Noname manuscript No.
(will be inserted by the editor)

A Parallel Line Search Subspace Correction Method for Composite Convex Optimization

Qian Dong · Xin Liu · Zaiwen Wen · Yaxiang Yuan

Received: date / Accepted: date

Abstract In this paper, we investigate a parallel subspace correction framework for composite convex optimization. The variables are first divided into a few blocks based on certain rules. At each iteration, the algorithms solve a suitable subproblem on each block simultaneously, construct a search direction by combining their solutions on all blocks, then identify a new point along this direction using a step size satisfying the Armijo line search condition. They are called PSCLN and PSCLO, respectively, depending on whether there are overlapping regions between two immediately adjacent blocks of variables. Their convergence is established under mild assumptions. We compare PSCLN and PSCLO with the parallel version of the fast iterative thresholding algorithm and the fixed-point continuation method using the Barzilar-Borwein step size and the greedy coordinate block descent method for solving the ℓ_1-regularized minimization problems. Our numerical results show that PSCLN and PSCLO can run fast and return solutions no worse than those from the state-of-the-art algorithms. It is also observed that the overlapping domain decomposition scheme is helpful when the data of the problem has certain special structures.

Keywords line search · block coordinate descent method · domain decomposition · Jacobian-type iteration · distributed optimization.

Qian Dong
State Key Laboratory of Scientific and Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, CHINA (dongqian@lsec.cc.ac.cn). Research supported in part by NSFC grants 11331012 and 11321061.

Xiu Liu
State Key Laboratory of Scientific and Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, CHINA (liuxin@lsec.cc.ac.cn). Research supported in part by NSFC grants 11101409, 11331012 and 91330115, and the National Center for Mathematics and Interdisciplinary Sciences, CAS.

Zaiwen Wen
Beijing International Center for Mathematical Research, Peking University, Beijing, CHINA (wenzw@math.pku.edu.cn). Research supported in part by NSFC grants 11322109 and 91330202.

Yaxiang Yuan
State Key Laboratory of Scientific and Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, CHINA (yyx@lsec.cc.ac.cn). Research supported in part by NSFC grants 11331012 and 11321061.

1 Introduction

In this paper, we consider the composite convex optimization problem

$$\min_{x\in\mathbb{R}^n} \varphi(x) := f(x) + h(x), \tag{1}$$

where $f(x)$ is differentiable convex function and $h(x)$ is a convex function that is possibly non-smooth. Many models in sparse optimization can be formulated as problem (1), such as the ℓ_1-regularized minimization (LASSO) [41] and the sparse logistic regression [38].

One simple and traditional parallel algorithm for solving (1) is to compute the gradient and other operations as many as possible in parallel in the fixed-point continuation method (also known as proximal gradient methods) and the (fast) iterative thresholding algorithm [2]. The alternating direction method of multipliers (ADMM) [4,6,14,15,17,18,22,24,43] can also be parallelized in a similar fashion. The block coordinate descent (BCD) method is widely used in parallel computation. It divides the variable x into a few small blocks, then minimizes the objective function with respect to one block of variables while fixes all other blocks at each iteration. The BCD method can be found under numerous names, including linear and nonlinear Gauss-Seidel methods, subspace correction methods [25,40] and alternating minimization approaches. Since the dimension of each block is often small, the cost of each iteration of the BCD method is relative cheap which enables it suitable for large scale problems [13,31,37]. When problem (1) restricted on a selected block of variables has no closed form solution, a popular strategy is the block coordinate gradient decent method or the block coordinate proximal gradient algorithm [3,37,42,45]. Instead of solving the restricted problem exactly, these two methods use one gradient step or one proximal gradient step, respectively. Nesterov's acceleration techniques have been extended to improve the numerical performance of the BCD method in [15,26,31]. Luo and Tseng established linear convergence rate of the BCD method for solving several types of problem (1) by estimating the local error bounds [28–30]. More detailed results on the convergence rate and iteration complexity of the BCD method and its variants are referred to [15,19,23, 26,34] and the references therein.

The blocks of the variables x are often updated by using several types of strategies in the BCD method. The cyclic (Gauss-Seidel) strategy updates blocks one by one in turn and the values of the newly updated blocks are used in the subsequent operations on other blocks. The essentially cyclic rule selects each block at least once every T successive iterations, where T is an integer equal to or greater than the number of blocks. The Gauss-Southwell rule computes a positive value q_i for every block i according some criteria and then chooses the block with the largest value of q_i to work on next, or chooses the k-th block to work on, where $q_k \geq \beta \max_i q_i$ for $\beta \in (0, 1]$. Recently, some randomized rules have been proposed in [27,31,34,37]. The greedy coordinate block descent method (GRock) [32] selects a few blocks of variables and then a few variables in each selected block, both by greedy means, and update the latter in parallel at each iteration. Numerical experiments on a computer cluster and Amazon EC2 demonstrate that GRock is an efficient parallel algorithm for LASSO and sparse logistic regression. Another type of scheme is the Jacobian-type iteration. It updates all the blocks simultaneously. Hence, it is suitable for parallel computation and distributed optimization [7,8,20,31–33,35].

1.1 The Domain Decomposition Methods

The domain decomposition method [36,25] is a popular method for solving partial differential equations. It can be designed as an efficient parallel computation method by splitting the spatial domain into several subdomains and solving the corresponding problems on these subdomains iteratively using certain strategies. The concept of domain decomposition has been extended to solve optimization problems, for example, the successive and parallel subspace correction methods [5,9,11,21,40]. These two methods are exactly the Gauss-Seidel-type and Jacobian-type BCD methods, respectively. Similar to the BCD methods, the parallel subspace correction (PSC) method has higher scalability than the successive subspace correction (SSC) method. For this reason, we only consider the PSC method in this paper.

Generally speaking, the domain decomposition approaches can be classified by whether there are overlapping blocks among subdomains. Suppose that $\mathbb{R}^n$ is split into p subspaces, namely,

$$\mathbb{R}^n = X_1 + X_2 + \cdots + X_p, \tag{2}$$

where

$$X_i = \{x \in \mathbb{R}^n | \text{supp}(x) \subset \mathcal{J}_i\}, \quad 1 \le i \le p,$$

such that $\mathcal{J} := \{1, ..., n\}$ and $\mathcal{J} = \bigcup_{i=1}^{p} \mathcal{J}_i$. It is a non-overlapping domain decomposition of $\mathbb{R}^n$ if $\mathcal{J}_i \bigcap \mathcal{J}_j = \emptyset$, for any $i \neq j, 1 \le i, j \le p$. Otherwise, it is an overlapping domain decomposition of $\mathbb{R}^n$ if there exist $i, j \in \{1, ..., p\}$ and $i \neq j$ such that $\mathcal{J}_i \bigcap \mathcal{J}_j \neq \emptyset$. The overlapping domain decomposition strategy has been shown to be useful for the total variation minimization in [11].

Throughout this paper, we make the following assumption which is commonly used in domain decomposition.

Assumption 1 *For any given $x \in \mathbb{R}^n$, there exist $x_i \in X_i$, $1 \le i \le p$, such that $x = \sum_{i=1}^{p} x_i$ and $||x_i||_2 \le C_0 ||x||_2$, $1 \le i \le p$, where C_0 is a constant.*

For the given x and x_i, $i = 1, \dots, p$ in Assumption 1, a short calculation shows that

$$\left(\sum_{i=1}^{p} ||x_i||_2^2 \right)^{\frac{1}{2}} \le C_1 ||x||_2; \tag{3}$$

$$\tau \sum_{i=1}^{p} ||x_i||_2^2 \ge ||x||_2^2, \tag{4}$$

where $C_1 = C_0 p^{\frac{1}{2}}$ and

$$\tau = \begin{cases} 1, & \text{if } \mathcal{J}_i \bigcap \mathcal{J}_j = \emptyset, \text{ for any } i \neq j, \\ 2, & \text{if } \mathcal{J}_i \bigcap \mathcal{J}_j = \emptyset, \text{ for any } |i - j| > 2, \\ p, & \text{otherwise.} \end{cases} \tag{5}$$

1.2 The PSC Method

The PSC framework for solving (1) can be formulated as follows:

$$\begin{aligned} d_i^k &= \underset{d_i \in X_i}{\operatorname{argmin}}\, \varphi_i^k(d_i), \ \ i = 1, ..., p, \\ x^{k+1} &= x^k + \sum_{i=1}^{p} \alpha_i^k d_i^k, \end{aligned} \tag{6}$$

where φ_i^k is a surrogate function of φ restricted to the i-th subspace at k-th iteration. Tai and Xu proved in [40] the linear convergence rate of the PSC method (6) under the assumption of the smoothness and strongly convexity of φ. Carstensen derived similar results for the SSC method in [5] when φ is a summation of a strongly smooth convex function and a nonsmooth separable convex function. The PSC methods have been proposed for LASSO [9,12] and total variation minimization [10–12,21]. However, the step size α_i^k $(1 \le i \le p)$ are required to satisfy the conditions: $\sum_{i=1}^{p} \alpha_i^k \le 1$ and $\alpha_i^k > 0$ $(1 \le i \le p)$ in order to ensure the convergence. Usually, the step size α_i^k is quite small under these conditions and convergence tends to be slow. For instance, a typical step size is $\alpha_i^k = \frac{1}{p}$, which becomes smaller and smaller as the number of subspaces increases.

