Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Block coordinate descent with time perturbation for nonconvex nonsmooth problems in real-world studies[*#]

Rui LIU[1], Wei-chu SUN[2], Tao HOU[1], Chun-hong HU[‡1], Lin-bo QIAO[3]

[1]*Department of Oncology, The Second Xiangya Hospital of Central South University, Changsha 410011, China*

[2]*First Clinical Medical College, University of South China, Hengyang 421001, China*

[3]*College of Computer, National University of Defense Technology, Changsha 410073, China*

E-mail: liuruirui@csu.edu.cn; smsysun@foxmail.com; houtao@csu.edu.cn; huchunhong@csu.edu.cn; qiao.linbo@nudt.edu.cn

Received July 9, 2019; Revision accepted Sept. 16, 2019; Crosschecked Oct. 25, 2019

Abstract: The era of big data in healthcare is here, and this era will significantly improve medicine and especially oncology. However, traditional machine learning algorithms need to be promoted to solve such large-scale real-world problems due to a large amount of data that needs to be analyzed and the difficulty in solving problems with nonconvex nonlinear settings. We aim to minimize the composite of a smooth nonlinear function and a block-separable nonconvex function on a large number of block variables with inequality constraints. We propose a novel parallel first-order optimization method, called asynchronous block coordinate descent with time perturbation (ATP), which adopts a time perturbation technique that escapes from saddle points and sub-optimal local points. The details of the proposed method are presented with analyses of convergence and iteration complexity properties. Experiments conducted on real-world machine learning problems validate the efficacy of our proposed method. The experimental results demonstrate that time perturbation enables ATP to escape from saddle points and sub-optimal points, providing a promising way to handle nonconvex optimization problems with inequality constraints employing asynchronous block coordinate descent. The asynchronous parallel implementation on shared memory multi-core platforms indicates that the proposed algorithm, ATP, has strong scalability.

Key words: Convergence analysis; Asynchronous block coordinate descent method; Time perturbation; Nonconvex nonsmooth optimization; Real-world study
https://doi.org/10.1631/FITEE.1900341 **CLC number:** TP391

1 Introduction

In real-world studies (RWS), healthcare is coming to the era of big data, which will significantly benefit oncology (Khozin et al., 2017). The last decade has witnessed a growing interest in the potential benefits and relevance of RWS (Bertsimas et al., 2008; Li et al., 2018; Sun et al., 2019) with increasing large-scale datasets and rapid development of machine learning algorithms. It has been reported in the literature that the nonconvex regularized objective function usually yields better statistic results (Fan and Li, 2001). However, it is beyond the ability of traditional machine learning algorithms to solve such large-scale RWS problems due to the large amount of data that needs to be analyzed and the difficulty in solving such problems, especially the nonconvex settings and inequality constraints.

In this study, we focus on solving the following

‡ Corresponding author

* Project supported by the National Key R&D Program of China (No. 2018YFB2101100) and the National Natural Science Foundation of China (Nos. 61806216 and 61702533)

# A preliminary version was presented at IEEE 21st International Conference on High Performance Computing and Communications, jointly with IEEE 17th International Conference on Smart City and IEEE 5th International Conference on Data Science and Systems, October 10–12, 2019, China

ORCID: Chun-hong HU, http://orcid.org/0000-0003-3857-4598 nonconvex nonsmooth optimization problem with inequality constraints on a large number of block variables:

$$\begin{array}{ll}\min\limits_{\boldsymbol{x}\in\mathbb{R}^d} & h(\boldsymbol{x}) \triangleq f(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n) + \sum\limits_{i=1}^{n} r_i(\boldsymbol{x}_i) \\ \text{s.t.} & \boldsymbol{A}_1\boldsymbol{x}_1 + \boldsymbol{A}_2\boldsymbol{x}_2 + \ldots + \boldsymbol{A}_n\boldsymbol{x}_n \leq \boldsymbol{b},\end{array} \tag{1}$$

where $\boldsymbol{x}_i \in \mathbb{R}^{d_i}$, $\boldsymbol{x} = [\boldsymbol{x}_1; \boldsymbol{x}_2; \ldots; \boldsymbol{x}_n] \in \mathbb{R}^d$, $\sum_{i=1}^n d_i = d$, $\boldsymbol{A}_i \in \mathbb{R}^{p\times d_i}$, and $\boldsymbol{b} \in \mathbb{R}^p$. Function $r(\boldsymbol{x}) \triangleq \sum_{i=1}^n r_i(\boldsymbol{x}_i)$ is a separable lower semi-continuous function (possibly nonconvex nonsmooth) and function f is a smooth nonlinear function (possibly nonconvex). Note that $r_i(\boldsymbol{x}_i)$ can be an indicator function of a closed set $\boldsymbol{\chi}_i \in \mathbb{R}^{d_i}$ (possibly nonconvex). Problem (1) covers many interesting and important applications in RWS and problems from various fields such as distributed optimization and coordination (Bertsekas and Tsitsiklis, 1989; Li et al., 2019), statistics learning (Friedman et al., 2001), resource allocation (Xiao et al., 2004), and network utility maximization (Palomar and Chiang, 2006). One typical example is the constrained Lasso problem (James et al., 2012), which is obtained by setting $f = \frac{1}{2}\|\boldsymbol{Y} - \boldsymbol{D}^{\mathrm{T}}\boldsymbol{x}\|_2^2$ and $r_i = \lambda\|\boldsymbol{x}_i\|_1$, where $\boldsymbol{D} = [D_1, D_2, \ldots, D_N]$ whose i^{th} column D_i $(i = 1, 2, \cdots, n)$ is the i^{th} data sample, and $\lambda > 0$ is a balancing parameter.

With respect to escaping from sub-optimal points, a promising approach for solving problem (1) is graduated optimization (Hazan et al., 2016), under the assumption that f is separable convex and r_i is a simple indicator function with a closed convex set $\boldsymbol{\chi}_i$ $(i = 1, 2, \ldots, n)$. However, it is computationally prohibitive to solve large-scale problems, because we need to compute the Hessian matrix of the self-concordant barriers (Nesterov and Nemirovskii, 1994) at each iteration. Problem (1) is an extension to multi-term convex optimization problems (Qiao et al., 2016b, 2018; Shen et al., 2017, 2018; Wang et al., 2017), an extension to nonconvex problems (Qiao et al., 2016a, 2017), and is highly related to the analysis of first-order optimization solvers for neural networks (Zou et al., 2018, 2019). Recent results (Ge et al., 2015; Anandkumar and Ge, 2016; Jin et al., 2017) demonstrated that first-order methods, such as stochastic gradient descent, can also provide the theoretical guarantee to escape from saddle points and achieve sub-optimal points for many nonconvex optimization problems. The first-order methods have been widely used and thoroughly studied due to their ability to solve large-scale machine learning problems. On the other hand, the block coordinate descent (BCD) method has been recognized as a very powerful algorithm for solving problem (1) when $\boldsymbol{b} \geq \boldsymbol{0}$ and $\boldsymbol{A}_i = \boldsymbol{0}$ $(i = 1, 2, \ldots, n)$. At each iteration of a BCD optimization algorithm, a single block variable is updated while the other block variables are fixed. Compared to the gradient- and Newton-type methods, the per-iteration memory and computational consumption of the BCD method are deficient. Therefore, BCD and its randomized variants are particularly suitable for huge-size optimization problems, confirmed by a variety of experiments (Nesterov, 2012; Razaviyayn et al., 2014; Xu and Yin, 2017). However, it is still an open problem to use BCD methods for escaping from saddle points. Another difficulty comes from the constraint for the optimization of problem (1). Hong et al. (2016) and Jiang et al. (2019) analyzed the optimization algorithm for solving nonconvex problems with affinity constraints. However, these methods focus on solving the equality constrained consensus and sharing problems, where the nonsmooth part of the objective function is assumed to be convex. In contrast, the algorithm proposed in this study can handle more general problems in the form of problem (1) with inequality constraints.

Sophisticatedly constructed noises were added to the gradient updates in the perturbed gradient method in Jin et al. (2017). In contrast, we propose a novel asynchronous BCD method that uses the time perturbation technique, which efficiently helps obtain high performance and escape from the saddle points in handling large-scale problems. An illustration of the asynchronous update manner is shown in Fig. 1.

The contributions of this work are listed below:

1. We propose a new asynchronous BCD with path-following and time perturbation method, ATP for short, for solving problem (1).

2. We present analyses for the theoretical convergence and iteration complexity of the proposed ATP method.

3. Experimental results demonstrate the convergence behavior of the proposed algorithm for the randomized block variable selection rule with time perturbation.

4. We conduct empirical comparisons of the

proposed algorithm under different settings, i.e., randomized variable selection rule vs. cyclic variable selection rule, synchronous vs. asynchronous, and with time perturbation vs. without time perturbation. The experimental results demonstrate the efficacy of ATP.

5. Experiments conducted on the real-world application of constrained folded concave penalized linear regression show that ATP can escape from saddle points and local optima. Besides, ATP shows strong scalability in handling large-scale machine learning problems in a distributed setting.

