Parallel Block Coordinate Minimization with Application to Group Regularized Regression

Giuseppe C. Calafiore[*]

Abstract

This paper proposes a method for parallel block coordinate-wise minimization for convex functions. Each iteration involves a first phase where n independent minimizations are performed over the n variable blocks, followed by a phase where the results of the first phase are coordinated to obtain the whole variable update. Convergence of the method to the global optimum is proved for functions composed of a smooth part plus a possibly non-smooth but separable term. The method is also proved to have linear rate of convergence, for functions that are smooth and strongly convex. The proposed algorithm can give computational advantage over the more standard serial block coordinate-wise minimization methods, when run over a parallel, multi-worker, computing architecture. The method is suitable for regularized regression problems, such as the group Lasso, group Ridge regression, and goup Elastic Net. Numerical tests are run on such type of regression problems to exemplify the performance of the proposed parallel method in comparison with the serial one.

Keywords: Block coordinate minimization, Nondifferentiable minimization, Parallel methods, Group Lasso, Ridge regression, Elastic Net, Regularized regression.

[*]Giuseppe C. Calafiore, Dipartimento di Automatica e Informatica, Politecnico di Torino, Italy. Tel.: +39-011-564.7071; Fax: +39-011-564.7099. E-mail: `giuseppe.calafiore@polito.it`

1 Introduction

Coordinate minimization methods are based on the idea of iteratively minimizing an objective function f over a cyclically selected single variable (or over a block of variables, for the so-called block-coordinate minimization methods), while all the other variables are held fixed. A multivariate optimization problem is thus reduced to a sequence of univariate (or single block-variate) minimizations over coordinate directions, each of which can often be solved very efficiently, or even in "closed form," for some specific problems. These methods offer the advantage of simplicity and of being typically derivative-free, which made them recently a very popular choice for the solution of several large-scale problems arising in machine learning and regularized regression, such as the Lasso [16], the group Lasso [15, 18], the Ridge regression and Elastic Net [19], and the sparse logistic regression problem [8, 10].

In the standard serial (or Gauss-Seidel type) approach to block coordinate minimization, given a current value $x^{(k)}$ of the decision variable x at iteration k, the objective is minimized with respect to a selected block, then the variable is updated, a new block is selected for minimization, and so on, according to the scheme

$$x_i^{(k+1)} = \arg\min_{x_i} f(x_1^{(k+1)}, \ldots, x_{i-1}^{(k+1)}, x_i, x_{i+1}^{(k)}, \ldots, x_n^{(k)}), \quad i = 1, \ldots, n. \tag{1}$$

Different methods vary essentially with respect to the way in which the blocks are selected (e.g., deterministically in a cyclic sequence, or in a randomized fashion), but these methods remain intrinsically serial, since the minimization with respect to a block needs to wait for the result of the minimization over the previous block in order to be carried out. Serial (a.k.a. *cyclic*) coordinate descent methods have been analyzed in a number of works. For convex and differentiable f, convergence to the optimum is guaranteed (see, e.g., [6]), and rates of convergence have been obtained, for randomized schemes in, e.g., [9]. For convex but possibly non differentiable f, the cyclic method does not converge in general, unless f has some additional properties. A special but important case where convergence can be guaranteed is when f is composed of a convex and differentiable term, plus a *separable* convex but possibly non-differentiable term. This case is studied in the seminal paper [17], where a proof of convergence for the cyclic coordinate minimization method is given. Determining rates of convergence of the method, however, is hard (Nesterov in [9] claimed that it is "almost impossible to estimate the rate of convergence" of cyclic coordinate descent methods, in the general case), and only few results are currently available in the literature; an excellent review in this respect is given in Section 10 of [14], to which the reader is referred for further details and references.

Serial block-coordinate descent is generally regarded as the technique of choice in many large-scale problems arising in machine learning. As the number and scale of the blocks increases, however, the need arises to exploit the potential of parallel computing architectures, in order to solve larger problems in reasonable time. A substantial speedup from parallelization would be achieved if the blocks could be minimized in parallel, instead of serially. In a purely parallel approach (also known as the Jacobi-type method), given a current value $x^{(k)}$ of the variable x at iteration k, the objective is minimized with respect to all blocks simultaneously, and the variable is then updated to the obtained values of the blocks, according to the scheme

$$x_i^{(k+1)} = \arg\min_{x_i} f(x_1^{(k)}, \ldots, x_{i-1}^{(k)}, x_i, x_{i+1}^{(k)}, \ldots, x_n^{(k)}), \quad i = 1, \ldots, n. \tag{2}$$

Such an approach, unfortunately, fails to converge, in general, even on convex and smooth functions. Also, very little material is available in the literature on parallel versions of coordinate minimization schemes, which are the subject of the present paper. To the best of our knowledge, the relevant contributions are rare: an early analysis, for smooth objectives, is given in the classical book by Tsitsiklis and Bertsekas, [4], and a recent one is given in [1]. This latter contribution, however, deals with a parallel coordinate descent method with only scalar blocks, and with a specific regularization function, given by the ℓ_1 norm of the variable. An important contribution recently also appeared in [13], where the authors propose a remarkably inclusive framework for analysis of parallel descent methods on a class of convex functions composed by a partially separable and smooth term, plus a "simple" separable term. The method in [13] relies on random sampling and iterative parallel descent steps based on a separable over-approximation of the expected value of the objective function. This method bears some resemblance to proximal methods, in that it requires the separable term to be of simple structure, as well as knowledge of some structural parameters of the objective function, such as Lipschitz constants and degree of separability, which, incidentally, are not required in the parallel method described in the present work. Also, we notice that in the literature there is a distinction between coordinate *minimization* (CM) methods and coordinate *descent* (CD) methods. In CM methods an ideally exact minimization over the selected coordinate block is performed at each step, whereas in CD methods an update step is more simply performed over the selected coordinate block, based on local information, such as the partial gradient or a global over-approximations of the objective function at the current point. To the best of this author's knowledge, the only existing results on algorithms' convergence rate (both serial and parallel) are available for CD methods; no result seems to be available for parallel CM methods, which are the focus ot the present paper.

In this paper, we propose a parallel version of a block coordinate minimization method for classes of convex objective functions, which include the objectives usually encountered in regularized regression problems. Every iteration in our scheme essentially works in two phases: given a current value of the variable x, in the first phase n independent block-coordinate minimizers are computed, as in a standard Jacobi approach. In a second phase, these minimizers are passed to a centralized coordinator, who suitably updates the current solution, and then passes it back to the workers for the next iteration. The first phase of the method allows ideally for an n_w-times speedup with respect to a sequential method, where n_w is the number of "workers" that are available in parallel for performing the n independent block minimizations. The second phase requires a (typically mild) additional effort for coordinating the results of the first phase; this coordination amounts to computing a suitable stepsize for the block updates. We prove convergence of this method to the global optimum, for convex functions satisfying a smooth plus separable condition (see Assumption 1), under no additional requirements. Further, we prove that, for smooth and strongly convex functions, the proposed method is guaranteed to have at least a linear convergence rate.

Also, we specialize the proposed algorithm to the group Lasso and group Ridge regression problems, and perform some numerical tests, showing the actual potential of the method for speedup in a parallel computing environment, with respect to the serial approach.

