Accelerating block coordinate descent methods with identification strategies*

Ronaldo Lopes[1], Sandra A. Santos[1], and Paulo J. S. Silva[1]

[1] *Institute of Mathematics, University of Campinas, Rua Sergio Buarque de Holanda, 651, 13083-859, Campinas, São Paulo, Brazil*
Emails: ronaldolps3@gmail.com, sandra@ime.unicamp.br, pjssilva@ime.unicamp.br

June 20, 2018

Abstract

This work is about active set identification strategies aimed at accelerating block-coordinate descent methods (BCDM) applied to large-scale problems. We start by devising an identification function tailored for bound-constrained composite minimization together with an associated version of the BCDM, called Active BCDM, that is also globally convergent. The identification function gives rise to an efficient practical strategy for Lasso and ℓ_1-regularized logistic regression. The computational performance of Active BCDM is contextualized using comparative sets of experiments that are based on the solution of problems with data from deterministic instances from the literature. These results have been compared with those of well-established and state-of-the-art methods that are particularly suited for the classes of applications under consideration. Active BCDM has proved useful in achieving fast results due to its identification strategy. Besides that, an extra second-order step was used, with favorable cost-benefit.

Key words: Block coordinate descent, active-set identification, large-scale optimization and ℓ_1 regularization.

AMS Classification: 60K05, 49M37, 90C30, 90C06, 90C25

*Partially supported by FAPESP grants 2014/14228-6, 2013/05475-7, and 2013/07375-0 and CNPq grants 306986/2016-7, and 302915/2016-8.

1 Introduction

Coordinate descent methods are among the simplest ones in optimization. They are based on a well-established idea employed by iterative methods for linear systems, like the ones of Jacobi and Gauss-Seidel. At each iteration, one component of the independent variable is updated, in a cyclic way. They also fit into the framework of first-order methods, having low cost per iteration and low memory requirements. Besides that, they are easy to implement. Moreover, coordinate descent methods turn out to be an interesting option for very large scale problems for which the low cost per iteration pays out, even considering the large number of iterations needed to achieve a reasonable precision. This is especially the case for accelerated versions, like the so-called block coordinate descent methods (BCDM), in which the variables are grouped in convenient blocks.

The renewed interest in these methods is justified by the crescent demand on very large-scale optimization problems that come from different areas such as machine learning [24], compressed sensing [14], group Lasso [40], matrix completion [7] and truss topology design [29], to name a few. The problem structure is very much favorable for BCDM. Recent research concerning BCDM involves convergence and complexity analysis [2, 28], inexact solution of the subproblems [34], parallelization [31] and other acceleration techniques (see e.g. [16, 25, 38]).

Our contribution is aligned with the acceleration of BCDM, particularly for large-scale bound-constrained problems. In such a setting, there are important cases for which the dimension of the active optimal face is expected to be significantly smaller than the problem dimension. Consequently, it is worth pursuing the identification of this face, what is precisely our aim. Moreover, even for huge-scale unconstrained problems, such as Lasso [35] and ℓ_1-regularized logistic regression [26], the search for a sparse solution provides a room for identifying the zero components by exploring reformulations based on the introduction of simple bounds. Additionally, in the context of ℓ_1-regularized problems, the identification tool may provide a room for improving hot starts for cross validation strategies that adjust the regularization parameter.

The outline of this text is the following. In Section 2 we introduce the problem of interest and the main ingredients to present the BCDM, which is stated in Section 3, together with some preliminary results. In Section 4 we present an identification function for the active constraints and show how to apply it to problems with ℓ_1 regularization. In Section 5 we introduce the algorithm Active BCDM, with the related convergence results. Our analysis allows for different probabilities to be used at each iteration in the selection of the blocks, what is a novelty, to the best of our knowledge. Section 6 is dedicated to the numerical experiments, with comparative results that put our approach into perspective. Finally, some conclusions are made in Section 7.

2 Background

The problem we are interested in is

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) + \psi(x) \\ \text{s.t.} \quad & l \leq x \leq u, \end{aligned} \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a smooth function and $\psi : \mathbb{R}^n \to \mathbb{R}$ is a convex function, possibly nonsmooth, with a block-separable structure given by

$$\psi(x) = \sum_{i=1}^{m} \psi_i(x_{(i)}), \tag{2}$$

where $x_{(i)} \in \mathbb{R}^{p_i}$ is a subset of coordinates of the vector $x \in \mathbb{R}^n$ with $\sum_{i=1}^{m} p_i = n$. The vectors l, u have n coordinates in $[-\infty, +\infty]$, with $l \leq u$ and infinite values used to denote the absence of a bound.

Starting from the block decomposition of the vector x in m subsets of coordinates satisfying (2), we define a set of matrices $U_i \in \mathbb{R}^{n \times p_i}$ such that

$$x = \sum_{i=1}^{m} U_i x_{(i)}.$$

We also choose a set of m symmetric positive definite matrices $B_i \in \mathbb{R}^{p_i \times p_i}$, denoted by $B_i \in \mathbb{S}^{p_i}_{++}$, $i = 1, \ldots, m$. We then define a set of norms with respect to $\mathbb{R}^{p_i}$, and their respective dual norms, by

$$\|x_{(i)}\|_{(i)} := \sqrt{x_{(i)}^T B_i x_{(i)}} \text{ and } \|x_{(i)}\|^*_{(i)} := \sqrt{x_{(i)}^T B_i^{-1} x_{(i)}}, \ \forall i \in \{1, \ldots, m\}.$$

The matrices B_i, besides providing second-order information to the models, may carry scaling information from the problem variables, especially if they are diagonal.

We assume that the gradient of the smooth function f is Lipschitz continuous by blocks. That is, for each $i = 1, \ldots, m$, there is a constant $L_i > 0$ such that

$$\|\nabla_i f(x + U_i h_i) - \nabla_i f(x)\|^*_{(i)} \leq L_i \|h_i\|_{(i)}, \ h_i \in \mathbb{R}^{p_i}, \ i = 1, \ldots, m, \tag{3}$$

for all x such that $l \leq x \leq u$ and with $\nabla_i f(y) = U_i^T \nabla f(y)$.

Let $B \in \mathbb{R}^{n \times n}$ be the block diagonal positive definite matrix defined as

$$B = \text{diag}(L_1 B_1, \cdots, L_m B_m),$$

in which, for $i = 1, \ldots, m$, the scalars L_i are as in (3), and the matrices B_i are those chosen above. We also define the following norm in $\mathbb{R}^n$, and its dual, respectively by

$$\|z\|_B = \sqrt{z^T B z} \quad \text{and} \quad \|z\|^*_B = \sqrt{z^T B^{-1} z}.$$

For the vectors x, y, l, u, which may be indexed by coordinates or by blocks, we have adopted the convention that the subindex (i) refers to the i-th block, whereas i refers to the i-th component. The remaining vectors, matrices, functions and scalars used along the text, namely, h_i, w_i, p_i, s_i, B_i, U_i, ψ_i, $\nabla_i f$, L_i, are only indexed by blocks. Therefore, we use simply i to refer to the i-th block for not overloading the notation.

Throughout the text, $\|\cdot\|$ denotes either the Euclidean vector norm or the induced matrix norm. We use $|\mathcal{I}|$ to refer to the cardinality of the set $\mathcal{I}$. We also denote the feasible set of problem (1) by

$$\mathcal{X} := \{x \in \mathbb{R}^n \mid l \leq x \leq u\}, \tag{4}$$

and its objective function by $F(x) = f(x) + \psi(x)$.

Finally, let us recall the classic definition of stationarity for a general optimization problem.

Definition 1. *Given a subset $\mathcal{C}$ of $\mathbb{R}^n$ and a vector x^* in $\mathcal{C}$, a vector $y \in \mathbb{R}^n$ is said to be a tangent of $\mathcal{C}$ at x^* if either $y = 0$ or there exists a sequence $\{x^k\} \subset \mathcal{C}$ such that $x^k \neq x^*$ for all k and*

$$x^k \to x^*, \qquad \frac{x^k - x^*}{\|x^k - x^*\|} \to \frac{y}{\|y\|}.$$

The set of all tangents of $\mathcal{C}$ at the point x^* is called the tangent cone of $\mathcal{C}$ at x^*, and is denoted by $T_{\mathcal{C}}(x^*)$.

Definition 2. *Consider the problem of minimizing a function $\varphi : \mathbb{R}^n \to \mathbb{R}$ on a set $\mathcal{C} \subset \mathbb{R}^n$. A point $x^* \in \mathcal{C}$ is said to be* stationary *for this problem if the one-sided directional derivative*

$$\varphi'(x^*; d) = \lim_{t \downarrow 0} \frac{\varphi(x^* + td) - \varphi(x^*)}{t} \tag{5}$$

exists and is nonnegative for all $d \in T_{\mathcal{C}}(x^*)$. This condition is also sufficient if $\mathcal{C}$ is a closed and convex set and φ is a convex function.

We can use the optimality condition above and apply it to our main problem (1) to see that the following result holds.

Lemma 1. *Let $x^* \in \mathcal{X}$ be a minimizer of* (1)*, then for all $x \in \mathcal{X}$*

$$\nabla f(x^*)^T (x - x^*) + \psi(x) - \psi(x^*) \geq 0.$$

Proof. Just apply the optimality condition of Definition 2 to the equivalent problem

$$\begin{aligned} \min_{x,z} \quad & f(x) + z \\ \text{s.t} \quad & \psi(x) \leq z \\ & x \in \mathcal{X}. \end{aligned}$$

□

3 Block coordinate descent methods

The aforementioned elements provide all the necessary ingredients to state the framework of the Block Coordinate Descent Method (BCDM), described in Algorithm 1.

Given a point $x \in \mathbb{R}^n$, we define the vector $h_i(x)$ as the minimizer of the subproblem

$$\begin{aligned} \min_{h \in \mathbb{R}^{p_i}} \quad & \nabla_i f(x)^T h + \frac{L_i}{2} h^T B_i h + \psi_i(x_{(i)} + h) - \psi_i(x_{(i)}), \\ \text{s.t.} \quad & l \leq x + U_i h \leq u. \end{aligned} \tag{6}$$

Algorithm 1 (General) Block Coordinate Descent Method (BCDM)

1: Choose an initial point $l \leq x^0 \leq u$ and the parameters $\epsilon \in \mathbb{R}_+$ and $\delta_{\max} \in \mathbb{N}$. Set a vector $v \in \mathbb{R}^n$ with 2ϵ in all positions. Set $k = 1$.
2: **repeat**
3: Choose a block $i \in \{1, \ldots, m\}$ by some strategy.
4: Find $\widehat{h}_i \equiv h_i(x^k)$, a solution to the subproblem (6)
5: Set $x^{k+1} = x^k + U_i \widehat{h}_i$ and $v_{(i)} = \widehat{h}_i$. Update $k = k + 1$
6: **until** $\|v\| \leq \epsilon$ or $k = \delta_{\max}$

Next, we establish an auxiliary result. It is based on [36, Lemma 3], where unconstrained problems are analyzed. We have adapted this lemma to take into account the bound constraints present in problem (1). The proof is included not only for completeness, but also because part of its reasoning is employed subsequently in the text.