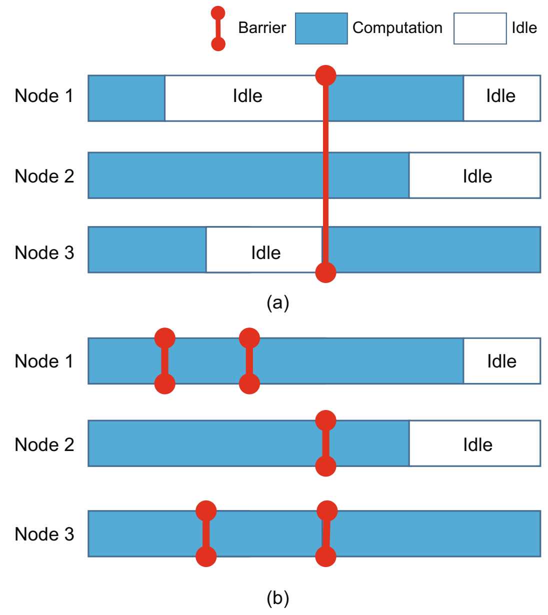

Fig. 1 Synchronous manner (a) and asynchronous manner (b)

2 Related work

Various approaches have been proposed to solve problem (1) without inequality constraints. One of the most potent methods is the BCD-type method. The storage cost and computational cost at each iteration of BCD-type methods are much lower than the counterparts of most existing gradient-based methods, because the BCD-type methods choose a single block to update at each iteration, making them be able to solve large-scale machine learning problems (Nesterov, 2012; Razaviyayn et al., 2014). In the last decade, the BCD-type methods (Sun et al., 2017; Xu and Yin, 2017) were developed with well-established convergence properties only when $\boldsymbol{A}_i = \mathbf{0}$ $(i = 1, 2, \ldots, n)$, and have been widely used in various real-world applications due to their ability to solve large-scale machine learning problems with provable theoretical analysis. However, to the best of our knowledge, the theoretical analysis of BCD methods in solving nonconvex problems with inequality constraints is still missing.

To handle the difficulty brought by inequality constraints of problem (1), the alternating direction method of multipliers (ADMM) (Hong et al., 2016; Wang et al., 2019) has been widely used, which relies on the Lagrangian relaxation and subgradient-type method of multipliers. Specifically, Hong et al. (2016) presented an analysis of ADMM with either a cyclic or a randomized block variable selection rule, but focused on solving consensus and sharing problems. Wang et al. (2019) presented asymptotic convergence results under the assumption that the objective function is a Kurdyka-Lojasiewicz function while the iteration complexity remains unknown.

However, none of these methods can effectively escape from sub-optimal points and they severely suffer from saddle points in the nonconvex setting. Furthermore, these methods need to restrict the properties of the objective function f, r_i's, and $\boldsymbol{A}_i$'s in problem (1). In this study, we are the first to consider an asynchronous BCD method with path-following and time perturbation to solve the nonconvex optimization problem with inequality constraints. The proposed ATP algorithm is the first one to demonstrate the efficacy of an asynchronous update manner with time perturbation for solving nonconvex optimization problems. In addition, the proposed algorithm can scale to multiple cores to handle large-scale machine learning problems.

With respect to solving large-scale problems in optimization, asynchronous-parallel methods have drawn more and more attention due to the increasing amount of large-scale data/variables involved in real-world machine learning applications. In distributed settings, asynchronous-parallel algorithms (Bertsekas and Tsitsiklis, 1989) keep all workers continuously running and eliminate the idle waiting time and costly synchronization communication. Recently, Cannelli et al. (2018) and Zhang et al. (2018) considered asynchronous algorithms for nonconvex cases, Xu (2019) considered asynchronous algorithms for problems with equality constraints, and Sun et al. (2017) and Peng et al. (2019) developed theoretical analysis of asynchronous methods for convex problems without affinity constraints under unbounded delay and mild assumptions. However, the theoretical convergence guarantee of asynchronous BCD with randomized BCD for inequality constraints has never been analyzed.

Note that this work is an extension of Liu (2019). In contrast, the algorithm proposed is different by adding the path-following technique, the theoretical analysis of the ATP method for nonconvex nonsmooth optimization problems is the major difference, and experiments are conducted to explore the effectiveness of the proposed algorithm.

3 Problem formulation

We consider the optimization of problem (1) and make the following assumptions:

Assumption 1 $f(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$ is continuously differentiable with respect to $\boldsymbol{x}_i$ for $1 \leq i \leq n$. The gradient is Lipschitz continuous with a constant $L_i > 0$ such that

$$\begin{aligned}&\|\nabla_i f(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n) - \nabla_i f(\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, \ldots, \tilde{\boldsymbol{x}}_n)\| \\ \leq & L_i \|\boldsymbol{x}_1 - \tilde{\boldsymbol{x}}_1, \boldsymbol{x}_2 - \tilde{\boldsymbol{x}}_2, \ldots, \boldsymbol{x}_n - \tilde{\boldsymbol{x}}_n\|, \ \forall \boldsymbol{x}, \tilde{\boldsymbol{x}} \in \mathbb{R}^d.\end{aligned}$$

Assumption 2 The domain of the objective function h is a compact and closed set. Specifically, $\|\boldsymbol{x}_i\| \leq D_i$ for $(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n) \in \mathrm{dom}(h)$.

Assumption 3 The proximal mapping of $r_i(\boldsymbol{x})$, i.e.,

$$\operatorname*{prox}_{r_i}(\bar{\boldsymbol{x}}) = \arg\min_{x} \left(r_i(\boldsymbol{x}) + \frac{\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2}{2} \right),$$

is easily obtained for $i = 1, 2, \ldots, n$.

Note that the least square and logistic functions satisfy Assumption 1, i.e.,

$$f(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n) = \frac{1}{2n} \left\| \sum_{i=1}^{n} \boldsymbol{d}_i^{\mathrm{T}} \boldsymbol{x}_i - b_i \right\|^2$$

or

$$f(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n) = \frac{1}{n} \sum_{i=1}^{n} \log \left(1 + \exp(-b_i \cdot \boldsymbol{d}_i^{\mathrm{T}} \boldsymbol{x}_i) \right),$$

where $[\boldsymbol{d}_1^{\mathrm{T}}; \boldsymbol{d}_2^{\mathrm{T}}; \ldots; \boldsymbol{d}_n^{\mathrm{T}}] \in \mathbb{R}^{n \times d}$ is a data matrix and $\boldsymbol{b} = [b_1, b_2, \ldots, b_n]^{\mathrm{T}} \in \mathbb{R}^n$. Furthermore, Assumption 3 is satisfied by many nonconvex penalty functions, such as smoothly clipped absolute deviation (SCAD) (Fan and Li, 2001), minimax concave penalty (Zhang CH, 2010), and capped-ℓ_1 penalty (Zhang T, 2010).

Next, we recall the definition of the generalized gradient (Rockafellar and Wets, 2009) to characterize the first-order optimality conditions of problem (1).

Definition 1 Let $g : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ be a proper lower semi-continuous function. Suppose $g(\bar{\boldsymbol{x}})$ is finite for a given $\bar{\boldsymbol{x}}$. For $\boldsymbol{v} \in \mathbb{R}^d$, we say the following:

1. $\boldsymbol{v}$ is a regular sub-gradient (or Frechet subdifferential) of h at $\bar{\boldsymbol{x}}$, written as $\boldsymbol{v} \in \hat{\partial} h(\bar{\boldsymbol{x}})$, if

$$\lim_{\boldsymbol{x} \neq \bar{\boldsymbol{x}}} \inf_{\boldsymbol{x} \to \bar{\boldsymbol{x}}} \frac{h(\boldsymbol{x}) - h(\bar{\boldsymbol{x}}) - \langle \boldsymbol{v}, \boldsymbol{x} - \bar{\boldsymbol{x}} \rangle}{\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|} \geq 0.$$

2. $\boldsymbol{v}$ is a general sub-gradient of h at $\bar{\boldsymbol{x}}$, written as $\boldsymbol{v} \in \partial h(\bar{\boldsymbol{x}})$, if there exist sequences $\{\boldsymbol{x}^k\}$ and $\{\boldsymbol{v}^k\}$ such that $\boldsymbol{x}^k \to \bar{\boldsymbol{x}}$ with $h(\boldsymbol{x}^k) \to h(\bar{\boldsymbol{x}})$ and $\boldsymbol{v}^k \in \hat{\partial} h(\boldsymbol{x}^k)$ with $\boldsymbol{v}^k \to \boldsymbol{v}$ as $k \to +\infty$.

When applying the first-order methods for general nonconvex optimization of problem (1), we claim that it has the highest possibility of obtaining a sequence of iterations that converges to a stationary point. Ge et al. (2015) showed that deterministic/stochastic gradient descent converges to a minimum point almost surely if the strict saddle property holds. Therefore, it has been recognized as a significant reference if the iterations generated by the proposed algorithm converge to a stationary point:

$$\boldsymbol{x}_i^{k,j+1} = \arg\min_{\boldsymbol{x}_i} \bar{\varPhi}_i \left(\boldsymbol{x}_{1 \leq l \leq i-1}^{k,j+1}, \boldsymbol{x}_i, \boldsymbol{x}_{i<l \leq n}^{k,j}, \mu \right). \tag{2}$$

4 Asynchronous block coordinate descent with time perturbation

In this section, we propose a novel asynchronous BCD method with a randomized block variable selection rule. We present the details of the algorithm running on the server (Algorithm 1) and on the worker (Algorithm 2).