The paper is organized as follows: Section 2 contains definitions, assumptions and preliminary results; the main Section 3 describes the parallel block coordinate minimization method and states a global convergence result, under Assumption 1. Section 4 provides a rate of convergence result, under Assumption 2. In Section 5 we provide the details of the algorithm's implementation for the group Lasso and Ridge regression problems, and in Section 6 we report the results of some numerical tests on these regression problems. Conclusions are drawn in Section 7.

2 Setup and preliminaries

Let $f : \mathbb{R}^N \to \mathbb{R}$ be a convex function taking values $f(x) = f(x_1, \ldots, x_n)$, where $x_i \in \mathbb{R}^{n_i}$ is a block of variables of dimension $n_i \geq 1$, with $\sum_{i=1}^n n_i = N$. We assume that $\operatorname{dom} f$ is open, and we denote with $E_i \in \mathbb{R}^{N,n_i}$ a stacked block matrix where the j-th block is a zero block of dimension $n_j \times n_i$, for $j \neq i$, and it is an identity block of dimension $n_i \times n_i$, for $j = i$.

Definition 1 A point $z \in \operatorname{dom} f$ is a *coordinate-wise minimum point* of f, if

$$f(z + E_i\xi_i) \geq f(z), \quad \forall \xi_i \in \mathbb{R}^{n_i},\ i = 1, \ldots, n;$$

it is a *minimum point* of f, if

$$f(z + w) \geq f(z), \quad \forall w \in \mathbb{R}^N.$$

For convex f, it is well known that $z \in \operatorname{dom} f$ is a minimum point of f if and only if

$$f'(z, v) \geq 0, \quad \forall v,$$

where $f'(z, v)$ is the directional derivative of f at z along direction v (we recall that the directional derivative of convex f exists at each $x \in \operatorname{int} \operatorname{dom} f$, even if f does not admit a standard gradient):

$$f'(z, v) = \lim_{\lambda \to 0_+} \frac{f(z + \lambda v) - f(z)}{\lambda} = \max_{g \in \partial f(x)} v^\top g,$$

where $\partial f(x)$ is the subdifferential of f at x. Also, $z \in \operatorname{dom} f$ is a coordinate-wise minimum point of f if and only if

$$f'(z, E_i v_i) \geq 0, \quad \forall v_i,\ i = 1, \ldots, n.$$

Clearly, if z is a minimum point of f, it is also a coordinate-wise minimum point. The converse is not true, in general. A relevant exception is discussed in the next section.

2.1 Differentiable plus separable structure

Assumption 1 *We assume that f has the following form:*

$$f(x) = \psi(x) + \phi(x), \tag{3}$$

where ψ is convex and differentiable on $\operatorname{dom} f$ *(which is assumed to be open), and*

$$\phi(x) = \sum_{i=1}^{n} \varphi_i(x_i),$$

where φ_i are convex (but possibly non differentiable) functions. Moreover, we assume that f is bounded below, and attains its minimum value f^*.

The following fact holds.

Proposition 1 *Let f satisfy Assumption 1, and suppose z is a coordinate-wise minimum point for f. Then, z is a minimum point of f.*

Proof. If z is a coordinate-wise minimum point for f, then, for all $v_i \in \mathbb{R}^{n_i}$ and all $i = 1, \ldots, n$, it holds that $f'(z, E_i v_i) \geq 0$. We thus have that

$$\begin{aligned} f'(z, v) &= \nabla\psi(z)^\top v + \sum_i \varphi_i'(z_i, v_i) \qquad (4) \\ &= \sum_i \nabla_i \psi(z)^\top v_i + \varphi_i'(z_i, v_i) \\ &= \sum_i f'(z, E_i v_i) \geq 0, \end{aligned}$$

which permits to conclude that z is a minimum point of f (in the above, $\nabla_i\psi(z)$ denotes the block of the gradient of ψ relative to i-th variable block).

We remark incidentally that a similar type of result actually holds for stationary points of more general function classes, see, e.g., Section 3 in [17]. □

2.1.1 Examples

Many problems of practical interest, especially those arising in the context of regularized loss minimization in machine learning, do satisfy Assumption 1. Notable examples (with scalar blocks) are the Lasso problem (see, e.g., [16]), for which $\psi(x) = \|y - Ax\|_2^2$, $\varphi_i(x_i) = \lambda|x_i|$, $i = 1, \ldots, n$; and the logistic regression problem (see [10]), where $\psi(x) = \sum_i \log\left(1 + \mathrm{e}^{-y_i a_i^\top x}\right)$, and $\varphi_i(x_i)$ are as in the Lasso. Similarly, examples of problems with non-scalar blocks are the so-called "group Lasso" (see [18]), where

$$\psi(x) = \|y - \sum_{i=1}^{n} A_i x_i\|_2^2, \quad \varphi_i(x_i) = \lambda\|x_i\|_2, \qquad (5)$$

being x_i, $i = 1, \dots, n$, blocks of variables of dimension $n_i \geq 1$; the sparse group Lasso (see [15]), where $\varphi_i(x_i) = \lambda_1 \|x_i\|_2 + \lambda_2 \|x_i\|_1$; and the group logistic regression (see [8]). Also, a group version of the Elastic Net problem (see [19, 3]) is obtained for

$$\psi(x) = \|y - \sum_{i=1}^{n} A_i x_i\|_2^2, \quad \varphi_i(x_i) = \lambda_1 \|x_i\|_2^2 + \lambda_2 \|x_i\|_2, \tag{6}$$

in which the special case with $\lambda_2 = 0$ yields the block Ridge regression (or Tikhonov regularized least-squares) problem.

3 Parallel block coordinate minimization

We describe in the following subsections the proposed parallel block-coordinate minimization method. For $x \in \operatorname{dom} f$, define

$$f_i(\nu; x) \doteq f(x + E_i \nu), \ \nu \in \mathbb{R}^{n_i}, \quad i = 1, \dots, n, \tag{7}$$

and

$$f_i^*(x) \doteq \min_{t \in \mathbb{R}^{n_i}} f_i(t - x_i; x), \quad i = 1, \dots, n. \tag{8}$$

Assuming the minimum is attained, we let ξ_i be a corresponding minimizer, i.e.,

$$\xi_i \in \arg \min_{t \in \mathbb{R}^{n_i}} f_i(t - x_i; x), \quad i = 1, \dots, n,$$

whence $f(x + E_i(\xi_i - x_i)) = f_i^*(x)$. Vector $\xi \doteq (\xi_1, \dots, \xi_n)$ is thus the result of application of a full parallel coordinate minimization step on $f(x)$. Observe that while, by definition,

$$f_i^*(x) = f(x + E_i(\xi_i - x_i)) \leq f(x + E_i(t - x_i)), \quad \forall t,$$

hence $f(x + E_i(\xi_i - x_i)) \leq f(x)$, it does not hold in general that $f(\xi) \leq f(x)$; thus a full parallel coordinate minimization step may not decrease f.

