Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

Stochastic Parallel Block Coordinate Descent for Large-Scale Saddle Point Problems

Zhanxing Zhu and **Amos J. Storkey**
School of Informatics, University of Edinburgh, UK
{zhanxing.zhu, a.storkey}@ed.ac.uk

Abstract

We consider convex-concave saddle point problems with a *separable* structure and *non-strongly convex* functions. We propose an efficient stochastic block coordinate descent method using *adaptive* primal-dual updates, which enables flexible parallel optimization for large-scale problems. Our method shares the efficiency and flexibility of block coordinate descent methods with the simplicity of primal-dual methods and utilizing the structure of the separable convex-concave saddle point problem. It is capable of solving a wide range of machine learning applications, including robust principal component analysis, Lasso, and feature selection by group Lasso, etc. Theoretically and empirically, we demonstrate significantly better performance than state-of-the-art methods in all these applications.

1 Introduction

A large number of machine learning (ML) models can be cast as convex-concave saddle point (CCSP) problems. There are two common cases. First, convex optimization problems with linear constraints can easily be reformulated as CCSP problems by introducing Lagrangian multipliers (Chen, Donoho, and Saunders 2001; Boyd et al. 2011; Wainwright 2014). Second, empirical risk minimization with regularization (ERM, (Hastie, Tibshirani, and Friedman 2009)) can be reformulated as CCSP problem by conjugate dual transformation. In machine learning applications, these two groups of CCSP problems often exhibit a separable additive structure. Developing efficient optimization methods for seperable CCSP problems is especially important for large-scale applications. Existing work, such as (Zhang and Xiao 2015; Zhu and Storkey 2015a), assumes the strong convexity of each of the separable functions, and applies to ERM problems. Although the strong convexity assumption can be relaxed, there is no guide on how to select the extra regularization parameters. We also find the relaxation significantly hinders convergence rates even for post-hoc optimal choices of parameters. Furthermore, inappropriate parameter selection dramatically deteriorates the practical performance. Even for strongly-convex systems the strong-convexity parameter is often hard to determine. Additionally, it is currently unclear how to adapt the stepsize for handling block separable problems.

In this work, we propose a novel stochastic and parallelizable approach for Sep-CCSP problem, which naturally handles convex cases that are not strongly convex and avoids any notorious hyperparameter selection issues. This method is also capable of dealing with block separable CCSP problem. In the following, we formally introduce the Sep-CSSP problem and consider the two common machine learning instantiations of this problem.

The generic **convex-concave saddle point problem** is written as

$$\min_{\mathbf{x}\in\mathbb{R}^n}\max_{\mathbf{y}\in\mathbb{R}^m}\left\{L(\mathbf{x},\mathbf{y})=f(\mathbf{x})+\langle\mathbf{y},\mathbf{A}\mathbf{x}\rangle-g^*(\mathbf{y})\right\}, \quad (1)$$

where $f(\mathbf{x})$ is a proper convex function, g^* is the convex conjugate of a convex function g, and $\mathbf{A}\in\mathbb{R}^{m\times n}$. Many machine learning tasks reduce to solving a problem of this form. One important subclass of (1) is where $f(\mathbf{x})$ or $g^*(\mathbf{y})$ exhibits an additive separable structure. We say $f(\mathbf{x})$ is *separable* when $f(\mathbf{x})=\sum_{j=1}^J f_j(\mathbf{x}_j)$, with $\mathbf{x}_j\in\mathbb{R}^{n_j}$, and $\sum_{j=1}^J n_j=n$. Separability for $g^*(\cdot)$ is defined likewise. We can also partition matrix $\mathbf{A}$ into J column blocks $\mathbf{A}_j\in\mathbb{R}^{m\times n_j}$, $j=1,\dots,J$, and $\mathbf{A}\mathbf{x}=\sum_{j=1}^J\mathbf{A}_j\mathbf{x}_j$, resulting in a problem of the form

$$\min_{\mathbf{x}\in\mathbb{R}^n}\max_{\mathbf{y}\in\mathbb{R}^m}\sum_{j=1}^J f_j(\mathbf{x}_j)+\sum_{j=1}^J\langle\mathbf{y},\mathbf{A}_j\mathbf{x}_j\rangle-g^*(\mathbf{y}). \quad (2)$$

We call problems of the form (2) **Separable Convex Concave Saddle point (*Sep-CCSP*)** problems. We develop an efficient optimization method for Sep-CCSP problems when $f(\cdot)$ and/or $g^*(\cdot)$ are *non-strongly convex*; many ML methods result in a non-strongly convex Sep-CCSP form.

Example 1 Separable function minimization with linear constraints takes the form

$$\min_{\mathbf{x}}\sum_{i=1}^J f_i(\mathbf{x}_i)\ \text{s.t.}\ \sum_{i=1}^J\mathbf{A}_i\mathbf{x}_i=\mathbf{b}, \quad (3)$$

leading to

$$\min_{\mathbf{x}}\max_{\mathbf{y}}L(\mathbf{x},\mathbf{y})=\sum_{i=1}^J f_i(\mathbf{x}_i)+\langle\mathbf{y},\sum_{i=1}^J\mathbf{A}_i\mathbf{x}_i\rangle-\mathbf{y}^T\mathbf{b} \quad (4)$$

when we introduce Lagrangian multipliers $\mathbf{y}$ for the linear constraints. Here $g^*(\mathbf{y}) = \mathbf{y}^T\mathbf{b}$ is non-strongly convex. A large number of machine learning problems can be expressed as linearly constrained optimization problems of this form (Chen, Donoho, and Saunders 2001; Boyd et al. 2011; Wainwright 2014), for instance, robust principal component analysis (RPCA) (Wright et al. 2009; Candès et al. 2011).