We first introduce the barrier function to penalize the inequality constraints, and then construct the following potential function:

$$\begin{aligned}\varPhi(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n; \mu) =& f(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n) + \sum_{i=1}^{n} r_i(\boldsymbol{x}_i) \\ & - \mu g \left(\boldsymbol{b} - \sum_{i=1}^{n} \boldsymbol{A}_i \boldsymbol{x}_i \right),\end{aligned} \tag{3}$$

where $g : \mathbb{R}^p \to \mathbb{R}$ is the barrier function, μ is defined as a barrier parameter, and a log barrier function

Algorithm 1 ATP for nonconvex optimization (server)

```
1: Initialize $\gamma \in (0,1)$, $\eta \in (1,+\infty)$, $\mu^0 > 0$, and $\boldsymbol{x}_i^0$ for $i = 1,2,\ldots,n$
2: for $k = 0,1,2,\ldots$ do
3:   Initialize $\mathcal{C}^{k+1,1} = \{1,2,\ldots,n\}$ and $\boldsymbol{x}_i^{k,0} = \boldsymbol{x}_i^k$ for $i = 1,2,\ldots,n$
4:   for $j = 1,2,\ldots$ do
5:     Choose an active set $\mathcal{C}^{k+1,j+1} \subset \mathcal{C}^{k+1,1}$ in inner loop $j+1$
6:     for $i \in \mathcal{C}^{k+1,j+1}$ do
7:       Receive $\boldsymbol{x}_i^{k,j+1}$ from worker
8:       Update $\boldsymbol{x}_i^{k+1} = \boldsymbol{x}_i^{k,j+1}$ (inconsistent write operation)
9:     end for
10:    if the stopping criterion is satisfied then
11:      Set the stopping flag
12:      Break
13:    end if
14:  end for
15:  $\mu^{k+1} = \gamma\mu^k$
16: end for
```

log(·) is adopted. The proposed ATP has two loops. In each outer loop, we decrease the barrier parameter $\mu > 0$ to 0. In each inner loop, we update each variable $\boldsymbol{x}_i$ by minimizing the potential function Φ (Eq. (3)) for fixed $\mu > 0$. However, it is intractable to exactly minimize Φ with respect to $\boldsymbol{x}_i$ since the smooth part of Φ is a nonconvex function which has multiple stationary points. Therefore, we define the following local tight approximation $\bar{\Phi}_i$ of Φ with respect to $\boldsymbol{x}_i$ at $(\hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_2, \ldots, \hat{\boldsymbol{x}}_n)$ through linearizing the smooth part of Φ:

$$\begin{aligned}\bar{\Phi}_i(\boldsymbol{x}_i, \hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_2, \ldots, \hat{\boldsymbol{x}}_n, \mu) = P(\boldsymbol{x}_i) + \frac{h_i}{2}\|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i\|^2 \\ + \Big\langle \boldsymbol{x}_i - \hat{\boldsymbol{x}}_i, \nabla f_i(\hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_2, \ldots, \hat{\boldsymbol{x}}_n) \\ + \mu \boldsymbol{A}_i^{\mathrm{T}} \cdot \frac{1}{\boldsymbol{b} - \sum_{l=1}^n \boldsymbol{A}_l \hat{\boldsymbol{x}}_l} \Big\rangle. \qquad (4)\end{aligned}$$

For fixed $(\hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_2, \ldots, \hat{\boldsymbol{x}}_n)$ and μ, $\bar{\Phi}_i$ is a strongly convex function of $\boldsymbol{x}_i$ for a sufficiently large proximal parameter $h_i > 0$, determined by an efficient line search criterion. Then we can reformulate Eq. (2) as follows:

$$\boldsymbol{x}_i^{k,j+1} = \arg\min_{\boldsymbol{x}_i} \left[\frac{1}{h_i}P(\boldsymbol{x}_i) + \frac{1}{2}\left\|\boldsymbol{x}_i - \left(\boldsymbol{x}_i^k - \frac{1}{h_i}\boldsymbol{z}\right)\right\|^2\right],$$

Algorithm 2 ATP for nonconvex optimization (worker)

```
1: repeat
2:   Pull $\mathcal{C}^{k+1,j+1}$, $\mu$, and $\boldsymbol{x}^k$ from server
3:   Randomly choose $i \in \mathcal{C}^{k+1,j+1}$
4:   Set $h_i \in [H_{\min}, H_{\max}]$ and $h_i > 0$
5:   repeat
6:     Update variable $\boldsymbol{x}_i$ by Eq. (2)
7:     Update the proximal parameter $h_i = \eta h_i$
8:   until the line search criterion is satisfied
9:   Push $\boldsymbol{x}_i^{k,j+1}$ to server after a random time delay
10: until the stopping flag is set
```

where

$$\begin{aligned}\boldsymbol{z} =& \nabla f_i\left(\boldsymbol{x}_{1\le l\le i-1}^{k,j+1}, \boldsymbol{x}_{i<l\le n}^{k,j}\right) \\ &+ \mu \boldsymbol{A}_i^{\mathrm{T}} \cdot \frac{1}{\boldsymbol{b} - \sum\limits_{l=1}^{i-1} \boldsymbol{A}_l \boldsymbol{x}_l^{k,j+1} - \sum\limits_{l=i}^{n} \boldsymbol{A}_l \boldsymbol{x}_l^{k,j}}. \qquad (5)\end{aligned}$$

This is equivalent to computing the proximal mapping of $P(\boldsymbol{x})$, hence admitting the closed-form expression (Hong et al., 2016). In what follows, we consider the following block variable selection rule: Let $i \in \{1,2,\ldots,n\}$ be the index of the i^{th} primal variable $\boldsymbol{x}_i$, and let $\mathcal{C}^{k+1,j+1} \subset \{1,2,\ldots,n\}$ be the active set of primal variables updated in inner loop $j+1$ of outer loop $k+1$.

4.1 Randomized block variable selection rule

A single primal variable $\boldsymbol{x}_i$ is selected randomly in the inner loop at each iteration, and the probability is

$$\begin{aligned}&\mathrm{Prob}(i \in \mathcal{C}^{k+1,j+1} | \boldsymbol{x}_1^{k,j}, \boldsymbol{x}_2^{k,j}, \ldots, \boldsymbol{x}_n^{k,j}) \\ &\quad = p_i^{k+1,j+1} \ge p_{\min} > 0.\end{aligned}$$

The proximal parameters $h_i > 0$ in Algorithms 1 and 2 are related to the Lipschitz constant L_i (which is difficult to compute in practice) and the value of the current iteration. Therefore, we use a line search criterion to determine the appropriate value of h_i to guarantee the efficiency of our algorithm in practical applications.

4.2 Line search criterion

One of the most widely used line search criteria simply requires that the value of the objective function be decreasing after updating the block variables $\boldsymbol{x}_i$ for $i = 1,2,\ldots,n$. Specifically, in this work we accept the value of proximal parameter h_i if $\boldsymbol{x}_i$ is updated when the following line search criterion is satisfied:

$$\begin{aligned}\varPhi\left(\boldsymbol{x}_{1\le l\le i-1}^{k,j+1}, \boldsymbol{x}_{i\le l\le n}^{k,j}, \mu^k\right) - \varPhi\left(\boldsymbol{x}_{1\le l\le i}^{k,j+1}, \boldsymbol{x}_{i+1\le l\le n}^{k,j}, \mu^k\right)\\ \ge \frac{\sigma h_i}{2}\left\|\boldsymbol{x}_i^{k,j} - \boldsymbol{x}_i^{k,j+1}\right\|^2, \qquad (6)\end{aligned}$$

where σ is a constant in the interval $(0,1)$.

4.3 Stopping criterion

We repeat the inner loop for fixed μ^k in outer loop $k+1$ until the following stopping criterion is satisfied for $1\le i\le n$:

$$\begin{aligned}\operatorname{dist}\Big(&-\nabla_i f\big(\boldsymbol{x}_1^{k,j+1}, \boldsymbol{x}_2^{k,j+1}, \ldots, \boldsymbol{x}_n^{k,j+1}\big)\\ &+\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{y}, \partial r_i\big(\boldsymbol{x}_i^{k,j+1}\big)\Big) \le \mu^k, \qquad (7)\end{aligned}$$

where $\boldsymbol{y}_i\left(\boldsymbol{b} - \sum_{j=1}^n \boldsymbol{A}_j\boldsymbol{x}_j^{k,j+1}\right)_i = \mu^k$ and $\mu^k > 0$ is a barrier parameter in outer loop $k+1$.

4.4 Time perturbation

Different from the gradient perturbation proposed in Jin et al. (2017), where sophisticatedly constructed noises were added to the gradient updates, we use the time perturbation technique in the asynchronous procedure within the BCD decomposition framework. Specifically, the read and write operations are set not to be locked, and the client will push the intermediate values to the server node after a random time delay of ΔT_i after updating the primal variables $\boldsymbol{x}_i$ for $i=1,2,\ldots,n$ at each iteration. Time perturbation can be viewed as an implementation of out-of-order updates, and the experiments demonstrate that the time perturbation technique can greatly help the proposed ATP algorithm escape from sub-optimal points.

5 Analyses of convergence and iteration complexity

5.1 Convergence analysis

We first present a lemma which guarantees that the line search criterion in inequality (6) is satisfied:

Lemma 1 Given the constant $\sigma\in(0,1)$ and $i\in\mathcal{C}^{k+1,j+1}$ in inner loop $j+1$ of outer loop $k+1$, the line search criterion in inequality (6) is satisfied when

$$h_i \ge (L_i + \mu^k\boldsymbol{s}^{\mathrm{T}}\boldsymbol{A}_i\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{s})/(1-\sigma), \qquad (8)$$

where $\boldsymbol{s} = 1\Big/\left(\boldsymbol{b} - \sum_{l=1}^{i-1}\boldsymbol{A}_l\boldsymbol{x}_l^{k,j+1} - \sum_{l=i}^{n}\boldsymbol{A}_l\boldsymbol{x}_l^{k,j}\right)$. In addition, h_i is bounded under the line search criterion.