3.1 Averaged parallel descent

Let $x(k) \in \operatorname{dom} f$ be a current point, and let $\xi(k)$ be the point obtained from a full parallel minimization step from $x(k)$. Define

$$v(k) \doteq \xi(k) - x(k),$$

and let $\theta_i(k)$, $i = 1, \ldots, n$, be any weights satisfying the following properties:

$$\begin{array}{l} \theta_i(k) \in [0,\, 1], \quad \sum_i \theta_i(k) = 1, \\ f(x(k)) - f_i^*(x(k)) > 0 \;\Rightarrow\; \theta_i(k) > 0 \end{array}, \quad \forall k = 0, 1, 2, \ldots, \text{ and for } k \to \infty. \tag{9}$$

Consider an updated point of the form

$$x(k+1) \;=\; \sum_i \theta_i(k)(x(k) + E_i v_i(k)) = x(k) + \sum_i \theta_i(k) E_i v_i(k). \tag{10}$$

Since f is convex, from Jensen's inequality, we have

$$\begin{aligned} f(x(k+1)) &\leq \sum_i \theta_i(k) f(x(k) + E_i v_i(k)) \\ &= \sum_i \theta_i(k) f_i^*(x(k)) \\ &= f(x(k)) - \sum_i \theta_i(k) \left(f(x(k)) - f_i^*(x(k)) \right) \qquad (11) \\ &\leq f(x(k)), \end{aligned}$$

since $\theta_i(k) \geq 0$, and $f(x(k)) - f_i^*(x(k)) \geq 0$ for all i. The following proposition holds.

Proposition 2 *Under Assumption 1 and weights $\theta_i(k)$ satisfying (9), the iterates produced by (10), initialized with any $x(0) \in \operatorname{dom} f$, converge to a minimum point of f.*

Proof. Let $f^{(k)} \doteq f(x(k))$, $k = 0, 1, \ldots$, for some initial point $x(0) \in \operatorname{dom} f$. Since $f^{(0)} < \infty$ and $f^{(k+1)} \leq f^{(k)}$, the sequence $\{f^{(k)}\}$ generated by (10) is non-increasing. Further, since f is bounded below, the sequence converges to some limit $\bar{f}$, that is $\lim_{k\to\infty} f^{(k)} = \bar{f}$, and $\lim_{k\to\infty} f^{(k+1)} - f^{(k)} = 0$. In turn, from (11), this implies that

$$\lim_{k\to\infty} -\sum_i \theta_i(k)(f(x(k)) - f_i^*(x(k))) \geq 0.$$

Since, from (9), the $\theta_i(k)$ are nonnegative and can be zero only if $f(x(k)) - f_i^*(x(k)) = 0$, from the previous inequality it follows that

$$\lim_{k\to\infty} f(x(k)) - f_i^*(x(k)) = 0, \quad i = 1, \ldots, n,$$

i.e., $x(k)$ tends to a coordinate-wise minimum point of f, which, by Proposition 1, is also a minimum point of f. $\square$

3.1.1 Choice of weights

The standard choice for the weights is $\theta_i(k) = 1/n$, for which we obtain iterates of the form

$$x(k+1) = x(k) + \frac{1}{n}\sum_i E_i v_i(k) = x(k) + \frac{1}{n}v(k) = x(k) + \frac{1}{n}(\xi(k) - x(k)).$$

Such iterates can be interpreted as standard descent steps, where $v(k) = \xi(k) - x(k)$ is the descent direction, and $s = 1/n$ is a fixed step size.

Other choices for $\theta_i(k)$ are also possible, for instance assigning larger weights to blocks along which a larger decrease of the objective is observed. Letting

$$\Delta_i(k) \doteq f(x(k)) - f_i^*(x(k)), \quad i = 1, \ldots, n,$$

we may for instance set

$$\theta_i(k) = \frac{1 + \Delta_i(k)}{n + \sum_j \Delta_j(k)}, \quad i = 1, \ldots, n. \tag{12}$$

3.2 Parallel descent with variable step sizes

One problem with the parallel method described in the previous section is that, irrespective of the choice of the weights, the block updates $v_i(k) = \xi_i(k) - x_i(k)$ are *averaged* over $i = 1, \ldots, n$, i.e., they are multiplied by weights $\theta_i(k)$ that decrease as n increases, and this may slow down convergence in the case of large n. We hence propose next a variant of the method whereby a variable step size is assigned according to a simple backtracking rule. We modify the update rule (10) as follows

$$\begin{aligned} x(k+1) &= (1 - ns_k)x(k) + ns_k \sum_i \theta_i(k)(x(k) + E_i v_i(k)) \\ &= x(k) + ns_k \sum_i \theta_i(k) E_i v_i(k) \\ &= x(k) + s_k w(k), \end{aligned} \tag{13}$$

where

$$w(k) \doteq n \sum_i \theta_i(k) E_i v_i(k),$$

$s_k \in (0, 1]$ is a step size, and $\theta_i(k)$ are weights satisfying (9). We next show that the search direction $w(k)$ is a descent direction for f at $x(k)$. Indeed, for $s_k \leq 1/n$, $x(k+1)$ is a convex combination of points, and we have again from Jensen's inequality that

$$\begin{aligned} f(x(k+1)) &\leq (1-ns_k)f(x(k)) + ns_k \sum_i \theta_i(k) f(x(k) + E_i v_i(k)) \\ &= (1-ns_k)f(x(k)) + ns_k \sum_i \theta_i(k) f_i^*(x(k)) \\ &= f(x(k)) - ns_k \sum_i \theta_i(k)(f(x(k)) - f_i^*(x(k))). \end{aligned} \tag{14}$$

Thus, for $s_k \leq 1/n$, it holds that

$$\frac{f(x(k) + s_k w(k)) - f(x(k))}{s_k} \leq -n \sum_i \theta_i(k)(f(x(k)) - f_i^*(x(k))) \leq 0,$$

whence we have for the directional derivative that

$$\begin{aligned} f'(x(k), w(k)) &= \lim_{s \to 0_+} \frac{f(x(k) + sw(k)) - f(x(k))}{s} \\ &\leq -n \sum_i \theta_i(k)(f(x(k)) - f_i^*(x(k))) \leq 0, \end{aligned} \tag{15}$$

which shows that $w(k)$ is indeed a non-increasing direction for f at $x(k)$. Moreover, $f'(x(k), w(k)) < 0$, unless $x(k)$ is a coordinate-wise minimum point for f, thus $w(k)$ is a direction of descent.

3.2.1 Backtracking

Clearly, setting $s_k = 1/n$ in (13) we recover the update (10). The point is however to try larger step sizes, in a backtracking fashion, until an acceptable rate of decrease of the objective is achieved. We simply start the backtracking procedure with $s = 1$, and iteratively reduce it via $s \leftarrow \beta s$, where $\beta \in (0, 1)$ is some given parameter, until we find a value such that

$$f(x(k) + sw(k)) \leq f(x(k)) - ns \sum_i \theta_i(k)(f(x(k)) - f_i^*(x(k))).$$

We are guaranteed by Eq. (14) that this backtracking procedure will terminate with a step size value no smaller than $1/n$, hence it will hold for the final step size that $sn \geq 1$, thus

$$f(x(k) + sw(k)) \leq f(x(k)) - \sum_i \theta_i(k)(f(x(k)) - f_i^*(x(k))). \tag{16}$$

The following proposition holds.

