51s	**29s**	55m20s	21m40s	26m36s
677,399;47,236			$\\|\psi\\|$	1.5E-04	1.4E-04	4.1E-14	9.8E-01	1.5E-01
	10^{-1}	176	time	**1s**	8s	50s	1m32s	18m32s
			$\\|\psi\\|$	2.0E-10	7.8E-07	8.9E-16	8.5E-01	1.7E-01
SUSY	10^{-3}	16	time	**3s**	57s	1m14s	31m22s	24m0s
5,000,000;18			$\\|\psi\\|$	8.6E-11	1.4E-04	4.2E-11	6.3E+03	2.6E+00
	10^{-2}	12	time	**2s**	39s	1m11s	27m0s	22m10s
			$\\|\psi\\|$	2.5E-12	1.3E-03	4.4E-12	5.3E+03	4.6E+00
url	10^{-3}	325	time	**12s**	9m54s	Error	OT	9h13m20s
2,396,130;3,231,961			$\\|\psi\\|$	7.7E-08	3.4E-06	Error	OT	2.3E-01
	10^{-2}	59	time	**1s**	1m22s	Error	OT	8h18m20s
			$\\|\psi\\|$	1.48E-09	7.2E-04	Error	OT	3.2E-01
webspam	10^{-4}	79	time	**13s**	5m21s	38s	1h12m00s	1h43m50s
350,000;16,609,143			$\\|\psi\\|$	1.4E-10	1.3E-06	9.8E-13	5.3E-01	2.5E-02
	10^{-3}	57	time	**4s**	2m31s	20s	51m0s	1h10m10s
			$\\|\psi\\|$	2.6E-10	1.1E-06	2.7E-13	4.6E-01	3.7E-02