Proof For $i\in\mathcal{C}^{k+1,j+1}$, it follows from the update of $\boldsymbol{x}_i^{k,j+1}$ that

$$\begin{aligned}r_i(\boldsymbol{x}_i^{k,j}) \ge{}& r_i(\boldsymbol{x}_i^{k,j+1}) + \Bigg\langle \boldsymbol{x}_i^{k,j+1} - \boldsymbol{x}_i^{k,j},\\ &\nabla f_i(\boldsymbol{x}_{1\le l\le i-1}^{k,j+1}, \boldsymbol{x}_{i\le l\le n}^{k,j}) + \frac{h_i}{2}\left\|\boldsymbol{x}_i^{k,j+1} - \boldsymbol{x}_i^{k,j}\right\|^2\\ &+\mu^k\boldsymbol{A}_i^{\mathrm{T}}\cdot\frac{1}{\boldsymbol{b} - \sum\limits_{l=1}^{i-1}\boldsymbol{A}_l\boldsymbol{x}_l^{k,j+1} - \sum\limits_{l=i}^{n}\boldsymbol{A}_l\boldsymbol{x}_l^{k,j}}\Bigg\rangle\\ \ge{}& r_i\big(\boldsymbol{x}_i^{k,j+1}\big) + f\big(\boldsymbol{x}_{1\le l\le i}^{k,j+1}, \boldsymbol{x}_{i+1\le l\le n}^{k,j}\big)\\ &-\mu^k\log\left(\boldsymbol{b} - \sum_{l=1}^{i}\boldsymbol{A}_l\boldsymbol{x}_l^{k,j+1} - \sum_{l=i+1}^{n}\boldsymbol{A}_l\boldsymbol{x}_l^{k,j}\right)\\ &-\Bigg[f_i(\boldsymbol{x}_{1\le l\le i-1}^{k,j+1}, \boldsymbol{x}_{i\le l\le n}^{k,j}) - \mu^k\log\Bigg(\boldsymbol{b}\\ &-\sum_{l=1}^{i-1}\boldsymbol{A}_l\boldsymbol{x}_l^{k,j+1} - \sum_{l=i}^{n}\boldsymbol{A}_l\boldsymbol{x}_l^{k,j}\Bigg)\Bigg]\\ &+\frac{h_i - L_i - \mu^k\boldsymbol{s}^{\mathrm{T}}\boldsymbol{A}_i\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{s}}{2}\left\|\boldsymbol{x}_i^{k,j+1} - \boldsymbol{x}_i^{k,j}\right\|^2.\end{aligned}$$

Therefore, we have

$$\begin{aligned}\varPhi\left(\boldsymbol{x}_{1\le l\le i-1}^{k,j+1}, \boldsymbol{x}_{i\le l\le n}^{k,j}, \mu^k\right) - \varPhi\left(\boldsymbol{x}_{1\le l\le i}^{k,j+1}, \boldsymbol{x}_{i+1\le l\le n}^{k,j}, \mu^k\right)\\ \ge \frac{h_i - \left(L_i + \mu^k\boldsymbol{s}^{\mathrm{T}}\boldsymbol{A}_i\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{s}\right)}{2}\left\|\boldsymbol{x}_i^{k,j+1} - \boldsymbol{x}_i^{k,j}\right\|^2.\end{aligned}$$

Combining the above inequality with inequality (8) yields $h_i - \left(L_i + \mu^k\boldsymbol{s}^{\mathrm{T}}\boldsymbol{A}_i\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{s}\right) \ge \sigma h_i$, which implies inequality (6).

Next, we prove that h_i is bounded under the line search criterion. It is trivial to show that h_i is lower-bounded, since $h_i \ge H_{\min} > 0$ ($H_{\min}$ is defined in our algorithm). We prove that h_i is upper-bounded by contradiction. Since the line search criterion is satisfied whenever $h_i \ge \left(L_i + \mu^k\boldsymbol{s}^{\mathrm{T}}\boldsymbol{A}_i\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{s}\right)/(1-\sigma)$, we have $h_i \le \eta\left(L_i + \mu^k\boldsymbol{s}^{\mathrm{T}}\boldsymbol{A}_i\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{s}\right)/(1-\sigma)$.

Then we present a lemma which guarantees that the stopping criterion of each of the inner loop is satisfied and $\left(\boldsymbol{x}_1^{k+1}, \boldsymbol{x}_2^{k+1}, \ldots, \boldsymbol{x}_n^{k+1}\right)$ approximates the stationary point of the potential function $\varPhi(\cdot, \mu^k)$ for fixed μ^k. Specifically, we define the stationary point of $\left(\bar{\boldsymbol{x}}_1^{k+1}, \bar{\boldsymbol{x}}_2^{k+1}, \ldots, \bar{\boldsymbol{x}}_n^{k+1}\right)$

of the potential function $\Phi(\cdot, \mu^k)$ for fixed μ^k, and prove that $\left(\boldsymbol{x}_1^{k+1}, \boldsymbol{x}_2^{k+1}, \ldots, \boldsymbol{x}_n^{k+1}\right)$ approximates $\left(\bar{\boldsymbol{x}}_1^{k+1}, \bar{\boldsymbol{x}}_2^{k+1}, \ldots, \bar{\boldsymbol{x}}_n^{k+1}\right)$ under the error level of μ^k.

Lemma 2 Let $(\bar{\boldsymbol{x}}_1^{k+1}, \bar{\boldsymbol{x}}_2^{k+1}, \ldots, \bar{\boldsymbol{x}}_n^{k+1})$ be the stationary point of the potential function $\Phi(\cdot, \mu^k)$, and the randomized block variable selection rule is employed. There exists some $j > 0$ such that the stopping criterion in inequality (7) is satisfied almost surely for the randomized block variable selection rule after a finite number of inner loops. Then $(\boldsymbol{x}_1^{k+1}, \boldsymbol{x}_2^{k+1}, \ldots, \boldsymbol{x}_n^{k+1})$ approximates $(\bar{\boldsymbol{x}}_1^{k+1}, \bar{\boldsymbol{x}}_2^{k+1}, \ldots, \bar{\boldsymbol{x}}_n^{k+1})$ under the error level of μ^k. Specifically, there exists $\bar{\boldsymbol{y}} > 0$ such that, for $i = 1, 2, \ldots, n$,

$$\begin{aligned}\mu^k \geq &\operatorname{dist}\Big(- \nabla_i f(\boldsymbol{x}_1^{k+1}, \boldsymbol{x}_2^{k+1}, \ldots, \boldsymbol{x}_n^{k+1}) \\ &+ \boldsymbol{A}_i^{\mathrm{T}} \bar{\boldsymbol{y}}, \partial r_i(\boldsymbol{x}_i^{k+1})\Big), \\ \mu^k = &\bar{\boldsymbol{y}}_i \left(\boldsymbol{b} - \sum_{i=1}^{n} \boldsymbol{A}_i \boldsymbol{x}_i^{k+1}\right)_i, \\ \boldsymbol{b} > &\sum_{i=1}^{n} \boldsymbol{A}_i \boldsymbol{x}_i^{k+1},\end{aligned}$$

almost surely for the randomized block variable selection rule.

Proof Using Lemma 1, we obtain inequality (6) in the line search criterion:

$$\begin{aligned}&\Phi\left(\boldsymbol{x}_{1\leq l\leq i-1}^{k,j+1}, \boldsymbol{x}_{i\leq l\leq n}^{k,j}, \mu^k\right) - \Phi\left(\boldsymbol{x}_{1\leq l\leq i}^{k,j+1}, \boldsymbol{x}_{i+1\leq l\leq n}^{k,j}, \mu^k\right) \\ &\geq \frac{\sigma h_i}{2} \left\|\boldsymbol{x}_i^{k,j} - \boldsymbol{x}_i^{k,j+1}\right\|^2.\end{aligned}$$

Summing inequality (6) over $i \in \mathcal{C}^{k+1,j+1}$ and combining the fact that $\boldsymbol{x}_i^{j+1} = \boldsymbol{x}_i^j$ for $i \notin \mathcal{C}^{k+1,j+1}$ yields

$$\begin{aligned}&\Phi\left(\boldsymbol{x}_{1\leq l\leq n}^{k,j}, \mu^k\right) - \Phi\left(\boldsymbol{x}_{1\leq l\leq n}^{k,j+1}, \mu^k\right) \\ &\geq \sum_{i=1}^{n} \left[\frac{\sigma h_i}{2} \left\|\boldsymbol{x}_i^{k,j} - \boldsymbol{x}_i^{k,j+1}\right\|^2\right] \\ &\geq \frac{\sigma H_{\min}}{2} \sum_{i=1}^{n} \left\|\boldsymbol{x}_i^{k,j} - \boldsymbol{x}_i^{k,j+1}\right\|^2. \qquad (9)\end{aligned}$$

Next, we show that the potential function $\Phi(\boldsymbol{x}_{1\leq i\leq n}, \mu)$ is lower-bounded. Since $\operatorname{dom}(h)$ is a compact set and $\mu^k \leq \mu^0$, we have

$$\begin{aligned}\Phi\left(\boldsymbol{x}_{1\leq l\leq n}^{k,j}, \mu^k\right) &= f(\boldsymbol{x}_{1\leq i\leq n}^{k,j}) + \sum_{i=1}^{n} r_i(\boldsymbol{x}_i^{k,j}) \\ &\quad - \mu \log\left(\boldsymbol{b} - \sum_{i=1}^{n} \boldsymbol{A}_i \boldsymbol{x}_i^{k,j}\right) \\ &\geq h^* - \mu^0 \log\Bigg(\|\boldsymbol{b}\| \\ &\quad + \sum_{i=1}^{n} \|\boldsymbol{A}_i\| \left\|\boldsymbol{x}_i^{k,j}\right\|\Bigg) \\ &\geq \underline{\Phi}^*. \qquad (10)\end{aligned}$$

Since k and μ^k are fixed in any inner loop of outer loop $k+1$ and $\boldsymbol{x}_i^k = \boldsymbol{x}_i^{k,0}$ for $i = 1, 2, \ldots, n$, we conclude that $\Phi(\boldsymbol{x}_{1\leq l\leq n}^{k,0}, \mu^k)$ is finite.