Lemma 2. *A point x^* that is feasible for* (1) *is stationary if, and only if, $h_i(x^*) = 0$ for all $i = 1, \ldots, m$.*

Proof. First, for each $i = 1, \ldots m$, let h_i^* denote $h_i(x^*)$ in this proof.

We start assuming that $h_i^* = 0$, for $i = 1, \ldots, m$. Let $s \in \mathbb{R}^n$ be such that $l \leq x^* + s \leq u$ and let $s_i \in \mathbb{R}^{p_i}$ be such that $s = \sum_{i=1}^m U_i s_i$. For each $i = 1, \ldots, m$, we can use the optimality of h_i^*, for the problem given in (6), to conclude that

$$\begin{aligned} 0 &= \nabla_i f(x^*)^T h_i^* + \frac{L_i}{2} \|h_i^*\|_{(i)}^2 + \psi_i(x_{(i)}^* + h_i^*) - \psi_i(x_{(i)}^*) \\ &\leq \nabla_i f(x^*)^T s_i + \frac{L_i}{2} \|s_i\|_{(i)}^2 + \psi_i(x_{(i)}^* + s_i) - \psi_i(x_{(i)}^*). \end{aligned}$$

Summing these inequalities for $i = 1, \ldots, m$ and rearranging terms, we get

$$\sum_{i=1}^m \psi_i(x_{(i)}^* + s_i) - \sum_{i=1}^m \psi_i(x_{(i)}^*) \geq -\sum_{i=1}^m \nabla_i f(x^*)^T s_i - \sum_{i=1}^m \frac{L_i}{2} \|s_i\|_{(i)}^2,$$

which implies

$$\psi(x^* + s) - \psi(x^*) \geq -\nabla f(x^*)^T s - \frac{1}{2}\|s\|_B^2. \tag{7}$$

Now, let $d \in \mathbb{R}^n$ be a direction for which there exists $t > 0$ such that $x + td$ is feasible for (1). We have

$$\begin{aligned} F'(x^*; d) &= \lim_{t \downarrow 0} \frac{F(x^* + td) - F(x^*)}{t} \\ &= \lim_{t \downarrow 0} \frac{f(x^* + td) + \psi(x^* + td) - f(x^*) - \psi(x^*)}{t} \\ &\geq \lim_{t \downarrow 0} \frac{f(x^* + td) - f(x^*) - t\nabla f(x^*)^T d - ((t^2)/2)\|d\|_B^2}{t} \quad \text{[Using (7)]} \\ &= 0. \end{aligned}$$

We conclude that x^* is stationary for (1).

Now, let us prove the remaining implication. We then assume that x^* is stationary for (1) and consider, for the sake of a contradiction, that there is a block $i \in \{1, \ldots, m\}$ for which $h_i^* \neq 0$. The definition of h_i^* implies that

$$\nabla_i f(x^*)^T h_i^* + \psi_i(x_{(i)}^* + h_i^*) - \psi_i(x_{(i)}^*) \leq -\frac{L_i}{2}\|h_i^*\|_{(i)}^2 < 0. \tag{8}$$

On the other hand, for $d = U_i h_i^*$ and $t \in (0, 1)$,

$$\begin{aligned} F(x^* + td) &= f(x^* + td) + \psi(x^* + td) \\ &\leq f(x^*) + t\nabla_i f(x^*)^T h_i^* + \frac{t^2 L_i}{2}\|h_i^*\|_{(i)}^2 \\ &\quad + \sum_{j=1, j\neq i}^{m} \psi_j(x_{(j)}^*) + t\psi_i(x_{(i)}^* + h_i^*) + (1 - t)\psi_i(x_{(i)}^*), \end{aligned}$$

where the last inequality uses the Lipschitz continuity by blocks of f and the convexity of ψ_i. Hence,

$$F(x^* + td) - F(x^*) \leq t\nabla_i f(x^*)^T h_i^* + \frac{t^2 L_i}{2}\|h_i^*\|_{(i)}^2 + t\psi_i(x_{(i)}^* + h_i^*) - t\psi_i(x_{(i)}^*). \tag{9}$$

Dividing by t, taking the limit for $t \downarrow 0$ and using (8), it follows that

$$F'(x^*; d) \leq -\frac{L_i}{2}\|h_i^*\|_{(i)}^2 < 0.$$

We see that $d = U_i h_i^*$ is a descent direction from x^*, contradicting the assumption that x^* is stationary for (1). □

Another related result is set below, concerning the sufficient decrease achieved by the minimizer of subproblem (6).

Lemma 3. *Let x be a feasible point for (1) such that, for some $i \in \{1, \ldots, m\}$, $h_i(x) \neq 0$. Then,*

$$F(x + U_i h_i(x)) \leq F(x) - \frac{L_i}{2}\|h_i(x)\|^2_{(i)}.$$

Proof. Let us again simplify the notation and use h_i to denote $h_i(x)$. We start from the definition of h_i to see that, for all $t \in (0, 1)$,

$$\nabla_i f(x)^T h_i + \frac{L_i}{2}\|h_i\|^2_{(i)} + \psi_i(x_{(i)} + h_i) \leq t\nabla_i f(x)^T h_i + \frac{t^2 L_i}{2}\|h_i\|^2_{(i)} + \psi_i(x_{(i)} + th_i).$$

Now, from the convexity of ψ_i we have

$$\psi_i(x_{(i)} + th_i) \leq t\psi_i(x_{(i)} + h_i) + (1 - t)\psi_i(x_{(i)}).$$

Combining the last two inequalities, grouping similar terms together and dividing by $1 - t$, it follows that

$$\nabla_i f(x)^T h_i + (1 + t)\frac{L_i}{2}\|h_i\|^2_{(i)} + \psi_i(x_{(i)} + h_i) - \psi_i(x_{(i)}) \leq 0.$$

Taking limit for $t \uparrow 1$ we get

$$\nabla_i f(x)^T h_i + \frac{L_i}{2}\|h_i\|^2_{(i)} + \ \psi_i(x_{(i)} + h_i) - \psi_i(x_{(i)}) \leq -\frac{L_i}{2}\|h_i\|^2_{(i)}. \tag{10}$$

On the other hand, following the same reasoning in the deduction of inequality (9) present in the proof of the previous lemma, we can conclude that for all $t \in (0, 1)$

$$F(x + tU_i h_i) - F(x) \leq t\nabla_i f(x)^T h_i + \frac{t^2 L_i}{2}\|h_i\|^2_{(i)} + t\psi_i(x_{(i)} + h_i) - t\psi_i(x_{(i)}).$$

Taking again the limit for $t \uparrow 1$, this says that

$$F(x + U_i h_i) - F(x) \leq \nabla_i f(x)^T h_i + \frac{L_i}{2}\|h_i\|^2_{(i)} + \psi_i(x_{(i)} + h_i) - \psi_i(x_{(i)}),$$

which together with (10) gives the desired result. □

4 Identification functions

Our objective in this section is to define an identification function to unveil the active constraints at stationary points of (1). The concept of identification function for the set of KKT pairs of a general smooth nonlinear programming problem was introduced in [15]. Using of the set

$$\mathcal{S} = \{x \in \mathbb{R}^n \mid x \text{ is a stationary point of the problem (1)}\},$$

we slightly adapt the definition from [15] below, so that only the primal part of the stationary points are taken into consideration.

Definition 3. *We say that $\rho(x; x^*) : \mathbb{R}^n \to \mathbb{R}_-$ is an identification function for a point $x^* \in \mathcal{S} \subset \mathbb{R}^n$, with respect to the set $\mathcal{S}$, if*

(i) *ρ is continuous in an open set containing x^*;*

(ii) *$\rho(x; x^*) = 0 \Rightarrow x \in \mathcal{S}$;*

(iii) *$\rho(x^*; x^*) = 0$;*

(iv) *$\lim\limits_{x^k \to x^*} \dfrac{-\rho(x^k; x^*)}{\|x^k - x^*\|} = +\infty$, for all $x^k \to x^*$, with $x^* \in \mathcal{S}$ and $x^k \notin \mathcal{S}$, $\forall k$.*

Identification functions are used in the context of optimization to uncover the set of active constraints at the limit of sequences generated by algorithms. In order to see this, let us consider a feasible set defined by inequality constraints

$$\mathcal{F} = \{x \in \mathbb{R}^n \mid g(x) \leq 0\},$$

where $g : \mathbb{R}^n \to \mathbb{R}^p$ is assumed to be of class C^1. In particular, this setting takes into account the feasible set $\mathcal{X}$ of problem (1), which may be described by the function $g : \mathbb{R}^n \to \mathbb{R}^{2n}$ given by

$$g_i(x) := \begin{cases} l_i - x_i, & \text{if } 1 \leq i \leq n; \\ x_i - u_i, & \text{if } n+1 \leq i \leq 2n. \end{cases}$$

Let $\mathcal{S} \subset \mathcal{F}$, that will play later the role of the set of possible limit points generated by an algorithm, and let $x^* \in \mathcal{S}$ be a feasible point for which we have an identification function ρ with respect to $\mathcal{S}$.

The next result states that it is possible to determine which constraints are active at x^*, based only on the information given by the identification function in a neighborhood of x^*. To simplify the presentation, we define first two index sets with respect to a point $x \in \mathbb{R}^n$.

Definition 4. *Given $x \in \mathbb{R}^n$, let $\mathcal{A}(x)$ and $\widehat{\mathcal{A}}(x)$ be the following sets*

$$\mathcal{A}(x) = \{i \in \{1, \ldots, m\} \mid g_i(x) = 0\},$$
$$\widehat{\mathcal{A}}(x) = \{i \in \{1, \ldots, m\} \mid g_i(x) \geq \rho(x; x^*)\}.$$

Proposition 1. *Let ρ be an identification function for $x^* \in \mathcal{S}$ with respect to $\mathcal{S}$. Then there exists an $\varepsilon > 0$ such that*

$$\mathcal{A}(x^*) \equiv \widehat{\mathcal{A}}(x), \forall x \in \mathcal{B}(x^*, \varepsilon), \text{with } x \notin \mathcal{S}.$$

Proof. As $g \in C^1$, it is also locally Lipschitz continuous. Thus, fixed an $\varepsilon_1 > 0$ there is a constant $c > 0$ such that

$$-g_i(x) \leq -g_i(x^*) + c\|x - x^*\|, \ i = 1, \ldots, p \tag{11}$$

for all $x \in \mathcal{B}(x^*, \varepsilon_1)$.

Let $i \in \mathcal{A}(x^*)$. By Definition 4, $g_i(x^*) = 0$, and by Definition 3 (iv), there exists $\varepsilon_2 > 0$ such that

$$c\|x - x^*\| \leq -\rho(x; x^*), \forall x \in \mathcal{B}(x^*, \varepsilon_2) \text{ and } x \notin \mathcal{S}. \tag{12}$$

Taking the intersection of the neighborhoods of (11) and (12), we have that

$$g_i(x) \geq \rho(x; x^*),$$

for all $x \notin \mathcal{S}$ and $x \in \mathcal{B}(x^*, \min\{\varepsilon_1, \varepsilon_2\})$. Therefore, $i \in \widehat{\mathcal{A}}(x)$.

