Efficiency of Accelerated Coordinate Descent Method on Structured Optimization Problems

Yurii Nesterov and Sebastian Stich
Efficiency of Accelerated Coordinate Descent Method on Structured Optimization Problems

Yu. Nesterov* and S. Stich†

February 1, 2016

Abstract

In this paper we prove a new complexity bound for a variant of Accelerated Coordinate Descent Method [7]. We show that this method often outperforms the standard Fast Gradient Methods (FGM, [3, 6]) on optimization problems with dense data. In many important situations, the computational expenses of oracle and method itself at each iteration of our scheme are perfectly balanced (both depend linearly on dimensions of the problem). As application examples, we consider unconstrained convex quadratic minimization, and the problems arising in Smoothing Technique [6]. On some special problem instances, the provable acceleration factor with respect to FGM can reach the square root of the number of variables. Our theoretical conclusions are confirmed by numerical experiments.

Keywords: Convex optimization, structural optimization, fast gradient methods, coordinate descent methods, complexity bounds.

*Center for Operations Research and Econometrics (CORE), Catholic University of Louvain (UCL), 34 voie du Roman Pays, 1348 Louvain-la-Neuve, Belgium.
†CORE/ICTEAM (UCL). This research was supported by Swiss Science Foundation (SNF).

The research results presented in this paper have been supported by a grant “Action de recherche concertée ARC 04/09-315” from the “Direction de la recherche scientifique - Communauté française de Belgique”. Scientific responsibility rests with the authors.
1 Introduction

Motivation. In the last years, coordinate descent methods attract more and more attention of the Optimization Community. Its popularity is based mainly on the fact that they can be applied to problems of a very big size. Starting from the paper [7], it became possible to provide the randomized variants of these schemes with very attractive worst-case efficiency guarantees, which take into account a very high sparsity of the data. Consequently, the further developments of these methods were naturally related to the needs of Big-Data machinery: parallelization, distributed computing, etc (see, for example, [4, 5]). However, in this paper we show that the coordinate descent strategies can be useful even for the problems of moderate-size when the data is dense.

In [7], there was proposed a variant of Fast Gradient Method [3], where the gradient step was replaced by a step along coordinate direction (we call this method Accelerated Coordinate Descent Method, ACDM for short). It was suggested to choose the corresponding active coordinate randomly, in accordance to uniform distribution. The expected complexity of this scheme for finding an \( \epsilon \)-solution for unconstrained minimization problem is of the order

\[
O \left( \frac{n}{\epsilon^2} \max_{1 \leq i \leq n} L_i \right)
\]

iterations, where \( L_i \) is the uniform upper bound on the \( i \)th diagonal element of the Hessian of the objective function, and \( n \) is the number of variables. At the same time, in [7] it was also mentioned that this scheme is not appropriate for Huge-Scale optimization problems since it needs at least one full-dimensional vector operation at each iteration.

Complexity bound (1.1) was improved in [2] up to the level

\[
O \left( \left[ \frac{n}{\epsilon} \sum_{i=1}^{n} L_i \right]^{1/2} \right)
\]

iterations. For choosing the active coordinate, the authors suggest to use probabilities

\[ L_i \left[ \sum_{k=1}^{n} L_k \right]^{-1}, \; i = 1, \ldots, n. \]

Finally, in our paper we get the further improvement in the complexity of ACDM, up to the level

\[
O \left( \frac{1}{\epsilon^2} \sum_{i=1}^{n} L_i^{1/2} \right)
\]

iterations. The probabilities we use now are defined as \( L_i^{1/2} \left[ \sum_{k=1}^{n} L_k^{1/2} \right]^{-1} \). This is the first time when we get the complexity estimate of ACDM, which does not depend explicitly in the dimension of the space of variables.

Another important result of our paper consists in finding interesting applications, where the new scheme becomes dominant. We show that in all unconstrained convex optimization problems obtained by Smoothing Technique [6], our method provably outperforms the standard Fast Gradient Methods. For some classes of problems, the gain in the computational time reaches the square root of the dimension. This improvement is mainly achieved due to the fact, that in many situations the computational expenses at
each iteration of our method are perfectly balanced with the computational time spent for updating the results of matrix-vector products (both depend linearly in the dimension of the problem). For the standard first-order methods, this is not true even if we apply them for unconstrained minimization of convex quadratic function with dense matrix. For the latter problem, the worst-case estimates of computational time of our method are provably better than the estimates of unbeatable Conjugate Gradients.\(^1\) Note also that for problems with explicit minimax structure, it is always possible to compute good bounds for the constants \(L_i, i = 1, \ldots, n\) (see Section 3.3).

Contents. In Section 2, we present a new version of ACD-method for solving the problem of unconstrained minimization of strongly convex function with Lipschitz continuous partial derivatives. The probability of choosing component \(i\) to be active is define as 

\[
L_i^{1/2} \left[ \sum_{k=1}^n L_k^{1/2} \right]^{-1},
\]

where \(L_i\) is the corresponding Lipschitz constant. Our scheme, complexity analysis, and efficiency estimates are nonstandard since they all are continuous in the convexity parameter of the objective function. In order to obtain the efficiency estimates and the rules of the method just for differentiable convex function, we need to pass to the limit in the corresponding expressions, tending the convexity parameter to zero.\(^2\)

In Section 3, we present some applications, where the new method has the best known worst-case bounds for the total computational time. In Section 3.1 we develop a general model of the objective function, which allows to update and compute efficiently the directional derivatives. Our key observation is that in many cases a single directional derivative can be easily computed, often in linear time. After that, we analyze the behavior of the new ACDM on the problems of quadratic minimization (Section 3.2) and in the framework of Smoothing Technique (Section 3.3). In both cases, we show that our method has better worst-case guarantees in computational time, as compared with the total computational time of the standard FGM.

We conclude the paper by presenting the results of preliminary computational experiments (Section 4). At our class of test problems, new ACDM always outperforms the standard Fast Gradient Method with automatic adjustment of the Lipschitz constant for the gradient.