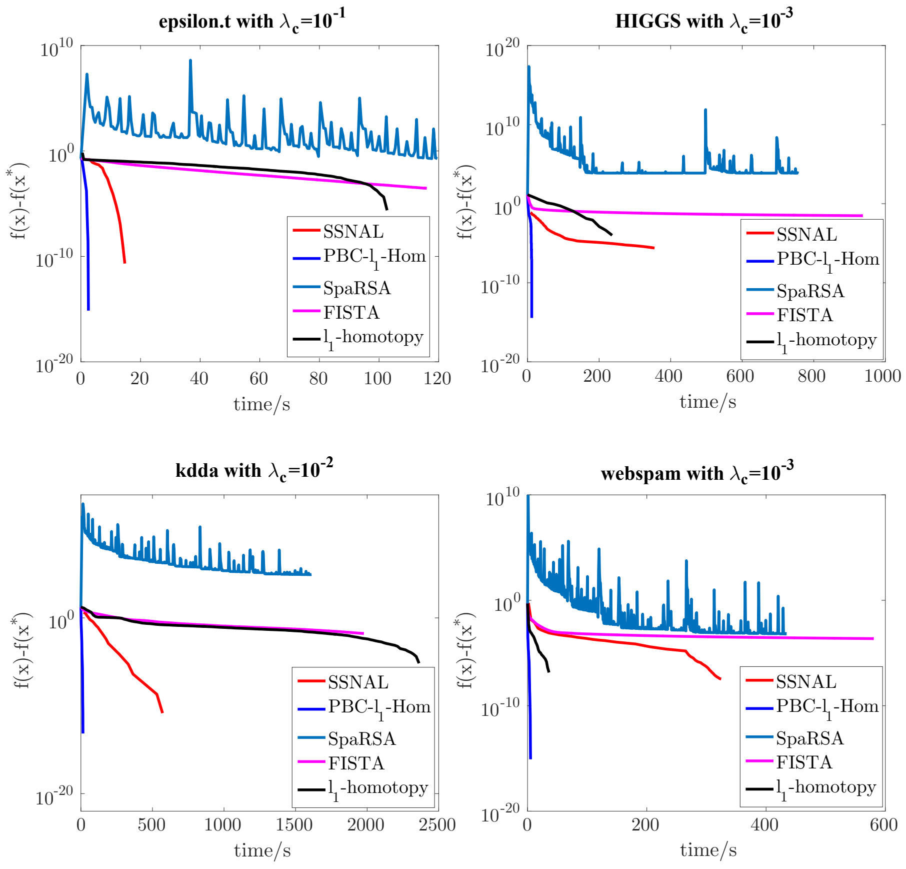

FIG. 6.1. *The running time versus the value error of the objective function.*

$O(q^3)$ flops. Moreover, from Theorem 2.3, PBC_l_1-Hom obtains a $\tilde{\tau}$-“precise” solution in $\frac{2f(x^0)(\tilde{L}+\delta)^2}{\delta\omega^2\tau^2 q}$ iterations. Hence, PBC_l_1-Hom takes $\frac{2f(x^0)(\tilde{L}+\delta)^2}{\delta\omega^2\tau^2 q}(O(mn)+O(q^3))$ flops in total. In Figure 6.2, we compare the computational complexity of PBC_l_1-Hom with the other algorithms. We terminated them with the same precision. We did not compare them with SpaRSA and FISTA, because SpaRSA and FISTA are much slower than them for these databases. The results show that the computational complexity on n of PBC_l_1-Hom is lower than the other algorithms and is much lower than linear computational complexity in practice.

From the results of **rcv1_test** and **rcv1_test-sub** in Table 6.2 and the results of **kdda** and **webspam** in Figure 6.2, we can see that the performance of PBC_l_1-Hom mainly depends on the sparsity of the solution and the number of features, while the number of samples does not have much influence. However, the other algorithms need more computation time as the number of samples increases. Note that when the number of nonzero components is much smaller than n, the l_1 homotopy method “wasted” much computation on computing $A_{W^c(t)}A_{W(t)}c$ and $A_{W^c(t)}A_{W(t)}d$. By decomposing the large-scale problem to a sequence of small-scale problems, PBC_l_1-Hom filters out most of the indices of $W^c(t)$ before the homotopy tracking steps. This means computing $A_{W^c(t)}A_{W(t)}c$ and $A_{W^c(t)}A_{W(t)}d$ in tracking the solution path of the subproblems takes a much smaller computational cost. Hence, when the number