Proposition 3 *Under Assumption 1 and weights $\theta_i(k)$ satisfying (9), the iterates produced by (13), initialized with any $x(0) \in \operatorname{dom} f$, and with step sizes obtained via the described backtracking procedure, converge to a minimum point of f.*

Proof. The proof follows the same line as the proof of Proposition 2: equation (16) ensures that the sequence $\{f^{(k)}\}$ is nonincreasing and, since it is bounded below, it reaches some limit, hence

$$0 = \lim_{k\to\infty} f^{(k+1)} - f^{(k)} \leq \lim_{k\to\infty} -\sum_i \theta_i(k)(f(x(k)) - f_i^*(x(k))).$$

Since $\theta_i(k)$ satisfy (9), it follows that

$$\lim_{k\to\infty} f(x(k)) - f_i^*(x(k)) = 0, \quad i = 1, \dots, n,$$

i.e., $x(k)$ tends to a coordinate-wise minimum point of f, which, by Proposition 1, is also a minimum point of f. $\square$

3.2.2 Parallel block coordinate minimization algorithm

We next summarize the described algorithm for parallel block coordinate minimization, under the standard setting with $\theta_i(k) = 1/n$, for $i = 1, \dots, n$.

1. (Initialization) Given $x(0) \in \operatorname{dom} f$, let $\mathtt{x} \leftarrow x(0)$, $\mathtt{f} \leftarrow f(x(0))$. Choose backtracking constant $\beta \in (0, 1)$.
2. Solve (possibly in parallel) the block coordinate minimization problems in (8) at current $\mathtt{x}$, and let $\mathtt{f}_i^*$ be the corresponding optimal values and ξ_i the minimizers, for $i = 1, \dots, n$. Set $\mathtt{v} \leftarrow \xi - \mathtt{x}$, $\Delta_i \leftarrow \mathtt{f} - \mathtt{f}_i^*$, $i = 1, \dots, n$.
3. Assign weights θ_i, $i = 1, \dots, n$ (in a standard implementation, we shall assume uniform weights $\theta_i = 1/n$), and compute descent direction $\mathtt{w} \leftarrow n \sum_i \theta_i E_i \mathtt{v}_i$.
4. (Backtracking)
   (a) Set $s \leftarrow 1$, and let $\eta = -n \sum_i \theta_i \Delta_i$.
   (b) If $s < 1/n$, set $s \leftarrow 1/n$ and exit backtracking procedure.
   (c) If $f(\mathtt{x} + s\mathtt{w}) \leq \mathtt{f} + s\eta$ exit backtracking procedure, else set $s \leftarrow \beta s$ and goto (b).

5. (Update) Set $\mathtt{x} \leftarrow \mathtt{x} + s\mathtt{w}$, $\mathtt{f} \leftarrow f(\mathtt{x})$. If stopping criterion is met, exit, else goto 2.

As a stopping criterion in the algorithm one can employ a standard check on minimal relative improvement on the objective function, or one on the deviation $\max_i \Delta_i$.

4 Rate of convergence for smooth and strongly convex f

The previous analysis showed that the algorithm proposed in Section 3.2.2 converges to a minimum of f, under the general setting of Assumption 1. In this section we develop an analysis of the rate of convergence, under a different assumption, as stated next.

Assumption 2 *Let f be twice differentiable and strongly convex on* $\operatorname{dom} f$.

The following result holds.

Proposition 4 *Let Assumption 2 be satisfied, let f^* denote the optimal value of f, and consider the sequence of points $x(k)$, $k = 0, 1, \ldots$, generated by the algorithm in Section 3.2.2. Then, for any $x(0) \in \operatorname{dom} f$ there exist a constant $c < 1$ such that*

$$f(x(k)) - f^* \leq c^k \left(f(x(0)) - f^*\right), \quad k = 1, 2, \ldots$$

Proof. For any given initial point $x(0) \in \operatorname{dom} f$, define

$$S_0 \doteq \{x \in \mathbb{R}^n : f(x) \leq f(x(0))\}.$$

The following facts are quite standard (see, e.g., Section 12.1.2.2 in [2]): under Assumption 2, S_0 is compact and there exist constants $M \geq m > 0$ (possibly depending on $x(0)$) such that, for all $x, y \in S_0$,

$$f(y) \leq f(x) + \nabla f(x)^\top (y - x) + \frac{M}{2}\|y - x\|_2^2 \tag{17}$$

$$f(y) \geq f(x) + \nabla f(x)^\top (y - x) + \frac{m}{2}\|y - x\|_2^2. \tag{18}$$

Further, for all $x \in S_0$ it holds that

$$f(x) - f^* \leq \frac{1}{2m}\|\nabla f(x)\|_2^2, \tag{19}$$

where $f^* = f(x^*)$ is the minimum value of f, and x^* is the corresponding (unique) minimizer.

We then observe that, since the algorithm in Section 3.2.2 is a descent algorithm, it holds that $x(k) \in S_0$ for all $k = 0, 1, \ldots$, and $x^* \in S_0$. Using (17) with $y = E_i h_i$, for $h_i \in \mathbb{R}^{n_i}$, we have that

$$f(x + E_i h_i) \leq f(x) + \nabla_i f(x)^\top h_i + \frac{M}{2}\|h_i\|_2^2, \quad \forall x, x + E_i h_i \in S_0.$$

Minimizing both sides of this equation we obtain

$$f_i^*(x) = \min_{h_i} f(x + E_i h_i) \leq \min_{h_i} f(x) + \nabla_i f(x)^\top h_i + \frac{M}{2}\|h_i\|_2^2,$$

where the minimum of the right side is $f(x) - \frac{1}{2M}\|\nabla_i f(x)\|_2^2$, which is attained at $h_i = -\frac{1}{M}\nabla_i f(x)$, therefore

$$f_i^*(x) = \min_{h_i} f(x + E_i h_i) \leq f(x) - \frac{1}{2M}\|\nabla_i f(x)\|_2^2, \quad \forall x \in S_0. \tag{20}$$

From (16), with $\theta_i(k) = 1/n$, we have that

$$f(x(k+1)) \leq f(x(k)) - \frac{1}{n}\sum_i \left(f(x(k)) - f_i^*(x(k))\right),$$

and, using (20),

$$f(x(k+1)) \leq f(x(k)) - \frac{1}{2nM}\sum_i \|\nabla_i f(x)\|_2^2 = f(x(k)) - \frac{1}{2nM}\|\nabla f(x)\|_2^2.$$

Then, using (19), we obtain that

$$\begin{aligned} f(x(k+1)) - f^* &\leq (f(x(k)) - f^*) - \frac{1}{2nM}\|\nabla f(x)\|_2^2 \\ &\leq (f(x(k)) - f^*) + \frac{m}{nM}(f^* - f(x(k))) \\ &= \left(1 - \frac{m}{nM}\right)(f(x(k)) - f^*). \end{aligned}$$

Letting $c \doteq 1 - \frac{m}{nM} < 1$, and $d(k) \doteq f(x(k)) - f^*$, it follows from the above equation that

$$d(k) \leq c^k d(0), \quad k = 1, 2, \ldots,$$

which proves the claim. $\square$

5 Application to group Lasso and Ridge regression

Consider the following regularized regression problem

$$\min_x f(x) \doteq \frac{1}{2}\|y - \sum_{i=1}^{n} A_i x_i\|_2^2 + \lambda \sum_{i=1}^{n} \|x_i\|_2^{\nu}, \tag{21}$$

where $A_i \in \mathbb{R}^{m,n_i}$ are given regressor matrices, $y \in \mathbb{R}^m$ is a given vector, $\lambda \geq 0$ is a regularization parameter, and either $\nu = 1$ or $\nu = 2$. For $\nu = 1$ we have a group Lasso problem, while for $\nu = 2$ we have a group Ridge regression problem. For $\nu = 1$ the objective function f satisfies Assumption 1, while for $\nu = 2$ it also satisfies Assumption 2. For $\nu = 1$, similar to the standard Lasso, the purpose of the regularization term $\lambda \sum_{i=1}^{n} \|x_i\|_2$ is to promote group sparsity in the solution, that is to obtain solutions in which many of the blocks x_i are zero; for $\nu = 2$ the regularization terms "shrinks" the solution towards zero, improving the numerical conditioning of the problem.