Notation. In what follows, we assume that the finite-dimensional linear vector space of variables \(E, \dim E = N\), is represented as a direct product of \(n\)-dimensional spaces \(E^{(i)}\), 

\[
\dim E^{(i)} = n_i,
\]

\[
E = \bigotimes_{i=1}^n E^{(i)}, \quad N = \sum_{i=1}^n n_i.
\]

We denote by \(E^*_i, i \in \{1: n\}\), the corresponding dual spaces. Thus, \(E_* = \bigotimes_{i=1}^n E^*_i\). Value

\(^1\) Of course, this result does not contradict to the well known fact on optimality of conjugate gradient methods. Note that coordinate descent methods belong to another family of optimization schemes, which do not generate minimization sequences belonging to Krylov spaces.

\(^2\) When this paper was already finished, we found a very recent paper [1], where there was analyzed a version of ACDM with the same distribution of probabilities. This version can be also used for minimizing strongly convex functions. However, it becomes inefficient as the convexity parameter goes to zero.
of linear function \( s^{(i)} \in E_s^{(i)} \) at point \( x^{(i)} \in E^{(i)} \) is denoted by \( \langle s^{(i)}, x^{(i)} \rangle \). We define
\[
\langle s, x \rangle \quad \text{def} \quad \sum_{i=1}^{n} \langle s^{(i)}, x^{(i)} \rangle, \quad x \in E, \ s \in E_s.
\]

We define also the partition operators \( U_i : E_i \to E, \ i = 1, \ldots, n \), by identity
\[
x = (x^{(1)}, \ldots, x^{(n)}) = \sum_{i=1}^{n} U_i x^{(i)}, \quad x^{(i)} \in E^{(i)}, \ i \in \{1 : n\}.
\]
If \( E = \mathbb{R}^N \), Then the matrices \( U_i \) are composed by columns of the unit \( N \times N \)-matrix:
\[
I_N = (U_1, \ldots, U_n).
\]

For a linear operator \( A \), acting from one linear vector space \( E' \) to another linear vector space \( E'' \), we define its adjoint operator by identity
\[
(\langle Au, v \rangle) = (A^* v, u), \quad u \in E', \ v \in E''.
\]

Clearly, \( A^* : E'' \to E' \).

For all spaces \( E^{(i)} \), we fix self-adjoint positive-definite operators \( B_i : E^{(i)} \to E_s^{(i)} \) (notation: \( B_i = B_i^* > 0 \), \( i = 1, \ldots, n \). Using these operators, we can introduce in these spaces the scalar products and Euclidean norms:
\[
\langle x^{(i)}, y^{(i)} \rangle_i \quad \text{def} \quad \langle B_i x^{(i)}, y^{(i)} \rangle, \quad ||x^{(i)}||^2_i \quad \text{def} \quad \langle B_i x^{(i)}, x^{(i)} \rangle, \quad x^{(i)}, y^{(i)} \in E^{(i)}, \ i \in \{1 : n\}.
\]

Similarly, for the dual spaces, we have the following definitions:
\[
\langle s^{(i)}, v^{(i)} \rangle_i^* \quad \text{def} \quad \langle s^{(i)}, B_i^{-1} v^{(i)} \rangle, \quad ||s^{(i)}||^*_i \quad \text{def} \quad \langle s^{(i)}, B_i^{-1} s^{(i)} \rangle, \quad s^{(i)}, v^{(i)} \in E_s^{(i)}, \ i \in \{1 : n\}.
\]

Thus, we get valid Cauchy-Schwartz inequalities:
\[
\langle s^{(i)}, x^{(i)} \rangle \leq ||s^{(i)}||^*_i \cdot ||x^{(i)}||_i, \quad x^{(i)} \in E^{(i)}, \ s^{(i)} \in E_s^{(i)}, \ i \in \{1 : n\}. \tag{1.4}
\]

In order to define the norms for the whole space \( E \), we use the scaling coefficients \( L = (L_1, \ldots, L_n) \) (to be defined later in (2.3)), and the tolerance parameter \( \alpha \in [0, 1] \). For \( x = (x^{(1)}, \ldots, x^{(n)}) \in E \) and \( s = (s^{(1)}, \ldots, s^{(n)}) \in E_s \) denote
\[
\langle s, x \rangle = \sum_{i=1}^{n} \langle s^{(i)}, x^{(i)} \rangle,
\]
\[
||x||^2_{[\alpha]} = \sum_{i=1}^{n} L_i^\alpha ||x^{(i)}||^2_i, \tag{1.5}
\]
\[
||s||^2_{[\alpha]*} = \sum_{i=1}^{n} L_i^{-\alpha} (||s^{(i)}||^*_i)^2.
\]

Clearly, for all \( x \in E \) and \( s \in E_s \) we have
\[
\langle s, x \rangle \leq ||x||_{[\alpha]} \cdot ||s||_{[\alpha]*}. \tag{1.6}
\]
In the case $E = \mathbb{R}^N$, we have $\|x\|_{[\alpha]}^2 = (B_\alpha x, x)$, $\|s\|_{[\alpha]}^2 = (s, B_\alpha^{-1}s)$, with

$$B_\alpha = \sum_{i=1}^n L_i^\alpha U_i B_i U_i^T, \quad B_\alpha^{-1} = \sum_{i=1}^n L_i^{-\alpha} U_i B_i^{-1} U_i^T.$$ 

For a differentiable function $f(x), x \in \text{dom} f \subseteq E$, denote by $\nabla f(x) \in E^*$ its gradient. Then, its partial derivatives are defined as follows:

$$\nabla_i f(x) \overset{\text{def}}{=} U_i^T - \nabla f(x) \in E_i^*, \quad i \in \{1:n\}.$$ 

If function $f$ is convex, then for any $x \in \text{dom} f$ and any partial displacement $h^{(i)} \in E^{(i)}$ satisfying condition $x + U_i h^{(i)} \in \text{dom} f$ (we call it feasible), we have

$$f(x + U_i h^{(i)}) \geq f(x) + \langle \nabla f(x), U_i h^{(i)} \rangle = f(x) + \langle U_i^T \nabla f(x), h^{(i)} \rangle$$

$$= f(x) + \langle \nabla_i f(x), h^{(i)} \rangle, \quad i \in \{1:n\}. \quad (1.7)$$

## 2 Accelerated Coordinate Descent Method

Consider the following optimization problem:

$$\min_{x \in E} f(x), \quad (2.1)$$

where function $f$ is convex and continuously differentiable on $E$. We assume that this problem is solvable and $x^* \in E$ is its optimal solution.