Now, we prove that $\widehat{\mathcal{A}}(x) \subset \mathcal{A}(x^*)$ for x close to x^*. Let $i \notin \mathcal{A}(x^*)$. Then $g_i(x^*) < 0$. By Definition 3, items (i) and (iii), we know that $\rho(\,\cdot\,; x^*)$ is continuous in an open set that contains x^*, and $\rho(x^*; x^*) = 0$. These facts, together with the continuity of the function g, assure the existence of an $\varepsilon_3 > 0$ for which, for all $x \in \mathcal{B}(x^*, \varepsilon_3)$, $g_i(x) < \rho(x; x^*)$. Hence $i \notin \widehat{\mathcal{A}}(x)$.

The result follows by taking $\varepsilon = \min\{\varepsilon_1, \varepsilon_2, \varepsilon_3\}$. □

We point out that identification functions always exist. Actually, it is straightforward to prove that, for a fixed $\alpha \in (0, 1)$, the function $\rho_\alpha^{\text{can}} : \mathbb{R}^n \to \mathbb{R}_-$, given by

$$\rho_\alpha^{\text{can}}(x; x^*) := -\|x - x^*\|^\alpha \tag{13}$$

is an identification function for a point $x^* \in \mathcal{S}$. This function, however, uses x^* in its formula, and typically the point of interest, x^*, is not readily available. Hence this ideal function is not computable. Below we will define an identification function for stationary points x^* of (1), that we are trying to approximate, using an optimization algorithm.

In order to introduce such computable identification function, given a feasible $x \in \mathcal{X}$, we will define as $h(x)$, the minimizer of

$$\begin{aligned} \min_{h \in \mathbb{R}^n} \quad & \sum_{i=1}^{m} \left(\nabla_i f(x)^T h_i + \frac{L_i}{2} \|h_i\|_{(i)}^2 + \psi_i(x_{(i)} + h_i) \right) \\ \text{s.t.} \quad & l_{(i)} \leq x_{(i)} + h_i \leq u_{(i)}, \quad i \in \{1, \ldots, m\}. \end{aligned} \tag{14}$$

Observe that this problem is closely related to (6). Actually, using the separability by blocks of (14) it is clear that the i-th block of $h(x)$ is exactly the solution to (6), i.e. $h_i(x)$.

With this notation, we assert that, for a fixed $\alpha \in (0, 1)$, the function $\rho_\alpha : \mathbb{R}^n \to \mathbb{R}_-$ defined as

$$\rho_\alpha(x) := -\|h(x)\|^\alpha \tag{15}$$

is an identification function for *any* stationary point x^* with respect to the set of all stationary points of (1), that we will denote from now on simply as $\mathcal{S}$. Since ρ_α does not depend on the specific $x^* \in \mathcal{S}$, we dropped the second argument from its definition. The role of the exponent α is to ensure that (15) satisfies the properties of Definition 3.

The following result provides useful bounds for the analysis, and it is similar to Lemma 2.1 of [37].

Proposition 2. *Let x^* be a stationary point for the problem* (1)*, and $L_i \in \mathbb{R}_{++}$ and $B_i \in \mathbb{S}^{p_i}_{++}$ be given. Assume that the function ∇f is locally Lipschitz continuous in a neighborhood of x^*, i.e., there exist $c > 0$ and $\varepsilon > 0$ such that*

$$\|\nabla f(x) - \nabla f(y)\| \leq c\|x - y\|, \forall x, y \in \mathcal{B}(x^*, \varepsilon), \tag{16}$$

and, for some $\sigma > 0$, the function ∇f satisfies

$$(\nabla f(y) - \nabla f(x^*))^T (y - x^*) \geq \sigma \|y - x^*\|^2, \ \forall y \in \mathcal{B}(x^*, \varepsilon). \tag{17}$$

Then, there exist $\zeta > 0$ and $\delta > 0$ such that

(i) $\|x + h(x) - y - h(y)\| \leq \zeta \|x - y\|, \forall x, y \in \mathcal{B}(x^*, \varepsilon)$;

(ii) $\|x - x^*\| \leq \delta \|h(x)\|, \forall x \in \mathcal{B}(x^*, \varepsilon)$.

Proof. Let us start by applying Lemma 1 to the problems that define $x^+ := x + h(x)$ and $y^+ := y + h(y)$. We get

$$\nabla f(x)^T (y^+ - x^+) + \sum_{i=1}^{m} L_i h_i(x)^T B_i (y^+_{(i)} - x^+_{(i)}) + \psi(y^+) - \psi(x^+) \geq 0,$$
$$\nabla f(y)^T (x^+ - y^+) + \sum_{i=1}^{m} L_i h_i(y)^T B_i (x^+_{(i)} - y^+_{(i)}) + \psi(x^+) - \psi(y^+) \geq 0.$$

Adding these two inequalities, it follows that

$$(\nabla f(x) - \nabla f(y))^T (y^+ - x^+) + \sum_{i=1}^{m} L_i (h_i(x) - h_i(y))^T B_i (y^+_{(i)} - x^+_{(i)}) \geq 0.$$

Recalling that $h_i(x) = x^+_{(i)} - x_{(i)}$ and $h_i(y) = y^+_{(i)} - y_{(i)}$, the inequality above is equivalent to

$$(\nabla f(x) - \nabla f(y))^T (y^+ - x^+) + \sum_{i=1}^{m} L_i (y_{(i)} - x_{(i)})^T B_i (y^+_{(i)} - x^+_{(i)}) \geq \sum_{i=1}^{m} L_i (y^+_{(i)} - x^+_{(i)})^T B_i (y^+_{(i)} - x^+_{(i)}).$$

We can now use the Cauchy-Schwarz inequality, and the definition of $\|\cdot\|_B$ to conclude that

$$\|\nabla f(x) - \nabla f(y)\| \|y^+ - x^+\| + \|x - y\| \|B(y^+ - x^+)\| \geq \sum_{i=1}^{m} L_i (y^+_{(i)} - x^+_{(i)})^T B_i (y^+_{(i)} - x^+_{(i)}) = \|y^+ - x^+\|_B^2. \tag{18}$$

From the consistency of the used norms, let c_1 be a positive constant such that

$$c_1 \|z\| \leq \|z\|_B, \quad \forall z \in \mathbb{R}^n.$$

Additionally, $\|Bz\| \leq \|B\|\|z\|$ holds for all $z \in \mathbb{R}^n$. Hence, the relationship (18) yields

$$\|\nabla f(x) - \nabla f(y)\|\|y^+ - x^+\| + \|x - y\|\|B\|\|y^+ - x^+\| \geq c_1^2\|y^+ - x^+\|^2.$$

Dividing it by $\|y^+ - x^+\|$, and applying the Lipschitz condition (16), we have

$$c\|x - y\| + \|B\|\|x - y\| \geq c_1^2\|x^+ - y^+\|,$$

which gives us (i), with $\zeta = (c + \|B\|)/c_1^2$.

Le us now prove (ii). We start again using Lemma 1 for the definition of x^* and x^+ to get

$$\nabla f(x^*)^T(x^+ - x^*) + \psi(x^+) - \psi(x^*) \geq 0,$$
$$\nabla f(x)^T(x^* - x^+) + \sum_{i=1}^{m} L_i h_i(x)^T B_i(x^*_{(i)} - x^+_{(i)}) + \psi(x^*) - \psi(x^+) \geq 0.$$

Adding these inequalities and recalling that $x^+ = x + h(x)$ we get

$$(\nabla f(x^*) - \nabla f(x))^T(x + h(x) - x^*) + \sum_{i=1}^{m} L_i h_i(x)^T B_i(x^*_{(i)} - x_{(i)} - h_i(x)) \geq 0.$$

Rearranging terms and using the fact that $B_i,\ i = 1, \ldots, m$, are positive definite

$$(\nabla f(x^*) - \nabla f(x))^T h(x) + \sum_{i=1}^{m} L_i h_i(x)^T B_i(x^*_{(i)} - x_{(i)}) \geq (\nabla f(x) - \nabla f(x^*))^T(x - x^*).$$

We can now use (17), the Cauchy-Schwarz inequality, and the Lipschitz continuity of ∇f to deduce from the latter inequality that

$$c\|x - x^*\|\|h(x)\| + \|h(x)\|\|B(x - x^*)\| \geq \sigma\|x - x^*\|^2.$$

Using the consistency of norms and dividing the inequality above by $\|x - x^*\|$ we obtain

$$\frac{(c + \|B\|)}{\sigma}\|h(x)\| \geq \|x - x^*\|,$$

which is (ii) with $\delta = (c + \|B\|)/\sigma$. The proof is complete. □

We are now ready to establish sufficient conditions under which ρ_α, defined in (15), is an identification function for $x^* \in \mathcal{S}$.

Proposition 3. *Assume that $x^* \in \mathcal{S}$ satisfies the hypotheses of Proposition 2. Then, ρ_α defined in* (15) *is an identification function for x^*, for any given $\alpha \in (0, 1)$.*

Proof. Lemma 2 ensures that items (ii) and (iii) of Definition 3 hold. From Proposition 2 there exists $\varepsilon > 0$ such that, for all $x, y \in \mathcal{B}(x^*, \varepsilon)$,

$$\begin{aligned} \big|\|h(x)\| - \|h(y)\|\big| &\leq \|h(x) - h(y)\| \\ &\leq \|x + h(x) - y - h(y)\| + \|x - y\| \\ &\leq \zeta\|x - y\| + \|x - y\| \\ &= (1 + \zeta)\|x - y\|. \end{aligned}$$

The previous inequality, together with the fact that, for any given $\alpha \in (0, 1)$, the function $\varphi(t) = -t^\alpha$ is continuous, imply that ρ_α is continuous in a neighborhood of x^*, ensuring the validity of item (i) of Definition 3.

Now, let $x^k \to x^*$ for which $x^k \notin \mathcal{S}$. Using item (i) of Proposition 2, we know that for k large enough

$$\frac{1}{\delta\|h(x^k)\|} \leq \frac{1}{\|x^k - x^*\|} \quad \Leftrightarrow \quad \frac{1}{\delta\|h(x^k)\|^{1-\alpha}} \leq \frac{\|h(x^k)\|^\alpha}{\|x^k - x^*\|}.$$

Taking the limit in the previous inequality, we obtain

$$+\infty = \lim_{x^k \to x^*} \frac{1}{\delta\|h(x^k)\|^{1-\alpha}} \leq \lim_{x^k \to x^*} \frac{-\rho_\alpha(x^k)}{\|x^k - x^*\|},$$

showing that item (iv) of Definition 3 is verified, concluding the proof. □

5 Active BCDM

Using the identification function defined in the last section, we can now present a simple variation of the BCDM that aims to incorporate the identification of active constraints. This is achieved by decreasing the probability of selecting variables that have already been considered active and fixed at a bound when selecting the coordinates to update. The main idea is that, after a given number of iterations, controlled by a frequency parameter δ_F, the identification function is computed, suggesting that some variables are already active and at their bounds. The probability of selecting these variables is thus decreased in the subsequent iterations, controlled by a preference factor δ_{DP}. The algorithm allows the selection of only a subset of the constraints suggested by the identification function. This flexibility will show beneficial in the numerical tests. This method will be called Active BCDM and it is fully described in Algorithm 2.