A usual approach for solving (21) is a sequential block minimization one, see, e.g., [15, 18]. In Section 6 we shall compare the performance of the standard sequential block method with the one obtained from the parallel block method described in this paper. We next give the computational details needed for application of the serial or parallel block coordinate minimization method to (21).

5.1 Solution of the block subproblem

The minimization problem with respect to a single block j has the form

$$\min_{x_j} f_j(x_j) \doteq \frac{1}{2}\|y^{(j)} - A_j x_j\|_2^2 + \lambda \|x_j\|_2^{\nu}, \tag{22}$$

where $y^{(j)} \doteq y - \sum_{i \neq j}^{n} A_i x_i$. Subproblem (22) can be solved for the block x_j as we describe next, for the two cases of $\nu = 2$ and $\nu = 1$.
Case $\nu = 2$**:** in this case f is smooth, and $\nabla f_j(x_j) = A_j^\top (A_j x_j - y^{(j)}) + 2\lambda x_j$. The optimal solution for the block satisfies $\nabla f_j(x_j) = 0$, whence

$$x_j = \left(A_j^\top A_j + 2\lambda I\right)^{-1} A_j^\top y^{(j)}. \tag{23}$$

Case $\nu = 1$**:** in this case f is non-differentiable. However, the subdifferential of f_j at x_j is

$$\partial f_j(x_j) = A_j^\top (A_j x_j - y^{(j)}) + \lambda \begin{cases} x_j / \|x_j\|_2 & \text{if } x_j \neq 0 \\ g \in \mathbb{R}^{n_j} : \|g\|_2 \leq 1 & \text{if } x_j = 0 \end{cases}$$

Thus, $x_j = 0$ is an optimal solution to (22) if and only if $0 \in \partial f_j(0)$, that is

$$A_j^\top y^{(j)} + \lambda g = 0, \quad \text{for some } g : \|g\|_2 \leq 1.$$

This latter condition is equivalent to $\|A_j^\top y^{(j)}\|_2 \leq \lambda$, thus

$$x_j = 0 \text{ is optimal} \quad \Leftrightarrow \quad \|A_j^\top y^{(j)}\|_2 \leq \lambda.$$

Consider then the case $\|A_j^\top y^{(j)}\|_2 > \lambda$. The optimal solution (which is nonzero) is characterized by the equation

$$A_j^\top A_j x_j + \lambda \frac{x_j}{\|x_j\|_2} = A_j^\top y^{(j)}, \tag{24}$$

thus

$$x_j = \left(A_j^\top A_j + \frac{\lambda}{\alpha} I \right)^{-1} A_j^\top y^{(j)},$$

where we let $\alpha \doteq \|x_j\|_2$. Taking the norm of both sides in the latter expression, we see that α must satisfy the scalar equation

$$\|(\alpha A_j^\top A_j + \lambda I)^{-1} A_j^\top y^{(j)}\|_2 = 1. \tag{25}$$

This scalar equation is readily solved for $\alpha > 0$ by any suitable numerical method, such as the Newton-Raphson method described below; once α is computed, it is plugged into (24) and then the desired x_j is obtained by solving this system of linear equations.

We remark that a special case of the present problem is the one discussed in [18], where it is assumed that $A_j^\top A_j = \sigma_j I$, $j = 1, \ldots, n$. In such case, α can be found in closed form as $\alpha = \frac{\|A_j^\top y^{(j)}\|_2 - \lambda}{\sigma_j}$, and the optimal x_j is $x_j = \frac{\|A_j^\top y^{(j)}\|_2 - \lambda}{\sigma_j \|A_j^\top y^{(j)}\|_2} A_j^\top y^{(j)}$. Also, we acknowledge that a method for the block subproblem similar to the one described here has been developed previously and independently in [11].

5.1.1 Newton-Raphson iterations for solving (25)

Let $\phi(\alpha) \doteq \|(\alpha A_j^\top A_j + \lambda I)^{-1} A_j^\top y^{(j)}\|_2^2 - 1$; we next describe Newton-Raphson (see, e.g., [5]) iterations for finding a positive root of the equation $\phi(\alpha) = 0$. This appears to be a convenient method for finding the desired root, since a "closed-form" expression for the derivative of ϕ with respect to α can be determined as follows. Define $B(\alpha) \doteq Q(\alpha)^2$, with $Q(\alpha) \doteq \alpha A_j^\top A_j + \lambda I$. By the rules of matrix derivatives (see, e.g., [7]), we have that

$$\frac{\partial B(\alpha)^{-1}}{\partial \alpha} = -B(\alpha)^{-1} \frac{\partial B(\alpha)}{\partial \alpha} B(\alpha)^{-1},$$

and

$$\frac{\partial B(\alpha)}{\partial \alpha} = \frac{\partial Q(\alpha)^2}{\partial \alpha} = 2Q(\alpha) A_j^\top A_j.$$

Therefore, we have

$$\begin{aligned} \phi'(\alpha) \doteq \frac{\mathrm{d}\phi(\alpha)}{\mathrm{d}\alpha} &= y^{(j)\top} A_j \frac{\partial B(\alpha)^{-1}}{\partial \alpha} A_j^\top y^{(j)} \\ &= -2y^{(j)\top} A_j Q(\alpha)^{-2} Q(\alpha) A_j^\top A_j Q(\alpha)^{-2} A_j^\top y^{(j)} \\ &= -2y^{(j)\top} A_j Q(\alpha)^{-1} A_j^\top A_j Q(\alpha)^{-2} A_j^\top y^{(j)} \\ &= -2y^{(j)\top} A_j A_j^\top A_j Q(\alpha)^{-3} A_j^\top y^{(j)}, \end{aligned}$$

where the last passage follows from the fact that $Q(\alpha)^{-1}$ commutes in the product with $A_j^\top A_j$. Observe that $Q(\alpha)^{-1}$ can be readily evaluated for different values of α, once an eigenvalue factorization $U\Sigma U^\top$ (U orthogonal, Σ diagonal and positive) is computed for the symmetric matrix $A_j^\top A_j$.