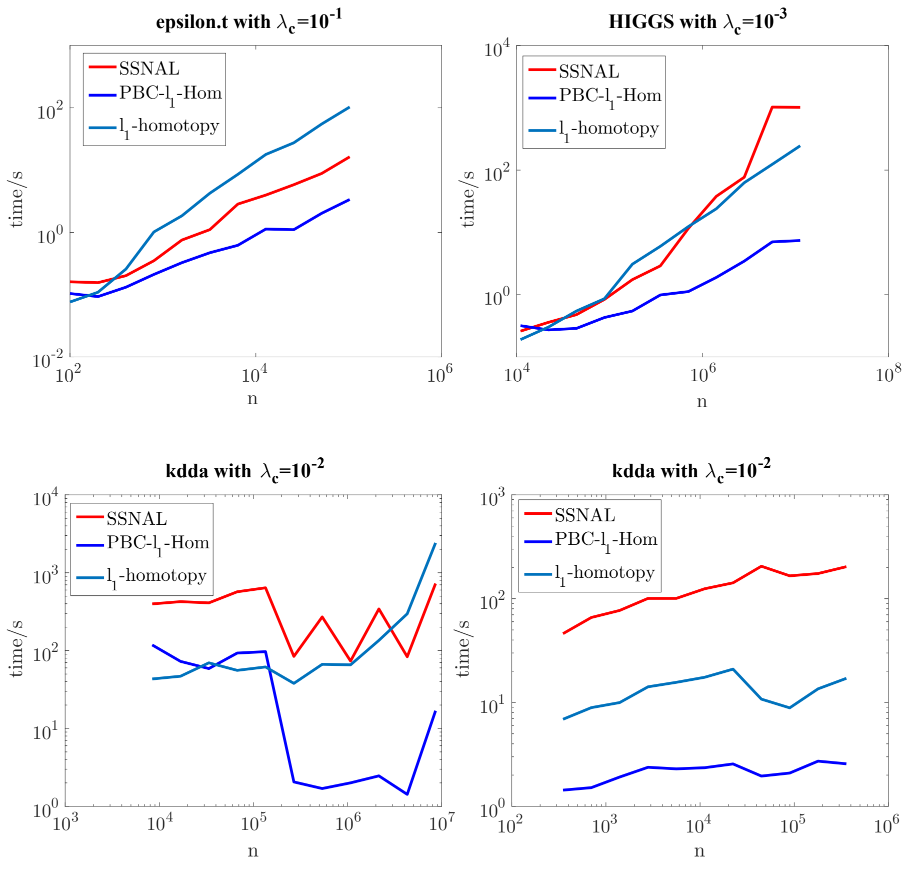

FIG. 6.2. *The dimension of the variables of the problem versus the running time.*

of samples is large, PBC_l_1-Hom is much faster and more robust than the original l_1-homotopy method.

Moreover, in Figure 6.3, we demonstrate the time complexity of PBC_l_1-Hom on the number of the nonzero components of the solution $(\text{nnz}(x^*))$, l_1-homotopy, and SSANL. We did not compare them with FISTA and SpaRSA, because they take much more time. As shown in Figure 6.3, PBC_l_1-Hom is more suitable for the problems for which solutions are sparse than are SSNAL and the l_1-homotopy method. Moreover, as $\text{nnz}(x^*)$ increases, the running time of PBC_l_1-Hom increases more slowly than the l_1-homotopy method.

6.2. l_{1-2}-minimization problem. To show that the presented algorithm framework is efficient to solve l_{1-2}-minimization problems with both incoherent and coherent matrices A (a coherent matrix A is unlikely to possess small restricted isometry constant), we follow Yin et al. [39] by randomly generating a Gaussian matrix which follows the independent identical distribution (i.i.d.)

$$A_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_m/m) \text{ for } i = 1, \ldots, n$$