Example 2 Another important case of Sep-CCSP is empirical risk minimization (ERM) of linear predictors, with a convex regularization function $f(\mathbf{x})$:

$$\min_{\mathbf{x}} f(\mathbf{x}) + \frac{1}{N}\sum_{i=1}^{N} g_i(\mathbf{a}_i^T\mathbf{x}) \tag{5}$$

where N labels the number of data points. Many well-known classification and regression problems are included in this formulation, such as group Lasso (Yuan and Lin 2006) with the regularizer as a sum of groupwise L_2-norm $f(\mathbf{x}) = \sum_{g=1}^{G} f_g(\mathbf{x}_g) = \lambda\sum_{g=1}^{G} w_g\|\mathbf{x}_g\|_2$. Reformulating the above regularized ERM by employing the conjugate dual of function g, i.e.,

$$g_i(\mathbf{a}_i^T\mathbf{x}) = \sup_{y_i\in\mathbb{R}} y_i\langle\mathbf{a}_i,\mathbf{x}\rangle - g_i^*(y_i), \tag{6}$$

we transform it into a Sep-CCSP problem,

$$\min_{\mathbf{x}}\max_{\mathbf{y}} \sum_{g=1}^{G} f_g(\mathbf{x}_g) + \frac{1}{N}\langle\sum_{i=1}^{N} y_i\mathbf{a}_i,\mathbf{x}\rangle - \frac{1}{N}\sum_{i=1}^{N} g_i^*(y_i). \tag{7}$$

If $g_i(\cdot)$ is not smooth (e.g. hinge or absolute loss), the conjugate dual $g^*(\cdot)$ is non-strongly convex.

Inspired by current active research on block coordinate descent methods (BCD, (Nesterov 2012; Richtárik and Takáč 2015; 2014)), we propose a Stochastic Parallel Block Coordinate Descent method (SP-BCD) for solving the separable convex-concave saddle point problems, particularly non-strongly convex functions. The key idea is to apply stochastic block coordinate descent of the separable primal space into the primal-dual framework (Chambolle and Pock 2014; Pock and Chambolle 2011) for the Sep-CCSP problem. We propose a novel *adaptive* stepsize for both the primal and dual updates to improve algorithm convergence performance. Compared with the standard primal-dual framework, our method enables the selected blocks of variables to be optimized in parallel according to the processing cores available. Without any assumption of strong convexity or smoothness, our method can achieve an $O(1/T)$ convergence rate, which is the best known rate for non-strongly (and non-smooth) convex problem. Also, in a wide range of applications, we show that SP-BCD can achieve significantly better performance than the aforementioned state-of-the-art methods. These results are presented in Section 4.

Wang, Banerjee, and Luo(2014) proposed a stochastic and parallel algorithm for solving the problem (3). However, their method is based on an augmented Lagrangian, often suffering from the selection of penalty parameter. As previously discussed, the methods for handling Sep-CCSP in (Zhang and Xiao 2015; Zhu and Storkey 2015a) focused on the ERM problem, and assumed that both $f(\mathbf{x})$ and $g^*(\mathbf{y})$ are strongly convex, or relaxed that constraint in ways that we show significantly hits performance, and required additional hyperparameter selection (as do augmented Lagrangian methods). Additionally, the method in (Zhang and Xiao 2015) is not capable of handling block separable CCSP problem. These all limit its applicability. Our approach SP-BCD can overcome these difficulties, which can (i) naturally handle non-strongly convex functions, and avoids any notorious hyperparameter selection issues; (ii) is capable of handling block separable CCSP problem.

2 Primal-dual Framework for CCSP

In (Chambolle and Pock 2011), the authors proposed a first-order primal-dual method for (non-smooth) convex problems with saddle-point structure, i.e., Problem (1). We refer this algorithm as PDCP. The update of PDCP in $(t+1)$-th iteration is as follows:

$$\mathbf{y}^{t+1} = \text{argmin}_{\mathbf{y}} g^*(\mathbf{y}) - \langle\mathbf{y},\mathbf{A}\overline{\mathbf{x}}^t\rangle + \frac{\sigma}{2}\|\mathbf{y}-\mathbf{y}^t\|_2^2 \tag{8}$$

$$\mathbf{x}^{t+1} = \text{argmin}_{\mathbf{x}} f(\mathbf{x}) + \langle\mathbf{y}^{t+1},\mathbf{A}\mathbf{x}\rangle + \frac{h}{2}\|\mathbf{x}-\mathbf{x}^t\|_2^2 \tag{9}$$

$$\overline{\mathbf{x}}^{t+1} = \mathbf{x}^{t+1} + \theta(\mathbf{x}^{t+1}-\mathbf{x}^t). \tag{10}$$

When the parameter configuration satisfies $\sigma h \geq \|\mathbf{A}\|^2$ and $\theta = 1$, PDCP can achieve a $O(1/T)$ convergence rate. For the general CCSP problem, PDCP does not consider the structure of matrix $\mathbf{A}$ and only applies constant stepsize for all dimensions of primal and dual variables. Based on PDCP, the authors in (Pock and Chambolle 2011) used the structure of matrix $\mathbf{A}$ and proposed a diagonal preconditioning technique for PDCP, which showed better performance in several computer vision applications. However, when the function $f(\mathbf{x})$ has separable structure with many blocks of coordinates, both these algorithms are batch methods and non-stochastic, i.e. they have to update all the primal coordinates in each iteration. This influences empirical efficiency.