We thus start the iterations with the value $\alpha = 0$, for which we have $\phi(0) = \|A_j^\top y^{(j)}\|_2^2/\lambda^2 - 1 > 0$, and $\phi'(0) = -\frac{2}{\lambda^3} y^{(j)\top} A_j (A_j^\top A_j) A_j^\top y^{(j)} < 0$, and we iteratively update the α value according to $\alpha \leftarrow \alpha - \frac{\phi(\alpha)}{\phi'(\alpha)}$, until convergence is reached to a desired precision.

6 Numerical test on regularized regression problems

We next present the results of some numerical tests we conducted by comparing the performance of the serial block minimization method and the proposed parallel block minimization method on the group Lasso and the group Ridge regression problems. In the following, we intend by "iteration" either a full serial sweep over the n blocks (in the serial method), or a full parallel pass over the n blocks, followed by variable update (in the parallel method). Therefore, in the serial method the computational effort per iteration is essentially n times the effort of minimization over a single block. In the parallel method, the computational effort per iteration is n times the effort of minimization over a single block, plus the effort of "coordination," due to computation of the descent direction w and backtracking for determining the step size s. The first term is the one who benefits from parallel implementation. In order to compare the computational times of the two algorithms, we used the following indices as proxies for evaluating approximately the performance of the two methods. Specifically, we defined the time for reaching a solution (to a desired accuracy) for the parallel method as

$$T_p = \frac{1}{n_w} \sum_{k=1}^{N_p} T_p(k) + \sum_{k=1}^{N_p} T_{\text{coord}}(k),$$

where N_p is the number of iterations needed to reach convergence in a run of the parallel method, $T_p(k)$ is the time needed for completing all the n block minimization at k-th iteration (which, in the actual test, are run serially on a single worker), n_w is the number of "workers" available in an ideal parallel computing environment, and $T_{\text{coord}}(k)$ is the time spent for variable update at the k-th iteration. Similarly, we defined the time for reaching a solution (to the same desired accuracy) for the serial method as

$$T_s = \sum_{k=1}^{N_s} T_s(k),$$

where N_s is the number of iterations needed to reach convergence in a run of the serial method, and $T_s(k)$ is the time needed for completing the k-th iteration.

In each of the following numerical tests, we generated $N_{\text{test}} = 100$ random problem instances, each involving $n = 100$ blocks with matrices A_i of size $m \times n_i$ each, with $m = 50$, $n_i = 50$, $i = 1, \ldots, n$. Both the entries in the A_i matrices and in vector y were generated in each problem instance as independent standard Normal random variables. We exited the algorithms when the relative improvement in the objective from one iteration to the previous one was below a threshold of 10^{-6}. The backtracking parameter was set to $\beta = 0.8$.

6.1 Ridge regression problem

The first set of experiments is performed on the group Ridge regression problem, i.e., on problem (21) with $\nu = 2$. The regularization parameter was set to $\lambda = 20$. Function f in this case is smooth and strongly convex, hence a linear convergence rate is guaranteed by Proposition 4.

Figure 1 shows histograms of the number of iterations needed for each of the N_{test} runs, for the serial algorithm (left) and for the parallel algorithm (right). The average number of iterations was 1205 for the serial method and 132 for the parallel one.

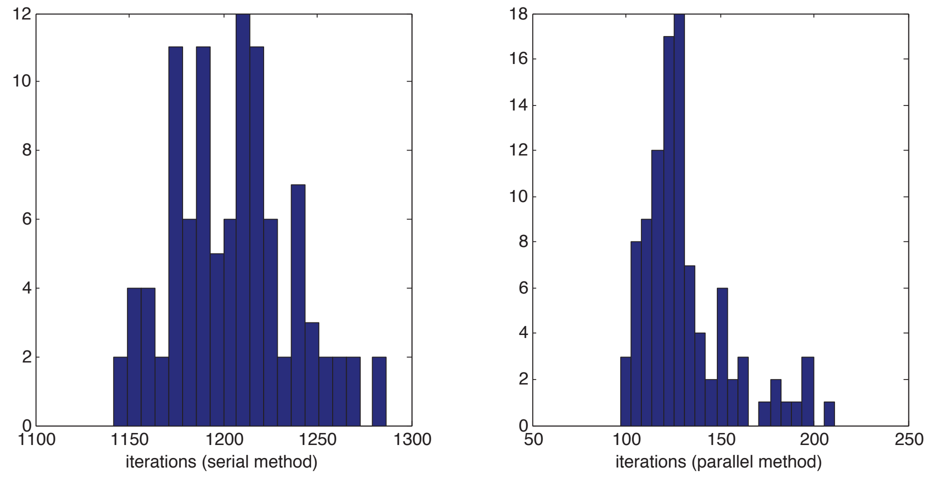

Figure 1: Histograms of iterations required to reach solution in N_{test} random instances, for the serial (left) and the parallel (right) method — Ridge example.

For each run of the parallel method, we recorded the step sizes used in each iteration, and then evaluated the average step size used in each run. Figure 2 shows the average step sizes obtained in each of the $N_{\text{test}} = 100$ random tests. These are much larger than the minimal step size $1/n = 0.01$, and indeed the parallel algorithm has been observed to often perform "full" steps, with $s = 1$.

We then compared the execution times, in terms of the T_s and T_p performance indices defined before. Tests were run under Matlab on a Window 7 workstation with a Xeon X5650 2.67 GHz CPU. Notice that Matlab was run in single worker mode, and that no particular effort was done in optimizing the codes for maximal efficiency; therefore attention should be given more to the relative figures between the serial and the parallel method rather than to the absolute values.

Assuming a virtual number of workers $n_w = 32$, the resulting computational times for the serial and the parallel methods are shown in the histograms of Figure 3. The average T_s was about 27 seconds for the serial method and the average T_p was about 0.6 seconds for the parallel one.

Finally, Figure 4 shows a comparison of the average plots for the relative

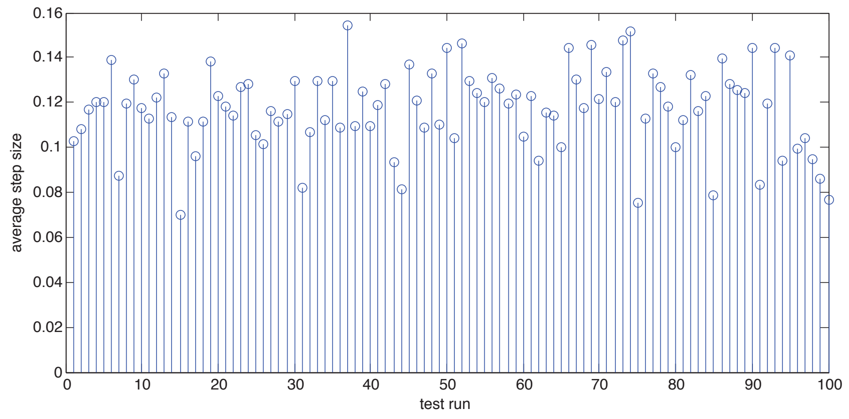

Figure 2: Average step sizes used in each of the N_{test} runs of the parallel algorithm — Ridge example.