and the random oversampled partial discrete cosign transform (DCT) matrix

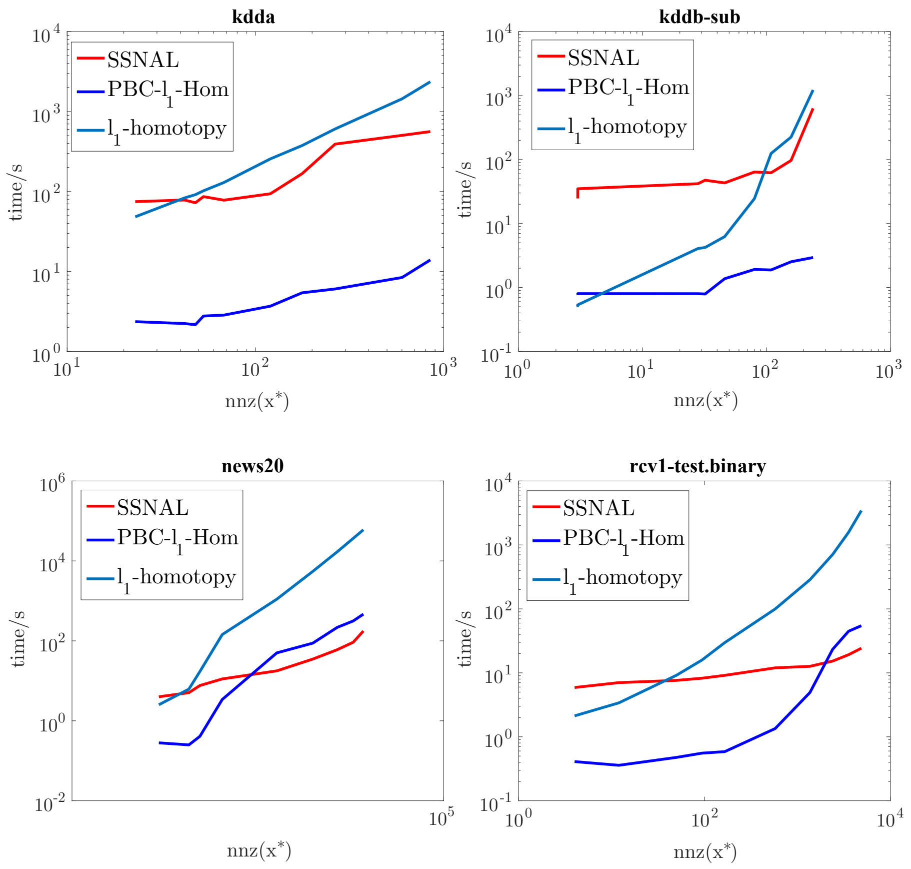

FIG. 6.3. *The number of the nonzero components of the solution versus the running time.*

$$A_i = \frac{1}{\sqrt{m}} \cos(2i\pi\xi/F) \text{ for } i = 1, \ldots, n,$$

where $\xi \in \mathbb{R}^n \sim \mathcal{U}[0,1]^m$, whose components are uniformly and independently sampled from [0,1]. Here, $F \in \mathbb{N}$ is the *refinement* factor. The Gaussian matrices fit for compressed sensing, being incoherent and having small RIP constants with high probability and the oversampled partial DCT matrices are highly coherent. Although the oversampled partial DCT matrices A sampled in this way do not have good RIP by any means, it is still possible to recover the sparse vector x^* provided its spikes are sufficiently separated. We follow Yin et al. [39] by randomly selecting the elements of supp(x^*)=$\{j \mid x_j^* \neq 0\}$ so as to satisfy the following condition:

$$\min_{j,k\in \text{supp}(x^*)} |j-k| \geq F.$$

We compared PBCDCA_l_1-Hom with the following algorithms:

- DCA_ADMM [39], which solves every DCA subproblem (3.1) with the ADMM method. Here, we terminate the ADMM iterations if $\frac{\|x^t - x^{t+1}\|}{\|x^t\|} < 1e{-}5$ or t reaches **maxiter** $= \min(\max(1000, 2 * n), 10^4)$. Here, t denotes the number of ADMM iterations.

- DCA_SSNAL, which solves every DCA subproblem (3.1) with the SSNAL method. Here, the parameter "**stoptol**" in SSNAL was set to $1e^{-6}$.
- DCA_l_1-homotopy, which solves every DCA subproblem (3.1) with l_1-homotopy method. The solution of the previous DCA subproblem was used to warm start the current one.

We terminate the outer iterations of the above three algorithms when $\frac{\|x^k - x^{k+1}\|}{\|x^k\|} < 1e^{-6}$ or k reaches **maxiter** $= 100$. Moreover, x^k is used to warm start the $(k+1)$th DCA subproblem. Here, k denotes the number of the DCA iterations. We use the KKT residual estimation $\|\psi\|$ and the relative error

$$\text{err} := \frac{\|x - x^*\|}{\|x^*\|} \tag{6.2}$$

to measure the precision of the solution x obtained by the above algorithms.

Tables 6.3 and 6.4 report the detailed numerical results for PBCDCA_l_1-Hom, DCA_SSNAL, DCA_l_1-homotopy, and DCA_ADMM solving the l_{1-2}-minimization problems with the Gaussian matrices and the oversampled DCT matrices, respectively. Figure 6.4 reports the computation complexity of these algorithms.

The numerical results demonstrate that PBCDCA_l_1-Hom is robust and efficient to handle l_{1-2}-minimization problems with both incoherent and highly coherent matrices. For large-scale problems, PBCDCA_l_1-Hom takes much less time and obtains much-higher-precision solutions. Moreover, it is clear that the computation complexity of PBCDCA_l_1-Hom is much lower than for the other algorithms from Figure 6.4, which means that PBCDCA_l_1-Hom is suitable for solving large-scale problems.