Inspired by the recent success of coordinate descent methods for solving separable optimization problems, we incorporate a stochastic block coordinate descent technique into above primal-dual methods and propose adaptive stepsizes for the chosen blocks via the structure of the matrix $\mathbf{A}$.

3 Our Method: SP-BCD for Sep-CCSP

The basic idea of our stochastic parallel block coordinate descent (SP-BCD) method for solving the saddle point problem (2) is simple; we optimize $L(\mathbf{x},\mathbf{y})$ by alternatively updating the primal and dual variables in a principled way. Thanks to the separable structure of $f(\mathbf{x})$, in each iteration we can randomly select K blocks of variables whose indices are denoted as S_t, and then we only update these selected blocks, given the current $y = y^t$, in the following way. If $j \in S_t$ then

$$\mathbf{x}_j^{t+1} = \text{argmin}_{\mathbf{x}_j} f_j(\mathbf{x}_j) + \langle\mathbf{y}^t,\mathbf{A}_j\mathbf{x}_j\rangle + \frac{1}{2}\|\mathbf{x}_j-\mathbf{x}_j^t\|_{\mathbf{h}_j}^2, \tag{11}$$

otherwise, we just keep $\mathbf{x}_j^{t+1} = \mathbf{x}_j^t$. In the blockwise update, we add a proximal term to penalize the deviation from last update $\mathbf{x}_j^t$, i.e.,

$$\frac{1}{2}\|\mathbf{x}_j - \mathbf{x}_j^t\|_{\mathbf{h}_j}^2 = \frac{1}{2}(\mathbf{x}_j - \mathbf{x}_j^t)^T \text{diag}(\mathbf{h}_j)(\mathbf{x}_j - \mathbf{x}_j^t), \quad (12)$$

where the diagonal matrix $\mathbf{H}_j = \text{diag}(\mathbf{h}_j)$ is applied for scaling each dimension of $\mathbf{x}_j$, and each $\mathbf{h}_j$ is a subvector of $\mathbf{h} = \left[\mathbf{h}_1^T, \ldots, \mathbf{h}_J^T\right]^T$. We configure the each dimension of $\mathbf{h}$ as

$$h_d = \sum_{j=1}^{m} |A_{jd}|, \quad d = 1, 2, \ldots, n. \quad (13)$$

Intuitively, h_d in our method can be interpreted as the coupling strength between the d-th dimension of the primal variable $\mathbf{x}$ and dual variable $\mathbf{y}$, measured by the L_1 norm of the vector $\mathbf{A}_{:,d}$ (i.e., the d-th column of matrix $\mathbf{A}$). Smaller coupling strength allows us to use smaller proximal penalty (i.e., larger stepsize) for updating the current primal variable block without caring too much about its influence on dual variable, and vice versa.

Then for those selected block variables, we use an extrapolation technique given in Eq.(10) to yield an intermediate variable $\overline{\mathbf{x}}^{t+1}$ as follows,

$$\overline{\mathbf{x}}_j^{t+1} = \begin{cases} \mathbf{x}_j^{t+1} + \theta\left(\mathbf{x}_j^{t+1} - \mathbf{x}_j^t\right) & \text{if } j \in S_t \\ \overline{\mathbf{x}}_j^t & \text{otherwise,} \end{cases} \quad (14)$$

where $\theta = K/J$ to account for there being only K blocks out of J selected in each iteration.

Assuming $g^*(\mathbf{y})$ is not separable, we update the dual variable as a whole. A similar proximal term is added with the diagonal matrix $\mathbf{\Sigma}^t = \text{diag}(\boldsymbol{\sigma}^t)$:

$$\mathbf{y}^{t+1} = \text{argmin}_{\mathbf{y}} g^*(\mathbf{y}) - \langle \mathbf{y}, \overline{\mathbf{r}}^t + \frac{J}{K}\sum_{j\in S_t} \mathbf{A}_j(\overline{\mathbf{x}}_j^{t+1} - \overline{\mathbf{x}}_j^t)\rangle + \frac{1}{2}\|\mathbf{y} - \mathbf{y}^t\|_{\boldsymbol{\sigma}^t}^2, \quad (15)$$

where $\overline{\mathbf{r}}^t = \sum_{j=1}^J \mathbf{A}_j \overline{\mathbf{x}}_j^t$. We configure the dual proximal penalty $\boldsymbol{\sigma}^t$ *adaptively* for each iteration,

$$\sigma_k^t = \frac{J}{K}\sum_{j\in S_t} |A_{kj}|, \quad k = 1, 2, \ldots, m. \quad (16)$$

This configuration adaptively accounts for the coupling strength between the dual variable and the chosen primal variable blocks in S_t through measuring the structure of the matrix $\mathbf{A}$. Later we show that the usage of the proposed adaptive proximal penalty for both primal and dual update contributes to significantly improve the convergence performance for many machine learning applications.

Another crucial component of the dual update is the construction of the term $\overline{\mathbf{r}}^t + \frac{J}{K}\sum_{j\in S_t} \mathbf{A}_j(\overline{\mathbf{x}}_j^{t+1} - \overline{\mathbf{x}}_j^t)$, which is inspired by a recently proposed fast incremental gradient method for non-strongly convex functions, SAGA (Defazio, Bach, and Lacoste-Julien 2014). We use the combination of the cached sum of all $\mathbf{A}_j\overline{\mathbf{x}}_j^t$, i.e., $\overline{\mathbf{r}}^t$, and the newly updated sample average $\frac{1}{K}\sum_{j\in S_t} \mathbf{A}_j(\overline{\mathbf{x}}_j^{t+1} - \overline{\mathbf{x}}_j^t)$ to obtain a variance reduced estimation of $\mathbb{E}[\overline{\mathbf{r}}]$, which is essentially the spirit of SAGA. After the dual update, $\overline{\mathbf{r}}^t$ is updated to $\overline{\mathbf{r}}^{t+1}$ using,

$$\overline{\mathbf{r}}^{t+1} = \overline{\mathbf{r}}^t + \sum_{j\in S_t} \mathbf{A}_j\left(\overline{\mathbf{x}}_j^{t+1} - \overline{\mathbf{x}}_j^t\right). \quad (17)$$