Next we proceed to prove that the (General) BCDM, Algorithm 1, globally converges with probability one to stationary points for problem (1) whenever the probability of choosing any block in a given iteration is bounded away from zero. In particular, the convergence result stated below allows for different probabilities of choosing a block, or subset of blocks, at each iteration, and encompasses the strategy adopted within the Active BCDM algorithm. Adaptive probabilities within coordinate descent methods were already considered e.g. in [10] and [19]. In the former, the probabilities of selecting the dual variables

Algorithm 2 BCDM with Identification of Active Variables (Active BCDM)

Choose an initial point $l \leq x^0 \leq u$, the parameters $\ell_{\max} \in \mathbb{N}$, $\varepsilon \in \mathbb{R}_+$ and $\alpha \in (0,1)$ and two natural numbers δ_F, δ_{DP}. Initialize a vector $v \in \mathbb{R}^n$ with 2ε in all positions, the sets $\mathcal{I} = \{1, \ldots, m\}$, $\mathcal{J} = \emptyset$, the initial cycle size $c_s = 10$, and the counter $\ell = 0$.
repeat
  for $k = \ell + 1, \ldots, \ell + c_s$ **do**
    Choose a block i that satisfies the probability distribution

$$P(i) = \begin{cases} \dfrac{\delta_{DP}}{\delta_{DP}|\mathcal{I}| + |\mathcal{J}|}, & \text{if } i \in \mathcal{I}, \\ \dfrac{1}{\delta_{DP}|\mathcal{I}| + |\mathcal{J}|}, & \text{if } i \in \mathcal{J}. \end{cases}$$

    Find $\widehat{h}_i \equiv h_i(x^k)$, a solution to the subproblem (6)
    Set $x^{k+1} = x^k + U_i \widehat{h}_i$ and $v_{(i)} = \widehat{h}_i$.
  end for
  Set $\ell = \ell + c_s$.
  Obtain the set $\mathcal{C}(x^\ell) \subset \{1, \ldots, m\}$ with the identification function $\rho_\alpha(x)$, where

$$\mathcal{C}(x^\ell) = \left\{ i \mid g_j(x^\ell) \geq \rho_\alpha(x^\ell), \forall x_j^\ell \text{ s.t. } j \text{ belongs to the } i\text{th-block} \right\}.$$

  Define $\mathcal{J} \subset \mathcal{C}(x^\ell)$ and $\mathcal{I} = \{1, \ldots, m\} - \mathcal{J}$. Set $c_s = \delta_F |\mathcal{I}|$.
until $\|v\| \leq \varepsilon$ or $\ell \geq \ell_{\max}$

are adaptively tuned along the iterative process, whereas the latter speeds up the method taking into account how much each coordinate contributes to the objective descent.

Assumption 1. *In this section, the function ψ is supposed to be continuous in the feasible set of problem* (1).

We will now prepare for the main convergence result, presenting two auxiliary lemmas.

Lemma 4. *Let x be a non-stationary point for* (1) *and $i \in \{1, 2, \ldots, m\}$ be a block index such that $h_i(x) \neq 0$. Then, there are $\delta, \gamma > 0$ such that for all feasible $y \in \mathcal{B}(x, \delta)$, $\|h_i(y)\|_{(i)} \geq \gamma \|h_i(x)\|_{(i)}$.*

Proof. Let us suppose by contradiction that the result is not true. Then, there must be a feasible sequence $y^k \to x$ and a sequence of scalars $\gamma_k \downarrow 0$ such that

$$\|h_i(y^k)\|_{(i)} \leq \gamma_k \|h_i(x)\|_{(i)}.$$

Therefore $h_i(y^k) \to 0$.

On the other hand, since the function that is minimized in the definition of $h_i(\cdot)$ is strictly convex, there exists $\delta > 0$, such that

$$\nabla_i f(x)^T h_i(x) + \frac{L_i}{2}\|h_i(x)\|^2_{(i)} + \psi_i(x_{(i)} + h_i(x)) \leq \psi(x_{(i)}) - 2\delta.$$

Now, as y^k is feasible, converges to x, and ψ is continuous in the feasible set, for k large enough it is true that

$$\nabla_i f(y^k)^T h_i(x) + \frac{L_i}{2}\|h_i(x)\|^2_{(i)} + \psi_i(y^k_{(i)} + h_i(x)) < \psi(y^k_{(i)}) - \delta.$$

Using the fact that $h_i(y^k) \to 0$, and once again the continuity of ψ, we also have that for k large enough

$$\psi(y^k_{(i)}) - \delta < \nabla_i f(y^k)^T h_i(y^k) + \frac{L_i}{2}\|h_i(y^k)\|^2_{(i)} + \psi_i(y^k_{(i)} + h_i(y^k)).$$

The last two inequalities combined contradict the optimality in the definition of $h_i(y^k)$. □

Lemma 5. *Let $\mathcal{K} \subset \mathbb{N}$ be an infinite set of indexes of iterations of the (General) BCDM and $i \in \{1, 2, \ldots, m\}$ a fixed block index that has probability at least $\epsilon > 0$ to be selected at every iteration in $\mathcal{K}$. The probability that i will be selected infinitely many times in $\mathcal{K}$ is 1.*

Proof. For k to be chosen a finite number of times, it must not be chosen after a last iteration $K \in \mathcal{K}$.

By the hypothesis, the probability that i is not chosen at a given iteration $k \in \mathcal{K}$ is at most $1 - \epsilon$. Therefore the probability that it is not chosen for all $\mathcal{K} \ni k > K$ is at most

$$\prod_{\mathcal{K} \ni k > K} (1 - \epsilon) = 0.$$

We have then proved that the probability that i is chosen only a finite number of times in $\mathcal{K}$ is at most zero, hence the probability that is chosen infinitely many times is one. □

We are ready to prove the main convergence theorem for the (General) BCDM that select the blocks with a probability bounded away from zero.

Theorem 1. *Let $\{x^k\}$ be a sequence generated by the (General) BCDM for which every block index $i \in \{1, 2, \ldots, m\}$ has probability at least ϵ to be chosen. Let x^* be one of its accumulation points, and suppose that the function F is bounded below in the set $\mathcal{X} = \{x \mid l \leq x \leq u\}$. The probability that x^* is stationary for problem* (1) *is 1.*

Proof. Let $\mathcal{K} \subset \mathbb{N}$ be the set of indexes associated with a subsequence of $\{x^k\}$ that converges to x^*.

If x^* is not stationary we can, without loss of generality, assume that for all $k \in \mathcal{K}$, $x^k \in \mathcal{B}(x^*, \delta)$, the neighborhood given by Lemma 4. Moreover, at

every iteration in which the block index i defined in the aforementioned lemma is considered by the Active BCDM we can see from the Lemma 3 that the objective function F decreases at least the constant value $\gamma^2 L_i/2\|h_i(x^*)\|^2_{(i)} > 0$.

As the (General) BCDM never allows F to increase, by Steps 4 and 5, it follows from the assumption that F is bounded below that the number of times this block index is considered has to be finite. But Lemma 5 says that this can only happen with probability 0. Therefore, x^* can only be non-stationary with probability 0 and has to be stationary with probability 1. □

In turn, the Active BCDM admits a strictly positive probability $\epsilon = 1/(m\delta_{DP})$ of choosing every index block $i \in \{1, 2, \ldots, m\}$, hence the next result follows directly from Theorem 1 and Proposition 1.

Corollary 1. *Let $\{x^k\}$ be a sequence generated by the Active BCDM, x^* be one of its accumulation points, and suppose that the function F is bounded below in the set $\mathcal{X} = \{x \mid l \leq x \leq u\}$. The probability that x^* is stationary for problem* (1) *is 1. Moreover, for k large enough, the index set defined at Step 9 verifies $\mathcal{C}(x^k) = \mathcal{A}(x^*)$ with probability 1.*

Due to the difficulty of estimating the required number of iterations for the stabilization of the set $\mathcal{C}(\cdot)$, the complexity analysis of the Algorithm ABCDM must take into account that the sets $\mathcal{I}$ and $\mathcal{J}$ might not have been properly selected yet, and the method is just taking the worst possible decision. A similar situation occurs if we try to adapt the local (linear) convergence analysis of [27, Theorem 3]. To encompass this worst case situation, the linear rate constant worsens, giving the false impression that the ABCDM should be slower than the regular BCDM. This is not the case when the algorithm is applied to problems with real data, as we extensively show in Section 6.

6 Numerical tests

The numerical experiments were performed on a DELL notebook with an Intel(R) Core(TM) i7-4610M CPU 3.00GHz, using MATLAB version 9.0.0.341360 (R2016a) under Ubuntu Linux 16.04 [1]. We have focused in two classes of problems. The first one, known as *Lasso* [35], is given by

$$\min_x \frac{1}{2}\|Ax - b\|_2^2 + \lambda\|x\|_1. \tag{19}$$

The second, called ℓ_1*-regularized logistic regression* [26], is described by

$$\min_x \sum_{i=1}^{m} \log(1 + \exp(-b_i A_{i*}x)) + \lambda\|x\|_1. \tag{20}$$

[1] To reproduce the experiments, that demand third-party codes, please contact us (`pjssilva@ime.unicamp.br`), for further instructions.

In both problems, $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $\lambda > 0$, and A_{i*} denotes the ith-row of the matrix A.

These problems are well tailored to be solved by a block-coordinate descent method, allowing for any separation by blocks. They yield closed formulas for the solutions of the related subproblems whenever the matrices B_i's are diagonal. The Lipschitz constants for the gradient by blocks in both cases are easily computable if the blocks are small. The gradient of the smooth part can be updated with little computational effort if a single block changes. For more details see e.g. [30].

We discuss in the next subsection possibilities for choosing the set $\mathcal{J}$ in Active BCDM that estimates the active constraints of the problem.

6.1 Choosing the set $\mathcal{J}$

The natural choice for the set $\mathcal{J}$ is to use Propositions 1 and 3 to define

$$\mathcal{J} \equiv \mathcal{C}(x), \tag{21}$$

where $\mathcal{C}$ is defined in Step 9 of the algorithm Active BCDM. This variation of the method will be called `BCDM+IF`, where `IF` stands for identification function. Hence, we need to focus on problems that fulfill the assumptions of Proposition 2. Therefore, we will restrict our attention in this subsection to problems of the form

$$\begin{aligned} \min_x \quad & \frac{1}{2}\|Ax-b\|_2^2 + \lambda\|x\|_1 \\ \text{s.t.} \quad & x \geq 0 \end{aligned} \tag{22}$$

with $\lambda := 0.1\|A^T b\|_\infty$, $A \in \mathbb{R}^{m\times n}$ and $m > n$. This choice for the regularization parameter has been adopted in the literature, see e.g. [21, Section V.C] and [5, Chapter 11].

We will compare the choice for $\mathcal{J}$ given in (21), setting $\alpha = 0.99$, with the alternative

$$\mathcal{J} = \{i \mid x_i = 0 \text{ and } h_i(x) = 0\} \subset \mathcal{C}(x). \tag{23}$$

The inclusion is valid because any identification function will classify as active a constraint that is satisfied as equality at the current point. This opens up the path for using the information of any identification function, even one that we cannot compute explicitly, like the expression given in (13). We add the condition $h_i(x) = 0$ to avoid fixing a coordinate when the method says that it can be improved. This care is specially important in our context, because the starting point is usually the null vector, a point whose coordinates are all active. This strategy is associated with ideas that have already been used in the BCDM context, see [30]. This variation of the Active BCDM method will be denoted by `BCDM+ST`, with `ST` standing for stationarity.