TABLE 6.3

The performance of PBCDCA_l_1-Hom, DCA_SSNAL, DCA_l_1-Hom, and DCA_ADMM on solving the l_{1-2}-minimization problems ($\alpha = 1$) with Gaussian matrices and $\lambda_c = 1e^{-3}$. n is the sample size and m is the dimension of features. In the table, "Da" = PBCDCA_l_1-Hom, "Db" = DCA_SSNAL, "Dc" = DCA_l_1-homotopy, and "Dd" = DCA_ADMM, respectively. "nnz" denotes the number of the nonzero components in the solution obtained by PBCDCA_l_1-Hom. "OM" denotes out of memory. "err" denotes the relative error of the solution obtained by PBCDCA_l_1-Hom which is calculated as (6.2).

n	m	nnz	$\|\psi\|$	err
			$Da\|Db\|Dc\|Dd$	$Da\|Db\|Dc\|Dd$
256	64	8	4.5E-11|3.2E-05|3.2E-05|3.6E-06	1.8E-03|2.5E-03|2.5E-03|1.5E-03
512	128	16	2.6E-11|2.1E-05|1.7E-05|2.0E-05	2.0E-03|3.0E-03|2.6E-03|4.0E-03
1024	256	32	1.4E-11|1.2E-05|7.4E-06|3.2E-05	2.2E-03|3.8E-03|2.6E-03|2.5E-01
2048	512	63	9.7E-11|9.8E-06|5.9E-06|4.5E-05	3.9E-03|5.4E-03|4.4E-03|6.8E-01
4096	1024	124	9.5E-11|1.2E-05|2.7E-06|6.4E-05	3.7E-03|9.7E-03|4.0E-03|7.8E-01
8192	2048	245	3.9E-11|1.7E-05|2.2E-06|8.9E-05	4.9E-03|2.6E-03|5.2E-03|8.5E-01
16384	4096	488	3.3E-09|1.5E-05|7.1E-07|1.2E-04	3.6E-03|4.9E-03|3.8E-03|8.6E-01
32768	8192	988	2.6E-09|1.8E-05|3.9E-07|1.8E-04	4.2E-03|7.1E-03|4.3E-03|3.4E-04
65536	16384	1968	5.0E-09|OM|3.0E-07|2.5E-04	6.1E-03|OM|6.2E-03|8.6E-01

Table 6.4

The performance of PBCDCA_l_1-Hom, DCA_SSNAL, DCA_l_1-Hom, and DCA_ADMM on solving the l_{1-2}-minimization problems ($\alpha = 1$) with oversampled DCT matrices and $\lambda_c = 1e^{-4}$. The matrices were generated with $F = 20$. The n is the sample size and m is the dimension of features. In the table, "Da" = PBCDCA_l_1-Hom, "Db" = DCA_SSNAL, "Dc" = DCA_l_1-homotopy, and "Dd" = DCA_ADMM, respectively. "nnz" denotes the number of the nonzero components in the solution obtained by PBCDCA_l_1-Hom. "OM" denotes out of memory. "err" denotes the relative error of the solution obtained by PBCDCA_l_1-Hom, which is calculated as (6.2).

n	m	nnz	$\|\psi\|$	err
			$Da\|Db\|Dc\|Dd$	$Da\|Db\|Dc\|Dd$
256	64	4	8.5E-11|1.5E-04|1.1E-04|2.2E-06	8.5E-05|1.5E-04|1.3E-04|6.7E-05
512	128	8	3.0E-11|2.3E-04|6.6E-05|1.9E-06	1.1E-04|3.9E-04|2.0E-04|1.5E-04
1024	256	14	5.8E-11|2.3E-04|8.3E-05|2.1E-06	2.0E-04|3.0E-04|2.0E-04|1.5E-04
2048	512	27	1.2E-11|1.0E-04|1.3E-04|3.8E-06	2.1E-04|2.7E-04|2.9E-04|2.5E-04
4096	1024	51	2.8E-11|1.5E-04|1.2E-04|7.0E-06	2.6E-04|3.4E-04|2.4E-04|2.2E-04
8192	2048	99	4.4E-11|8.6E-05|1.3E-04|1.1E-05	2.3E-04|2.8E-04|3.0E-04|2.8E-04
16384	4096	198	1.3E-11|4.4E-04|1.6E-04|1.2E-05	2.6E-04|2.7E-04|3.6E-04|3.4E-04
32768	8192	389	2.3E-11|7.4E-04|1.7E-04|1.8E-05	3.7E-04|2.7E-04|3.6E-04|3.5E-04
65536	16384	780	1.3E-11|OM|1.9E-04|2.3E-05	3.4E-04|OM|4.0E-04|3.9E-04