For the randomized block variable selection rule, we take the conditional expectation from both sides of inequality (9), implying

$$\begin{aligned}&\Phi\left(\boldsymbol{x}_{1\leq l\leq n}^{k,j}, \mu^k\right) - E\left[\Phi\left(\boldsymbol{x}_{1\leq l\leq n}^{k,j+1}, \mu^k\right) | \boldsymbol{x}_{1\leq l\leq n}^{k,j}\right] \\ &\geq \sum_{i=1}^{n} \frac{\sigma H_{\min}}{2} E\left[\left\|\boldsymbol{x}_i^{k,j} - \boldsymbol{x}_i^{k,j+1}\right\|^2\right].\end{aligned}$$

By the definition of the randomized block variable selection rule and $p_i^{j+1} \geq p_{\min}$, we have

$$\begin{aligned}&\Phi\left(\boldsymbol{x}_{1\leq l\leq n}^{k,j}, \mu^k\right) - E\left[\Phi\left(\boldsymbol{x}_{1\leq l\leq n}^{k,j+1}, \mu^k\right) \Big| \boldsymbol{x}_{1\leq l\leq n}^{k,j}\right] \\ &\geq \frac{\sigma p_{\min} H_{\min}}{2} \sum_{i=1}^{n} \left[\left\|\boldsymbol{x}_i^{k,j} - \tilde{\boldsymbol{x}}_i^{k,j+1}\right\|^2\right],\end{aligned}$$

where $\tilde{\boldsymbol{x}}_{1\leq i\leq n}^{k,j+1}$ is a "virtual" iteration assuming that all variables are updated in this inner loop. Thus, $\{\Phi(\boldsymbol{x}_{1\leq l\leq n}^{k,j}, \mu^k)\}$ is a supermartingale with respect to the natural history. By the supermartingale convergence theorem, $\{\Phi(\boldsymbol{x}_{1\leq l\leq n}^{k,j}, \mu^k)\}$ converges and we almost surely have

$$\sum_{i=1}^{n} \left\|\boldsymbol{x}_i^{k,j} - \boldsymbol{x}_i^{k,j+1}\right\|^2 \to 0, \ j \to +\infty, \qquad (11)$$

since $\sigma > 0$, $p_{\min} > 0$, and $H_{\min} > 0$. By the same argument, we conclude that it holds almost surely for the randomized block variable selection rule that

$$\sum_{i=1}^{n} \left\|\boldsymbol{x}_i^{k,j} - \boldsymbol{x}_i^{k,j+1}\right\| < n \sum_{i=1}^{n} \left\|\boldsymbol{x}_i^{k,j} - \boldsymbol{x}_i^{k,j+1}\right\|^2 \to 0. \qquad (12)$$

Noting that $\boldsymbol{x}_i^{k,j+1} \to \boldsymbol{x}_i^{k+1}$ for $i = 1, 2, \ldots, n$ if the stopping criterion in inequality (7) is satisfied, we derive from the first-order optimality condition that

$$\operatorname{dist}\Bigg(-\nabla_i f(\bar{\boldsymbol{x}}_1^{k+1}, \bar{\boldsymbol{x}}_2^{k+1}, \ldots, \bar{\boldsymbol{x}}_n^{k+1}) + \boldsymbol{A}_i^{\mathrm{T}} \bar{\boldsymbol{y}}_i^k, \quad (13)$$

$$\partial r_i(\bar{\boldsymbol{x}}_i^{k+1}) \Bigg) \leq C^k \sum_{i=1}^{n} \left\| \boldsymbol{x}_i^{k,j+1} - \boldsymbol{x}_i^{k,j} \right\|,$$

$$\bar{\boldsymbol{y}}_i^k \left(\boldsymbol{b} - \sum_{i=1}^{n} \boldsymbol{A}_i \bar{\boldsymbol{x}}_i^{k+1} \right)_i = \mu^k, \quad (14)$$

$$\sum_{i=1}^{n} \boldsymbol{A}_i \bar{\boldsymbol{x}}_i^{k+1} < \boldsymbol{b}, \quad (15)$$

for $i = 1, 2, \ldots, n$. $\left(\boldsymbol{x}_1^{k,j+1}, \boldsymbol{x}_2^{k,j+1}, \ldots, \boldsymbol{x}_n^{k,j+1}\right)$ is equal to $\left(\boldsymbol{x}_1^{k+1}, \boldsymbol{x}_2^{k+1}, \ldots, \boldsymbol{x}_n^{k+1}\right)$. Following Eq. (11), we conclude that the stopping criterion is satisfied after a finite number of inner loops.

Then we are ready to state our main result about the limiting behavior of our algorithm. As $\mu^k \to 0$, we show that the sequence $\left(\boldsymbol{x}_1^{k+1}, \boldsymbol{x}_2^{k+1}, \ldots, \boldsymbol{x}_n^{k+1}\right)$ converges to the stationary point of problem (1).

Theorem 1 Suppose the randomized block variable selection rule is employed. Then $\left(\boldsymbol{x}_1^{k+1}, \boldsymbol{x}_2^{k+1}, \ldots, \boldsymbol{x}_n^{k+1}\right)$ converges to the stationary point of problem (1) almost surely for the randomized block variable selection rule.

Proof Since $\left(\boldsymbol{x}_1^k, \boldsymbol{x}_2^k, \ldots, \boldsymbol{x}_n^k\right)$ lies in the compact set for $k = 0, 1, 2, \ldots$, the set of its limiting points is not empty. Consider a limiting point $(\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, \ldots, \tilde{\boldsymbol{x}}_n)$ with the sub-sequence $\{\boldsymbol{x}_1^{r_k}, \boldsymbol{x}_2^{r_k}, \ldots, \boldsymbol{x}_n^{r_k}\}_{k=0}^{\infty}$ converging to $(\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, \ldots, \tilde{\boldsymbol{x}}_n)$. Then we almost surely have

$$\operatorname{dist}\Bigg(-\nabla_i f(\bar{\boldsymbol{x}}_1, \bar{\boldsymbol{x}}_2, \ldots, \bar{\boldsymbol{x}}_n) + \boldsymbol{A}_i^{\mathrm{T}} \bar{\boldsymbol{y}}, \partial r_i(\bar{\boldsymbol{x}}_i) \Bigg) \leq \mu^{r_k},$$

$$\bar{\boldsymbol{y}}_i \left(\boldsymbol{b} - \sum_{i=1}^{n} \boldsymbol{A}_i \bar{\boldsymbol{x}}_i \right)_i \leq \mu^{r_k},$$

$$\sum_{i=1}^{n} \boldsymbol{A}_i \bar{\boldsymbol{x}}_i < \boldsymbol{b},$$

for the randomized block variable selection rule. As $k \to +\infty$, we have $\mu^{r_k} \to 0$. Therefore, we conclude that $\left(\boldsymbol{x}_1^{k+1}, \boldsymbol{x}_2^{k+1}, \ldots, \boldsymbol{x}_n^{k+1}\right)$ converges to the stationary point of problem (1) almost surely for the randomized block variable selection rule. This completes the proof of Theorem 1.

5.2 Iteration complexity analysis

In this subsection, we present the iteration complexity analysis of our algorithm with the randomized block variable selection rule. Note that the iterations may not converge to the global optimum point in nonconvex optimization. Therefore, the closeness to the optimal solution cannot be used to measure the iteration complexity. Inspired by Razaviyayn et al. (2014), we use the size of proximal gradient of the potential function as a measure of optimality. More precisely, we define

$$\begin{aligned} \hat{\nabla}\Phi(\boldsymbol{x}, \mu) = & \boldsymbol{x} - \arg\min \Bigg\{ \Big\langle \nabla f(\boldsymbol{x}), \boldsymbol{y} - \boldsymbol{x} \Big\rangle \\ & + \sum_{i=1}^{n} \Bigg[r_i(\boldsymbol{x}_i) + \Big\langle \mu \boldsymbol{A}_i^{\mathrm{T}} \cdot \frac{1}{\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}}, \\ & \boldsymbol{y}_i - \boldsymbol{x}_i \Big\rangle + \frac{h_i}{2} \left\| \boldsymbol{y}_i - \boldsymbol{x}_i \right\|^2 \Bigg] \Bigg\}, \end{aligned} \quad (16)$$

where $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$ and $\boldsymbol{y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_n)$. For fixed $\mu > 0$, $\hat{\nabla}\Phi(\boldsymbol{x}, \mu) = 0$ implies that

$$\operatorname{dist}\left(-\nabla_i f(\boldsymbol{x}) + \boldsymbol{A}_i^{\mathrm{T}} \boldsymbol{y}, \partial r_i(\boldsymbol{x}_i) \right) = 0,$$
$$\boldsymbol{y}_i \left(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x} \right)_i = \mu, \ \boldsymbol{A}\boldsymbol{x} < \boldsymbol{b}.$$

Then if we let $\mu < \epsilon$, the iterations generated reach the ϵ-stationary point of problem (1). Therefore, the decrease rate of $\left\| \hat{\nabla}\Phi(\boldsymbol{x}, \mu) \right\|$ and μ can be viewed as an iteration complexity analysis of our algorithm. In the asynchronous update manner, we focus mainly on the decrease rate of $\left\| \hat{\nabla}\Phi(\boldsymbol{x}, \mu) \right\|$ within outer loop $k + 1$ under the stopping criterion (inequality (7)) used in practice.

Theorem 2 Suppose the randomized block variable selection rule is employed. To reach the ϵ-stationary point of problem (1), the stopping criterion in inequality (7) is satisfied almost surely for the randomized block variable selection rule within $O(1/\epsilon)$ inner loops, and the number of iterations in the outer loop is $O(\log(1/\epsilon))$.