1.3 Our Contribution

One of our main contributions is the adaption of the Armijo backtracking line search in the PSC method for a large step size. The modified algorithm is called a parallel line search subspace correction method (PSCL) and it is outlined in Algorithm 1.

Algorithm 1: Parallel Line Search Subspace Correction Method

```
1 Determine a domain decomposition (2) of $\mathbb{R}^n$ that satisfies Assumption 1.
2 Initialize $x^0 \in \mathbb{R}^n$ and set $k := 0$.
3 while not converge do
4     Choose the surrogate functions $\varphi_i^k$ for each block, $i = 1, 2, ..., p$;
5     Solve the subproblem for each block: $d_i^k = \operatorname{argmin}_{d_i \in X_i} \varphi_i^k(d_i)$, $i = 1, 2, ..., p$;
6     Compute the summation of the direction $d^k = \sum_{i=1}^{p} d_i^k$ ;
7     Set $x^{k+1} = x^k + \alpha_k d^k$, where $\alpha_k$ satisfies the Armijo backtracking conditions;
8     Set $k := k + 1$.
```

When $h(x) = 0$ and $f(x)$ is strongly convex, the surrogate function in Algorithm 1 is set to the original objective φ, namely,

$$\varphi_i^k(d_i) = f(x^k + d_i), \ \text{for } d_i \in X_i. \tag{7}$$

Otherwise, it is set to a summation of a linear proximal function of the smooth part $f(x)$ and the nonsmooth part $h(x)$:

$$\varphi_i^k(d_i) = \nabla f(x^k)^{\mathrm{T}} d_i + \frac{1}{2\lambda_i} ||d_i||_2^2 + h(x^k + d_i), \ \text{for } d_i \in X_i. \tag{8}$$

We should point out that GRock also minimizes all $\varphi_i^k(d_i)$, $i = 1, \dots, p$, at each iteration for solving LASSO, but only some variables in a few selected blocks are updated and the line search procedure is not used.

The global convergence of PSCL is established by following the convergence analysis of the subspace correction methods for strongly convex problem [40], the active-set method for l_1 minimization [44] and the BCD method for nonsmooth separable minimization [42]. Specifically, linear convergence rate is proved for the strongly convex case and convergence to the solution set of problem (1) globally is obtained for the general nonsmooth case. Both non-overlapping and overlapping schemes (denoted by PSCLN and PSCLO, respectively) are proposed for PSCL. Parallel versions of these algorithms are implemented in C using MPI. Our numerical experiments demonstrate that PSCLO can be more efficient than PSCLN under certain circumstances. The numerical efficiency of the our algorithms is further confirmed in the comparison to the state-of-the-art algorithms including GRock.

1.4 Organization

The rest of this paper is organized as follows. The convergence analysis of PSCL for the strongly convex problems is presented in Section 2. Then the convergence properties of PSCL for the general nonsmooth problems are discussed in Section 3. The algorithmic issues and numerical results are reported in Section 4. Finally, some concluding remarks are given in Section 5.

2 The Strongly Convex Case

In this section, we consider the convergence properties of PSCL for minimizing a strongly convex function φ. Namely, $\varphi(x) = f(x)$ satisfies the following assumptions.

Assumption 2 *The function f is differentiable and there exist constants $K, L > 0$ such that*

$$(\nabla f(x) - \nabla f(y))^{\mathrm{T}}(x - y) \geq K||x - y||_2^2, \ \textit{for any}\, x, y \in \mathbb{R}^n, \tag{9}$$

$$||\nabla f(x) - \nabla f(y)||_2 \leq L||x - y||_2, \ \textit{for any}\, x, y \in \mathbb{R}^n. \tag{10}$$

Assumption 2 guarantees the strongly convexity of $f(x)$ and the Lipschitz continuity of the gradient $\nabla f(x)$. For notational brevity, we denote $g(x) = \nabla f(x)$, and $g^k = \nabla f(x^k)$. The next lemma shows that d^k is a descent direction and establishes the relationship between $(g^k)^{\mathrm{T}} d^k$ and $||d^k||_2^2$.

Lemma 1 *Suppose that Assumption 2 holds. Then it holds that*

$$(g^k)^{\mathrm{T}} d^k \leq -\frac{K}{\tau} ||d^k||_2^2,$$

where τ is defined in (5).

Proof Since $d_i^k \in X_i, \ 1 \leq i \leq p$ is a minimizer of problem (6) with (7) and f is differentiable, we have $g(x^k + d_i^k)^{\mathrm{T}} d_i^k = 0$. Then it follows from Assumption 2 and relationship (4) that

$$(g^k)^{\mathrm{T}} d^k = \sum_{i=1}^{p} (g(x^k) - g(x^k + d_i^k))^{\mathrm{T}} d_i^k \ \leq -K \sum_{i=1}^{p} ||d_i^k||_2^2 \ \leq -\frac{K}{\tau} ||d^k||_2^2. \tag{11}$$

This completes the proof.

At the k-th iteration of PSCL, the Armijo backtracking line search [39] determines a step size $\alpha_k = \eta\rho^l$, where $\eta > 0$ is given and l is the smallest integer satisfying the condition:

$$f(x^k + \eta\rho^l d^k) \leq f(x^k) + \sigma\eta\rho^l (g^k)^{\mathrm{T}} d^k, \tag{12}$$

where $\sigma, \rho \in (0, 1)$. The existence and boundedness of the step size α_k are ensured by the standard analysis. We include the proofs for completeness.

Lemma 2 *Suppose that Assumption 2 holds. Then*

$$f(x^k + \alpha d^k) \leq f(x^k) + \sigma\alpha(g^k)^{\mathrm{T}} d^k$$

holds for any $\alpha \in \left[0, \frac{2K(1-\sigma)}{L\tau}\right]$, *where* τ *is defined by* (5).

Proof Using the Lipschitz continuity of ∇f (namely g) and Lemma 1, we derive, for any $\alpha \in \left[0, \frac{2K(1-\sigma)}{L\tau}\right]$,

$$\begin{aligned} & f(x^k + \alpha d^k) - f(x^k) - \sigma\alpha(g^k)^{\mathrm{T}} d^k \\ \leq\ & (1-\sigma)\alpha(g^k)^{\mathrm{T}} d + \frac{L}{2}||d^k||_2^2\alpha^2 \ \leq\ \alpha\left[\frac{L}{2}\alpha - \frac{K(1-\sigma)}{\tau}\right]||d^k||_2^2 \ \leq\ 0. \end{aligned}$$

This completes the proof.

Lemma 3 *Suppose that Assumption 2 holds. Then we have*

$$\alpha_k \geq \min\left\{\frac{2K(1-\sigma)\rho}{L\tau}, \eta\right\},$$

where τ *is defined by* (5).

Proof The lemma holds if $\alpha_k = \eta$ satisfies the Armijo condition (12). Otherwise, we obtain:

$$f\left(x^k + \frac{\alpha_k}{\rho} d^k\right) > f(x^k) + \sigma\frac{\alpha_k}{\rho}(g^k)^{\mathrm{T}} d^k. \tag{13}$$

The Lipschitz continuity of ∇f yields

$$f\left(x^k + \frac{\alpha_k}{\rho} d^k\right) \leq f(x^k) + \frac{\alpha_k}{\rho}(g^k)^{\mathrm{T}} d^k + \frac{L}{2}||d^k||_2^2\left(\frac{\alpha_k}{\rho}\right)^2. \tag{14}$$

It follows from (13) and (14) that

$$\frac{\alpha_k}{\rho}(g^k)^{\mathrm{T}} d^k + \frac{L}{2}||d^k||_2^2\left(\frac{\alpha_k}{\rho}\right)^2 \geq \sigma\frac{\alpha_k}{\rho}(g^k)^{\mathrm{T}} d^k.$$

Then the boundedness of α_k holds by combining this inequality with Lemma 1.

Lemma 3 provides a lower bound for the line search step size α_k. According to the definition of τ in (5), this lower bound is independent of the number of blocks p in the non-overlapping case or when there is only overlapping between two immediate adjacent blocks. The next theorem establish the convergence of the PSCL method.