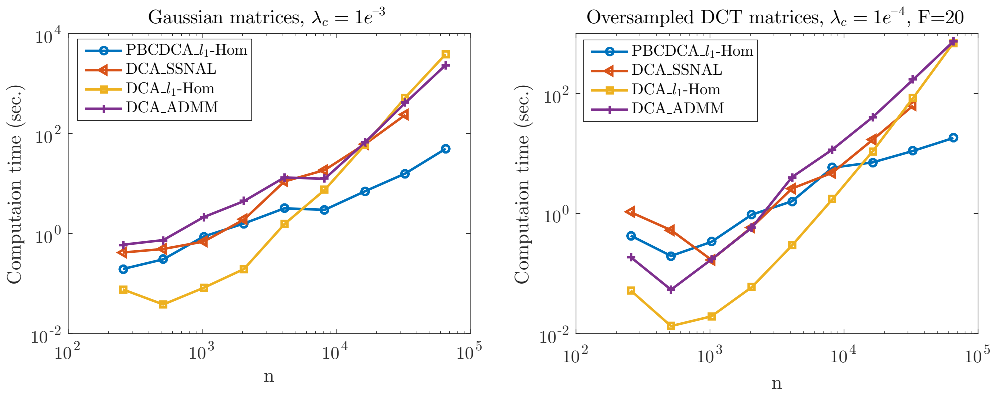

Fig. 6.4. *The computation time for PBCDCA_l_1-Hom, DCA_SSNAL, DCA_l_1-Hom, and DCA_ADMM solving the l_{1-2}-minimization problems ($\alpha = 1$) with the Gaussian matrices and the oversampled DCT matrices. The n is the sample size.*

7. Conclusion.

In this paper, we have proposed an algorithm framework for large-scale sparse least squares problems, including l_1- and l_{1-2}-minimization problems. The algorithm framework which takes the advantages of the sparsity of the solutions was proved to converge to a $\tilde{\tau}$-"precise" solution in a finite number of steps. Moreover, the value of the objective function of the iterative points is proved to monotonically decrease and linearly converges before a $\tilde{\tau}$-"precise" solution is obtained. With the improved l_1-homotopy method solving every subproblem, the algorithm framework is shown to be highly efficient and robust for large-scale ill-conditioned problems, especially when the solutions are highly sparse.

Compared with the original l_1-homotopy method, the PBC_l_1-Hom method mainly has four advantages. First, by decomposing the original problem into a sequence of small-scale subproblems, PBC_l_1-Hom filters out most of the zero components. Hence, in the homotopy tracking steps, we do not need to calculate large-scale matrix-vector multiplication $A_{W^c(t)}A_{W(t)}c$ and $A_{W^c(t)}A_{W(t)}d$ in the homotopy tracking steps. Moreover, by filtering out most of the zero components, the number of the steps needed in the homotopy tracking is greatly reduced. This is because when $|W^c(t)|$ is large, many indices of $W^c(t)$ may be moved to $W(t)$ in the homotopy tracking steps, which is especially serious when the complementarity is weak and the problem is ill-conditioned. Second, the value of $\lambda_{\max}(A_B^T A_B)$ is smaller than $\lambda_{\max}(A^T A)$. Since a small $\lambda_{\max}(A_B^T A_B)$ is beneficial to the efficiency of FISTA, it is easier to do warm-start for the subproblems than the original problem. Third, the PBC_l_1-Hom solves smaller-scale linear systems; hence it is more robust. Moreover, by adding proximal terms, the linear systems are better-conditioned. Fourth, the ε-precision verification-correction technique for the incorrect update case makes the homotopy tracking steps more robust for problems which are ill-conditioned and weakly complementary.