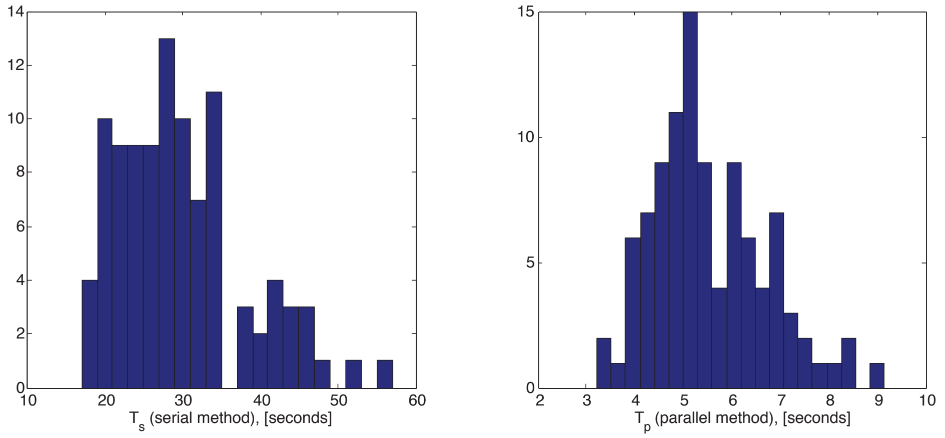

Figure 3: Histograms of total computation times T_s, T_p required to reach solution in each of the N_{test} random instances, for the serial (left) and the parallel (right) method (assuming $n_w = 32$ workers) — Ridge example.

rate of decrease $\delta(k)$ of the objective function for the serial and the parallel methods, where

$$\delta(k) \doteq \frac{f(x(k)) - f^*}{f(x(0)) - f^*}.$$

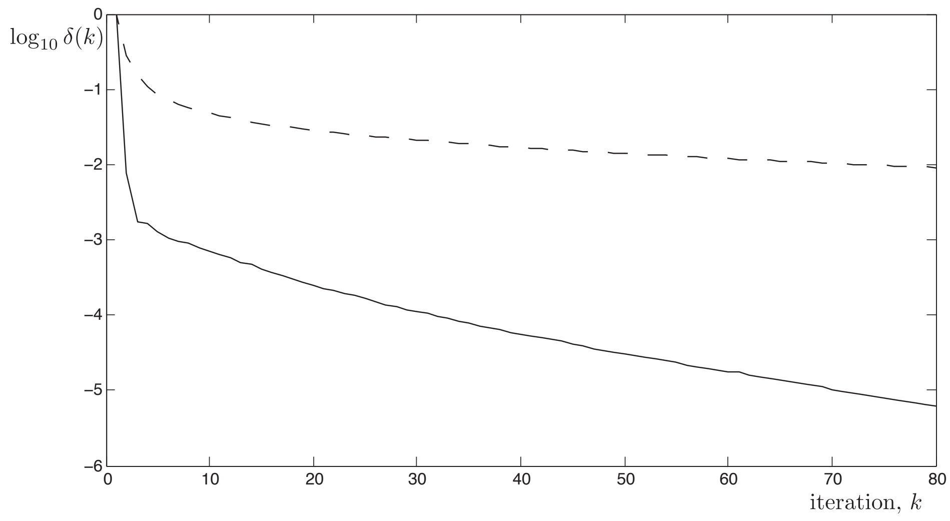

Figure 4: Relative reduction of objective value on the first 80 iterations of the serial (dashed line) and the parallel (solid line) methods (average over the N_{test} random runs) — Ridge example.

6.2 Group Lasso problem

The second set of experiments was performed on the group Lasso problem, i.e., on problem (21) with $\nu = 1$. Function f is nonsmooth in this case, hence the present theoretical analysis guarantees convergence (Proposition 3), albeit possibly not at a linear rate.

Figure 5 shows histograms of the number of iterations needed for each of the N_{test} runs, for the serial algorithm (left) and for the parallel algorithm (right). The average number of iterations was 618 for the serial method and 642 for the parallel one.

Figure 6 shows the average step sizes obtained in each of the $N_{\text{test}} = 100$ random tests.

Assuming a virtual number of workers $n_w = 32$, the resulting computational times for the serial and the parallel methods are shown in the histograms of Figure 7. The average T_s was about 57 seconds for the serial method and the average T_p was about 5.3 seconds for the parallel one. Finally, Figure 8 shows a comparison of the average plots for the relative rate of decrease $\delta(k)$ of the objective function for the serial and the parallel methods.

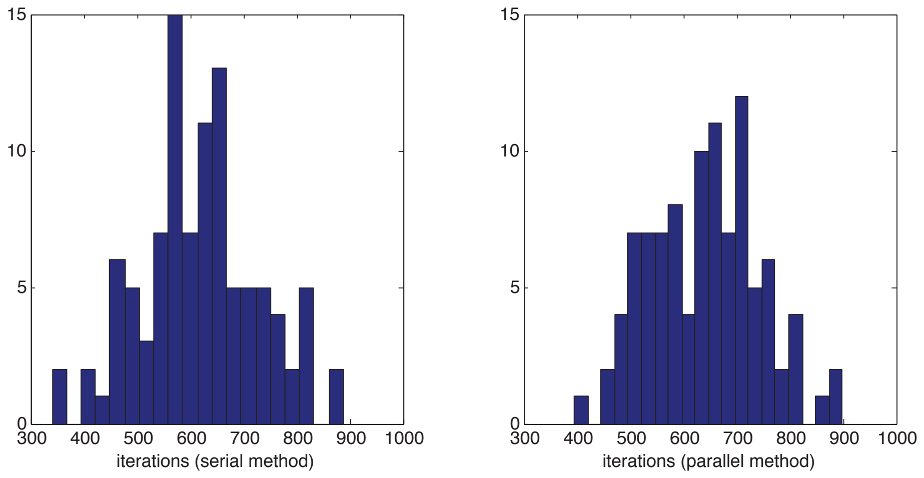

Figure 5: Histograms of iterations required to reach solution in N_{test} random instances, for the serial (left) and the parallel (right) method — group Lasso example.

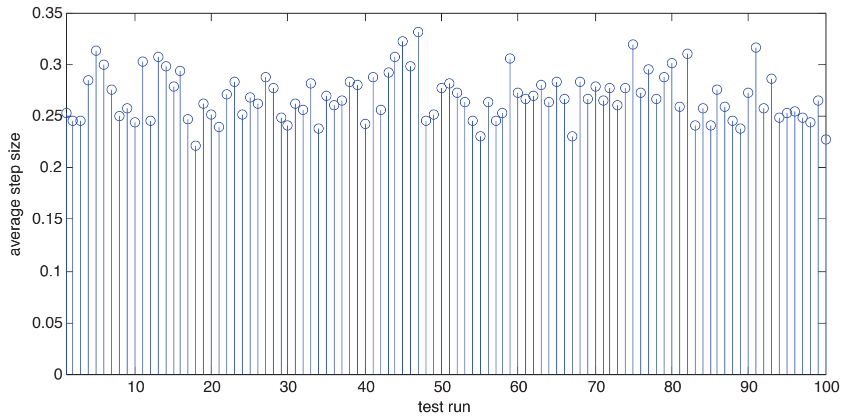

Figure 6: Average step sizes used in each of the N_{test} runs of the parallel algorithm — group Lasso example.