Theorem 3 *Suppose that Assumption 2 holds. Then the sequence $\{x^k\}_{k\in N}$ generated by the PSCL method satisfies*

$$\lim_{k\to\infty} \nabla f(x^k) = 0, \tag{15}$$

namely, the sequence $\{x^k\}$ converges to the global minimizer.

Proof It follows from Lemma 1 and the Armijo condition (12) that

$$f(x^k + \alpha_k d^k) \le f(x^k) - \sigma\alpha_k K \sum_{i=1}^{p} ||d_i^k||_2^2, \tag{16}$$

which yields $\sum_{k=0}^{\infty}\sum_{i=1}^{p} ||d_i^k||_2^2 < \infty$ due to the boundedness of $\{\alpha_k\}_{k\in N}$. Therefore, we obtain

$$\lim_{k\to\infty} d_i^k = 0,\ 1 \le i \le p. \tag{17}$$

Using the first-order optimality condition of the subproblems (6) and Assumption 2, we have, for any $y_i \in X_i$,

$$\begin{aligned} |(g^k)^{\mathrm{T}} y_i| &= |(g^k)^{\mathrm{T}} y_i - g(x_i^k + d_i^k)^{\mathrm{T}} y_i| \\ &\le ||g(x^k) - g(x_i^k + d_i^k)||_2 ||y_i||_2 \ \le\ L||y_i||_2 ||d_i^k||_2. \end{aligned} \tag{18}$$

Combining the relationships (17) and (18),

$$\lim_{k\to\infty} (g^k)^{\mathrm{T}} y_i = 0,\ \text{for any}\ y_i \in X_i,\ 1 \le i \le p.$$

Consequently, we have

$$\lim_{k\to\infty} (g^k)^{\mathrm{T}} y = 0,\ \text{for any}\ y \in \mathbb{R}^n.$$

Suppose that $\{s^1, s^2, ..., s^n\}$ is an orthonormal basis of $\mathbb{R}^n$. Then we can obtain

$$\lim_{k\to\infty} (g^k)^{\mathrm{T}} s^i = 0,\ 1 \le i \le n,$$

which implies (15). The strongly convexity of $f(x)$ ensures the global convergence of the sequence $\{x^k\}_{k\in N}$ to the unique minimizer. This completes the proof.

Let x^* be the unique minimizer under Assumption 2. We next prove the linear convergence rate in terms of $e_k = f(x^k) - f(x^*)$ similar to the proofs in [40].

Theorem 4 *Suppose that Assumption 1 and 2 hold. There exists a constant $\gamma \in (0, 1)$ depending on $C_1, p, L, K, \sigma, \eta$ such that*

$$e_k \le \gamma e_{k-1} \le \gamma^k e_0,\ \textit{for any}\ k \ge 1.$$

Proof Denote $w_j^k = x^k + \alpha_k \sum_{i=1}^{j} d_i^k,\ 0 \le j \le p$. According to Assumption 2, it holds that

$$\frac{(g^{k+1} - g(x^*))^{\mathrm{T}}(x^{k+1} - x^*)}{||x^{k+1} - x^*||_2} \ge K \left[\frac{2}{L}(f(x^{k+1}) - f(x^*))\right]^{\frac{1}{2}}. \tag{19}$$

By Assumption 1, there exists $v_i \in X_i,\ 1 \le i \le p$ such that $x^{k+1} - x^* = \sum_{i=1}^{p} v_i$ and $(\sum_{i=1}^{p} ||v_i||_2^2)^{\frac{1}{2}} \le C_1 ||x^{k+1} - x^*||_2$. Then, we have

$$\begin{aligned}
&(g^{k+1} - g(x^*))^{\mathrm{T}}(x^{k+1} - x^*) = (g^{k+1})^{\mathrm{T}}(x^{k+1} - x^*) = \sum_{i=1}^{p}(g^{k+1})^{\mathrm{T}} v_i \\
&= \sum_{i=1}^{p}(g(x^k + \alpha_k \sum_{j=1}^{p} d_j^k) - g(x^k))^{\mathrm{T}} v_i + \sum_{i=1}^{p}(g(x^k) - g(x^k + d_i^k))^{\mathrm{T}} v_i \\
&= \sum_{i=1}^{p}\sum_{j=1}^{p}(g(w_j^k) - g(w_{j-1}^k))^{\mathrm{T}} v_i + \sum_{i=1}^{p}(g(x^k) - g(x^k + d_i^k))^{\mathrm{T}} v_i \\
&\le \alpha_k L p \left(\sum_{i=1}^{p} ||d_i^k||_2^2\right)^{\frac{1}{2}} \left(\sum_{i=1}^{p} ||v_i||_2^2\right)^{\frac{1}{2}} + L \sum_{i=1}^{p} ||d_i^k||_2 ||v_i||_2 \\
&\le L(\alpha_k p + 1) \left(\sum_{i=1}^{p} ||d_i^k||_2^2\right)^{\frac{1}{2}} \left(\sum_{i=1}^{p} ||v_i||_2^2\right)^{\frac{1}{2}} \\
&\le C_1 L\,(\alpha_k p + 1) \left(\sum_{i=1}^{p} ||d_i^k||_2^2\right)^{\frac{1}{2}} ||x^{k+1} - x^*||_2.
\end{aligned}$$

Due to (16), the Armijo line search gives

$$\frac{(g^{k+1} - g(x^*))^{\mathrm{T}}(x^{k+1} - x^*)}{||x^{k+1} - x^*||_2} \le C_1 L(\alpha_k p + 1) \left[\frac{1}{\sigma K \alpha_k}(f(x^k) - f(x^{k+1}))\right]^{\frac{1}{2}}.$$

Combining the above relationship and (19), we obtain $e_{k+1} \le \hat{\zeta}_k (e_k - e_{k+1})$, where $\hat{\zeta}_k = \frac{C_1^2(\alpha_k p+1)^2 L^3}{2\sigma K^3 \alpha_k}$. Then we have $e_{k+1} \le \frac{\hat{\zeta}_k}{\hat{\zeta}_k + 1} e_k$. Lemma 3 implies that $\min\left\{\frac{2K(1-\sigma)\rho}{L\tau}, \eta\right\} \le \alpha_k \le \eta$. Therefore, there exists $\gamma \in (0, 1)$ such that $e_{k+1} \le \gamma e_k$. This completes the proof.

3 The General Convex Case

In this section, we consider the convergence properties of the PSCL method for solving (1) with $h(x) \neq 0$. In this case, the surrogate function takes the following form

$$\varphi_i^k(d_i) := \nabla f(x^k)^{\mathrm{T}} d_i + \frac{1}{2\lambda_i}||d_i||_2^2 + h(x^k + d_i), \tag{20}$$

where the parameter λ_i are constants only related to the convex part f. Without loss of generality, we assume

$$\lambda = \max_{1 \le i \le p} \lambda_i, \quad \underline{\lambda} = \min_{1 \le i \le p} \lambda_i. \tag{21}$$

We make the following assumption for $\varphi(x)$.

Assumption 5 *(i) The functions $f(x)$ and $h(x)$ are convex and φ is bounded from below. The function f is differentiable and its gradient $\nabla f(x)$ is Lipschitz continuous with a constant L.*
(ii) There exists a constant $Q > 0$ for the function $h(x)$ such that

$$|h(x) - h(y)| \leq Q||x - y||_2, \qquad \textit{for any}\, x, y \in \mathbb{R}^n. \tag{22}$$

(iii) Given a domain decomposition satisfying Assumption 1, the function $h(x)$ satisfies, for any $x, y \in \mathbb{R}^n$, there exist $y_i \in X_i$ $(i = 1, ..., p)$ such that

$$h(x + y) - h(x) = \sum_{i=1}^{p}(h(x + y_i) - h(x)). \tag{23}$$

Assumption 5 (iii) describes the separability of $h(x)$ under certain domain decomposition $X = \sum_{i=1}^{p} X_i$. Assumption 5 (iii) holds if $h(x)$ is separable and the domain decomposition is grouped by the coordinates of x. Specifically, we point out that Assumption 5 (ii) and (iii) hold, if the nonsmooth part $h(x)$ takes the form $||x||_1$.