We have selected problems that were used in [20, 33] to compare these two strategies. We have chosen those with $m > n$, and for which the data (A, b)

Table 1: Tests of Lasso instances with non-negative constraints

Label	Name	(m, n)	F^*	nz(x^*)
NN_1	[4] `illc1033`	$(1033, 320)$	1.098×10^7	88.12%
NN_2	[4] `illc1850`	$(1850, 712)$	1.771×10^7	94.52%
NN_3	[4] `well1033`	$(1033, 320)$	7.093×10^6	93.75%
NN_4	[4] `well1850`	$(1850, 712)$	8.295×10^6	96.91%
NN_5	[8] `real-sim`	$(72309, 20958)$	3.525×10^4	99.93%
NN_6	[8] `mnist`	$(60000, 717)$	3.241×10^5	95.53%
NN_7	[8] `webspam-unigram`	$(350000, 138)$	3.241×10^5	98.55%
NN_8	[11] `Maragal-3`	$(1690, 858)$	1.009×10^1	99.06%
NN_9	[11] `Maragal-4`	$(1964, 1027)$	1.865×10^1	99.22%
NN_{10}	[11] `Maragal-5`	$(4654, 3296)$	4.381×10^1	99.72%
NN_{11}	[11] `Maragal-6`	$(21255, 10144)$	5.968×10^1	99.90%
NN_{12}	[11] `Maragal-7`	$(46845, 26525)$	1.414×10^2	99.95%

were available in a deterministic way, summing up to 12 problems. The tests are originally least-squares problems with non-negativity constraints, i.e.

$$\begin{aligned} \min_x \quad & \frac{1}{2}\|Ax - b\|_2^2 \\ \text{s.t.} \quad & x \geq 0. \end{aligned} \tag{24}$$

In order to obtain a problem of the form (22), we added the extra term $\lambda\|x\|_1$. Table 1 presents the list of problems with the respective name, dimension, the original source, the target value for the stopping criteria (F^*), and the percentage of null coordinates at x^*, the solution of the problem (nz(x^*)).

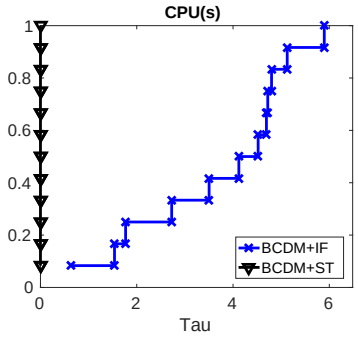
One main difficulty when trying to compare variations of the coordinate descent method, and other first-order methods, is the termination criterion. In this paper we will use a target objective value for termination, like in [12]. To define the target, we compute a high quality solution of each problem by letting a standard method, like the BCDM, to run for a very large number of iterations. We then round the final objective function to four significant digits and we add one to this last, fourth, digit. This procedure creates an upper bound for the optimal value that is correct up the third digit and has error of one in the fourth digit. Hence, any method that is able to compute a feasible point with objective smaller than this bound will solve the problem with relative error smaller than 10^{-4} when considering the objective value. This is a reasonable goal for a first-order method that struggles to achieve very high precision in real world problems. We adopt this stopping criterion in all our numerical experiments. As already mentioned, this target value is presented in Table 1 at the column labeled as F^*.

Furthermore, to completely define `BCDM+IF` and `BCDM+ST` it is necessary to select values for the parameters δ_{DP} and δ_F that are used in Algorithm 2. We did this by testing 42 combinations of these parameters, six values for $\delta_F \in \{0.2, 1, 5, 25, 125, 625\}$ and seven values for $\delta_{DP} \in \{1, 10, 10^2, 10^3, 10^4, 10^5, 10^6\}$.

We selected the best combination for each method by looking at the performance profiles [13] based on the total CPU time obtained after 10 runs of both methods in the 12 test problems, setting the size of the first cycle to 10, and the subsequent ones as $\max\{\min\{\delta_F|\mathcal{I}|, n\}, 10\}$. The best variations were `BCDM+IF` with $\delta_{DP} = 10$ and $\delta_F = 125$, and `BCDM+ST` with $\delta_{DP} = 10^5$ and $\delta_F = 5$.

Next, we compare the choices for $\mathcal{J}$, using again a performance profile confronting `BCDM+IF` and `BCDM+ST` (Figure 1, with $\log_2$ scale in the horizontal axis, for better visualization). It is clear from this figure that the best choice for $\mathcal{J}$ is the one used by `BCDM+ST`. From now on we will use `BCDM+ST` as the default implementation of Algorithm 2 and we will call it simply `ActiveBCDM`.

Figure 1: Performance profiles for `BCDM+IF` and `BCDM+ST` on the problems of Table 1



Finally, we compare `ActiveBCDM` with the original BCDM method without any identification. This is equivalent to setting $\mathcal{J} = \emptyset$ for all iterations. We call this variation `UBCDM`, as it always uses a uniform distribution to select the blocks. We still need to tune the parameter δ_F, as `UBCDM` needs to compute the objective function to stop when it achieves the target F^*. Again, we started by testing $\delta_F \in \{0.001, 0.01, 0.1, 1\}$ and then compared the performance profiles. The best variation turned out to be $\delta_F = 0.1$. After that, we compared the tuned `UBCDM` with `ActiveBCDM` (Figure 2). Once again, the method that tries to identify the active constraints is clearly faster.

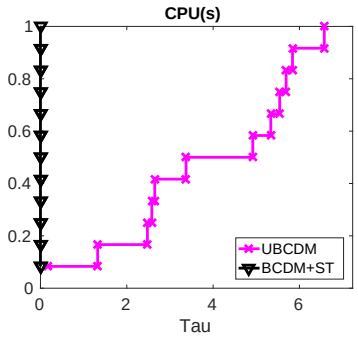
One important observation for the choice of $\mathcal{J}$ defined by (23) is that it opens up the possibility of applying identification ideas for unconstrained problems with ℓ_1 regularization. In fact, the problem

$$\min_x f(x) + \lambda\|x\|_1 \tag{25}$$

can be reformulated as a problem with simple constraints

$$\begin{aligned} \min_{x_+, x_-} \quad & f(x_+ - x_-) + \lambda e^T(x_+ + x_-) \\ \text{s.t.} \quad & x_+ \geq 0 \\ & x_- \geq 0, \end{aligned} \tag{26}$$

Figure 2: Performance profiles for `UBCDM` and `ActiveBCDM` on the problems of Table 1



where $e = (1, \ldots, 1)^T$. In particular, all solutions $\overline{x}$ of the problem (25) can be written in terms of solutions $(\overline{x}_+, \overline{x}_-)$ of (26), using $\overline{x} = \overline{x}_+ - \overline{x}_-$.

Let x^* be a solution of (25). A null coordinate $x_i^* = 0$ will be associated with two active constraints $(x_+^*)_i = 0$ and $(x_-^*)_i = 0$ in (26). Hence, we can use the estimate of the active set $\mathcal{J}$ given by (23) implicitly for (26) to identify such coordinates and speed up the convergence, even for unconstrained problems with ℓ_1 regularization. In the next subsections, we will apply `ActiveBCDM` aiming to explore these ideas.

6.2 Lasso

6.2.1 Data set and stopping criteria

Now that we have a choice for the set $\mathcal{J}$ in `ActiveBCDM` that works for problems in the form (25), we can revert our attention back to the unconstrained Lasso problem.

For testing our ideas, we have selected a set of 49 real problems from the literature. We tried to include not only problems for which A has more lines than columns, that have a stronger least-squares flavor, but also problems for which the matrix has more columns than rows, that are closer to variable selection. The problems are listed in Tables 2 and 3, together with information about source, dimensions, target objective F_{Las}^* and the percentage of null coordinates at x_{Las}^*, the solution of the problem, denoted by $\text{nz}(x_{\text{Las}}^*)$. The information in the last two columns of the aforementioned tables will be detailed and used in Subsection 6.3. The target was computed to be an upper bound on the real objective that is correct up to the third digit, like in the last section, and it was used to define the stopping criterion in the implementations. The problems were preprocessed, with the deletion of any null column that appeared in the data. Moreover, as in the previous section, we use $\lambda = 0.1\|A^T b\|_\infty$, i.e. $\lambda = 0.1\|\nabla f(0)\|_\infty$, for all tests.

Table 2: Test problems with more rows than columns

Label	Name	(m, n)	F^*_{Las}	$\text{nz}(x^*_{\text{Las}})$	F^*_{Log}	$\text{nz}(x^*_{\text{Log}})$
SR_1	[8] a1a.t	$(30956, 123)$	1.061×10^4	95.12%	1.605×10^4	95.12%
SR_2	[8] a2a.t	$(30296, 123)$	1.037×10^4	95.12%	1.569×10^4	95.12%
SR_3	[8] a4a.t	$(27780, 123)$	9.499×10^3	95.12%	1.439×10^4	95.12%
SR_4	[8] connect-4	$(67557, 126)$	2.537×10^4	91.26%	* * * * *	* * * * *
SR_5	[8] dna.scale	$(2000, 180)$	2.355×10^3	32.77%	* * * * *	* * * * *
SR_6	[8] mnist	$(60000, 717)$	3.241×10^5	95.53%	* * * * *	* * * * *
SR_7	[8] mushrooms	$(8124, 112)$	2.552×10^3	96.42%	3.741×10^3	98.21%
SR_8	[8] plishing	$(11055, 68)$	1.034×10^3	88.23%	4.265×10^3	86.76%
SR_9	[8] protein	$(17766, 356)$	7.108×10^3	77.80%	* * * * *	* * * * *
SR_{10}	[8] w2a	$(3470, 293)$	1.069×10^3	94.53%	1.551×10^3	95.22%
SR_{11}	[8] w4a.t	$(42383, 300)$	1.301×10^4	95.33%	1.880×10^4	95.33%
SR_{12}	[8] w5a.t	$(39861, 300)$	1.224×10^4	95.33%	1.769×10^4	95.33%
SR_{13}	[8] w6a.t	$(32561, 300)$	9.978×10^3	95.33%	1.442×10^4	95.33%
SR_{14}	[8] w8a.t	$(14951, 300)$	4.588×10^3	95.33%	6.634×10^3	95.33%
SR_{15}	[8] w5a	$(2833, 299)$	9.196×10^2	95.98%	1.342×10^3	95.98%
SR_{16}	[8] real-sim	$(72309, 20958)$	2.562×10^4	99.68%	3.641×10^4	99.75%
SR_{17}	[8] rcv1-test-bin	$(677399, 42735)$	2.428×10^5	99.83%	3.546×10^5	99.85%
SR_{18}	[8] rcv1-test-mult	$(518571, 41400)$	3.473×10^7	99.84%	3.505×10^5	99.78%
SR_{19}	[22] J-Lee	$(181395, 105353)$	1.271×10^2	99.88%	1.137×10^5	98.20%
SR_{20}	[8] webspam-unigram	$(350000, 138)$	1.413×10^5	92.02%	2.063×10^5	92.02%
SR_{21}	[11] Maragal-4	$(1964, 1027)$	1.865×10^1	99.22%	* * * * *	* * * * *
SR_{22}	[11] Maragal-5	$(4654, 3296)$	4.381×10^1	99.72%	3.077×10^3	95.35%
SR_{23}	[11] Maragal-6	$(21255, 10144)$	5.968×10^1	99.90%	1.425×10^4	97.39%
SR_{24}	[11] Maragal-7	$(46845, 26525)$	1.414×10^2	99.95%	3.181×10^4	99.18%

6.2.2 Extra subspace optimization and parameters selection

Once more, we need to set the values of the parameters δ_{DP} and δ_F to fully define the `ActiveBCDM` algorithm. We did this as before, testing different values in a few problems that are eliminated from the subsequent analysis. We used the problems from SR_1 to SR_9 and from SC_1 to SC_9 to tune this parameters. The same options of Section 6.1 were used for δ_{DP}, and for the length of the cycles, but we have used $\delta_F \in \{0.2, 1, 5, 25, 125\}$, as large values of δ_F did not produce satisfactory outcomes. These choices amounted to 35 combinations.