Proof We know from inequality (10) that for any $k \geq 0$ and $j \geq 0$, there is

$$\Phi(\boldsymbol{x}_{1 \leq l \leq n}^{k,j}, \mu^k) \geq \underline{\Phi}^*. \quad (17)$$

Next, we need to derive the upper bound of $\Phi(\boldsymbol{x}_{1 \leq l \leq n}^{k,j}, \mu^k)$ and show that this bound does not depend on $k \geq 0$ and $j \geq 0$. Since $\operatorname{dom}(h)$ is a compact set, $h(\boldsymbol{x}_{1 \leq l \leq n}^k)$ is bounded. Therefore, it is sufficient to show that $-\mu^k \log(\boldsymbol{b} - \sum_{l=1}^n \boldsymbol{A}_l \boldsymbol{x}_l^k)$ or $-\mu^k E\left[\log(\boldsymbol{b} - \sum_{l=1}^n \boldsymbol{A}_l \boldsymbol{x}_l^k)\right]$ is upper-bounded.

We first prove that $-\mu^k \log(\boldsymbol{b} - \sum_{l=1}^n \boldsymbol{A}_l \boldsymbol{x}_l^k)$ is upper-bounded for the randomized variable selection rule. Suppose the contrary that $-\mu^k \log(\boldsymbol{b} - \sum_{l=1}^n \boldsymbol{A}_l \boldsymbol{x}_l^k) \to +\infty$ as $k \to +\infty$. There exists $M > 0$ such that $-\mu^k \log(\boldsymbol{b} - \sum_{l=1}^n \boldsymbol{A}_l \boldsymbol{x}_l^k) > 0$ whenever $k \geq M$. In this case, we have

$$-\mu^{k-1} \log\left(\boldsymbol{b} - \sum_{l=1}^n \boldsymbol{A}_l \boldsymbol{x}_l^k\right) > -\mu^k \log\left(\boldsymbol{b} - \sum_{l=1}^n \boldsymbol{A}_l \boldsymbol{x}_l^k\right).$$

Therefore, we conclude that

$$\Phi(\boldsymbol{x}_{1\leq l\leq n}^k, \mu^{k-1}) \geq \Phi(\boldsymbol{x}_{1\leq l\leq n}^k, \mu^k). \tag{18}$$

From inequality (9), we have

$$\Phi(\boldsymbol{x}_{1\leq l\leq n}^{k-1}, \mu^{k-1}) \geq \Phi(\boldsymbol{x}_{1\leq l\leq n}^k, \mu^{k-1}). \tag{19}$$

Therefore, combining inequalities (18) and (19) we have

$$\begin{aligned} -\mu^k \log\left(\boldsymbol{b} - \sum_{l=1}^n \boldsymbol{A}_l \boldsymbol{x}_l^k\right) &\leq \Phi(\boldsymbol{x}_{1\leq l\leq n}^k, \mu^k) - h^* \\ &\leq \Phi(\boldsymbol{x}_{1\leq l\leq n}^{M-1}, \mu^{M-1}) - h^* \\ &< +\infty, \end{aligned} \tag{20}$$

which contradicts with that $-\mu^k \log(\boldsymbol{b} - \sum_{l=1}^n \boldsymbol{A}_l \boldsymbol{x}_l^k) \to +\infty$ as $k \to +\infty$.

Therefore, we conclude that the upper bound of $E\left[\Phi(\boldsymbol{x}_{1\leq l\leq n}^{k,j}, \mu^k)\right]$ exists for the randomized block variable selection rule.

We denote this upper bound as $\bar{\Phi}^*$ and the number of inner loops in outer loop $k+1$ as T_{inner}^k. Then the following holds for the randomized block variable selection rule:

$$\frac{\sigma H_{\min}}{2} \sum_{j=0}^{T_{\text{inner}}^k - 1} E\left[\sum_{i=1}^n \left\|\boldsymbol{x}_i^{k,j} - \boldsymbol{x}_i^{k,j+1}\right\|^2\right] \leq \bar{\Phi}^* - \underline{\Phi}^*.$$

Therefore, when $T_{\text{inner}}^k = O(1/\mu^k)$, there exists some $j \leq T_{\text{inner}}^k$ such that

$$E\left[\sum_{i=1}^n \left\|\boldsymbol{x}_i^{k,j} - \boldsymbol{x}_i^{k,j+1}\right\|^2\right] \leq \mu^k \tag{21}$$

for the randomized block variable selection rule.

On the other hand, we obtain the number of iterations required in the inner loop as

$$T = \sum_{k=0}^{\log(1/\epsilon)} \frac{C}{\gamma^k \delta_0} = \frac{C}{\delta_0} \cdot \frac{1/\epsilon - 1}{1/\gamma - 1} \leq \frac{C\gamma}{\delta_0 - \gamma\delta_0} \cdot \frac{1}{\epsilon},$$

which implies that the number of iterations required in the outer loop is $O(\log(1/\epsilon))$ and $\mu^k < \epsilon$.

6 Experiments

In large-scale RWS applications, a range of exciting and essential models can be reformulated as nonconvex optimization problems in the form of Eq. (1). Experiments were conducted on a shared memory multiprocessing platform to validate the efficacy of the proposed ATP algorithm in solving the problem of constrained folded concave penalized linear regression, which is a classical problem in sparse learning proposed in James et al. (2012):

$$\begin{aligned} \min_{\boldsymbol{\beta}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|^2 + \lambda \sum_{i=1}^d r(\boldsymbol{\beta}_i) \\ \text{s.t. } \boldsymbol{C}\boldsymbol{\beta} \leq \boldsymbol{b}, \end{aligned} \tag{22}$$

where $r(\cdot)$ is the SCAD (Fan and Li, 2001), $\boldsymbol{X} \in \mathbb{R}^{n\times d}$, $\boldsymbol{y} \in \mathbb{R}^n$, $\boldsymbol{C} \in \mathbb{R}^{m\times d}$, and $\boldsymbol{b} \in \mathbb{R}^m$ are the given data, and $\lambda > 0$ is the regularization parameter.

To explore the effectiveness of the proposed algorithm, we conducted experimental comparisons among ATP variants under different settings, i.e., randomized variable selection rule vs. cyclic variable selection rule, with time perturbation vs. without time perturbation, and synchronous vs. asynchronous. We implemented two baselines to solve problem (22), the inexact augmented Lagrangian method (IALM) and three-block linearized alternating direction method of multipliers (LADMM). Note that these two algorithms were implemented without theoretical convergence guarantee, while the proposed algorithm has been guaranteed to converge as analyzed in Section 5.

The algorithms were implemented in C++ with OpenMP (Dagum and Menon, 1998) and Eigen (Guennebaud and Jacob, 2010), and the public accessible code (https://github.com/linboqiao/ATP) is available online. OpenMP (open multi-processing, http://www.openmp.org/wp-content/uploads/openmp-4.5.pdf) is an application programming interface (API) that supports multi-platform shared memory multiprocessing programming. Eigen (http://eigen.tuxfamily.org/dox/) is a high-level C++ library for linear algebra, matrix and vector operations, geometrical transformations, numerical solvers, and related algorithms. Experiments were conducted on a server with an Intel® Xeon® CPU E5-2620, 2.4 GHz, and 32 GB memory, running Ubuntu 16.04 Server. OpenMP version 4.5 and Eigen version 3.3.2 were used.

Matrices $\boldsymbol{X}$ and $\boldsymbol{C}$ were generated with independent standard Gaussian entries, and the columns of $\boldsymbol{X}$ were standardized to have unit norms. Groups of the matrix were constructed to validate the robustness of the proposed algorithm. Then $\boldsymbol{b}$ was obtained via $\boldsymbol{b} = \boldsymbol{C\beta} + \hat{\boldsymbol{\epsilon}}$ with the first five entries of the coefficient vector to be 1 and the rest to be 0, and $\boldsymbol{y} = \boldsymbol{X\beta} + \boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \delta^2 I_d)$. We selected different regularization parameters in $\{1.4, 1.6, 1.8, 2.0\}$ to show that our algorithms are robust. Parameters δ, λ, γ, and μ were set to 2, 2, 0.8, and 1, respectively. ATP terminated when the relative changes in the last six consecutive objective function values were less than 10^{-7} or the number of iterations exceeded MAX-ITER, which was set to be 1000 in this work.

6.1 Randomized variable selection rule vs. cyclic variable selection rule

To explore the effectiveness of different variable selection rules, i.e., randomized variable selection rule and cyclic variable selection rule, synchronous BCD with these two kinds of variable selection rules was used to solve problem (22). Table 1 presents the performance comparison among synchronous BCD with time perturbation under the setting of the randomized variable selection rule (Sync-BCD-TP-R) and that under the setting of the cyclic variable selection rule (Sync-BCD-TP-C) as well as IALM and LADMM. The experimental results show that the proposed algorithms with time perturbation have better performance than IALM and LADMM. Among the proposed algorithms, the randomized variable selection rule performs better than the cyclic variable selection rule.