Proposition 1 *Assumption 5 (ii) and (iii) hold for $h(x) = ||x||_1$.*

Proof (1) Let $y_{(j)}$ be the j-th entry of y. By using the triangle inequality, we have, for any $x, y \in \mathbb{R}^n$,

$$|h(x) - h(y)| \leq \sum_{j=1}^{n} \big||x_{(j)}| - |y_{(j)}|\big| \leq \sum_{j=1}^{n} |x_{(j)} - y_{(j)}| \leq \sqrt{n}||x - y||_2.$$

(2) Assumption 1 implies that there exist $y_i \in X_i$ such that $y = \sum_{i=1}^{p} y_i$ and there is only one non-zero element in $\{y_{1,(j)}, ..., y_{p,(j)}\}$, where $y_{i,(j)}$ is the j-th component of y_i. Hence, we have $y_{i,(j)} = 0$ if $j \notin \mathcal{J}_i$, $\sum\limits_{i=1}^{p} y_{i,(j)} = y_{(j)}$, and

$$\sum_{j=1}^{n} (|x_j + y_j| - |x_j|) = \sum_{i=1}^{p} \left(\sum_{j \in \mathcal{J}_i} (|x_j + y_j| - |x_j|) \right),$$

which gives

$$\begin{aligned} h(x + y) - h(x) = ||x + y||_1 - ||x||_1 = \sum_{i=1}^{p} \left(\sum_{j \in \mathcal{J}_i} (|x_j + y_j| - |x_j|) \right) \\ = \sum_{i=1}^{p} \left(\sum_{j \in \mathcal{J}_i} (|x_j + y_j| - |x_j|) + \sum_{j \notin \mathcal{J}_i} (|x_j| - |x_j|) \right) \\ = \sum_{i=1}^{p}(||x + y_i||_1 - ||x||_1) = \sum_{i=1}^{p}(h(x + y_i) - h(x)). \end{aligned}$$

This completes the proof.

Suppose that Assumption 5 (i) holds. It is well known that x^* is a minimizer of (1) if and only if

$$\nabla f(x^*)^{\mathrm{T}}(y - x^*) + h(y) - h(x^*) \geq 0, \quad \text{for any}\, y \in \mathbb{R}^n. \tag{24}$$

Define

$$\Delta_k = (d^k)^{\mathrm{T}} \nabla f(x^k) + p\left(h\left(x^k + \frac{1}{p}d^k\right) - h(x^k)\right). \tag{25}$$

Given some constants $\eta > 0$ and $\sigma, \rho \in (0,1)$, the Armijo backtracking line search chooses a step size

$$\alpha_k = \eta\rho^l, \tag{26}$$

where l is the smallest positive integer satisfied the condition

$$\varphi(x^k + \eta\rho^l d^k) \leq \varphi(x^k) + \sigma\eta\rho^l \Delta_k. \tag{27}$$

The next lemma shows the relationship between Δ_k and $||d^k||_2^2$.

Lemma 4 *It holds that* $\Delta_k \leq -\frac{1}{\lambda\tau}||d^k||_2^2$*, where* λ *and* τ *are defined by* (21) *and* (5)*, respectively.*

Proof It follows from the first-order optimality condition of the subproblems (6) that

$$y_i^{\mathrm{T}}\left(\nabla f(x^k) + \frac{1}{\lambda_i}d_i^k\right) + h(x^k + d_i^k + y_i) - h(x^k + d_i^k) \geq 0, \text{ for any}\, y_i \in X_i. \tag{28}$$

Let $y_i = -d_i^k$. The following inequalities holds for any $i = 1, ..., p$,

$$(d_i^k)^{\mathrm{T}} \nabla f(x^k) + h(x^k + d_i^k) - h(x^k) \leq -\frac{1}{\lambda_i}||d_i^k||_2^2.$$

Using the convexity of $h(x)$ and the inequality (4), we can derive

$$\begin{aligned}\Delta_k &= (d^k)^{\mathrm{T}} \nabla f(x^k) + p\left(h\left(x^k + \frac{1}{p}d^k\right) - h(x^k)\right) \\ &\leq \sum_{i=1}^{p}\left[(d_i^k)^{\mathrm{T}} \nabla f(x^k) + h(x^k + d_i^k) - h(x^k)\right] \\ &\leq -\sum_{i=1}^{p}\frac{1}{\lambda_i}||d_i^k||_2^2 \leq -\frac{1}{\lambda\tau}||d^k||_2^2.\end{aligned} \tag{29}$$

This completes the proof.

It can be shown that the Armijo condition (26) holds for certain step sizes and α_k is bounded from below.

Lemma 5 *Suppose that Assumption 5 holds. It holds*

$$\begin{aligned}&\varphi(x^k + \alpha d^k) \leq \varphi(x^k) + \sigma\alpha\Delta_k, \quad \textit{for any}\, \alpha \in \left[0, \min\left\{\frac{1}{p}, \frac{2(1-\sigma)}{\lambda L\tau}\right\}\right]; \\ &\alpha_k \geq \min\left\{\frac{2(1-\sigma)\rho}{\lambda L\tau}, \eta\right\}, \quad \textit{for any}\, \eta \in \left(0, \frac{1}{p}\right], \quad k \in N.\end{aligned} \tag{30}$$

The proof of Lemma 5 is similar to those of Lemma 2 and 3 and hence it is omitted. We next prove the global convergence of the PSCL method.

Theorem 6 *Suppose that Assumptions 1 and 5 hold. Let $\{x^k\}$ be a sequence generated by the PSCL method. For any given $y \in \mathbb{R}^n$, it holds*

$$\liminf_{k\to\infty} \left[y^{\mathrm{T}}\nabla f(x^k) + h(x^k+y) - h(x^k)\right] \geq 0. \tag{31}$$

Moreover, we have $\lim_{k\to\infty} \varphi(x^k) = \varphi^*$, where φ^* is the minimum of problem (1).

Proof Using Lemma 4 and the Armijo condition (26), we have

$$\varphi(x^k + \alpha_k d^k) \leq \varphi(x^k) - \frac{\sigma\alpha_k}{\lambda}\sum_{i=1}^{p} ||d_i^k||_2^2.$$

Since $\varphi(x)$ is bounded from below and $\{\alpha_k\}_{k\in N}$ is bounded, we obtain $\sum_{k=0}^{\infty} ||d^k||_2^2 < \infty$, which implies

$$\lim_{k\to\infty} d^k = 0. \tag{32}$$

Using Assumption 5 and the optimality condition (28), we obtain, for any $y \in \mathbb{R}^n$,

$$\begin{aligned}
&y^{\mathrm{T}}\nabla f(x^k) + h(x^k+y) - h(x^k)\\
&= \left(y^{\mathrm{T}}\nabla f(x^k) + h(x^k+d^k+y) - h(x^k)\right) + \left(h(x^k+y) - h(x^k+d^k+y)\right)\\
&\geq \sum_{i=1}^{p}\left(y_i^{\mathrm{T}}\left(\nabla f(x^k) + \frac{1}{\lambda_i}d_i^k\right) + h(x^k+d_i^k+y_i) - h(x^k)\right) - \sum_{i=1}^{p}\frac{1}{\lambda_i}y_i^{\mathrm{T}}d_i^k - Q||d^k||_2\\
&\geq \sum_{i=1}^{p}\left(y_i^{\mathrm{T}}\left(\nabla f(x^k) + \frac{1}{\lambda_i}d_i^k\right) + h(x^k+d_i^k+y_i) - h(x^k+d_i^k)\right)\\
&\quad + \sum_{i=1}^{p}(h(x^k+d_i^k) - h(x^k)) - \sum_{i=1}^{p}\frac{1}{\lambda_i}||y_i||_2||d_i^k||_2 - Q||d^k||_2\\
&\geq -\sum_{i=1}^{p}\left(Q||d_i^k||_2 + \frac{1}{\underline{\lambda}}||y_i||_2||d_i^k||_2\right) - Q||d^k||_2\\
&\geq -\left(\sqrt{p}QC_1 + \frac{C_1^2}{\underline{\lambda}}||y|| + Q\right)||d^k||_2,
\end{aligned} \tag{33}$$

where the last inequality follows from the inequality (3). The inequality (33) together with (32) gives (31).

Recalling the convexity of f, we have, for any $y \in \mathbb{R}^n$,

$$\varphi(x^k+y) - \varphi(x^k) \geq y^{\mathrm{T}}\nabla f(x^k) + h(x^k+y) - h(x^k). \tag{34}$$

Since $\varphi(x)$ is bounded below and $\{\varphi(x^k)\}$ is monotonically decreasing, $\lim_{k\to\infty}\varphi(x^k)$ exists. Without loss of generality, we assume that $\lim_{k\to\infty}\varphi(x^k) = \tilde{\varphi}$. Taking limits from both sides of (34) and using (33), we obtain that $\varphi(y) \geq \tilde{\varphi}$ holds for any $y \in \mathbb{R}^n$, which implies $\tilde{\varphi} = \varphi^*$. This completes the proof.

If the level set $\{x \mid \varphi(x) \leq \varphi(x^0)\}$ is bounded in Theorem 6, any accumulation point of $\{x^k\}$ is a minimizer of problem (1). The boundedness of the level set can be guaranteed if $f(x)$ is bounded from below and the nonsmooth part $h(x)$ is of the form $||x||_1$.