Acknowledgments. The authors would like to thank their colleagues for their valuable suggestions that led to the improvement of this paper.

REFERENCES

[1] M. S. Asif and J. Romberg, *Sparse recovery of streaming signals using l_1-homotopy*, IEEE Trans. Signal Process., 62 (2014), pp. 4209–4223.

[2] A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM J. Imaging Sci., 2 (2009), pp. 183–202.

[3] E. Van Den Berg and M. P. Friedlander, *Probing the pareto frontier for basis pursuit solutions*, SIAM J. Sci. Comput., 31 (2008), pp. 890–912.

[4] A. Bjorck, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996.

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

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Found. Trends Machine learning, 3 (2011), pp. 1–122.

[7] R. H. Byrd, G. M. Chin, J. Nocedal, and F. Oztoprak, *A family of second-order methods for convex l_1-regularized optimization*, Math. Program., 159 (2016), pp. 435–467.

[8] R. H. Byrd, J. Nocedal, and F. Oztoprak, *An inexact successive quadratic approximation method for l-1 regularized optimization*, Math. Program., 157 (2016), pp. 375–396.

[9] E. J. Candès and J. Romberg, *Quantitative robust uncertainty principles and optimally sparse decompositions*, Found. Comput. Math., 6 (2006), pp. 227–254.

[10] E. J. Candès, J. Romberg, and T. Tao, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information*, IEEE Trans. Inform. Theory, 52 (2006), pp. 489–509.

[11] E. J. Candes and T. Tao, *Decoding by linear programming*, IEEE Trans. Inform. Theory, 51 (2005), pp. 4203–4215.

[12] C.-C. Chang and C.-J. Lin, *LIBSVM: A library for support vector machines*, ACM Trans. Intelligent Systems Technology, 2 (2011), 27.

[13] S. S. Chen, D. L. Donoho, and M. A. Saunders, *Atomic decomposition by basis pursuit*, SIAM Rev., 43 (2001), pp. 129–159.

[14] I. Daubechies, M. Defrise, and C. De Mol, *An iterative thresholding algorithm for linear inverse problems with a sparsity constraint*, Commun. Pure Appl. Math., 57 (2004), pp. 1413–1457.

[15] D. L. Donoho, *Compressed sensing*, IEEE Trans. Inform. Theory, 52 (2006), pp. 1289–1306.

[16] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani, et al., *Least angle regression*, Ann. Statist., 32 (2004), pp. 407–499.

[17] E. Esser, *Applications of lagrangian-based alternating direction methods and connections to split bregman*, CAM Report 9, UCLA, 2009.

[18] E. Esser, Y. Lou, and J. Xin, *A method for finding structured sparse solutions to nonnegative least squares problems with applications*, SIAM J. Imaging Sci., 6 (2013), pp. 2010–2046.

[19] J. Fan and R. Li, *Variable selection via nonconcave penalized likelihood and its oracle properties*, J. Amer. Statist. Assoc., 96 (2001), pp. 1348–1360.

[20] M. A. Figueiredo, R. D. Figueiredo, and S. J. Wright, *Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems*, IEEE J. Selected Topics Signal Process., 1 (2007), pp. 586–597.

[21] S. Foucart, *Hard thresholding pursuit: An algorithm for compressive sensing*, SIAM J. Numer. Anal., 49 (2011), pp. 2543–2563.

[22] S. Foucart, *Recovering jointly sparse vectors via hard thresholding pursuit*, in Proceedings of Sampling Theory and Applications, 2011.