In a typical implementation of a coordinate descent method for Lasso, the gradient is not fully computed after each (coordinate) step; the coordinates of the gradient are computed only when needed [30]. However, when implementing `ActiveBCDM`, after every cycle of c_s iterations, except for the initial warming up cicle, we need to fully compute the gradient to estimate the set $\mathcal{J}$ of possibly active constraints. Since this computation is a big effort when compared to a single coordinate iteration, it is natural for us to try to exploit this opportunity to perform an extra step to speed up convergence.

To achieve this, we take inspiration in [17], a work that aims to incorporate second-order information in the model that defines the subproblem (6). After computing $\mathcal{J}$, let $\mathcal{I}$ be the associated set of free variables. We try to perform a

Table 3: Test problems with more columns than rows

Label	Name	(m, n)	F^*_{Las}	$\text{nz}(x^*_{\text{Las}})$	F^*_{Log}	$\text{nz}(x^*_{\text{Log}})$
SC_1	[6] peppers05-6-6	$(32768, 65536)$	7.277×10^4	92.53%	* * * * *	* * * * *
SC_2	[6] peppers05-12-12	$(32768, 65536)$	1.343×10^5	92.27%	* * * * *	* * * * *
SC_3	[6] peppers025-12-12	$(16384, 65536)$	1.184×10^5	88.83%	* * * * *	* * * * *
SC_4	[3] SparcoProblem401	$(29166, 57344)$	1.605×10^2	98.66%	* * * * *	* * * * *
SC_5	[3] SparcoProblem402	$(29166, 57344)$	1.605×10^2	98.66%	* * * * *	* * * * *
SC_6	[3] SparcoProblem603	$(1024, 4096)$	1.099×10^2	99.75%	* * * * *	* * * * *
SC_7	[6] finance1000	$(30465, 216842)$	1.846×10^5	99.99%	2.110×10^4	99.99%
SC_8	[23] dbworld-bodies	$(64, 4702)$	5.134×10^0	99.72%	2.104×10^1	99.44%
SC_9	[23] dexter-train	$(300, 7751)$	1.175×10^2	99.75%	1.683×10^2	99.78%
SC_{10}	[23] dexter-valid	$(300, 7847)$	1.147×10^2	99.65%	1.667×10^2	99.71%
SC_{11}	[23] dorothea-train	$(800, 88119)$	2.043×10^2	99.89%	3.238×10^2	99.92%
SC_{12}	[23] dorothea-valid	$(350, 72113)$	8.833×10^1	99.88%	1.410×10^2	99.91%
SC_{13}	[8] news20-binary	$(19996, 1355191)$	7.334×10^3	99.99%	1.078×10^4	99.99%
SC_{14}	[8] news20-scale	$(15935, 60346)$	6.586×10^5	99.98%	1.075×10^4	99.86%
SC_{15}	[8] news20-t-scale	$(3993, 39128)$	1.652×10^5	99.98%	2.699×10^3	99.90%
SC_{16}	[8] rcv1-train-bin	$(20242, 44504)$	7.064×10^3	99.82%	1.042×10^4	99.84%
SC_{17}	[8] rcv1-train-mult	$(15564, 36842)$	1.054×10^6	99.79%	1.022×10^4	99.35%
SC_{18}	[8] sector-scale	$(6412, 49087)$	6.009×10^6	99.90%	4.335×10^3	99.94%
SC_{19}	[8] sector-t-scale	$(3207, 39234)$	2.994×10^6	99.89%	2.182×10^3	99.89%
SC_{20}	[22] Blanc-Mel	$(186414, 685568)$	2.596×10^2	99.99%	1.189×10^5	99.84%
SC_{21}	[23] farm-ads-vect	$(4143, 54877)$	1.444×10^3	99.91%	2.129×10^3	99.92%
SC_{22}	[6] mug05-12-12	$(12410, 24820)$	5.363×10^4	93.73%	* * * * *	* * * * *
SC_{23}	[6] mug025-12-12	$(6205, 24820)$	4.982×10^4	92.19%	* * * * *	* * * * *
SC_{24}	[6] mug075-12-12	$(13651, 24820)$	5.837×10^4	94.40%	* * * * *	* * * * *
SC_{25}	[11] Maragal-8	$(33212, 60845)$	2.037×10^2	99.76%	2.180×10^4	99.65%

step along the associated subspace

$$h_{\mathcal{I}}(x) = \underset{x+h\in\mathcal{X}}{\operatorname{argmin}} \left\{ \nabla_{\mathcal{I}} f(x)^T h_{\mathcal{I}} + \frac{1}{2} h_{\mathcal{I}}^T B h_{\mathcal{I}} + \psi_{\mathcal{I}}(x_{\mathcal{I}} + h_{\mathcal{I}}) \right\}, \tag{27}$$

with $h = U_{\mathcal{I}} h_{\mathcal{I}}$. The matrix B conveys second-order information from the smooth function f. Following the numerical results from [17], we tested two choices for B, namely $B = \text{diag}(\nabla^2_{\mathcal{I}} f(x))$ and $B = \nabla^2_{\mathcal{I}} f(x)$, where $\text{diag}(\cdot)$ is a function that returns the diagonal matrix whose diagonal is extracted from the argument. The convergence of the resulting algorithm was preserved by enforcing the monotonicity of the objective function along the generated sequence of iterates. To achieve this, we have employed a sufficient descent condition inspired by SpaRSA [39]. To decide whether $h_{\mathcal{I}}(x)$ is acceptable, we perform the test

$$F(x + U_{\mathcal{I}} h_{\mathcal{I}}) \leq F(x) - \frac{\gamma}{2} \|h_{\mathcal{I}}\|_2^2, \text{ with } \gamma = 10^{-6},$$

and accept the step if the test is satisfied. Otherwise, the extra step is ignored. We will call this variation of ActiveBCDM+SO, in which SO stands for second order.

Finally, we performed the tests to tune the parameters δ_{DP}, δ_F, for both ActiveBCDM and ActiveBCMD+SO. We should mention that we have used the slightly different range $\delta_F \in \{0.04, 0.2, 1, 5, 25\}$ for the ActiveBCDM+SO, including an even smaller option and excluding the largest one adopted for ActiveBCDM.

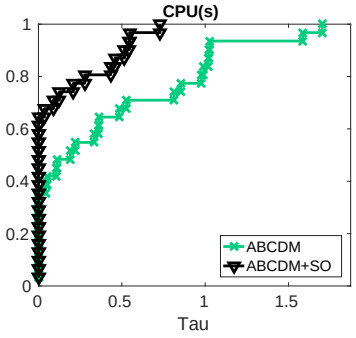
We used the mean CPU time of 10 simulations for each test problem and looked at the performance profiles to compare the results. The best choices for `ActiveBCDM` were $\delta_{DP} = 10^4$ and $\delta_F = 5$. As for `ActiveBCDM+SO`, the best values were $\delta_{DP} = 10^5$, $\delta_F = 1$ and $B = \text{diag}(\nabla^2_{\mathcal{I}} f(x))$.

In the next section we compared the tuned versions of `ActiveBCDM` and `ActiveBCDM+SO` between each other and with other well-known methods from the literature for Lasso.

6.2.3 Comparison with other methods

Figure 3 shows the performance profiles of the tuned versions of `ActiveBCDM` and `ActiveBCDM+SO` in the remaining 31 problems obtained after discarding SR_1 to SR_9 from Table 2 and SC_1 to SC_9 from Table 3, used in the tuning stage. The extra step clearly proved beneficial, speeding up the convergence.

Figure 3: Performance profiles for `ActiveBCDM` and `ActiveBCDM+SO` on 31 Lasso problems of Tables 2 and 3



Next, we tested the tuned version of the `ActiveBCDM+SO` for the same 31 problems against other methods from the literature. The first method was `SpaRSA` [39], based on the iteration

$$x^{k+1} = \text{argmin}_z \nabla f(x^k)^T (z - x^k) + \frac{\alpha_k}{2} \|z - x^k\|_2^2 + \lambda \|z\|_1,$$

with step length α_k selected using a nonmonotone rule. For further details, see [39].

The second method is called `FAST-BCD2-E`. Proposed in [12], it is a block coordinate descent method using directions obtained with the same rule of Algorithm 1. It rests upon an identification strategy, different from ours, that also tries to find the null coordinates at the minimizer. After this identification step, a subset of the inactive coordinates that violates the optimality condition the most is updated. For details, see [12].

The third method is called `OWL-QN` [1]. It rests upon a limited-memory BFGS strategy applied to the problem (19), using the subgradient of minimum norm of the objective function F in the place of the gradient of the model,

and minimizing the model over a specific orthant in which the function F is differentiable. We run an implementation coded by Mark Schmidt, described in [32].

Two additional methods, `PSSas` and `PSSgb`, were considered. Both are implementations of the limited-memory BFGS strategy and were developed by Mark Schmidt in his Ph.D thesis [32], for which good numerical results are described in the context of the ℓ_1-regularized logistic regression.

Last, but not the least, we have also compared our approach with `glmnet`, a fast method used by many practioners, based on cyclical coordinate descent, computed along a regularization path [18].

We used the implementations made available by the original authors for the third-party methods (in MATLAB), and we used them with their default parameters. For all tests, we compare the total CPU time to find a point whose objective is below F^*, starting from the null vector. We use the total CPU time because none of the methods are trying to explicitly exploit parallelism. Since some methods are nondeterministic, we consider the average time of 10 runs to proceed with the analysis.

Figure 4 presents performance profiles comparing all methods based on the BFGS updating against `ActiveBCDM+SO`. The figure on the left compares the methods among themselves in solving our test set and the one on the right compares the best BFGS-based method, namely `PSSgb`, against `ActiveBCDM+SO`. Although `ActiveBCDM+SO` shows a slight advantage in terms of efficiency, both are equally robust.

Figure 5 compares `ActiveBCDM+SO` with `FAST-BCD2-E` (left) and with `SpaRSA` (right). The profiles indicate that `ActiveBCDM+SO` outperforms both methods for this set of test problems.

Finally, the comparison between `ActiveBCDM+SO` and `glmnet` is depicted in Figure 6. One can see that both solvers are equally robust, with `ActiveBCDM+SO` in the lead as far as efficiency is concerned. To offer a complete comparative perspective, in Figure 7 we show together profiles of all the methods considered in this section.