Proposition 2 *Suppose that Assumptions 1 and 5 hold. Let $\{x^k\}$ be a sequence generated by the PSCL method starting from $x^0 \in \mathbb{R}^n$. If the level set $\{x \mid \varphi(x) \leq \varphi(x^0)\}$ is bounded, then any accumulation point of $\{x^k\}$ is a minimizer of problem (1).*

Proof It follows from the monotonicity of $\{\varphi(x^k)\}$ and the boundedness of the level set that the sequence $\{x^k\}$ is bounded. Therefore, there exists accumulation point x^*. According to (31), we have $y^{\mathrm{T}}\nabla f(x^*) + h(x^* + y) - h(x^*) \geq 0$ for any $y \in \mathbb{R}^n$, which is the first-order optimality condition (24). Hence, x^* is a minimizer of problem (1) due to the convexity of $\varphi(x)$. This completes the proof.

Remark 1 Two specific formulations of problem (1) are discussed in [11,12]. They showed that any accumulation point of the sequence generated by their algoirthm is a minimizer of problem (1), under the same assumptions as Theorem 3 and an additional assumption called coercive condition. Note that the coercive condition is not needed in our proof.

Remark 2 In [9–12,21], it requires that $\lambda < \frac{2}{L}$, where L is the Lipschitz constant of $\nabla f(x)$. We show that our step size is $O(1/\tau)$ at the worst case without any assumption on λ. If $\lambda < \frac{2}{L}$ holds, the global convergence of the PSCL method using a step size not less than $\frac{1}{\tau}$ is guaranteed.

4 Numerical Implementation

In this section, we demonstrate the efficiency of the PSCLN and PSCLO methods for solving LASSO [41]:

$$\min_{x\in\mathbb{R}^n} \ \varphi(x) := \frac{1}{2}||Ax - b||_2^2 + \mu||x||_1, \tag{35}$$

where $A \in \mathbb{R}^{m\times n}$ and μ is a parameter. We compare PSCLN and PSCLO with the classic PSC method, the parallel versions of FISTA [2] using backtracking line search and FPC_BB [16] by computing the gradient in parallel (denoted by P-FISTA, and P-FPC_BB, respectively) and the GRock method[1]. The potential of PSCLN and PSCLO is further illustrated in solving two huge scale problems. Our experiments are run on a computer cluster LSSC-III at the State Key Laboratory of Scientific and Engineering Computing (LSEC), CAS. This cluster has 282 nodes. Every node has two quad-core 2.67GHz CPUs (Intel X5550) and 24 GB memory.

4.1 Algorithmic Issues of PSCL

The non-overlapping domain decomposition scheme is designed as follows. Suppose that the variables $x \in \mathbb{R}^n$ are divided into blocks as

$$x = (x_{\mathcal{I}_1}^{\mathrm{T}}, x_{\mathcal{I}_2}^{\mathrm{T}}, ..., x_{\mathcal{I}_p}^{\mathrm{T}})^{\mathrm{T}}, \tag{36}$$

[1] Here an old version of GRock is used. We notice that a robust new version of GRock is released in November 2014 (http://www.math.ucla.edu/∼wotaoyin/papers/GRock/cdes.html), which is reported to be much efficient than the old version.

where $x_{\mathcal{I}}$ contains the components of x corresponding to the index set I. Similarly, the matrix A is also split by columns according to the division of x as

$$A = [A_{\mathcal{I}_1}, A_{\mathcal{I}_2}, ..., A_{\mathcal{I}_p}],$$

where $A_{\mathcal{I}}$ contains the columns of A corresponding to the index set I. The subsets $\mathcal{I}_1, ..., \mathcal{I}_p$ are chosen to have the same size.

Our overlapping domain decomposition scheme is chosen as follows. We split $\mathcal{I}_i$ into three parts $\mathcal{I}_{l_i}, \mathcal{I}_{c_i}$ and $\mathcal{I}_{r_i}$ by the order of coordinates. Then a new subset $\tilde{\mathcal{I}}_i$ of coordinates is constructed from $\mathcal{I}_{r_{i-1}}, \mathcal{I}_i$ and $\mathcal{I}_{l_{i+1}}$. More specifically, we have

$$\begin{aligned}
\tilde{\mathcal{I}}_1 &= \mathcal{I}_1 \bigcup \mathcal{I}_{l_2}, \\
\tilde{\mathcal{I}}_i &= \mathcal{I}_{r_{i-1}} \bigcup \mathcal{I}_i \bigcup \mathcal{I}_{l_{i+1}}, \; 2 \le i \le p-1 \\
\tilde{\mathcal{I}}_p &= \mathcal{I}_{r_{p-1}} \bigcup \mathcal{I}_p.
\end{aligned}$$

Consequently, two immediately adjacent blocks have a overlapping part, which is denoted by $\mathcal{O}_i = \mathcal{I}_{r_i} \bigcup \mathcal{I}_{l_{i+1}}, 1 \le i \le p-1$. The sizes of $\mathcal{O}_i(1 \le i \le p-1)$ are set to the same. When $\mathcal{O}_i = 0$ (for all i), it reduces to the non-overlapping case.

The computation of Ax^k done in parallel. Communication between nodes is needed for assembling d^k and computing the step size α_k. Different from the P-FISTA method, the parameters λ_i $(1 \le 1 \le p)$ are not necessary the same among every blocks in the PSCL method. We choose them as the following Barzilai-Borwein step [1]:

$$\lambda_i^k = \begin{cases} \min\left\{1 + 1.665\left(1 - \frac{m}{\eta}\right), 1.999\right\}, & k = 0, \\ \max\left\{\frac{1.7\|x^k_{\tilde{\mathcal{I}}_i} - x^{k-1}_{\tilde{\mathcal{I}}_i}\|_2^2}{(x^k_{\tilde{\mathcal{I}}_i} - x^{k-1}_{\tilde{\mathcal{I}}_i})^{\mathrm{T}}(g^k_{\tilde{\mathcal{I}}_i} - g^{k-1}_{\tilde{\mathcal{I}}_i})}, 1\right\}, & k > 0. \end{cases}$$

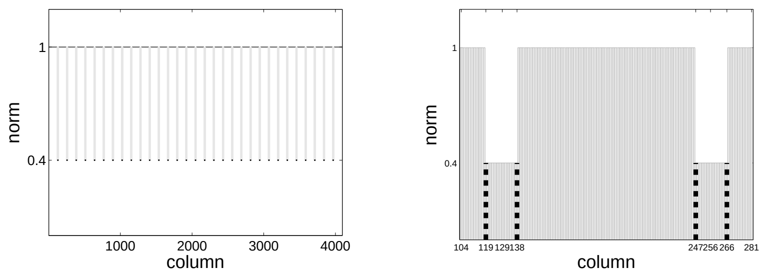
For the Armijo line search (27), we set $\rho = 0.5$, $\sigma = 0.5$ and $\eta = 2$.

4.2 Data Preparation

In our numerical experiments, the data matrix A and the exact solution x^* are generated randomly. Then the right-hand-side b is calculated in the same fashion as [32]. The detailed information of the data sets is described in Table 1. Note that there are two types of data sets in Table 1. For the data sets I, III, IV and VII, the 2-norm of each column of the matrix A is set to one. For the data sets II, V, VI and VIII, the columns of A are designed to have a special structure in favor of the PSCLO method. For example, the matrix A in the data set II has 32 blocks. Each immediately adjacent pair of blocks has 20 overlapping columns. The 2-norm of the columns in the overlapping regions are 0.4 and the 2-norm of other columns is 1. For brevity, we use "32/20/0.4" to describe the structure of matrix A. The distribution of the norms of the columns is illustrated in Figure 1(a). The vertical lines in Figure 1(a) show the difference between the norms of the overlapping and non-overlapping parts. An amplified view of the difference is depicted in Figure 1(b) more clearly. A full block of variables is located between the first and the last dashed vertical lines. The regions between the first two dashed vertical lines and between the last two dashed vertical lines represent the overlapping domains. Our numerical experience shows that the PSCLO method works better than the PSCLN method under this special pattern.