Global behavior of function $f(\cdot)$ is described by the following characteristics:

- **Parameter of strong convexity** $\sigma_\alpha \geq 0$, such that
  $$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{1}{2} \sigma_{1-\alpha} \|y - x\|_{[1-\alpha]}^2, \quad \forall x, y \in E. \quad (2.2)$$

- **Lipschitz constants** $L_i$ for partial derivatives:
  $$\|\nabla_i f(x + U_i h^{(i)}) - \nabla_i f(x)\|_i^* \leq L_i \|h^{(i)}\|_i, \quad \forall x \in E, h^{(i)} \in E^{(i)}, \quad i \in \{1:n\}. \quad (2.3)$$

These inequalities are equivalent to the following conditions:

$$f(x + U_i h^{(i)}) \leq f(x) + \langle \nabla_i f(x), h^{(i)} \rangle + \frac{1}{2} L_i \|h^{(i)}\|_i^2,$$

$$\forall x \in E, h^{(i)} \in E^{(i)}, \quad i \in \{1:n\}. \quad (2.4)$$

For the sake of simplicity, we assume that parameters $\sigma_\alpha$ and $L \overset{\text{def}}{=} (L_1, \ldots, L_n)$ are known.

Let us define now the partial gradient step at point $x \in E$ along the active coordinate $i \in \{1:n\}$:

$$h^{(i)}(x) \overset{\text{def}}{=} -B_i^{-1} \nabla_i f(x). \quad (2.5)$$

4
In view of inequality (2.4), for any stepsize $\tau \in \mathbb{R}$, we have

$$f(x + \tau U_i h^{(i)}(x)) - f(x) \leq \tau (\nabla_i f(x), h^{(i)}(x)) + \frac{\tau^2}{2} L_i \|h^{(i)}(x)\|^2$$

(6.2)

Finally, we need to define a random generator $j = \mathcal{R}_\beta(L)$, $\beta \in [0,1]$, which generates random numbers $j \in \{1:n\}$ with the following probabilities:

$$\pi_\beta[i] \equiv \text{Prob}(j = i) \overset{\text{def}}{=} \frac{1}{\sum_{i=1}^{n} L_i^\beta}, \quad i \in \{1:n\},$$

(7.7)

where $S_\beta = \sum_{i=1}^{n} L_i^\beta$.

For solving the problem (2.1), consider the following method.

**Method $ACDM_\alpha(x_0)$**

1. Define $v_0 = x_0 \in \mathbb{E}$, $A_0 = 0$, $B_0 = 1$, and $\beta = \frac{\alpha}{2}$.

2. For $t \geq 0$, iterate:
   
   1) Choose active coordinate $i_t = \mathcal{R}_\beta(L)$.

   2) Find parameter $a_{t+1} > 0$ from equation $a_{t+1}^2 S_\beta^2 = A_{t+1} B_{t+1}$, where $A_{t+1} = A_t + a_{t+1}$ and $B_{t+1} = B_t + \sigma_1 a_{t+1}$.

   3) Define $\alpha_t = \frac{a_{t+1}}{\beta_{t+1}}$, $\beta_t = \frac{\sigma_1 - a_{t+1}}{B_{t+1}}$, and $y_t = \frac{(1-\alpha_t) x_t + \alpha_t (1-\beta_t) v_t}{1-\alpha_t \beta_t}$.

   4) Compute $\nabla_i f(y_t)$. Update $x_{t+1} = y_t + \frac{1}{\tau_{i_t}} U_i h^{(i_t)}(y_t)$, and $v_{t+1} = (1 - \beta_t) v_t + \beta_t y_t + \frac{a_{t+1}}{L_{i_t}} \frac{1-\alpha_t}{B_{t+1} + \sigma_1 [a_t]} U_i h^{(i_t)}(y_t)$.

Denote $w_t = (1 - \beta_t) v_t + \beta_t y_t$. Then

$$y_t = \frac{(1-\alpha_t) x_t + \alpha_t (1-\beta_t) y_t}{1-\alpha_t \beta_t} = \frac{(1-\alpha_t) x_t + \alpha_t w_t}{1-\alpha_t \beta_t} - \frac{\alpha_t \beta_t y_t}{1-\alpha_t \beta_t}.$$

Thus, in method (2.8) we have the following representation:

$$y_t = (1 - \alpha_t) x_t + \alpha_t w_t.$$

(9.9)

Method (2.8) generates random output, which depends on particular implementation of the collection of i.i.d.-variables $\mathcal{I}_t = \{i_0, \ldots, i_t\}$ (define $\mathcal{I}_{-1} = \emptyset$). In what follows, notation $E_{\mathcal{I}_t}(\cdot)$ denotes the expectation of corresponding random variables.
Theorem 1 Let sequences \( \{x_t\}_{t \geq 0} \) and \( \{v_t\}_{t \geq 0} \) be generated by method (2.8). Then, for any \( t \geq 0 \) we have

\[
2A_t E_{t-1}(f(x_t) - f(x_*)) + B_t E_{t-1}(\|v_t - x_*\|_{1-\alpha}^2) \leq \|x_0 - x_*\|_{1-\alpha}^2,
\]

(2.10)

where

\[
A_t \geq \frac{1}{4\gamma} \left[ (1 + \gamma)^t - (1 - \gamma)^t \right]^2 \geq \frac{1}{4\gamma^2} t^2,
\]

(2.11)

and \( \gamma = \frac{\sigma^{1/2}}{2S_{\alpha/2}} \).