6.2 Synchronous block coordinate descent with time perturbation vs. without time perturbation

To demonstrate the effectiveness of time perturbation, synchronous type of BCD was implemented to solve problem (22) under different settings, i.e., synchronous BCD with time perturbation and without time perturbation. Fig. 2 shows the convergence behavior of ATP with synchronous updating rules (Sync-BCD-TP) in solving the nonconvex folded concave linear regression problem, running with 1, 2, 4, 8, 16, 22, 24, 26, 28, and 30 threads. The results demonstrate that time perturbation has a significant effect in helping the algorithm find a better solution. Besides, Sync-BCD-TP converges significantly faster with multi-thread settings since it uses

Table 1 Performance comparison among Sync-BCD-TP-R, Sync-BCD-TP-C, IALM, and LADMM

Method	(n, d)	σ	(Objective, Time (s))			
			$\lambda = 1.4$	$\lambda = 1.6$	$\lambda = 1.8$	$\lambda = 2.0$
Sync-BCD-TP-R	(2000, 5000)	0.1	(**126.73**, **7.31**)	(**120.12**, **7.61**)	(**141.27**, **7.25**)	(**160.17**, **7.11**)
Sync-BCD-TP-C			(720.09, 23.39)	(807.96, 23.52)	(942.65, 23.92)	(950.82, 23.79)
IALM			(1048.47, 46.70)	(1146.34, 46.60)	(1171.69, 48.32)	(1170.91, 46.33)
LADMM			(1348.56, 30.61)	(1534.81, 30.39)	(1717.74, 30.66)	(1896.04, 30.64)
Sync-BCD-TP-R	(2000, 5000)	0.3	(**93.17**, **8.15**)	(**119.38**, **7.57**)	(**148.05**, **7.39**)	(**173.73**, **7.13**)
Sync-BCD-TP-C			(679.48, 23.48)	(672.42, 24.47)	(627.10, 24.08)	(1158.11, 24.39)
IALM			(1150.52, 47.84)	(1304.51, 47.51)	(1450.57, 47.67)	(1590.68, 46.76)
LADMM			(1370.99, 31.49)	(1560.77, 31.55)	(1746.86, 31.64)	(1931.58, 31.56)
Sync-BCD-TP-R	(1000, 10 000)	0.1	(276.60, **15.12**)	(286.44, **15.04**)	(384.07, **14.45**)	(417.84, **15.00**)
Sync-BCD-TP-C			(**84.84**, 33.19)	(**85.20**, 37.54)	(**57.03**, 37.11)	(**123.92**, 37.25)
IALM			(3109.88, 64.10)	(3553.76, 61.16)	(3997.23, 60.27)	(4440.59, 56.81)
LADMM			(3761.72, 24.75)	(4299.11, 24.47)	(4836.50, 24.71)	(5373.87, 24.93)
Sync-BCD-TP-R	(1000, 10 000)	0.3	(283.47, **14.34**)	(376.80, **14.92**)	(405.42, **14.41**)	(430.40, **15.85**)
Sync-BCD-TP-C			(**134.16**, 38.10)	(**114.69**, 37.68)	(**58.43**, 40.84)	(**92.20**, 38.34)
IALM			(3132.08, 111.95)	(3579.23, 122.77)	(4026.25, 140.42)	(4472.96, 126.79)
LADMM			(3774.29, 35.96)	(4313.47, 36.26)	(4852.62, 36.26)	(5391.79, 36.42)

Bold values represent the best results

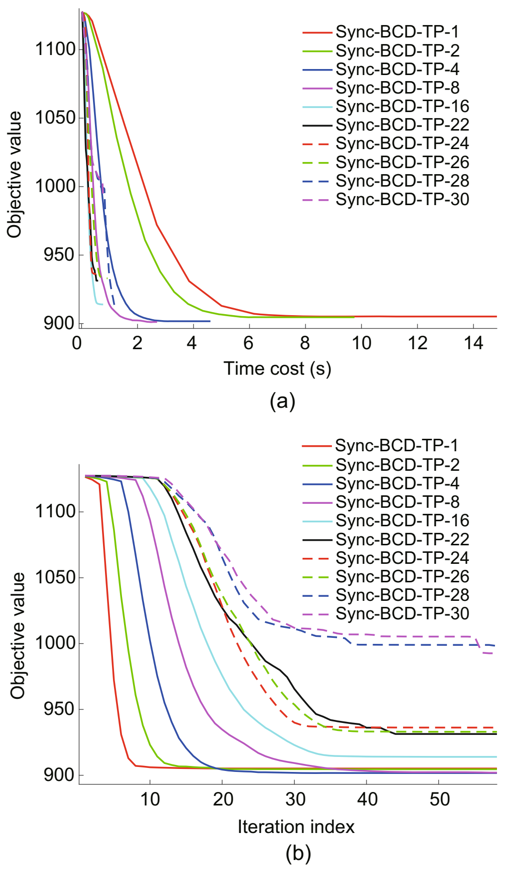

Fig. 2 Convergence behavior of synchronous block coordinate descent with time perturbation (Sync-BCD-TP) with the randomized selection rule: (a) objective value as a function of time cost with different numbers of threads; (b) objective value as a function of the iteration index with different numbers of threads. References to color refer to the online version of this figure

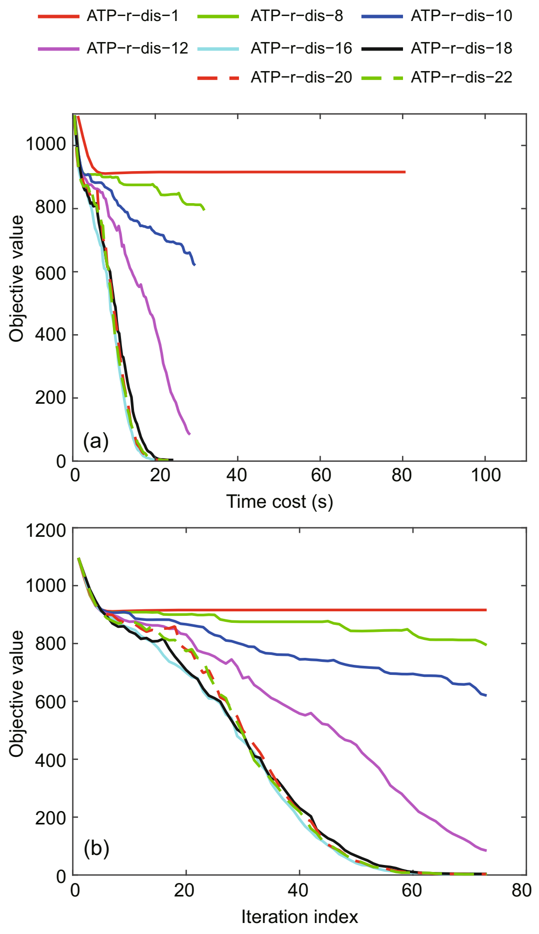

Fig. 3 Convergence behavior of ATP escaping from local optima and saddle points: (a) objective value as a function of time cost; (b) objective value as a function of the iteration index. References to color refer to the online version of this figure

the randomized BCD method, which is known as the best algorithm for large-scale problems.

6.3 Synchronous vs. asynchronous

Read lock or write lock is disabled when a worker is reading or writing shared variables to eliminate the costly synchronous update operation. The barrier parameter μ was set to be vanishing, as proposed in the ATP algorithm. An efficient line search criterion was adopted in the implementation. Other parameters were set according to our theoretical analysis.

Fig. 3 shows the convergence behavior of Algorithm 1 in solving the nonconvex folded concave linear regression problem, running with 1, 8, 10, 12, 16, 18, 20, and 22 threads. When the algorithm was updated synchronously, namely one-thread ATP, without the effect of time perturbation, it got stuck in bad saddle points in the nonconvex setting, with a final objective value of 906.7278. When the asynchronous update manner was adopted with time perturbation, the algorithm escaped from saddle points to achieve a final objective value of 2.6764 when tested with 22 threads. In Table 2, the asynchronous BCD with time perturbation technique shows significant improvement in escaping from local optimum points and saddle points. Besides, ATP converged fast with a multi-thread environment with the randomized variable selection rule. Furthermore, Fig. 4 demonstrates that ATP is scalable to multi-core platforms, and has an almost linear speedup ratio against the number of threads. We observe that the algorithm scales up to 22 threads for the tested problem. Degenerated convergence can be observed with 23 and 24 threads. This is mostly due to the following reasons:

1. Since there are only 24 processors (two CPUs, with each CPU having six physical cores and each physical core having two simulated processors due to the hyper threading feature from Intel) available on the server, the computation resource is limited. We further used the Intel VTune to analyze the performance of Sync-BCD-TP with the thread number varying from 1 to 8 on a notebook, and the results are publicly available (https://github.com/linboqiao/ATP/tree/master/Perf_analysis).

2. The total computational cost may vary due to the randomness of the randomized selection rule and the stopping criterion.

3. The variable $\boldsymbol{x}$ used for the current update is more staled when a relatively large number of threads are used, hence leading to slower convergence.

4. High cache miss rates and false sharing would significantly slow down the program execution.

Table 2 Objective values with time perturbation (TP) and without time perturbation (No-TP) on the problem of constrained folded concave penalized linear regression

d	n	m	Update manner	Objective value
1000	100	10	No-TP	928.16
1000	100	10	TP	19.65
1000	100	20	No-TP	910.85
1000	100	20	TP	2.48
1000	200	10	No-TP	826.64
1000	200	10	TP	436.61
1000	200	20	No-TP	829.04
1000	200	20	TP	2.35
2000	100	10	No-TP	1895.75
2000	100	10	TP	2.28
2000	100	20	No-TP	1919.50
2000	100	20	TP	2.22
2000	200	10	No-TP	1790.53
2000	200	10	TP	2.35
2000	200	20	No-TP	1824.84
2000	200	20	TP	2.47

7 Conclusions

In RWS, the last decade has witnessed a growing interest in big data in healthcare systems. However, traditional machine learning algorithms were not designed for solving such large-scale RWS problems due to a large amount of data that needs to be analyzed and the difficulty in solving problems with nonconvex nonlinear settings. In this paper, we have proposed a novel asynchronous block coordinate descent method with time perturbation, named ATP, to tackle large-scale nonconvex optimization problems with inequality constraints. We have presented convergence analysis of the proposed optimization algorithm with the randomized block variable selection rule and time perturbation, as well as iteration complexity analysis. Experiments conducted on real-world machine learning problems validated the efficacy of our proposed method. The experimental results demonstrated that time perturbation enables ATP to escape from saddle points and sub-optimal points, providing a promising way to handle nonconvex optimization problems with inequality constraints employing asynchronous block coordinate descent. The asynchronous parallel implementation on shared memory multi-core platforms indicated that the proposed algorithm ATP has strong scalability.