Table 1 Test data sets for LASSO

Data Set	A Type	A Size	μ	Sparsity of x^*	Size	Overlap Str.
I	Gaussian	2048×4096	0.1	200	64.00MB	—
II	Gaussian	2048×4096	0.1	200	64.00MB	32\20\0.4
III	Gaussian	10240×20480	0.05	2000	1.56 GB	—
IV	Gaussian	15360×30720	0.01	2000	3.52 GB	—
V	Gaussian	10240×20480	0.1	1000	1.56 GB	128\24\0.4
VI	Gaussian	5120×10240	0.1	500	0.39 GB	128\12\0.5
VII	Gaussian	50000×100000	0.01	4000	37.25 GB	—
VIII	Gaussian	50000×100000	0.01	4000	37.25 GB	160\100\0.4



(a) Column Norm of A in Data Set II (b) Column Norm of the Second Block

Fig. 1 The structure of the matrix A in the data set II

The initial point x^0 is set to 0. All algorithms are terminated if the stopping criterion

$$||x^k - x^*||_2 < \epsilon ||x^*||_2$$

is satisfied for a tolerance $\epsilon = 10^{-7}$. The blocks used by both PSCLN or PSCLO have the same size. For PSCLO, the size of the overlapping blocks between two immediately adjacent blocks is the same as the pattern of the matrix A on the data sets II, V, VI and VIII, and it is set to 24, 36 and 100 (around $n/1000$), respectively, on the data sets III, IV and VII.

4.3 Numerical Comparisons between PSCL and PSC

The PSCLN and PSCLO methods using a fixed step size $\frac{1}{p}$ are denoted by PSCAN and PSCAO, respectively. They are equivalent to the classic PSC methods. The maximal number of iterations is set to 200. We compare the performance of PSCAN, PSCAO, PSCLN and PSCLO on the data sets I and II. The relative error $||x^k - x^*||_2$ versus iteration history is shown in Figure 2. We can see that PSCAN and PSCAO converge slowly and the relative error is of order $O(10^{-1})$ after 200 iterations. Both PSCLN and PSCLO achieve an accuracy of order $O(10^{-7})$ within 100 iterations. In particular, PSCLO takes fewer iterations than PSCLN on the data set II.

The step sizes used at each iteration of the four algorithms are presented in Figure 3. The red dash lines in Figure 3 correspond to the step size $\frac{1}{p}$ in PSCAN and PSCAO. It is clear that the step sizes determined by the Armijo rule can be much larger than $\frac{1}{p}$. This partly explains why PCSLN and PSCLO perform better than PSCAN and PSCAO.

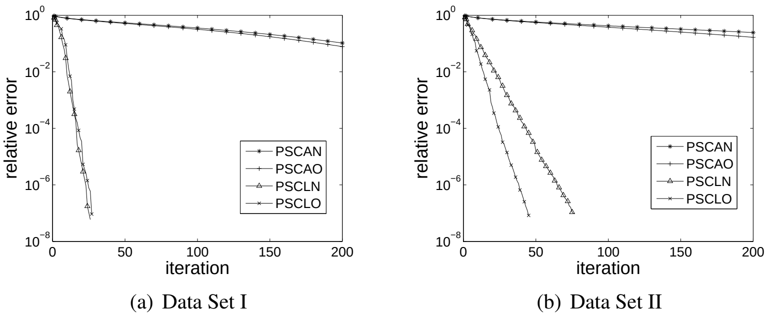


Fig. 2 The relative error history on the data sets I and II

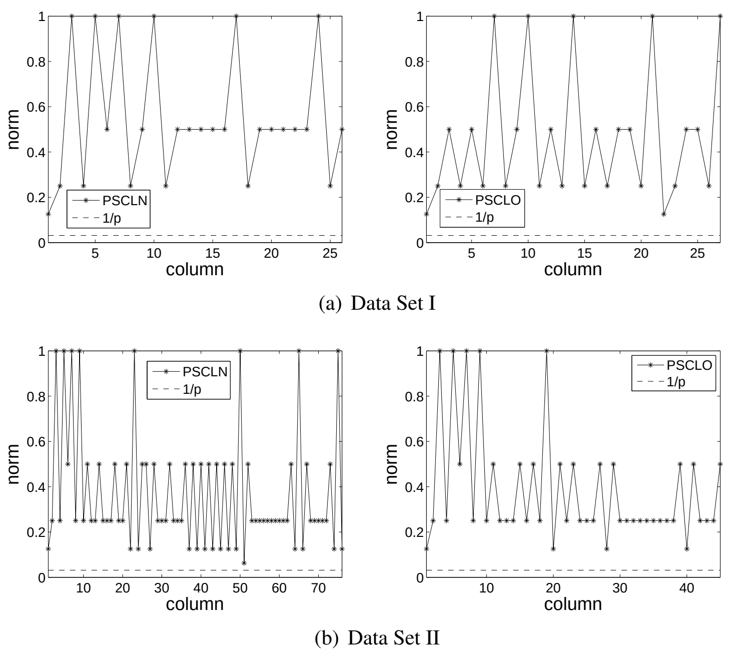


Fig. 3 Illustration of the step sizes chosen by the Armijo rule on the data sets I and II

Figure 3 shows that the step sizes taken by PSCL are mostly 0.25 or 0.5. Therefore, we compare PSCLN and PSCLO with PSCAN and PSCAO using a fixed step size 0.25 and 0.5 instead of $\frac{1}{p}$. The computational results are summarized in Table 2. From the table, we can observe that PSCLO and PSCLN are still more efficient than PSCAN and PSCAO. Moreover, the overlapping scheme PSCLO is better than the non-overlapping scheme PSCLN on the data set II.

Table 2 Numerical results of PSC using $\alpha_k = 0.25, 0.5$

data	Iteration No.	Total Time (s)	Comm. Time (s)	Iteration No.	Total Time (s)	Comm. Time (s)
	PSCAN			PSCAO		
	$\alpha_k = 0.25$					
I	122	0.1243	0.0189	116	0.1440	0.0294
II	146	0.1449	0.0284	122	0.1357	0.0242
	$\alpha_k = 0.5$, failed to find the solutions					
	PSCLN			PSCLO		
I	26	0.0236	0.0092	27	0.0298	0.0090
II	76	0.0616	0.0240	45	0.0411	0.0128

4.4 Comparisons of Complexity with the State-Of-The-Art Algorithms

In this subsection, we compare the complexities of the arithmetic operations of P-FISTA, P-FPC_BB, GRock, PSCLN and PSCLO for LASSO. Here we analyze P-FISTA using a constant step size and similar results can be obtained for P-FISTA using backtracking line search. Among all of the above parallel algorithms, P-FISTA is the simplest one. The computational cost of each iteration of P-FISTA includes two matrix-vector multiplications $O(mn/p)$ for computing the gradient of the smooth part and the shrinkage step $O(n/p)$, which is dominated by the matrix-vector multiplication. P-FPC_BB, PSCLN and PSCLO take inexact or exact Armijo line search strategies compared with P-FISTA. These line search strategies are efficient and always terminate after a few trials. Each line search step evaluates one objective function value whose cost is $O(n/p)$. Different from the others, a sorting algorithm is needed in GRock for the greedy strategy, and the cost of the quick sort is $O((n/p)\log(n/p))$. Consequently, the computational cost per iteration of all algorithms is $O(mn/p)$.

As for the communication cost, P-FISTA needs $O(m\log(p))$ for the matrix-vector multiplication in each iteration. The evaluation of the objective function in the Armijo line searches (P-FPC_BB, PSCLN and PSCLO) and the greedy strategy (GRock) is $O(\log(p))$ and $O(p\log(p))$, respectively. PSCLO also needs $O(\mathcal{O}_i)$ for updating the iteration but it is cheaper than the cost in matrix-vector multiplication. Hence, the communication cost of all algorithms in each iteration is $O(m\log(p))$.

Therefore, the complexity per iteration of PSCLN and PSCLO is comparable with that of the state-of-the-art algorithms.

4.5 Numerical Comparisons to the State-Of-The-Art Algorithms

In this subsection, we compare the numerical performance of P-FISTA, P-FPC_BB, GRock, PSCLN and PSCLO on the data sets III, IV, V and VI using $1, 2, 4, \dots, 128$ processors, respectively. The total number of iterations, the total wall-clock time and the time spend on communication of these algorithms are reported in Table 3.