Algorithm 1 SP-BCD for Separable Convex-Concave Saddle Point Problems

Input: number of blocks picked in each iteration K, $\theta = K/J$, the configuration of $\mathbf{h}$ and $\boldsymbol{\sigma}^t$ as given in Eq. (13) and (16), respectively.
Initialize: $\mathbf{x}^0$, $\mathbf{y}^0$, $\overline{\mathbf{x}}^0 = \mathbf{x}^0$, $\overline{\mathbf{r}}^0 = \sum_{j=1}^J \mathbf{A}_j\overline{\mathbf{x}}_j^0$
for $t = 1, 2, \ldots, T$ **do**
  Randomly pick set S_t of K blocks from $\{1, \ldots, J\}$ each chosen with probability K/J.
  for each block in parallel **do**
    Update each primal variable block using Eq.(11), and extrapolate it using Eq.(14);
  end for
  Update dual variables using Eq.(15) and update $\overline{\mathbf{r}}^{t+1}$ using Eq. (17).
end for

The whole procedure for solving Sep-CCSP problem (2) using SP-BCD is summarized in Algorithm 1. There are several notable characteristics of our algorithm:

1. This algorithm is amenable to parallelism for large-scale optimization, which is suitable for modern computing clusters. Our method possesses one of key advantages of stochastic parallel coordinate descent method (Richtárik and Takáč 2015): providing the flexibility that in each iteration the number of selected blocks can be optimized completely in parallel according to available number of machines or computational cores. This could make use of all the computational availability as effectively as possible.

2. The related non-stochastic primal-dual algorithms (Chambolle and Pock 2011; 2014) need evaluation of the norm of $\mathbf{A}$. For large problem size, the norm evaluation can be time-consuming. The parameter configuration in our algorithm avoids norm estimation, but maintains a $O(1/T)$ convergence rate.

3. Although an augmented Lagrangian framework, such as ADMM, can implement an effective optimization for many problems with linear constraints (3), the selection of the penalty parameter has a dramatic influence on its performance. Current selection rules rely on various heuristics or exhaustive search, and no theoretical justifications exist. This difficulty also occurs with other recent work (Zhang and Xiao 2015) when $f(\mathbf{x})$ and $g^*(\mathbf{y})$ are not strongly convex. Our method avoids this issue.

Convergence Analysis

For a convergence analysis, we employ the following gap for any saddle point $(\mathbf{x}, \mathbf{y})$, $\mathcal{G}(\mathbf{x}', \mathbf{y}') \triangleq \max_{\mathbf{y}} L(\mathbf{x}', \mathbf{y}) - \min_{\mathbf{x}} L(\mathbf{x}, \mathbf{y}')$. As discussed by (Chambolle and Pock 2011), this gap will practically measure the optimality of the algorithm if the domain of the $(\mathbf{x}', \mathbf{y}')$ is " large enough" such that $(\mathbf{x}', \mathbf{y}')$ could lie in the interior of their domains.

The following theorem establishes the convergence of our algorithm.

Theorem 1. *Given that all $f_i(\mathbf{x}_i)$ and $g^*(\mathbf{y})$ are convex functions, and we set $\theta = K/J$, proximal parameters for primal and dual update as Eq.(13) and (16), respectively. Then for any saddle point $(\mathbf{x}, \mathbf{y})$, the expected gap decays as the following rate:*

$$\mathbb{E}\left[L\left(\sum_{t=1}^{T}\mathbf{x}^t/T, \mathbf{y}\right) - L\left(\mathbf{x}, \sum_{t=1}^{T}\mathbf{y}^t/T\right)\right] \leq \frac{1}{T}M(0),$$

where $M(0) =$

$$\frac{J}{2K}\|\mathbf{x}^0 - \mathbf{x}\|_{\mathbf{h}}^2 + \frac{1}{2}\|\mathbf{y}^0 - \mathbf{y}\|_{\boldsymbol{\sigma}^0}^2 - \langle \mathbf{y}^0 - \mathbf{y}, \mathbf{A}\left(\mathbf{x}^0 - \mathbf{x}\right)\rangle$$
$$+ \frac{J-K}{K}\left(f(\mathbf{x}^0) + \langle \mathbf{y}, \mathbf{A}\mathbf{x}^0\rangle - (f(\mathbf{x}) + \langle \mathbf{y}, \mathbf{A}\mathbf{x}\rangle)\right).$$

The proof of the above theorem is technical and given in the full version of this paper in (Zhu and Storkey 2015b).