Proof: Denote \( r_t^2 = \|v_t - x_*\|_{1-\alpha}^2 \). Then

\[
\|v_{t+1} - x_*\|_{1-\alpha}^2 = \sum_{i \neq i_t} L_i^{-1-\alpha} \|w_t^{(i)} - x_*\|_i^2 + L_{i_t}^{1-\alpha} \|w_t^{(i_t)} - x_*^{(i_t)} + \frac{a_{t+1}k(t+i)(y_t)}{L^{1-\alpha}_{i_t}B_{t+1}\pi_{\beta}[i_t]}\|_i^2
\]

(2.6)

\[
= \|w_t - x_*\|_{1-\alpha}^2 - \frac{2a_{t+1}}{B_{t+1}\pi_{\beta}[i]}(\nabla_{i_t} f(y_t), w_t^{(i_t)} - x_*^{(i_t)}) + \frac{a_{t+1}^2}{L_{i_t}^{1-\alpha}B_{t+1}\pi_{\beta}[i]}(\|\nabla_{i_t} f(y_t)\|_{i_t}^2)
\]

(2.7)

Since \( \|w_t - x_*\|_{1-\alpha}^2 \leq (1 - \beta_t)r_t^2 + \beta_t\|y_t - x_*\|_{1-\alpha}^2 \), we can continue as follows:

\[
B_{t+1}r_{t+1}^2 \leq B_tr_t^2 + \beta_tB_{t+1}\|y_t - x_*\|_{1-\alpha}^2 - \frac{2a_{t+1}}{\pi_{\beta}[i]}(\nabla_{i_t} f(y_t), w_t^{(i_t)} - x_*^{(i_t)})
\]

(2.8)

\[
+ \frac{a_{t+1}^2}{B_{t+1}\pi_{\beta}[i]}(f(y_t) - f(y_t + \frac{1}{L_{i_t}}U_{i_t}h^{(i_t)}(y_t)))
\]

Note that \( E_{i_t} f(x_{t+1}) = \sum_{i=1}^n \pi_{\beta}[i] f(y_t + \frac{1}{L_{i_t}}U_{i_t}h^{(i_t)}(y_t)) \). Therefore, taking expectation of the above inequality in random variable \( i_t \), we obtain

\[
E_{i_t}(B_{t+1}r_{t+1}^2) \leq B_tr_t^2 + a_{t+1}\sigma_{1-\alpha}\|y_t - x_*\|_{1-\alpha}^2 + 2a_{t+1}\|\nabla f(y_t), x_* - w_t\|
\]

(2.9)

\[
+ 2\frac{a_{t+1}^2}{B_{t+1}\pi_{\beta}}(f(y_t) - E_{i_t}(f(x_{t+1}))).
\]

Since \( w_t = y_t + \frac{1}{\alpha_{t-1}}(y_t - x_t) \), we obtain

\[
a_{t+1}(\nabla f(y_t), x_* - w_t) = a_{t+1}(\nabla f(y_t), x_* - y_t + \frac{1}{\alpha_{t-1}}(x_t - y_t))
\]

(2.10)

\[
\leq a_{t+1}(f(x_t) - f(y_t)) - \frac{1}{2}a_{t+1}\sigma_{1-\alpha}(y_t - x_*|_{1-\alpha} + a_{t+1}\frac{1-\alpha}{\alpha_{t-1}}(f(x_t) - f(y_t))
\]

(2.11)

\[
= a_{t+1}f(x_t) - A_{t+1}f(y_t) + A_tf(x_t) - \frac{1}{2}a_{t+1}\sigma_{1-\alpha}(y_t - x_*|_{1-\alpha}^2).
\]

6
Substituting this inequality in (2.12), we obtain

\[ E_i(B_{t+1}^2) \leq B_t r_0^2 + 2 A_t (f(x_t) - f(x_\ast)) - 2 A_{t+1} (E_i(f(x_{t+1}) - f(x_\ast))). \]

It remains to take the expectation in \( I_{t-1} \) and sum up all previous inequalities. We obtain

\[ 2 A_t E_{I_{t-1}}(f(x_t) - f(x_\ast)) + B_t E_{I_{t-1}}(r_t^2) \leq r_0^2 = \|x_0 - x_\ast\|_{[1-\alpha]}^2. \]

Let us estimate now the growth of coefficients \( A_t \) and \( B_t \). Note that \( B_t = 1 + \sigma_1 A_t \). Therefore, equation for finding parameter \( a_{t+1} \) in method (2.8) looks as follows:

\[ (A_{t+1} - A_t) S_{[2]}^2 = A_{t+1}(1 + \sigma_1 A_{t+1}). \]

Denote \( C_t = \sigma_{1-\alpha} A_t^{1/2} \), \( t \geq 0 \). Then

\[ \sigma_{1-\alpha}^{-1} C_{t+1}^2 (1 + C_{t+1}^2) = \sigma_{1-\alpha}^{-2} S_{[2]}^2 (C_{t+1}^2 - C_t^2) \leq 4 \sigma_{1-\alpha}^{-2} S_{[2]}^2 (C_{t+1} - C_t)^2 C_{t+1}^2. \]

Thus, \( C_{t+1} - C_t \geq \gamma (1 + C_t^2)^{1/2} \geq C_t + \gamma (1 + C_t^2)^{1/2} \) with \( \gamma = \frac{\sigma_{1-\alpha}^{1/2}}{2 S_{[2]}} \). Now, by induction we can easily check that \( C_t \geq \frac{1}{4} [(1 + \gamma)^t - (1 - \gamma)^t] \geq \gamma^t t \) for \( t \geq 0 \). Indeed, in this case,

\[ 1 + C_t^2 \geq 1 + \frac{1}{4} (1 + \gamma)^{2t} + \frac{1}{4} (1 - \gamma)^{2t} - \frac{1}{2} (1 - \gamma^2)^t \geq \frac{1}{4} [(1 + \gamma)^t + (1 - \gamma)^t]^2. \]