6.3 ℓ_1-regularized logistic regression

6.3.1 Data sets and stopping criteria

In this section we describe the results of testing the variants of `ActiveBCDM` to solve ℓ_1-regularized logistic regression problems (20), from now on shortened referred as ℓ_1-RLR. We set the parameter $\lambda = 0.1\|\nabla f(x^0)\|$, mimicking the choice used for Lasso to achieve sparse solutions. We have started the methods from $x^0 = 0$, as previously.

The ℓ_1-RLR problem is well suited for binary classification, that is, problems in which the response vector b has entries $b_i \in \{-1, 1\}, i = 1, \ldots, n$. Only problems 1–3, 8, 10–17, 19, and 20 from Table 2 and problems 8–13, 16, 20, and 21, from Table 3 have this feature, amounting to 23 binary classification test problems.

Figure 4: Performance profiles among the quasi-Newton methods (left) and between the best quasi-Newton method and `ActiveBCDM+SO` (right) for 31 Lasso problems of Tables 2 and 3.

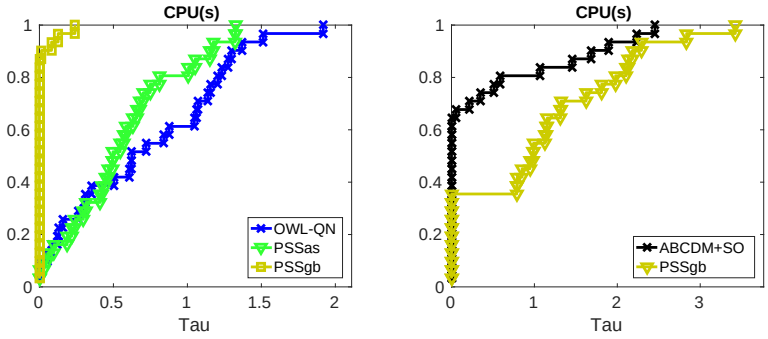


Figure 5: Performance profiles for `ActiveBCDM+SO` and `FAST-BCD2-E` (left) and for `ActiveBCDM+SO` and `SpaRSA` (right) on 31 Lasso problems of Tables 2 and 3.

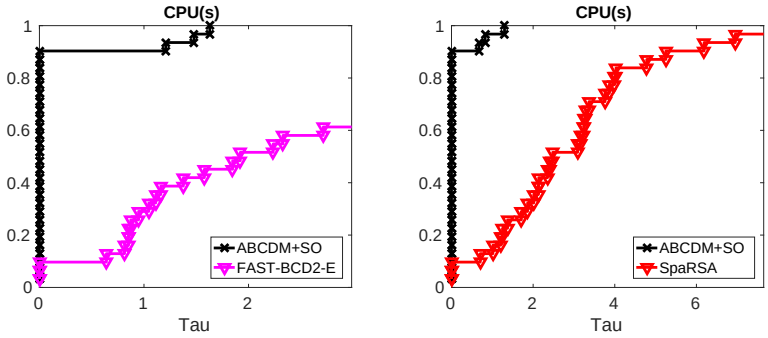


Figure 6: Performance profiles for `ActiveBCDM+SO` and `glmnet` on 31 Lasso problems of Tables 2 and 3

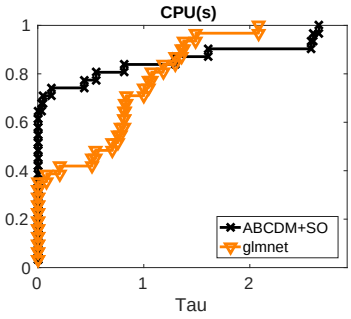
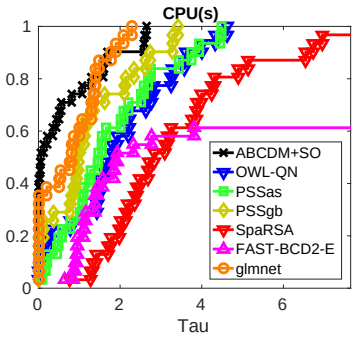


Figure 7: Perfomance profiles for all the considered solvers on 31 Lasso problems of Tables 2 and 3



In order to try to use the available information of the matrix A of the 26 problems that have been left out, we created a random binary right-hand side using the following procedure. We started by generating a random vector x with standard normally distributed coordinates. Next we nullified roughly 50% of the entries of x, once again randomly, and computed b as the sign vector of Ax. We then flipped approximately 10% of the entries of b to introduce misclassification in the data. Finally, we checked, by solving the problem with `PSSgb`, whether the solution had at least 90% of null coordinates. The problems for which this goal was achieved were SR_7, SR_{18}, SR_{22} to SR_{24}, SC_7, SC_{14}, SC_{15}, SC_{17} to SC_{19}, SC_{25}. We add such problems to the original set of 23 binary classification problems, totaling 35 test problems. The emphasis on problems with sparse solution is justified by the fact that the identification strategy may be effective only in this case.

Again, the stopping criterion is based on an upper bound on the objective function that is correct up to an error of one in the fourth digit. These upper bounds are provided at the penultimate column of Tables 2 and 3. The last column of the tables contains information about the percentage of the number of null coordinates of the solution of the problem $\text{nz}(x^*_{\text{Log}})$. We have used $*****$ to indicate the problems with inappropriate data for ℓ_1-RLR, i.e., the problems for which we could not generate a satisfactory binary vector b following the aforementioned procedure.

6.3.2 Extra subspace optimization and parameters selection

In this section we will tune the parameters δ_{DP}, δ_F and the choice for the matrix B as in Section 6.2.1. We have used 12 problems in this tuning procedure, namely SR_1, SR_2, SR_3, SR_8, SR_{10}, SR_{11}, SC_8, SC_9, SC_{10}, SC_{11}, SC_{12}, SC_{13}. We use this set of problems because they originally have a binary response vector b.

We compared the CPU time obtained by means of the average of 10 runs

among all variants of the methods `ActiveBCDM` and `ActiveBCDM+SO`, using the same range for the parameters for δ_{DP}, and the same length for the cycles described in Section 6.1. We have tunned $\delta_F \in \{0.2, 1, 5, 25, 125\}$ for `ActiveBCDM`, and $\delta_F \in \{0.04, 0.2, 1, 5, 25\}$ for `ActiveBCDM+SO`.

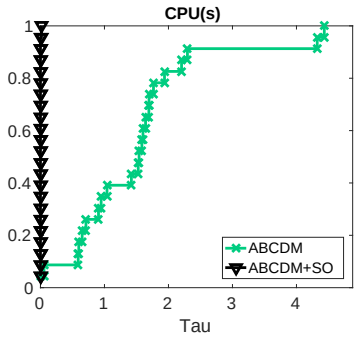
The best performances were attained with the values $\delta_{DP} = 10^6$ and $\delta_F = 1$ for `ActiveBCDM`, and $\delta_{DP} = 10^5$, $\delta_F = 0.2$, and $B = \nabla^2_I f(x)$ for `ActiveBCDM+SO`. These choices of parameters were used for the tests of the next subsection.

6.3.3 Comparison with other methods

Here, we compare the tuned versions of `ActiveBCDM` and `ActiveBCDM+SO` and other six methods of the literature in the remaining 23 problems, after excluding the 12 problems used to tune the parameters. Once again, for the third-party methods we have used the original implementations (in MATLAB) and the default parameters. The main performance metric is the total CPU time.

We start comparing `ActiveBCDM` and `ActiveBCDM+SO` (Figure 8). We can see that for the ℓ_1-RLR problems the second-order step is effective for solving most of the problems of our testing set.

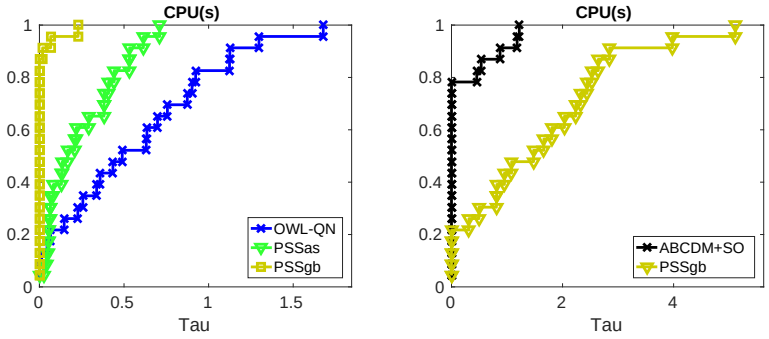
Figure 8: Performance profiles for `ActiveBCDM` and `ActiveBCDM+SO` on 23 ℓ_1-RLR problems of Tables 2 and 3



Next we compare `ActiveBCDM+SO` with other methods. We start by comparing it with the three BFGS-based strategies that were used in the previous section, namely `OWL-QN`, `PSSas`, and `PSSgb`. Figure 9 shows, on the left, the performance profiles of these three methods solving the aforementioned 23 logistic regression test problems. `PSSgb` is the best method, as expected. It is a limited memory BFGS that was specially tailored to solve the ℓ_1-RLR problem [32]. The profiles on the right compare `PSSgb` with `ActviveBCDM+SO`. Both methods have similar performance, with just a slight advantage for `PSSgb`, in spite of the fact that `ActviveBCDM+SO` performs mostly simple first-order iterations.

We have also compared `ActiveBCDM+SO` with the methods `FCDv.1` and `FCDv.2`, introduced in [17]. The latter two methods are also block-coordinate descent methods that update the coordinate blocks similarly to the Active BCDM, but

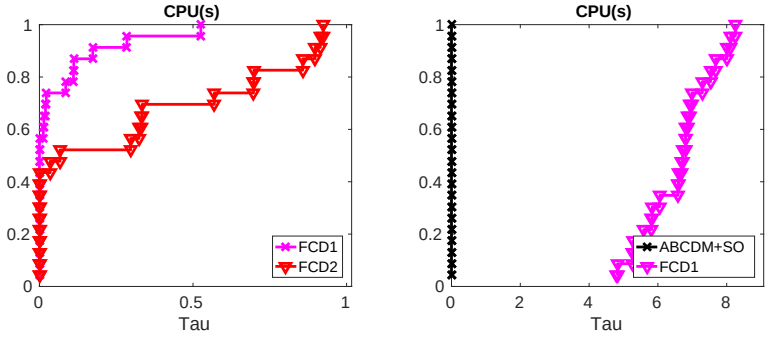
Figure 9: Performance profiles among the quasi-Newton methods (left) and between the best quasi-Newton method and `ActiveBCDM+SO` (right) for 23 ℓ_1-RLR problems of Tables 2 and 3



solve each subproblem inexactly, using information of the diagonal of the Hessian (`FCDv.1`) and of the full Hessian (`FCDv.2`) of the smooth objective function f by blocks of size $\lfloor 0.001n \rfloor$.

Again, we start comparing these two methods to select the best one, which was in turn compared to `ActiveBCDM+SO`. In this case, `ActiveBCDM+SO` clearly outperformed `FCDv.1` and `FCDv.2`.