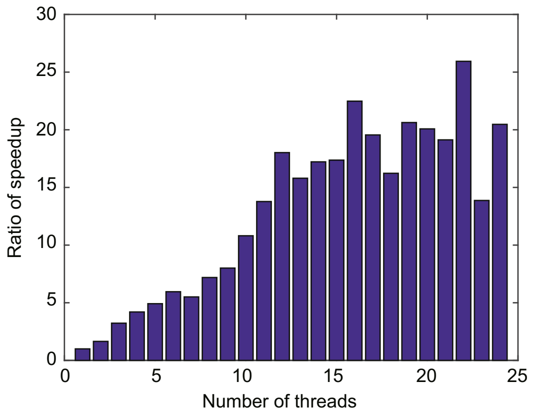

Fig. 4 Strong scaling results of ATP

Compliance with ethics guidelines

Rui LIU, Wei-chu SUN, Tao HOU, Chun-hong HU, and Lin-bo QIAO declare that they have no conflict of interest.

References

Anandkumar A, Ge R, 2016. Efficient approaches for escaping higher order saddle points in non-convex optimization. Proc 29th Annual Conf on Learning Theory, p.81-102.

Bertsekas DP, Tsitsiklis JN, 1989. Parallel and Distributed Computation: Numerical Methods. Prentice Hall, Englewood Cliffs, NJ, USA.

Bertsimas D, Bjarnadóttir MV, Kane MA, et al., 2008. Algorithmic prediction of health-care costs. *Oper Res*, 56(6):1382-1392. https://doi.org/10.1287/opre.1080.0619

Cannelli L, Facchinei F, Kungurtsev V, et al., 2018. Asynchronous parallel algorithms for nonconvex optimization. https://arxiv.org/abs/1607.04818

Dagum L, Menon R, 1998. OpenMP: an industry standard API for shared-memory programming. *IEEE Comput Sci Eng*, 5(1):46-55. https://doi.org/10.1109/99.660313

Fan JQ, Li RZ, 2001. Variable selection via nonconcave penalized likelihood and its oracle properties. *J Am Stat Assoc*, 96(456):1348-1360. https://doi.org/10.1198/016214501753382273

Friedman J, Hastie T, Tibshirani R, 2001. The Elements of Statistical Learning. Springer, Berlin, Germany.

Ge R, Huang FR, Jin C, et al., 2015. Escaping from saddle points—online stochastic gradient for tensor decomposition. Proc 28th Conf on Learning Theory, p.797-842.

Guennebaud G, Jacob B, 2010. Eigen v3. http://eigen.tuxfamily.org

Hazan E, Levy KY, Shalev-Shwartz S, 2016. On graduated optimization for stochastic non-convex problems. Proc 33rd Int Conf on Machine Learning, p.1833-1841.

Hong MY, Luo ZQ, Razaviyayn M, 2016. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. *SIAM J Optim*, 26(1):337-364. https://doi.org/10.1137/140990309

James GM, Paulson C, Rusmevichientong P, 2012. The Constrained Lasso. Working Paper, University of Southern California, Los Angeles, California, USA.

Jiang B, Lin TY, Ma SQ, et al., 2019. Structured nonconvex and nonsmooth optimization: algorithms and iteration complexity analysis. *Comput Optim Appl*, 72(1):115-157. https://doi.org/10.1007/s10589-018-0034-y

Jin C, Ge R, Netrapalli P, et al., 2017. How to escape saddle points efficiently. Proc 34th Int Conf on Machine Learning, p.1724-1732.

Khozin S, Blumenthal GM, Pazdur R, 2017. Real-world data for clinical evidence generation in oncology. *J Nat Cancer Inst*, 109(11), Article djx187. https://doi.org/10.1093/jnci/djx187

Li D, Lai Z, Ge K, et al., 2019. HPDL: towards a general framework for high-performance distributed deep learning. Proc 39th IEEE Int Conf on Distributed Computing Systems, p.1742-1753.

Li JQ, Vachani A, Epstein A, et al., 2018. A doubly robust approach for cost-effectiveness estimation from observational data. *Stat Methods Med Res*, 27(10):3126-3138. https://doi.org/10.1177/0962280217693262

Liu R, 2019. Asynchronous block coordinate descent method for large-scale nonconvex problem in real world study. Proc IEEE 21st Int Conf on High Performance Computing and Communications, jointly with IEEE 17th Int Conf on Smart City and IEEE 5th Int Conf on Data Science and Systems, p.2033-2038. https://doi.org/10.1109/HPCC/SmartCity/DSS.2019.00281

Nesterov Y, 2012. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM J Optim*, 22(2):341-362. https://doi.org/10.1137/100802001

Nesterov Y, Nemirovskii A, 1994. Interior-Point Polynomial Algorithms in Convex Programming. SIAM, Philadelphia, USA. https://doi.org/10.1137/1.9781611970791

Palomar DP, Chiang M, 2006. A tutorial on decomposition methods for network utility maximization. *IEEE J Sel Areas Commun*, 24(8):1439-1451. https://doi.org/10.1109/JSAC.2006.879350

Peng ZM, Xu YY, Yan M, et al., 2019. On the convergence of asynchronous parallel iteration with unbounded delays. *J Oper Res Soc China*, 7(1):5-42. https://doi.org/10.1007/s40305-017-0183-1

Qiao LB, Zhang BF, Su JS, et al., 2016a. Linearized alternating direction method of multipliers for constrained nonconvex regularized optimization. Proc 8th Asian Conf on Machine Learning, p.97-109.

Qiao LB, Lin TY, Jiang YG, et al., 2016b. On stochastic primal-dual hybrid gradient approach for compositely regularized minimization. Proc 22nd European Conf on Artificial Intelligence, p.167-174. https://doi.org/10.3233/978-1-61499-672-9-167

Qiao LB, Zhang BF, Lu XC, et al., 2017. Adaptive linearized alternating direction method of multipliers for non-convex compositely regularized optimization problems. *Tsinghua Sci Technol*, 22(3):328-341. https://doi.org/10.23919/TST.2017.7914204

Qiao LB, Lin TY, Qin Q, et al., 2018. On the iteration complexity analysis of stochastic primal-dual hybrid gradient approach with high probability. *Neurocomputing*, 307:78-90. https://doi.org/10.1016/j.neucom.2018.03.066

Razaviyayn M, Hong MY, Luo ZQ, et al., 2014. Parallel successive convex approximation for nonsmooth nonconvex optimization. Proc 27th Int Conf on Neural Information Processing Systems, p.1440-1448.

Rockafellar RT, Wets RJB, 2009. Variational Analysis. Springer, New York, USA.

Shen L, Liu W, Yuan G, et al., 2017. GSOS: Gauss-Seidel operator splitting algorithm for multi-term nonsmooth convex composite optimization. Proc 34th Int Conf on Machine Learning, p.3125-3134.

Shen L, Sun P, Wang YT, et al., 2018. An algorithmic framework of variable metric over-relaxed hybrid proximal extra-gradient method. https://arxiv.org/abs/1805.06137

Sun MZ, Jiang Y, Sun C, et al., 2019. The associations between smoking and obesity in northeast China: a quantile regression analysis. *Sci Rep*, 9, Article 3732. https://doi.org/10.1038/s41598-019-39425-6

Sun T, Hannah R, Yin W, et al., 2017. Asynchronous coordinate descent under more realistic assumptions. Proc Advances in Neural Information Processing Systems, p.6182-6190.

Wang X, Ma SQ, Goldfarb D, et al., 2017. Stochastic quasi-Newton methods for nonconvex stochastic optimization. *SIAM J Optim*, 27(2):927-956. https://doi.org/10.1137/15M1053141

Wang Y, Yin W, Zeng J, 2019. Global convergence of ADMM in nonconvex nonsmooth optimization. *J Sci Comput*, 78(1):29-63. https://doi.org/10.1007/s10915-018-0757-z

Xiao L, Johansson M, Boyd SP, 2004. Simultaneous routing and resource allocation via dual decomposition. *IEEE Trans Commun*, 52(7):1136-1144. https://doi.org/10.1109/TCOMM.2004.831346

Xu YY, 2019. Asynchronous parallel primal-dual block coordinate update methods for affinely constrained convex programs. https://arxiv.org/abs/1705.06391

Xu YY, Yin W, 2017. A globally convergent algorithm for nonconvex optimization based on block coordinate update. *J Sci Comput*, 72(2):700-734. https://doi.org/10.1007/s10915-017-0376-0

Zhang CH, 2010. Nearly unbiased variable selection under minimax concave penalty. *Ann Stat*, 38(2):894-942. https://doi.org/10.1214/09-AOS729

Zhang T, 2010. Analysis of multi-stage convex relaxation for sparse regularization. *J Mach Learn Res*, 11:1081-1107.

Zhang X, Liu J, Zhu ZY, 2018. Taming convergence for asynchronous stochastic gradient descent with unbounded delay in non-convex learning. https://arxiv.org/abs/1805.09470

Zou FY, Shen L, Jie ZQ, et al., 2018. A sufficient condition for convergences of Adam and RMSProp. Proc IEEE Conf on Computer Vision and Pattern Recognition, p.11127-11135. https://arxiv.org/abs/1811.09358

Zou FY, Shen L, Jie ZQ, et al., 2019. Weighted AdaGrad with unified momentum. https://arxiv.org/abs/1808.03408