Hence,

\[ C_{t+1} \geq \frac{1}{2} [(1 + \gamma)^t - (1 - \gamma)^t] + \frac{1}{2} [(1 + \gamma)^t + (1 - \gamma)^t] \]

\[ = \frac{1}{2} [(1 + \gamma)^{t+1} + (1 - \gamma)^{t+1}]. \]

Thus, \( A_t \geq \frac{1}{4 \sigma_{1-\alpha}} [(1 + \gamma)^t - (1 - \gamma)^t]^2 \geq \frac{1}{4 \sigma_{1-\alpha}} t^2 \), and

\[ B_t = 1 + \sigma_1 A_t \geq 1 + \frac{1}{4} [(1 + \gamma)^t - (1 - \gamma)^t]^2 \geq \frac{1}{4} [(1 + \gamma)^t + (1 - \gamma)^t]^2. \]

\[ 7 \]

Note that method (2.8) and its efficiency bounds (2.10), (2.11) are continuous in the convexity parameter \( \sigma_{1-\alpha} \). As \( \sigma_{1-\alpha} \to 0 \), we get a monotone decrease of values \( B_t \) to one, and values \( A_t \) go to their lower bounds \( \frac{S_{[2]}^2}{4 \sigma_{1-\alpha}} \).

**Remark 1** The first coordinate descent version of method (2.8) with \( \alpha = 0 \) (uniform distribution) was suggested in [7]. In [2], this method was extended onto arbitrary values of \( \alpha \in [0, 1] \). However, in [2] the authors used another random strategies \((\pi_i = L_{\alpha}/S_{\alpha})\). As a result, they get weaker complexity bounds. Indeed, in order to solve problem (2.1) with accuracy \( \epsilon \), they need \( O \left( \frac{S_{[2]}^2}{\sigma_{1-\alpha}} \ln \frac{1}{\epsilon} \right) \) iterations (see Theorem 4 in [2]). Our method requires \( O \left( \frac{S_{[2]}^2}{\sigma_{1-\alpha}^2} \ln \frac{1}{\epsilon} \right) \) iterations. It is easy to see that we always have

\[ \sqrt{nS_{[2]}^2} \geq S_{[2]/2}; \]

and sometimes the gain can reach a factor of order \( \sqrt{n} \). We give the corresponding examples in Section 3.
3 Examples of applications

3.1 Favorable structure of objective function

Let us compare now the complexity bounds of the Accelerated Coordinate Descent Method (2.8) with complexity bounds of the standard Fast Gradient Methods (e.g. [6]). For the sake of simplicity, we assume that in problem (2.1) we have $\dim \mathcal{E}(i) = 1$, $i \in \{1 : n\}$. Thus, $\dim \mathcal{E} = N \equiv n$. Moreover, let us assume that the objective function in (2.1) is twice continuously differentiable. Therefore,

$$L_i(f) = \sup_{x \in \mathcal{E}} \langle \nabla^2 f(x)e_i, e_i \rangle, \quad x \in \mathcal{E}, \ i \in \{1 : n\},$$

(3.1)

where $e_i$ is the $i$th coordinate vector in $\mathcal{E}$.

Let us define also the Lipschitz constant for the gradient of objective function in (2.1):

$$L(f) = \sup_{x \in \mathcal{E}} \max_{\|h\| \leq 1} \langle \nabla^2 f(x)h, h \rangle.$$  (3.2)

Assuming that $\|e_i\| \leq 1$ for all $i = 1, \ldots, n$, we clearly have $L_i(f) \leq L(f), \ i \in \{1 : n\}$.

For our comparison, let us choose $\alpha = 1$. Then all distances in $\mathcal{E} \equiv \mathbb{R}^n$ are measured in the standard Euclidean norm, which does not depend on the Lipschitz constants for the derivatives. For the sake of notation, denote $\|\cdot\| \equiv \|\cdot\|_0$. Denote $R = \|x_0 - x^*\|$ and let us assume that $\sigma_0 = 0$ (no strong convexity).

In this situation, fast gradient methods solve problem (2.1) up to accuracy $\epsilon$ in $O \left(I_{FGM} \overset{\text{def}}{=} \frac{L^{1/2}(f)}{\epsilon^{1/2}} R\right)$ iterations (e.g. [8]). At each iteration, they need to update $n$-dimensional vectors and to call oracle (a constant number of times). Denoting the corresponding computational expenses by $T_{FGM}$, we get the following bound for total computational cost:

$$C_{FGM} = I_{FGM} \cdot T_{FGM} = \frac{L^{1/2}(f)}{\epsilon^{1/2}} R \cdot T_{FGM}.$$  

Similarly, in view of Theorem 1, for solving problem (2.1) up to accuracy $\epsilon$, method (2.8) needs $O \left(I_{ACDM} \overset{\text{def}}{=} \frac{S_{1/2}(f)}{\epsilon^{1/2}} R\right)$ iterations. Thus, its total computational cost is

$$C_{ACDM} = I_{ACDM} \cdot T_{ACDM} = \frac{S_{1/2}(f)}{\epsilon^{1/2}} R \cdot T_{ACDM}.$$  

Note that $S_{1/2} \leq nL^{1/2}(f)$. Therefore, in order to ensure $C_{ACDM} \leq C_{FGM}$, we need to find problems, for which $T_{ACDM} \leq \frac{1}{n} T_{FGM}$.

Let the objective function $f$ in problem (2.1) has the following structure:

$$f(x) = F(Ax, x),$$

(3.3)

where $F(s, x) : \mathbb{R}^{m+n} \rightarrow \mathbb{R}$ is a convex differentiable function, and $A$ is an $m \times n$-matrix.