Figure 10: Performance Profile between the methods `FCDv.1` and `FCDv.2` (left) and between the best `FCD` method and `ActiveBCDM+SO` (right) for 23 ℓ_1-RLR problems of Tables 2 and 3



Additionally, `ActiveBCDM+SO` has been compared to `FaRSA`, an algorithm introduced in [9] for the unconstrained optimization of ℓ_1-regularized functionals. `FaRSA` is also based on subspace acceleration strategies, and on the possibility of identifying the set of zero variables at the solution. The performance profiles considering the whole set of 23 ℓ_1-RLR problems of Tables 2 and 3 are presented in Figure 11, showing that both methods have a remarkably similar performance. However, if we split the test set into two groups, one for which the matrices have more columns than lines ($m > n$), and the other for prob-

lems with more columns than lines ($m < n$), a definite difference emerges. See Figure 12. Clearly, for the problems SR_7, SR_{12} to SR_{20}, and SR_{22} to SR_{24}, with $m > n$, `FaRSA` has superior performance, due to the limited room for improvement when selecting columns. For the problems SC_7, SC_{14} to SC_{21} and SC_{25}, on the other hand, `ActiveBCDM+SO` has better performance, evincing the impact of our identification strategy.

As we have done for the Lasso instances, Figure 13 depicts the comparative performance of all the methods considered for solving the ℓ_1-RLR instances.

Figure 11: Performance profiles for `ActiveBCDM+SO` and `FaRSA` for 23 ℓ_1-RLR problems of Tables 2 and 3

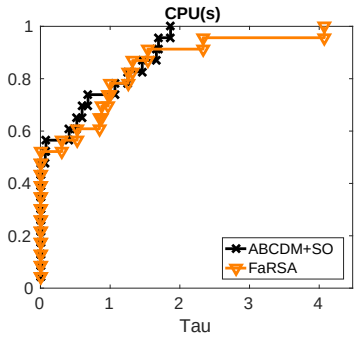
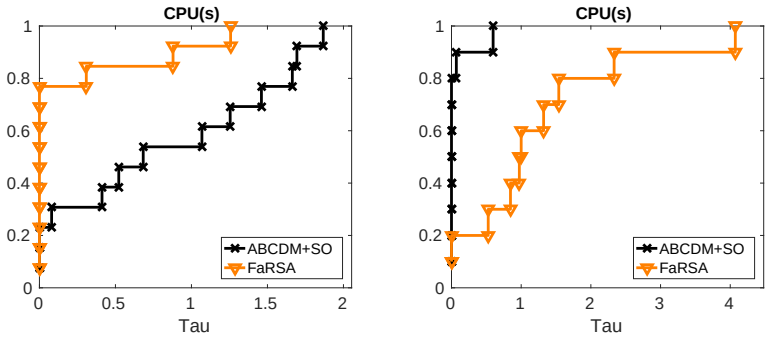


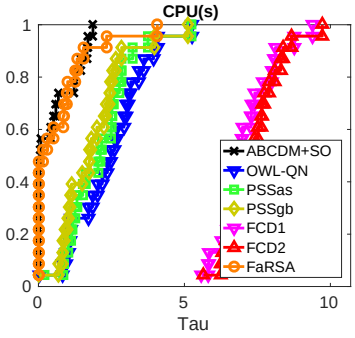
Figure 12: Performance profiles for `ActiveBCDM+SO` and `FaRSA` for 13 ℓ_1-RLR problems of Table 2 (left) and 10 ℓ_1-RLR problems of Table 3 (right)



7 Final remarks

We have proposed identification strategies for accelerating block coordinate descent methods, which rest upon the idea of identification functions. An enhanced version of the algorithm BCDM was presented, the so-called Active BCDM algorithm, with global convergence results. Our analysis encompasses the usage

Figure 13: Perfomance profiles for all the considered solvers on 23 ℓ_1-RLR problems of Tables 2 and 3



of different probabilities in the selection of blocks at each cycle of iterations, with complete freedom on the choices, as long as the probabilities are bounded away from zero.

Although our primary focus was to address the bound-constrained minimization of composite functions, our strategy also proved valuable for solving unconstrained problems with ℓ_1 regularization. Indeed, this class of problems has a bound-constrained equivalent formulation that guides the probabilistic selection of the adequate sets of coordinates to work with for achieving the desired sparse solution.

There are two key parameters in our approach, namely the frequency δ_F of iterations for keeping the probability function fixed, and the preference factor δ_{DP} of the blocks within the current set $\mathcal{I}$ of estimated inactive constraints. The proposed algorithm allows the flexibility of selecting just a subset of constraints to declare as active. This strategy showed advantageous practical results for Lasso problems with simple bounded variables arising from 12 test instances originated from least-squares problems with non-negativity constraints extracted from [4, 8, 11].

Additionally, we have worked with two classes of test problems: Lasso and ℓ_1-regularized logistic regression. Instead of solving problems with randomly generated data, we preferred to assess our strategy comparatively by means of 49 test problems with deterministic data from the literature, selected from [3, 4, 6, 8, 11, 22, 23].

It is worth mentioning that a careful tuning had been performed concerning the parameters δ_F and δ_{DP}. Moreover, we have investigated the performance of potentially adopting a second-order step, following recent ideas from [17], with effective results for the set of test problems considered.

Nine third-party solvers were used to put our MATLAB implementation of the `ActiveBCDM` in perspective: `OWL-QN` [1]; `PSSas` and `PSSgb` [32]; `FAST-BCD2-E` [12]; `SpaRSA` [39]; `glmnet` [18]; `FaRSA` [9]; `FCDv.1` and `FCDv.2` [17]. The first three were used for addressing both classes of problems (Lasso and ℓ_1-RLR).

The fourth, the fifth, and the sixth were used just for Lasso, whereas the last three were applied only for solving logistic regression problems.

Future work includes the investigation of a parallel version of Active BCDM, whose main difficulty is to adapt existing convergence results, due to the changing of sets of probabilities along the cycle of iterations.

Acknowledgements. We are thankful to the comments and suggestions of two anonymous referees, which helped us to improve the presentation of our work.

References

[1] G. Andrew and J. Gao. Scalable training of L1-regularized log-linear models. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 33–40, New York, NY, USA, 2007. ACM.

[2] A. Beck and L. Tetruashvili. On the convergence of block coordinate descent type methods. *SIAM Journal on Optimization*, 23(4):2037–2060, 2013.

[3] E. V. Berg, M. P. Friedlander, G. Hennenfent, F. Herrmann, R. Saab, and Ö. Yılmaz. SPARCO: A testing framework for sparse reconstruction. Technical Report TR-2007-20, Dept. Computer Science, University of British Columbia, Vancouver, October 2007.

[4] R. F. Boisvert, R. Pozo, K. Remington, R. F. Barrett, and J. J. Dongarra. *Matrix Market: a web resource for test matrix collections*, pages 125–137. Springer US, Boston, MA, 1997.

[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2010.

[6] J. K. Bradley, A. Kyrola, D. Bickson, and C. Guestrin. Parallel coordinate descent for ℓ_1-regularized loss minimization. In ICML2011, editor, *Proceedings of the 28th International Conference on Machine Learning*, pages 1–8, Bellevue, Washington, USA, June-July 2011. The International Machine Learning Society.

[7] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

[8] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[9] T. Chen, F. Curtis, and D. Robinson. A reduced-space algorithm for minimizing ℓ_1-regularized convex functions. *SIAM Journal on Optimization*, 27(3):1583–1610, 2017.

[10] D. Csiba, Z. Qu, and P. Richtárik. Stochastic dual coordinate ascent with adaptive probabilities. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning*, volume 37 of *ICML'15*, pages 674–683, Lille, France, 2015. JMLR.org.

[11] T. A. Davis and Y. Hu. The University of Florida Sparse Matrix Collection. *ACM Transactions on Mathematical Software*, 38(1):1:1–1:25, Dec. 2011.

[12] M. De Santis, S. Lucidi, and F. Rinaldi. A fast active set block coordinate descent algorithm for ℓ_1-regularized least squares. *SIAM Journal on Optimization*, 26(1):781–809, 2016.

[13] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213, 2002.

[14] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[15] F. Facchinei, A. Fischer, and C. Kanzow. On the accurate identification of active constraints. *SIAM Journal on Optimization*, 9(1):14–32, 1998.

[16] O. Fercoq and P. Richtárik. Accelerated, parallel and proximal coordinate descent. *SIAM Journal on Optimization*, 25(4):1997–2013, 2015.

[17] K. Fountoulakis and R. Tappenden. A flexible coordinate descent method. *Computational Optimization and Applications*, 70(2):351–394, 2018.

[18] J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

[19] T. Glasmachers and U. Dogan. Accelerated coordinate descent with adaptive coordinate frequencies. In *Proceedings of the 5th Asian Conference on Machine Learning (ACML)*, volume 29 of *Proceedings of Machine Learning Research*, pages 72–86, Australian National University, Canberra, Australia, 2013. PMLR.

[20] D. Kim, S. Sra, and I. S. Dhillon. A non-monotonic method for large-scale non-negative least squares. *Optimization Methods Software*, 28(5):1012–1039, Oct. 2013.

[21] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale ℓ_1-regularized least squares. *IEEE Journal on Selected Topics in Signal Processing*, 1(4):606–617, 2007.

[22] P. Komarek. Paul Komarek's webpage. `http://komarix.org/ac/ds/`. Accessed: 29-January-2017.

[23] M. Lichman. UCI Machine Learning Repository, University of California, Irvine, School of Information and Computer Sciences. `http://archive.ics.uci.edu/ml`. Last updated 23-July-2017; accessed 01-September-2017.

[24] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning*. The MIT Press, Cambridge, USA, 2012.

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

[26] A. Y. Ng. Feature selection, L_1 vs. L_2 regularization and rotational invariance. In *Proceedings of the 21st International Conference on Machine Learning*, page paper 354, 2004.

[27] A. Patrascu and I. Necoara. Efficient random coordinate descent algorithms for large-scale structured nonconvex optimization. *Journal of Global Optimization*, 61(1):19–46, 2015.

[28] Z. Qu and P. Richtárik. Coordinate descent with arbitrary sampling i: algorithms and complexity. *Optimization Methods and Software*, 31(5):829–857, 2016.

[29] P. Richtárik and M. Takáč. *Efficient Serial and Parallel Coordinate Descent Methods for Huge-Scale Truss Topology Design*, pages 27–32. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

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

[31] P. Richtárik and M. Takáč. Parallel coordinate descent methods for big data optimization. *Mathematical Programming*, 156(1):433–484, 2016.

[32] M. Schmidt. *Graphical Model Structure Learning with L1-Regularization*. PhD thesis, University of British Columbia, Vancouver, 2010.

[33] M. Slawski. Problem-specific analysis of non-negative least squares solvers with a focus on instances with sparse solutions (working paper). `https://sites.google.com/site/slawskimartin/publications`, 2013. accessed 01-September-2017.

[34] R. Tappenden, P. Richtárik, and J. Gondzio. Inexact coordinate descent: Complexity and preconditioning. *Journal of Optimization Theory and Applications*, 170(1):144–176, 2016.

[35] R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society, Series B*, 58(1):267–288, 1996.

[36] P. Tseng and S. Yun. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, 117(1):387–423, 2009.

[37] Z. Wen, W. Yin, H. Zhang, and D. Goldfarb. On the convergence of an active-set method for ℓ_1 minimization. *Optimization Methods and Software*, 27(6):1127–1146, 2012.

[38] S. J. Wright. Coordinate descent algorithms. *Mathematical Programming*, 151(1):3–34, 2015.

[39] S. J. Wright, R. D. Nowak, and M. A. T. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009.

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