Remark. For the parameter configuration in Theorem 1, when $\theta = K/J$, the key point for obtaining the convergence of our algorithm is that we select one particular configuration of $\mathbf{h}$ and $\boldsymbol{\sigma}^t$ to guarantee the positive semidefiniteness of the following matrix,

$$\mathbf{P} = \begin{bmatrix} \mathrm{diag}(\mathbf{h}_{S_t}) & -\mathbf{A}_{S_t}^T \\ -\mathbf{A}_{S_t} & \frac{K}{J}\mathrm{diag}(\boldsymbol{\sigma}^t) \end{bmatrix} \succeq 0. \qquad (18)$$

Under the parameter configuration of $\mathbf{h}$ and $\boldsymbol{\sigma}^t$ in Theorem 1, we can guarantee matrix $\mathbf{P}$ is diagonally dominant, directly leading positive semidefiniteness. However, the parameter configuration to make $\mathbf{P} \succeq 0$ is not unique. We find that other configurations are also valid, for instance, for each block j, $\mathbf{h}_j = \|\mathbf{A}_j\|\mathbf{I}$ and $\boldsymbol{\sigma} = \frac{J}{K}\sigma\mathbf{I}$, where $\sigma = \max\{\|\mathbf{A}_j\|\}_{j=1}^J$. Different parameter configuration might provide some influence on the performance of the algorithm. We leave the comparison between them and further theoretical analysis as future work.

4 Applications

In this section, we provide examples of Sep-CCSP problems in machine learning. In each application, we select different methods to compare with that have already shown strong performance in that particular scenario. Note that, since the method in (Zhang and Xiao 2015) cannot handle block separable CCSP problem, it is not applicable for the first and third experiment. To provide a fair comparison with other methods, all the experiments are implemented in one core/machine. Each experiment is run 10 times and the average results are reported to show statistical consistency.

Robust Principal Component Analysis

Robust Principal Component Analysis (RPCA) is a variant of PCA to obtain a low rank and sparse decomposition of an observed data matrix $\mathbf{B}$ corrupted by noise (Wright et al. 2009; Candès et al. 2011), which could help to handle outliers existing in datasets. RPCA aims to solve the following optimization problem,

$$\min_{\{\mathbf{X}_i\}_{i=1}^3} \frac{1}{2}\|\mathbf{X}_1\|_{\mathrm{F}}^2 + \mu_2\|\mathbf{X}_2\|_1 + \mu_3\|\mathbf{X}_3\|_* \quad \text{s.t. } \mathbf{B} = \sum_{i=1}^{3}\mathbf{X}_i,$$

Table 1: RPCA problem: performance of all compared methods (with ADMM, GSADMM and PDMM hyperparameters set to the post-hoc optimal).

Methods	Iteration	Time (s)	Frobenus norm of residual (10^{-4})	Objective (10^8)
ADMM	149	2191	9.71	1.924
GSADMM	**23**	**448**	8.69	1.924
PDCP	59	911	7.80	1.924
PDMM1	125	927	9.92	1.924
PDMM2	73	750	**4.55**	1.924
PDMM3	67	834	8.56	1.924
SP-BCD1	104	784	7.63	1.924
SP-BCD2	48	**492**	**6.17**	1.924
SP-BCD3	42	553	6.72	1.924

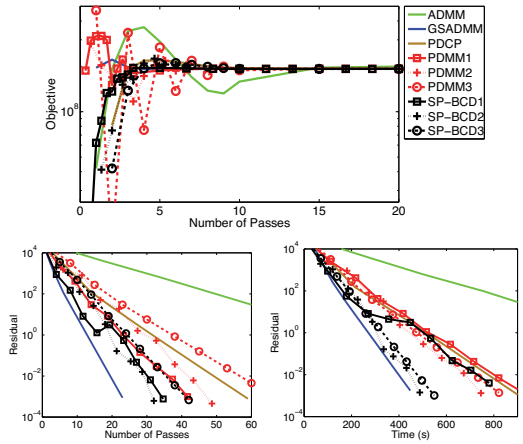

Figure 1: RPCA problem: our method (with $K = \{1, 2, 3\}$) versus ADMM, GSADMM, PDCP and PDMM (with $K = \{1, 2, 3\}$).

where $\mathbf{B} \in \mathbb{R}^{m\times n}$, $\mathbf{X}_1$ is a noise matrix, $\mathbf{X}_2$ is a sparse matrix, $\mathbf{X}_3$ is a low rank matrix, and $\|\cdot\|_*$ is the nuclear norm of a matrix. We generate the observation matrix $\mathbf{B}$ in the same way as (Parikh and Boyd 2014), where we have $m = 2000$, $n = 5000$ and the rank is $r = 100$. The regularization parameters are set as $\mu_2 = 0.15\|\mathbf{B}\|_\infty$ and $\mu_3 = 0.15\|\mathbf{B}\|$. Note that RPCA problem with this matrix size is non-trivial since there are in total $30,000,000$ variables and $10,000,000$ equality constraints to handle.

In this particular application, the parameter configuration for SP-BCD with each different number of blocks K chosen from the possible 3 in each iteration can be obtained: (1) $K = 1$, $(\theta, h, \sigma^t) = (1/3, 1, 1)$; (2) $K = 2$, $(\theta, h, \sigma^t) = (2/3, 1, 2)$; (3) $K = 3$, $(\theta, h, \sigma^t) = (1, 1, 3)$.

Our method SP-BCD is compared with (1) ADMM implemented by (Parikh and Boyd 2013); (2) Gauss-Seidel ADMM (GSADMM) (Hong and Luo 2012), which solves the problem (3) in a cyclic block coordinate manner. However, GSADMM with multiple blocks is not well understood and there is no theory guarantee, and GSADMM has to be implemented sequentially and cannot be paral-

lel; (3) PDCP (Chambolle and Pock 2011), for which the recommended parameter configuration can be computed as $(\theta, h, \sigma) = \left(1, \sqrt{3}, \sqrt{3}\right)$; (4) PDMM (Wang, Banerjee, and Luo 2014) with $K = \{1, 2, 3\}$. For each of the three competing methods (ADMM, GSADMM and PDMM) we run extensive experiments using different penalty parameter values ρ, and report the results for best performing ρ, despite the fact that knowledge of which ρ is optimal is not available to the algorithms a priori. Hence the real-world performance of SP-BCD relative to these methods is significantly greater than these figures suggest.