Our main structural assumption on function $F$ is that the complexity $T_F$ of its first-order oracle is linear:

$$T_F = O(m + n).$$  (3.4)
This time is required for computing the function value \( F(s, x) \) and the gradient
\[
\nabla F(s, x) = (\nabla_s F(s, x), \nabla_x F(s, x)) \in \mathbb{R}^m \times \mathbb{R}^n.
\]

Note that \( \nabla f(x) = \nabla_x F(Ax, x) + A^T \nabla_s F(Ax, x) \). Let us estimate now the complexity of one iteration of our methods, assuming that matrix \( A \) is dense and
\[
m \geq O(n). \tag{3.5}
\]

For Fast Gradient Method, the most expensive computation at each iteration is the call of oracle. In accordance of our assumptions, computation of the function value and the gradient needs \( O(mn) \) arithmetic operations. All other costs (update of \( n \)-dimensional vectors, computation of scalar products, etc.) need \( O(m + n) \) operations. Thus, we conclude that
\[
T_{FGM} = O(mn). \tag{3.6}
\]

For ACD-method (2.8), at each iteration we need to know only the value of directional derivative \( \nabla_t f(y_t) \). If the vector \( Ay_t \) is already computed, this needs \( O(m + n) \) operations. Therefore, during the process (2.8) we need to update recursively these vectors. For this, we need to update also the products \( Ax_t, AV_t \), and \( AW_t \). These operations need just computation of convex combinations of some already computed vectors with the cost \( O(n) \). Only two operations for computing \( Ax_{t+1} \) and \( AV_{t+1} \) need addition of \( i \)-th column of matrix \( A \) with some factors, and their cost is \( O(m) \). Thus, we conclude that in our case
\[
T_{ACDM} = O(m + n) \overset{(3.5)}{=} \frac{1}{n} T_{FGM}. \tag{3.7}
\]

Hence, for all optimization problem (2.1) with above structure we have \( C_{ACDM} \leq C_{FGM} \).

In the next two parts of this section we give examples of objective functions, for which ACD-method (2.8) can outperform the standard schemes by a dimensionally dependent factor. For these examples, we can guarantee that \( L_i(f) \ll L(f), i \in \{1 : n\} \).

### 3.2 Unconstrained minimization of quadratic function

Let \( A \in \mathbb{R}^{n \times n} \) be a symmetric positive-definite matrix, and \( F(s, x) = \frac{1}{2} \langle s, x \rangle - \langle b, x \rangle \). Then, all structural assumptions of Section 3.1 are satisfied, and we conclude that for problem
\[
\min_{x \in \mathbb{R}^n} [f(x) = \frac{1}{2} \langle Ax, x \rangle - \langle b, x \rangle] \tag{3.8}
\]
we have \( C_{ACDM} \leq C_{FGM} \).

Let us assume now that matrix \( A \) has positive elements, which have same order of magnitude:
\[
0 < \kappa_1 \leq A^{(i,j)} \leq \kappa_2, \quad i, j \in \{1 : n\}, \tag{3.9}
\]
and \( \kappa_2 \leq O(\kappa_1) \). Then,
\[
S_{1/2} \leq n\kappa_2^{1/2}. \tag{3.10}
\]

On the other hand,
\[
L(f) = \lambda_{\max}(A) \geq \kappa_1 \lambda_{\max}(1_n 1_n^T) = n\kappa_1, \tag{3.11}
\]
where $1_n \in \mathbb{R}^n$ is the vector of all ones. This implies that

$$S_{1/2} \leq \sqrt{\frac{\mu_2}{\kappa_1} \cdot L^{1/2}(f)}.$$  

In other words, assumption (3.9) implies $C_{ACDM} \leq O \left( \frac{1}{\sqrt{n\mu_2}} \right) C_{FGM}$.

### 3.3 Smoothing Technique

Smoothing technique [6] can be applied to objective functions with sufficiently simple dual representation:

$$f(x) = \max_{u \in Q} \{ \langle Ax, u \rangle - \phi(u) \},$$  

where $Q \subset \mathbb{R}^m$ is a closed convex bounded set, and function $\phi$ is convex on $Q$. Let us measure distances in $\mathbb{R}^m$ by some norm $\| \cdot \|_X$. We assume that

$$\| e_i \|_X \leq 1, \quad i = 1, \ldots, n,$$  

where $e_i$ is the $i$th coordinate vector in $\mathbb{R}^n$.

Function $f$ defined by (3.12) is typically nonsmooth. However, optimization problem in (3.12) must be simple enough since we assume it solvable in a closed form (otherwise, the value $f(x)$ is not computable). In this situation, it is often possible to approximate $f$ by a convex function with Lipschitz continuous gradient.

Indeed, let prox-function $d(u)$ be differentiable and strongly convex on $Q$ in some norm $\| \cdot \|_U$ with convexity parameter one:

$$\langle \nabla d(u_1) - \nabla d(u_2), u_1 - u_2 \rangle \geq \| u_1 - u_2 \|_U^2, \quad u_1, u_2 \in U.$$  

Assume that $d(u) \geq 0$ for all $u \in Q$ and $d(u_0) = 0$ at some prox-center $u_0 \in Q$.

Denote

$$f_\mu(x) = \max_{u \in Q} \{ \langle Ax, u \rangle - \phi(u) - \mu d(u) \},$$  

where $\mu > 0$ is the smoothness parameter. Then $f_\mu$ approximates $f$ with accuracy $O(\mu)$, and its gradient is Lipschitz continuous with constant $L(f_\mu) = \frac{\mu}{\| A \|}$, where

$$\| A \| = \max_{x,u} \{ \langle Ax, u \rangle : \| x \|_X \leq 1, \| u \|_U \leq 1 \}.$$  