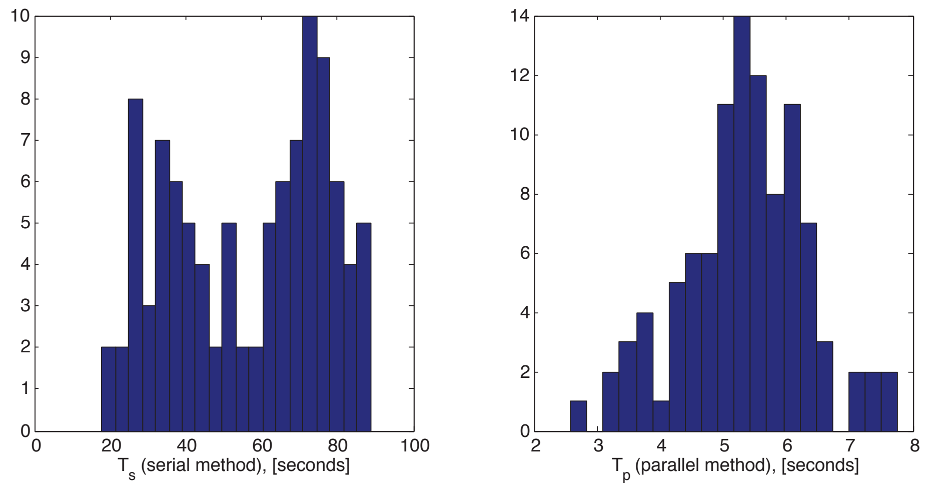

Figure 7: Histograms of total computation times T_s, T_p required to reach solution in each of the N_{test} random instances, for the serial (left) and the parallel (right) method (assuming $n_w = 32$ workers) — group Lasso example.

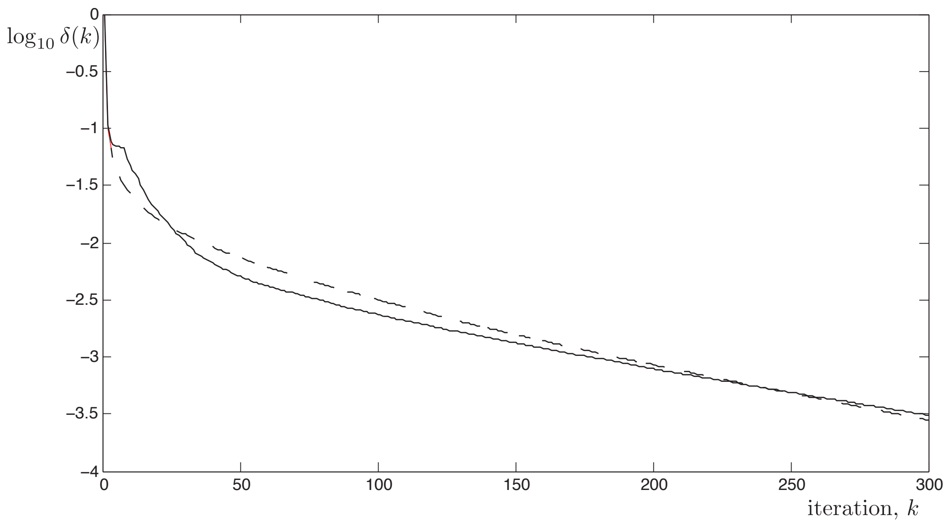

Figure 8: Relative reduction of objective value on the first 300 iterations of the serial (dashed line) and the parallel (solid line) methods (average over the N_{test} random runs) — group Lasso example.

All the previous tests have been performed using uniform weights $\theta_i(k) = 1/n$. We also tested the parallel algorithm using the non-uniform weights in (12). On this example's data, the use of non-uniform weights did not change significantly the average numerical performance of the algorithm; in particular, the average objective decrease curve resulted to be essentially identical to the solid line shown in Figure 8 for uniform weights.

7 Conclusions

In this paper, we presented a parallel version of the popular block coordinate minimization method. While a pure parallel (Jacobi-type) block minimization method fails to converge in general, we proved that convergence can be guaranteed if the parallel block minimization phase is followed by a suitable averaging and step-size selection phase, for the class of convex functions composed of a smooth part plus a possibly nonsmooth but separable part. Further, we proved that a linear convergence rate can be guaranteed, for smooth and strongly convex functions.

Numerical tests suggested that the number of iterations needed for reaching convergence is at most of the same order as the one needed in the standard serial implementation, and that substantial computational time improvement can ideally be gained by exploiting parallel workers.

Both the serial and this parallel method can however be unsuitable for Big Data and huge-scale problems, where the number n of blocks is very large, since both approaches require full sweeps over all blocks at each iteration. However, similar to what has been done in the serial case (see [12] and the references therein) and in the parallel case in [13], a randomized approach could be applied to our method, by iteratively randomly selecting $p \ll n$ blocks in each parallel minimization step. Analysis of this idea is subject to ongoing research.

References

[1] J.K. Bradley, A. Kyrola, D. Bickson, and C. Guestrin. Parallel coordinate descent for ℓ_1-regularized loss minimization. In *28th Int. Conf. on Machine Learning*, 2011.

[2] G. Calafiore and L. El Ghaoui. *Optimization Models*. Cambridge Univ. Press, 2014.

[3] C. De Mol, E. De Vito, and L. Rosasco. Elastic-net regularization in learning theory. *Journal of Complexity*, 25:201–230, 2009.

[4] J.N.Tsitsiklis and D. Bertsekas. *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, 1989.

[5] C.T. Kelley. *Solving Nonlinear Equations with Newton's Method*, volume 1 of *Fundamentals of Algorithms*. SIAM, 2003.

[6] Z.Q. Luo and P. Tseng. On the convergence of the coordinate descent method for convex differentiable minimization. *Journal of Optimization theory and applications*, 72(1):7–35, 1992.

[7] J.R. Magnus and H. Neudecker. *Matrix differential calculus with applications in statistics and econometrics*. John Wiley & Sons, 1988.

[8] L. Meier, S. van de Geer, and P. Bühlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society*, 70:53–71, 2008.

[9] Yu. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22:341–362, 2012.

[10] A.Y. Ng. Feature selection, ℓ_1 vs. ℓ_2 regularization, and rotational invariance. In *Proceedings of the 21st Int. Conf. on Machine Learning*, 2004.

[11] Z. Qin, K. Scheinberg, and D. Goldfarb. Efficient block-coordinate descent algorithms for the group lasso. *Mathematical Programming Computation*, 5(2):143–169, 2013.

[12] P. Richtárik and M. Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, pages 1–38, 2012.

[13] P. Richtárik and M. Takáč. Parallel coordinate descent methods for big data optimization. *ArXiv.org*, 1212.0873v2, 2013.

[14] A. Saha and A. Tewari. On the nonasymptotic convergence of cyclic coordinate descent methods. *SIAM Journal on Optimization*, 23(1):576–601, 2013.

[15] N. Simon, J. Friedman, T. Hastie, and R. Tibshirani. A sparse-group lasso. *Journal of Computational and Graphical Statistics*, 22(2):231–245, 2013.

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

[17] P. Tseng. Convergence of a block coordinate descent method for non-differentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494, 2001.

[18] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society*, 68(1):49–67, 2007.

[19] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *J. R. Statist. Soc. B*, 67(2):301–320, 2005.