Figure 1 depicts the performance of all the methods on the evolution of the objective and the residual (i.e., the deviation from satisfied constraints measured by $\|\mathbf{X}_1 + \mathbf{X}_2 + \mathbf{X}_3 - \mathbf{B}\|_{\text{fro}}$) w.r.t. number of passes and consumed time. All methods quickly achieve the consensus objective value in 20 passes. The key difference in performance is how fast they satisfy the equality constraint. Our method SP-BCD with $K = 2$ is the fastest, achieving almost the same performance with GASDMM, while being fully parallelizable whereas GSADMM can only be run sequentially. Although PDMM2 obtains the lowest residual (measured by Frobenus Norm of deviation of satisfied constraints), it spends much longer time $750s$, compared with $492s$ for SP-BCD2. When we run the SP-BCD2 with the same amount of time as that of PDMM2, SP-BCD2 could achieve Frobenus Norm of residual as $\mathbf{2.36 \times 10^{-4}}$, which shows better performance than PDMM2. The real difference in performance is greater as optimal hyperparameters are not actually available to the competing methods.

Lasso

Lasso is an important l_1 regularized linear regression, solving the optimization problem,

$$\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \lambda\|\mathbf{x}\|_1 \tag{19}$$

where λ is a regularization parameter, and $\mathbf{A} \in \mathbb{R}^{m \times n}$ is an observed feature matrix. In typical applications, there are many more features than number of training examples, i.e., $m < n$. By dualizing the first quadratic loss function in (19), we can have its Sep-CCSP form

$$\min_{\mathbf{x}\in\mathbb{R}^n} \max_{\mathbf{y}\in\mathbb{R}^m} \lambda\|\mathbf{x}\|_1 + \langle \mathbf{y}, \mathbf{A}\mathbf{x}\rangle - \sum_{i=1}^{m}\left(\frac{1}{2}y_i^2 + b_i y_i\right). \tag{20}$$

Since $\|\mathbf{x}\|_1$ is totally separable and non-strongly convex, we can apply our SP-BCD method to the above saddle point problem, i.e., in each iteration we randomly select K coordinates of primal variable $\mathbf{x}$ to update. For the dual update, the corresponding problem has a simple close-formed solution that can be updated directly.

Due to the vast literature for the Lasso problem, we only choose several representative methods to compare with our method, (1) ISTA (Iterative Shrinkage-Thresholding Algorithm); (2) FISTA (Fast ISTA, (Beck and Teboulle 2009)); (3) ADMM (Boyd et al. 2011, Chap 6.4), note that the formulation of ADMM for Lasso problem is different from Eq.(19). ADMM splits the loss function and regularization term using two separable variables, which needs to solve a linear system in each iteration. When the problem size is very large, the time complexity is high and even computationally inacceptable. (4) PDCP (Chambolle and Pock 2011), which needs estimation of norm of matrix $\mathbf{A}$. (5) SPDC (Zhang and Xiao 2015) needs an extra regularization parameter to adapt non-strong convexity. We choose optimal regularization parameter by post-hoc selection.

We generate the data as in (Boyd et al. 2011, Chap 11.1). For SP-BCD and SPDC, we randomly choose $K = 100$ coordinates per iteration to run the experiments.

Table 2 reports the performance of all these methods on two problems with different sizes and sparsity. We can observe that SP-BCD uses the least number of passes and time to achieve same objective value with other methods. For smaller sized problems, ADMM also performs very well. However, when the problem size is rising, the computational burden from solving large linear systems becomes a serious issue for ADMM. The issue of scalability also influences the performance of PDCP since it needs the estimation of norm of matrix $\mathbf{A}$. Our method SP-BCD is not restricted heavily by a large problem size. SPDC (Zhang and Xiao 2015) even with optimal regularization parameter (by post-hoc selection) still dramatically deteriorates its performance.

Table 2: Lasso problem: performance of all compared methods.

Methods	m, n, d	Time (s)	Passes	Objective
ISTA	$\{1, 5, 0.5\} \times 10^3$	2.27	100	111.405
	$\{5, 20, 2\} \times 10^3$	45.67	100	448.351
FISTA	$\{1, 5, 0.5\} \times 10^3$	1.16	56	111.320
	$\{5, 20, 2\} \times 10^3$	19.00	49	448.271
ADMM	$\{1, 5, 0.5\} \times 10^3$	**0.69**	63	**111.318**
	$\{5, 20, 2\} \times 10^3$	19.83	51	**448.258**
PDCP	$\{1, 5, 0.5\} \times 10^3$	1.40	100	**111.318**
	$\{5, 20, 2\} \times 10^3$	26.80	100	448.263
SPDC	$\{1, 5, 0.5\} \times 10^3$	3.76	100	117.518
	$\{5, 20, 2\} \times 10^3$	70.10	100	473.806
SP-BCD	$\{1, 5, 0.5\} \times 10^3$	**0.70**	**30**	**111.318**
	$\{5, 20, 2\} \times 10^3$	**13.32**	**30**	**448.263**

Feature Selection with Group Lasso

We consider solving the following group Lasso problem (Yuan and Lin 2006):

$$\min_{\mathbf{x}} \lambda \sum_{g=1}^{G} \sqrt{d_g}\|\mathbf{x}_g\|_2 + \frac{1}{N}\sum_{i=1}^{N} g_i(\mathbf{a}_i^T\mathbf{x}, z_i), \tag{21}$$

where $\mathbf{x}$ is partitioned according to feature grouping, i.e., $\mathbf{x} = [\mathbf{x}_1^T, \mathbf{x}_2^T, \ldots, \mathbf{x}_G^T]^T$, each $\mathbf{a}_i$ is d-dimensional feature vector, $z_i \in \{-1, 1\}$ is the label, and $g_i(\mathbf{a}_i^T\mathbf{x}, z_i)$ is a convex loss function, such as the squared loss, logit loss, or hinge loss. The regularizer is the sum of groupwise L_2-norm $\|\mathbf{x}_g\|_2$, and the trade-off constant λ is to balance between the loss and the regularization term. The value d_g accounts for the varying group sizes. We use hinge loss function $g_i(\mathbf{a}_i^T\mathbf{x}, z_i) = \max(0, 1 - z_i\mathbf{a}_i^T\mathbf{x})$ for demonstration.