Note that $\| A \| \geq \max_u \{ \langle Ae_i, u \rangle : \| u \|_U \} = \| Ae_i \|_U^*$ for all $i = 1, \ldots, n$. Therefore,

$$\sum_{i=1}^n \| Ae_i \|_U^* \leq n\| A \|.$$  

Recall that the gradient of function $f_\mu$ is defined as

$$\nabla f_\mu(x) = A^T u_\mu(x),$$  

where $u_\mu(x)$ is the unique solution of the optimization problem in definition (3.15).
Let us justify now the bounds for $L_i(f_\mu)$, $i \in \{1 : n\}$. Consider two points $x_1$ and $x_2 = x_1 + h$, where $h$ is an arbitrary direction in $\mathbb{R}^n$. Denote $u_i = u_\mu(x_i)$, $i = 1, 2$. From the optimality conditions for optimization problem in (3.15), we have
\[
\langle Ax_1 - \nabla \phi(u_1) - \mu \nabla d(u_1), u_2 - u_1 \rangle \leq 0,
\]
\[
\langle Ax_2 - \nabla \phi(u_2) - \mu \nabla d(u_2), u_1 - u_2 \rangle \leq 0.
\]
Adding these two inequalities, we get
\[
\mu \|u_1 - u_2\|_U^2 \leq \mu \langle \nabla d(u_1) - \nabla d(u_2), u_1 - u_2 \rangle \leq \langle Ax_1 - Ax_2 - (\nabla \phi(u_1) - \nabla \phi(u_2)), u_1 - u_2 \rangle \leq \langle Ah, u_2 - u_1 \rangle.
\]
Taking now $h = \tau e_i$, where $e_i$ is the $i$th coordinate vector in $\mathbb{R}^n$, we obtain:
\[
\tau \langle \nabla_i f_\mu(x_2) - \nabla_i f_\mu(x_1) \rangle \geq \tau \langle e_i, A^T(u_2 - u_1) \rangle \geq \mu \|u_1 - u_2\|_U^2 \geq \langle A_i, u_1 - u_2 \rangle^2 \geq \frac{\mu}{\|A_i\|_U^2} \langle A e_i, u_1 - u_2 \rangle^2 = \frac{\mu}{\|A_i\|_U^2} \langle \nabla_i f_\mu(x_1) - \nabla_i f_\mu(x_2) \rangle^2.
\]
Thus, we can take $L_i(f_\mu) = \frac{1}{\mu} \langle |A e_i|_U \rangle^2$, $i \in \{1 : n\}$. Consequently,
\[
\sum_{i=1}^n L_i^{1/2}(f_\mu) \leq n L^{1/2}(f_\mu).
\]
If the set $Q$ and function $\phi$ in (3.15) are simple, then $f_\mu$ satisfies all conditions of Section 3.1 (in particular, with known product $Ax$, vector $u_\mu(x)$ is computable in $O(m)$ operations). Therefore, for its unconstrained minimization, efficiency estimates of ACD-method (2.8) are always not worse than the bounds of any Fast Gradient Method.

Let us present an example, where ACD-method (2.8) is much better than FGM (since $L_i(f_\mu) << L(f_\mu)$, $i \in \{1 : n\}$). Assume that all elements of matrix $A$ are positive and have the same order of magnitude:
\[
0 < \kappa_1 \leq A^{(i,j)} \leq \kappa_2, \quad i \in \{1 : m\}, \quad j \in \{1 : n\},
\]
and $\kappa_2 \leq O(\kappa_1)$. Then, clearly $L_i(f_\mu) \leq \frac{m}{\mu} \kappa_2^2$. Therefore, $S_1/2 \leq n \kappa_2 \left(\frac{m}{\mu}\right)^{1/2}$.

On the other hand,
\[
L(f_\mu) = \frac{1}{\mu} \lambda_{\text{max}}(A^T A) \geq \frac{1}{\mu} \kappa_1^2 m \lambda_{\text{max}}(1^T_n 1^T_n) = \frac{1}{\mu} \kappa_1^2 mn.
\]
Thus, comparing the bounds
\[
C_{ACDM} = O\left(m \cdot S_1/2 \cdot \frac{R}{\epsilon^{1/2}}\right) \leq O\left(n m^{3/2} \cdot \frac{R}{\mu^{1/2} \epsilon^{1/2}}\right),
\]
and
\[
C_{FGM} = O\left(m n \cdot L^{1/2}(f_\mu) \cdot \frac{R}{\epsilon^{1/2}}\right) \geq O\left(n^{3/2} m^{3/2} \cdot \frac{R}{\mu^{1/2} \epsilon^{1/2}}\right),
\]
we can see that the bound for ACD-method (2.8) is at least in $O(n^{1/2})$ times better.
4 Preliminary computational experiments

In our computational experiments, we solved the following problem with randomly generated data:

$$
\min_{x \in \mathbb{R}^M} \left\{ f_\mu(x) = \sum_{i=1}^{N} \phi_\mu \left( (a_i, x) - c^{(i)} \right) \right\},
$$

(4.1)

where

$$
\phi_\mu(\tau) = \begin{cases} 
\frac{\tau^2}{2\mu}, & \text{if } |\tau| \leq \mu, \\
|\tau| - \frac{1}{2\mu}, & \text{if } \tau > \mu.
\end{cases}
$$

Coefficients of dense vectors $a_i$, $i = 1, \ldots, N$, are uniformly distributed in the interval $[1, 2]$. Coefficients of vector $c = (c^{(1)}, \ldots, c^{(N)})^T \in \mathbb{R}^N$ are chosen as $c^{(i)} = (a_i, \vec{y})$, where the entries of vector $\vec{y} \in \mathbb{R}^M$ are uniformly distributed in the interval $[-1, 1]$.

Thus, the optimal value of function $f_\mu$ is zero. Therefore, for all methods we use the termination criterion $f_\mu(x) \leq \epsilon$ with $\epsilon = 10^{-2}$. We choose also $\mu = \epsilon$.

Among numerous variants of Fast Gradient Methods, we choose the method with the maximal adaptivity to the unknown Lipschitz constant for the gradient of objective function. Its scheme is as follows.

FGM: Choose $x_0 \in \mathbb{E}$ and $L_0 > 0$. Set $v_0 = x_0$.