Since P-FISTA and P-FPC_BB are sequential algorithms whose operations are parallelized as many as possible, their number of iterations are the same when different numbers

Table 3 Numerical results of LASSO: iteration number, total time and communication time

data	core no.	Iteration No.					Total Time (s)					Comm. Time (s)				
		P-FISTA	P-FPC_BB	GRock	PSCLN	PSCLO	P-FISTA	P-FPC_BB	GRock	PSCLN	PSCLO	P-FISTA	P-FPC_BB	GRock	PSCLN	PSCLO
III	1	90	227	*5000	69	69	51.1716	86.3665	1638.6536	29.1741	29.1810	0.0041	0.0039	0.1041	0.0012	0.0013
	2	90	227	*5000	65	67	28.6598	48.4448	933.9151	15.5991	16.3330	0.0165	0.0171	0.2722	0.0062	0.0076
	4	90	227	*5000	75	69	15.1800	26.4313	518.4890	9.7874	8.9497	0.0291	0.0255	0.4490	0.0070	0.0062
	8	90	227	3433	74	70	8.2043	14.1053	190.9595	5.1537	5.1588	0.1353	0.0736	1.4160	0.0238	0.0105
	16	90	227	1777	74	72	4.1499	7.1174	50.5900	2.7352	2.5909	0.2112	0.1573	1.2846	0.0304	0.0523
	32	90	227	889	72	78	2.2301	3.7146	13.3647	1.3282	1.4389	0.2540	0.2141	0.9371	0.0991	0.1084
	64	90	227	484	73	81	1.2305	2.0154	3.9686	0.7288	0.8452	0.2786	0.2786	0.5727	0.0964	0.0944
	128	90	227	282	65	86	0.8020	1.1939	1.4028	0.3885	0.5545	0.2973	0.3041	0.3937	0.0865	0.1361
IV	1	99	197	*5000	107	107	193.8786	257.0118	4248.2967	144.7893	143.8059	0.0070	0.0051	0.1361	0.0027	0.0030
	2	99	197	*5000	114	112	97.7049	130.4537	2218.6784	79.7377	77.3608	0.0255	0.0188	0.4453	0.0108	0.0117
	4	99	197	4994	106	111	67.5096	92.0222	1619.1294	51.6309	54.6253	0.3063	0.0292	0.7350	0.0196	0.6081
	8	99	197	2568	104	102	27.6192	35.6501	320.8674	19.0977	19.5880	0.9565	0.3535	0.6588	0.2342	0.5440
	16	99	197	1271	109	113	13.8061	18.6568	81.3370	10.6888	11.6383	0.5585	0.7141	2.5432	0.4997	0.5244
	32	99	197	654	107	109	7.2666	9.5667	21.7961	5.4443	5.5921	0.7152	0.7222	1.2619	0.4483	0.3680
	64	99	197	349	116	106	3.9461	5.0214	6.1636	3.1111	2.9299	0.6010	0.4863	0.6760	0.2895	0.3128
	128	99	197	215	106	124	2.2421	2.7023	2.2111	1.5021	2.0532	0.5587	0.3739	0.4516	0.2452	0.3517
V	1	171	115	*5000	77	77	65.7541	39.1495	1581.2636	26.2855	26.3564	0.0079	0.0020	0.1048	0.0014	0.0014
	2	171	115	4267	82	86	36.7643	21.8967	758.1660	15.5775	16.3405	0.1269	0.0083	0.6577	0.0048	0.0052
	4	171	115	2175	80	81	20.0329	12.0982	214.4555	8.7594	8.8763	0.1149	0.0636	0.6097	0.1754	0.1585
	8	171	115	1118	82	77	10.8228	6.4566	59.1331	4.5887	4.3388	0.2837	0.0516	0.5717	0.0559	0.0539
	16	171	115	565	79	81	5.7849	3.2727	16.5432	2.2805	2.3192	0.4907	0.0794	1.7729	0.0760	0.0411
	32	171	115	305	80	77	3.1679	1.7308	4.4449	1.2319	1.2015	0.5279	0.1282	0.3545	0.1092	0.0969
	64	171	115	180	79	72	1.7967	0.9322	1.4442	0.6478	0.6333	0.4918	0.1191	0.2314	0.0916	0.1020
	128	171	115	110	73	47	1.2556	0.5883	0.5344	0.3620	0.2701	0.5609	0.1687	0.1364	0.0947	0.0691
VI	1	124	79	4227	56	56	12.2770	6.8332	336.2071	4.9732	4.9721	0.0042	0.0010	0.0697	0.0008	0.0007
	2	124	79	2112	56	52	6.8449	3.8013	93.7195	2.7366	2.5616	0.0121	0.0025	0.0670	0.0018	0.0018
	4	124	79	1110	59	58	3.7216	2.0919	27.2833	1.6322	1.6093	0.0523	0.0038	0.1096	0.0031	0.0073
	8	124	79	580	50	54	2.0332	1.1265	7.7368	0.7271	0.7897	0.0876	0.0153	0.1149	0.0100	0.0156
	16	124	79	307	52	52	1.0527	0.5748	2.1003	0.3859	0.3903	0.1061	0.0256	0.0841	0.0194	0.0203
	32	124	79	181	52	52	0.6315	0.3206	0.6899	0.2176	0.2224	0.1653	0.0433	0.0848	0.0331	0.0318
	64	124	79	103	55	47	0.4032	0.1872	0.2333	0.1319	0.1293	0.1742	0.0474	0.0529	0.0342	0.0368
	128	124	79	67	52	34	0.3438	0.1314	0.1105	0.0885	0.0701	0.2265	0.0588	0.0461	0.0409	0.0309

of cpu processors are used. The number of iterations of other three algorithms are not the same when the number of processors invoked are different. This is most obvious in GRock since the number of updated blocks at each iteration is equal to the number of processors. Table 3 shows that GRock is faster than P-FISTA and P-FPC_BB in most of cases, but it is slower than the PSCL algorithms. PSCLN and PSCLO always take fewer iterations than the others since they are more suitable for the structures of the large problems in our setting. They seem to be able to take advantage of the local information more than P-FISTA and P-FPC_BB and also preserve more global information than GRock. Consequently, they consumed less total wall-clock time and communication time than other algorithms. Moreover, PSCLO is faster than PSCLN on data sets V and VI when 128 processors are invoked.

Figure 4 shows the parallel speedup factor (PSF(p)) and parallel efficiency (PE(p)) of P-FISTA, P-FPC_BB, PSCLN and PSCLO. They are defined as

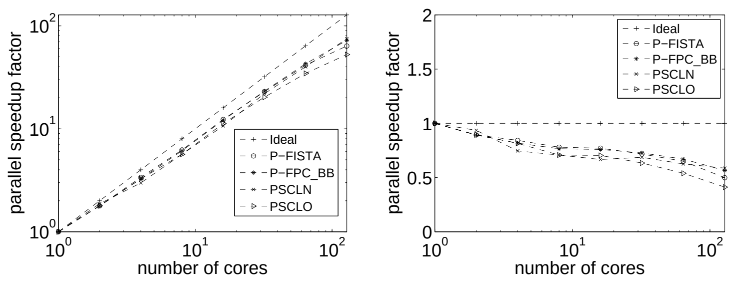
$$\text{PSF}(p) = \frac{T_1}{T_p}, \quad \text{PE}(p) = \frac{T_1}{pT_p}, \tag{37}$$

where T_p is the total time using p cores to run a algorithm. Since the number of updated blocks are different when different numbers of processors are used in GRock, the definitions (37) are not suitable to measure its efficiency. Hence, the figures on GRock are not reported.

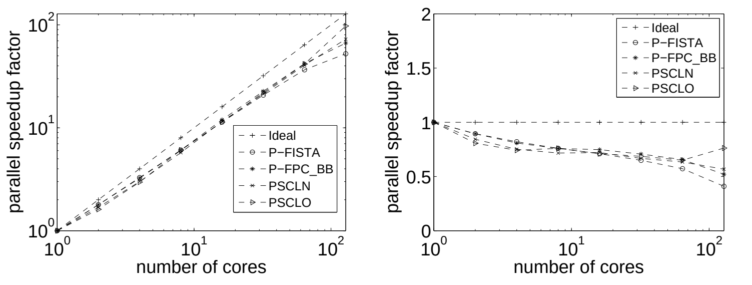
Finally, we test all algorithms on the huge data sets VII and VIII. A summary of computational results is presented in Table 4. We can observe the similar numerical performance as Table 3. The advantage of PSCLO can be seen from the data set VIII.

5 Conclusion

The SSC and PSC methods are two well known domain decomposition methods for solving partial differential equations. The SSC method is not suitable for parallel computation since



(a) Data Set III



(b) Data Set V

Fig. 4 Parallel analysis on the data sets III and V

Table 4 Numerical results on the large data sets

	P-FISTA	P-FPC_BB	GRock	PSCLN	PSCLO
data set/size	VII/37.25GB				
accuracy tolerance	1e-7				
core no.	160				
iteration no.	90	127	230	74	86
comm. time (s)	5.4461	5.5142	8.2352	3.2796	5.7410
relative error	8.9257e-08	6.6115e-08	9.3870e-08	7.3944e-08	1.4595e-08
total time (s)	20.1560	20.4863	24.7613	11.9848	16.9488
data set/size	VIII/37.25GB				
accuracy tolerance	1e-7				
core no.	160				
iteration no.	212	191	243	128	86
comm. time (s)	12.1959	7.7972	9.1687	5.0911	4.8956
relative error	9.5259e-08	9.5288e-08	9.7444e-08	8.3672e-08	9.1491e-08
total time (s)	35.5718	26.9156	26.6191	17.7854	14.7187

the blocks are updated sequentially. The PSC method may not be efficient due to the conservative step size in order to ensure global convergence of the algorithm. In order to accelerate the PSC method, we propose a new parallel line search subspace correction framework for composite convex optimization. Two versions called PSCLN and PSCLO are proposed depending on whether there are overlapping between two immediately adjacent blocks. Global convergence are established under mild conditions.