[23] S. Foucart and M.-J. Lai, *Sparsest solutions of underdetermined linear systems via l_q-minimization for $0 < q \leq 1$*, Appl. Comput. Harmon. Anal., 26 (2009), pp. 395–407.

[24] K. Fountoulakis, J. Gondzio, and P. Zhlobich, *Matrix-free interior point method for compressed sensing problems*, Math. Program. Comput., 6 (2014), pp. 1–31.

[25] T. Goldstein and S. Osher, *The split Bregman method for l1-regularized problems*, SIAM J. Imaging Sci., 2 (2009), pp. 323–343.

[26] G. H. Golub and C. F. Van Loan, *Matrix computations*, Vol. 3, Johns Hopkins University Press, Baltimore, 2012.

[27] J. J. Júdice and F. M. Pires, *Basic-set algorithm for a generalized linear complementarity problem*, J. Optim. Theory Appli., 74 (1992), pp. 391–411.

[28] N. Keskar, J. Nocedal, F. Öztoprak, and A. Waechter, *A second-order method for convex 1-regularized optimization with active-set prediction*, Optim. Methods Softw., 31 (2016), pp. 605–621.

[29] J. Kim and H. Park, *Fast active-set-type algorithms for l1-regularized linear regression*, in Proceedings of the 13th International Conference on Artificial Intelligence and Statistics, 2010, pp. 397–404.

[30] X. Li, D. Sun, and K.-C. Toh, *A highly efficient semismooth newton augmented lagrangian method for solving lasso problems*, SIAM J. Optim., 28 (2018), pp. 433–458.

[31] A. Milzarek and M. Ulbrich, *A semismooth Newton method with multidimensional filter globalization for l_1-optimization*, SIAM J. Optim., 24 (2014), pp. 298–333.

[32] R. Tibshirani, *Regression shrinkage and selection via the lasso*, J. R. Stat. Soc. Ser. B Methodol., 58 (1996), pp. 267–288.

[33] R. Tibshirani, *Regression shrinkage and selection via the lasso: A retrospective*, J. R. Stat. Soc. Ser. B Stat. Methodol., 73 (2011), pp. 273–282.

[34] Z. Wen, W. Yin, D. Goldfarb, and Y. Zhang, *A fast algorithm for sparse reconstruction based on shrinkage, subspace optimization, and continuation*, SIAM J. Sci. Comput., 32 (2010), pp. 1832–1857.

[35] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, *Robust face recognition via sparse representation*, IEEE Trans. Pattern Anal. Machine Intelligence, 31 (2009), pp. 210–227.

[36] S. J. Wright, R. D. Nowak, and M. A. Figueiredo, *Sparse reconstruction by separable approximation*, IEEE Trans. Signal Process., 57 (2009), pp. 2479–2493.

[37] L. Xiao and T. Zhang, *A proximal-gradient homotopy method for the l_1-regularized least-squares problem*, in Proceedings of the 29th International Conference on Machine Learning, 2012, pp. 839–846.

[38] Z. Xu, X. Chang, F. Xu, and H. Xu, *$l_{1/2}$ regularization: A thresholding representation theory and a fast solver*, IEEE Trans. Neural Networks Learning Systems, 23 (2012), pp. 1013–1027.

[39] P. Yin, Y. Lou, Q. He, and J. Xin, *Minimization of l_{1-2} for compressed sensing*, SIAM J. Sci. Comput., 37 (2015), pp. A536–A563.

[40] G.-X. Yuan, K.-W. Chang, C.-J. Hsieh, and C.-J. Lin, *A comparison of optimization methods and software for large-scale l1-regularized linear classification*, J. Mach. Learn. Res., 11 (2010), pp. 3183–3234.

[41] J. Zeng, S. Lin, Y. Wang, and Z. Xu, *$l_{1/2}$ regularization: Convergence of iterative half thresholding algorithm*, IEEE Trans. Signal Process., 62 (2014), pp. 2317–2329.

[42] H. Zou, *The adaptive lasso and its oracle properties*, J. Amer. Statist. Assoc., 101 (2006), pp. 1418–1429.