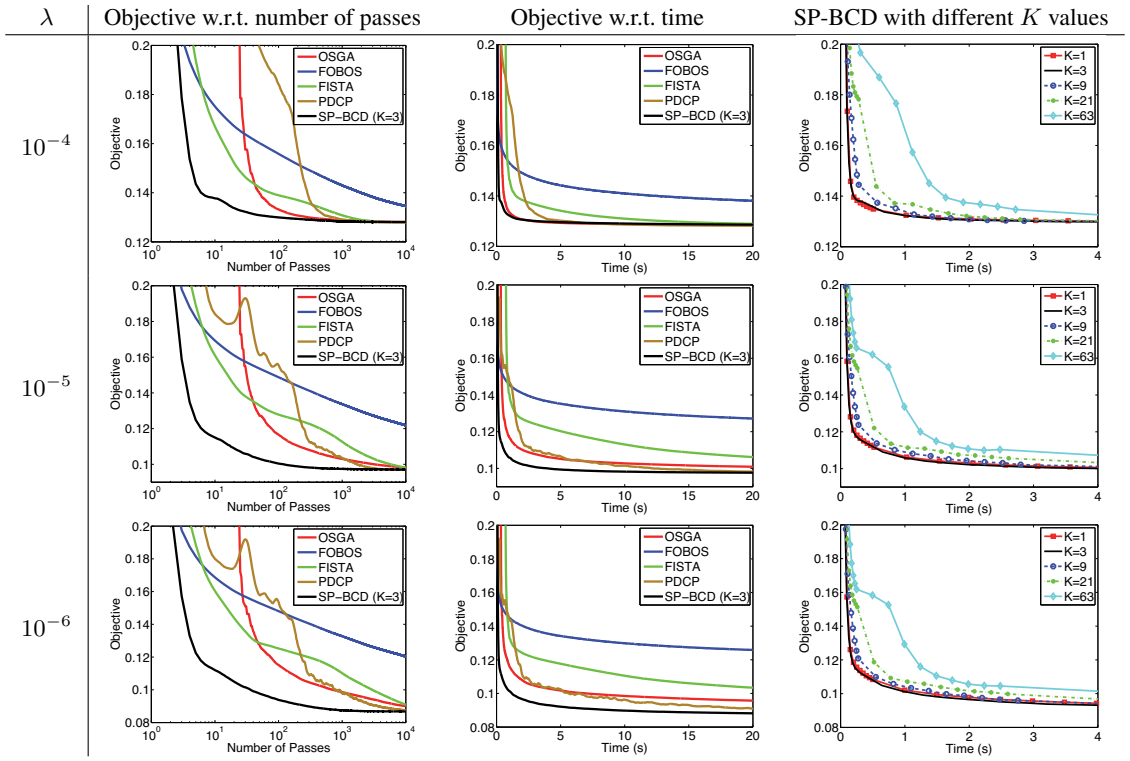

Figure 2: Group Lasso on MEMset dataset with different regularization parameter λ.

By the conjugate dual transformation of hinge loss,

$$g_i(\mathbf{a}_i^T\mathbf{x}, z_i) = \max_{y_i \in [0,1]} \langle -y_i z_i \mathbf{a}_i, \mathbf{x} \rangle + y_i, \qquad (22)$$

we can transform the group Lasso problem into the following saddle point problem,

$$\min_{\mathbf{x}} \max_{\mathbf{y} \in [0,1]^N} \lambda \sum_{g=1}^{G} \sqrt{d_g} \|\mathbf{x}_g\|_2 + \frac{1}{N} \langle -\sum_{i=1}^{N} y_i z_i \mathbf{a}_i, \mathbf{x} \rangle + \frac{1}{N} \sum_{i=1}^{N} y_i \qquad (23)$$

This reformulation of group Lasso makes both the dual and primal update extremely simple and efficient, both of which have closed-formed solution and can be easily derived.

To evaluate the performance of our method for the group Lasso problem, we apply it to a real-world dataset for splice site detection, which plays an important role in gene finding. The MEMset Donor dataset is widely used to demonstrate the advantages of the group Lasso models (Meier, Van De Geer, and Bühlmann 2008; Roth and Fischer 2008). From the original training set, we construct a balanced training set with $8,415$ true and $8,415$ false donor sites. Group lasso on this data with up to 2nd order interactions and up to 4 order interactions has been analyzed by (Meier, Van De Geer, and Bühlmann 2008; Roth and Fischer 2008), respectively. As shown in (Roth and Fischer 2008), there is not much improvement using higher order interactions. Therefore we only consider all three-way and lower order interactions. This forms $G = 63$ groups or $d = 2604$-dimensional feature space with $\{7, 21, 35\}$ groups of $\{4, 16, 64\}$-dimensional coordinate block, respectively.