We compare our proposed algorithms with the state-of-the-art parallel optimization algorithms including P-FISTA, P-FPC_BB, and GRock for solving the LASSO problem. Our numerical results show that PSCLN and PSCLO can perform better than other algorithms to achieve similar accuracy. Although PSCLN is slightly better than PSCLO on most cases, PSCLO can perform better than PSCLN when the test problem has certain special structures. It remains an interesting topic to develop a more efficient general overlapping domain decomposition scheme and to investigate its efficiency on practical applications.

Acknowledgements The authors thank Prof. Wotao Yin for the insightful discussion, and Mr. Zhimin Peng for providing the source codes of GRock.

References

1. Barzilai, J., Borwein, J.M.: Two-point step size gradient methods. IMA Journal of Numerical Analysis **8**(1), 141–148 (1988)
2. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM Journal on Imaging Sciences **2**(1), 183–202 (2009)
3. Beck, A., Tetruashvili, L.: On the convergence of block coordinate descent type methods. SIAM Journal on Optimization **23**(4), 2037–2060 (2013)
4. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends® in Machine Learning **3**(1), 1–122 (2011)
5. Carstensen, C.: Domain decomposition for a non-smooth convex minimization problem and its application to plasticity. Numerical linear algebra with applications **4**(3), 177–190 (1997)
6. Deng, W., Lai, M.J., Peng, Z., Yin, W.: Parallel multi-block admm with o (1/k) convergence. arXiv preprint arXiv:1312.3040 (2013)
7. Fercoq, O., Richtárik, P.: Accelerated, parallel and proximal coordinate descent. arXiv preprint arXiv:1312.5799 (2013)
8. Fercoq, O., Richtárik, P.: Smooth minimization of nonsmooth functions with parallel coordinate descent methods. arXiv preprint arXiv:1309.5885 (2013)
9. Fornasier, M.: Domain decomposition methods for linear inverse problems with sparsity constraints. Inverse Problems **23**(6), 2505 (2007)
10. Fornasier, M., Kim, Y., Langer, A., Schönlieb, C.B.: Wavelet decomposition method for l_2/tv-image deblurring. SIAM Journal on Imaging Sciences **5**(3), 857–885 (2012)
11. Fornasier, M., Langer, A., Schönlieb, C.B.: A convergent overlapping domain decomposition method for total variation minimization. Numerische Mathematik **116**(4), 645–685 (2010)
12. Fornasier, M., Schönlieb, C.B.: Subspace correction methods for total variation and l_1-minimization. SIAM Journal on Numerical Analysis **47**(5), 3397–3428 (2009)
13. Friedman, J., Hastie, T., Tibshirani, R.: Regularization paths for generalized linear models via coordinate descent. Journal of statistical software **33**(1), 1 (2010)
14. Gabay, D., Mercier, B.: A dual algorithm for the solution of nonlinear variational problems via finite element approximation. Computers & Mathematics with Applications **2**(1), 17–40 (1976)
15. Glowinski, R., Marroco, A.: Sur l'approximation, par lments finis d'ordre un, et la rsolution, par pnalisation-dualit d'une classe de problmes de dirichlet non linaires. ESAIM: Mathematical Modelling and Numerical Analysis - Modlisation Mathmatique et Analyse Numrique **9**(R2), 41–76 (1975). URL `http://eudml.org/doc/193269`
16. Hale, E.T., Yin, W., Zhang, Y.: Fixed-point continuation for l_1-minimization: Methodology and convergence. SIAM Journal on Optimization **19**(3), 1107–1130 (2008)
17. He, B., Xu, H.K., Yuan, X.: On the proximal jacobian decomposition of alm for multipleblock separable convex minimization problems and its relationship to admm (2013)
18. He, B., Yuan, X.: On the direct extension of admm for multi-block separable convex programming and beyond: From variational inequality perspective
19. Hong, M., Wang, X., Razaviyayn, M., Luo, Z.Q.: Iteration complexity analysis of block coordinate descent methods. arXiv preprint arXiv:1310.6957 (2013)
20. Kyrola, A., Bickson, D., Guestrin, C., Bradley, J.K.: Parallel coordinate descent for l_1-regularized loss minimization. In: Proceedings of the 28th International Conference on Machine Learning (ICML-11), pp. 321–328 (2011)
21. Langer, A., Osher, S., Schönlieb, C.B.: Bregmanized domain decomposition for image restoration. Journal of Scientific Computing **54**(2-3), 549–576 (2013)
22. Li, X., Sun, D., Toh, K.C.: A schur complement based semi-proximal admm for convex quadratic conic programming and extensions. arXiv preprint arXiv:1409.2679 (2014)

23. Lin, Q., Lu, Z., Xiao, L.: An accelerated proximal coordinate gradient method and its application to regularized empirical risk minimization. arXiv preprint arXiv:1407.1296 (2014)
24. Lin, T., Ma, S., Zhang, S.: On the convergence rate of multi-block admm. arXiv preprint arXiv:1408.4265 (2014)
25. Lions, P.L.: On the schwarz alternating method. i. In: First international symposium on domain decomposition methods for partial differential equations, pp. 1–42. Paris, France (1988)
26. Lu, Z., Xiao, L.: On the complexity analysis of randomized block-coordinate descent methods. arXiv preprint arXiv:1305.4723 (2013)
27. Lu, Z., Xiao, L.: Randomized block coordinate non-monotone gradient method for a class of nonlinear programming. arXiv preprint arXiv:1306.5918 (2013)
28. Luo, Z.Q., Tseng, P.: On the convergence of the coordinate descent method for convex differentiable minimization. Journal of Optimization Theory and Applications **72**(1), 7–35 (1992)
29. Luo, Z.Q., Tseng, P.: On the linear convergence of descent methods for convex essentially smooth minimization. SIAM Journal on Control and Optimization **30**(2), 408–425 (1992)
30. Luo, Z.Q., Tseng, P.: Error bounds and convergence analysis of feasible descent methods: a general approach. Annals of Operations Research **46**(1), 157–178 (1993)
31. Nesterov, Y.: Efficiency of coordinate descent methods on huge-scale optimization problems. SIAM Journal on Optimization **22**(2), 341–362 (2012)
32. Peng, Z., Yan, M., Yin, W.: Parallel and distributed sparse optimization. preprint (2013)
33. Richtárik, P., Takáč, M.: Parallel coordinate descent methods for big data optimization. arXiv preprint arXiv:1212.0873 (2012)
34. Richtárik, P., Takáč, M.: Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. Mathematical Programming **144**(1-2), 1–38 (2014)
35. Scherrer, C., Tewari, A., Halappanavar, M., Haglin, D.: Feature clustering for accelerating parallel coordinate descent. In: NIPS, pp. 28–36 (2012)
36. Schwarz, H.A.: Ueber einige abbildungsaufgaben. Journal für die reine und angewandte Mathematik **70**, 105–120 (1869)
37. Shalev-Shwartz, S., Tewari, A.: Stochastic methods for l_1-regularized loss minimization. The Journal of Machine Learning Research **12**, 1865–1892 (2011)
38. Shevade, S.K., Keerthi, S.S.: A simple and efficient algorithm for gene selection using sparse logistic regression. Bioinformatics **19**(17), 2246–2253 (2003)
39. Sun, W., Yuan, Y.X.: Optimization theory and methods: nonlinear programming, vol. 1. springer (2006)
40. Tai, X.C., Xu, J.: Global and uniform convergence of subspace correction methods for some convex optimization problems. Mathematics of Computation **71**(237), 105–124 (2002)
41. Tibshirani, R.: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological) pp. 267–288 (1996)
42. Tseng, P., Yun, S.: A coordinate gradient descent method for nonsmooth separable minimization. Mathematical Programming **117**(1-2), 387–423 (2009)
43. Wang, X., Hong, M., Ma, S., Luo, Z.Q.: Solving multiple-block separable convex minimization problems using two-block alternating direction method of multipliers. arXiv preprint arXiv:1308.5294 (2013)
44. Wen, Z., Yin, W., Zhang, H., Goldfarb, D.: On the convergence of an active-set method for l_1 minimization. Optimization Methods and Software **27**(6), 1127–1146 (2012)
45. Zhang, H., Jiang, J., Luo, Z.Q.: On the linear convergence of a proximal gradient method for a class of nonsmooth convex minimization problems. Journal of the Operations Research Society of China **1**(2), 163–186 (2013)