We compare our SP-BCD with several recent competitive optimization methods for the non-smooth regularized problem: (1) OSGA (Neumaier 2014), a fast subgradient algorithm with optimal complexity; (2) FOBOS (Duchi and Singer 2009) based on Forward-Backward splitting; (3) FISTA (Beck and Teboulle 2009), using a smoothing technique to make it applicable with smoothing parameter $\epsilon = 5 \times 10^{-4}$; (4) PDCP (Chambolle and Pock 2011).

In this application, we evaluate the performance of these methods under different regularization parameter $\lambda = \{10^{-4}, 10^{-5}, 10^{-6}\}$. The first two columns in Figure 2 compares our method SP-BCD (with $K = 3$) with other methods in terms of the evolution of the objective function in Eq.(21) both w.r.t. the number of passes and w.r.t time. In all these test cases, SP-BCD demonstrates its superiority on both number of passes and consumed time. When the regularization is strong with large $\lambda = 10^{-4}$, all the methods tend to converge fast, but SP-BCD is the fastest one. PDCP performs poorly in first hundreds or thousands of passes, since it only applies the constant stepsize $1/\|A\|$. Compared with PDCP, our method considers the structure of matrix $\mathbf{A}$ and scales each dimension of primal and dual updates, which can achieve better empirical performance.

In order to investigate the effect of the number of chosen blocks for our method, we implement it using different K

values, $K = \{1, 3, 9, 21, 63\}$. The results are shown in the third column of Figure 2. In all the tested cases, a smaller number of blocks yields faster convergence, which shows the advantage of the flexible stochastic update of our method compared with (Pock and Chambolle 2011).

5 Conclusion and Future Work

The proposed SP-BCD for Sep-CCSP with non-strongly convex functions shares the efficiency and flexibility of block coordinate descent methods while keeping the simplicity of primal-dual methods and utilizing the structure of matrix $\mathbf{A}$. Many machine learning models are covered and we compare SP-BCD with other competitive methods in various applications. An immediate future direction is to investigate other valid parameter configurations.

References

Beck, A., and Teboulle, M. 2009. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences* 2(1):183–202.

Boyd, S.; Parikh, N.; Chu, E.; Peleato, B.; and Eckstein, J. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning* 3(1):1–122.

Candès, E. J.; Li, X.; Ma, Y.; and Wright, J. 2011. Robust principal component analysis? *Journal of the ACM (JACM)* 58(3):11.

Chambolle, A., and Pock, T. 2011. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision* 40(1):120–145.

Chambolle, A., and Pock, T. 2014. On the ergodic convergence rates of a first-order primal-dual algorithm. *Optimization-online preprint*.

Chen, S.; Donoho, D.; and Saunders, M. A. 2001. Atomic decomposition by basis pursuit. *SIAM review* 43(1):129–159.

Defazio, A.; Bach, F.; and Lacoste-Julien, S. 2014. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, 1646–1654.

Duchi, J., and Singer, Y. 2009. Efficient online and batch learning using forward backward splitting. *The Journal of Machine Learning Research* 10:2899–2934.

Hastie, T.; Tibshirani, R.; and Friedman, J. 2009. *The elements of statistical learning*, volume 2. Springer.

Hong, M., and Luo, Z. 2012. On the linear convergence of the alternating direction method of multipliers. *arXiv preprint arXiv:1208.3922*.

Meier, L.; Van De Geer, S.; and Bühlmann, P. 2008. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 70(1):53–71.

Nesterov, Y. 2012. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization* 22(2):341–362.

Neumaier, A. 2014. Osga: A fast subgradient algorithm with optimal complexity. *arXiv preprint arXiv:1402.1125*.

Parikh, N., and Boyd, S. 2013. Proximal algorithms. *Foundations and Trends in Optimization* 1(3):123–231.

Parikh, N., and Boyd, S. 2014. Block splitting for distributed optimization. *Mathematical Programming Computation* 6(1):77–102.

Pock, T., and Chambolle, A. 2011. Diagonal preconditioning for first order primal-dual algorithms in convex optimization. In *2011 IEEE International Conference on Computer Vision (ICCV)*, 1762–1769. IEEE.

Richtárik, P., and Takáč, M. 2014. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming* 144(1-2):1–38.

Richtárik, P., and Takáč, M. 2015. Parallel coordinate descent methods for big data optimization. *Mathematical Programming* 1–52.

Roth, V., and Fischer, B. 2008. The group-lasso for generalized linear models: uniqueness of solutions and efficient algorithms. In *Proceedings of the 25th international conference on Machine learning*, 848–855. ACM.

Wainwright, M. J. 2014. Structured regularizers for high-dimensional problems: Statistical and computational issues. *Annual Review of Statistics and Its Application* 1:233–253.

Wang, H.; Banerjee, A.; and Luo, Z. 2014. Parallel direction method of multipliers. In *Advances in Neural Information Processing Systems 27*, 181–189.

Wright, J.; Ganesh, A.; Rao, S.; Peng, Y.; and Ma, Y. 2009. Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In *Advances in neural information processing systems*, 2080–2088.

Yuan, M., and Lin, Y. 2006. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68(1):49–67.

Zhang, Y., and Xiao, L. 2015. Stochastic primal-dual coordinate method for regularized empirical risk minimization. In *International Conference on Machine Learning (ICML)*.

Zhu, Z., and Storkey, A. J. 2015a. Adaptive stochastic primal-dual coordinate descent for separable saddle point problems. In *Machine Learning and Knowledge Discovery in Databases*. Springer. 645–658.

Zhu, Z., and Storkey, A. J. 2015b. Stochastic parallel block coordinate descent for large-scale saddle point problems. *arXiv preprint arXiv:1511.07294*.