For $t \geq 0$ iterate:

1) Find the smallest $i_t \geq 0$ such that for

$$
a_{t,i_t} = \frac{1}{2^{t+1}L_t} \left( 1 + \sqrt{1 + 2^{t+2}L_tA_t} \right), \quad \tau_{t,i_t} = \frac{a_{t,i_t}}{a_{t,i_t} + A_t},
$$

$$
y_{t,i_t} = (1 - \tau_{t,i_t})x_t + \tau_{t,i_t}v_t, \quad \text{and} \quad x_{t+1,i_t} = y_{t,i_t} - \frac{1}{2^{t}L_t} \nabla f(y_{t,i_t})
$$

we have

$$
f(y_{t,i_t}) - f(x_{t+1,i_t}) \geq \frac{1}{2^{t+1}L_t} \| \nabla f(y_{t,i_t}) \|^2.
$$

2) Set $x_{t+1} = x_{t+1,i_t}$, $v_{t+1} = v_t - a_{t,i_t} \nabla f(y_{t,i_t})$, $A_{t+1} = A_t + a_{t,i_t}$, and $L_{t+1} = 2^{t-1}L_t$.

On the contrary, for Accelerated Coordinate Descent Method (2.8) with parameters $\alpha = 1$ and $\sigma_0 = 0$, we choose the fixed worst-case estimates for the coordinate Lipschitz constants

$$
L_i(f_\mu) = \frac{1}{\mu} \| A^T e_i \|^2, \quad i = 1, \ldots, M,
$$

(4.3)

where $A = (a_1, \ldots, a_N)$ and the norm is standard Euclidean. Since we take $\beta \equiv \alpha/2 = \frac{1}{2}$, we get the following distribution of probabilities:

$$
\pi_{1/2}[i] = \frac{\| A^T e_i \|}{\sum_{k=1}^{M} \| A^T e_k \|}, \quad i = 1, \ldots, M.
$$

(4.4)
At the same time, $S_{1/2}^2 = \frac{1}{\mu} \left( \sum_{i=1}^{M} \| A e_i \| \right)^2$.

In all our experiments we use the staring point $x_0 = 0 \in \mathbb{R}^M$. In the method below, notation $Ax$ (or, $Ay$, $Av$) is used for the value of the linear operator in (4.1), computed at point $x \in \mathbb{R}^M$:

$$Ax \equiv A^T x - c \in \mathbb{R}^N.$$ 

The scheme of ACD-method for problem (4.1) looks as follows.

<table>
<thead>
<tr>
<th>ACDM for (4.1): Define $v_0 = x_0 = 0 \in \mathbb{R}^M$, $Av_0 = Ax_0 = -c$, and $A_0 = 0$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>For $t \geq 0$, iterate:</td>
</tr>
<tr>
<td>1) Find parameter $a_{t+1} &gt; 0$ from equation $a_{t+1}^2 S_{\beta}^2 = A_{t+1} + a_{t+1}$. Set $A_{t+1} = A_t + a_{t+1}$, and $\tau_t = \frac{a_{t+1}}{\lambda_{t+1}}$.</td>
</tr>
<tr>
<td>2) Define $y_t = (1 - \tau_t)x_t + \tau_tv_t$. Update $Ay_t = (1 - \tau_t)Ax_t + \tau_tAv_t$.</td>
</tr>
<tr>
<td>3) Choose $i_t$ in accordance to distribution (4.4) and compute $\nabla_i f(y_t)$.</td>
</tr>
<tr>
<td>4) Update $x_{t+1} = y_t - \frac{1}{\tau_t} \nabla_i f(y_t)e_{i_t}$, $Ax_{t+1} = Ay_t - \frac{1}{\tau_t} \nabla_i f(y_t)A^T e_{i_t}$, $v_{t+1} = v_t - a_{t+1} \nabla_i f(y_t)e_{i_t}$, and $Av_{t+1} = Av_t - \frac{a_{t+1}}{\pi_{1/2}[v_t]} \nabla_i f(y_t)A^T e_{i_t}$.</td>
</tr>
</tbody>
</table>

Note that the computational cost of all operations in the above method, including the computation of directional derivative $\nabla_i f(y_t) = \langle A^T e_i, \nabla f(y_t) \rangle$, is linear in the dimensions of problem (4.1).

In view of its adaptivity, in our experiments, FGM is a priori in a much better position.
than ACDM. Nevertheless, the computational results are as follows.

\[ \begin{array}{|c|c|c|c|c|c|c|} \hline N & M & IT & NF & \text{Time (sec)} & IT/M & \text{Time (sec)} \\ \hline 100 & 50 & 4727 & 18916 & 0.547 & 2024 & 0.578 \\ 50 & 100 & 4889 & 19566 & 0.578 & 2305 & 0.672 \\ 200 & 100 & 11244 & 44986 & 4.750 & 3070 & 4.000 \\ 100 & 200 & 12859 & 51450 & 5.250 & 3750 & 4.203 \\ 400 & 200 & 25473 & 101902 & 40.234 & 5495 & 23.125 \\ 200 & 400 & 26184 & 104750 & 40.719 & 6345 & 30.157 \\ 800 & 400 & 55511 & 222056 & 358.234 & 8789 & 302.203 \\ 400 & 800 & 61994 & 247992 & 397.656 & 11461 & 245.657 \\ 1600 & 800 & 122542 & 490184 & 3185.953 & 13899 & 1652.733 \\ 800 & 1600 & 126748 & 507008 & 3213.156 & 19139 & 2360.719 \\ \hline \end{array} \]

Table 1. Performance of FGM and ACDM on problem (4.1).

In this table, first two columns display the dimensions of problem (4.1). In all our tests, the matrix \( A \) is dense. Therefore, for the largest problem we have more than one million nonzero coefficients. Columns \( IT \) and \( NF \) show the number of iterations and number of function evaluation of FGM. Column \( IT/M \) shows the number of blocks of \( M \) iterations in method ACDM. Finally, the column \( \text{Time} \) displays the total computational time in seconds.

For us, the main characteristics of complexity of the problem for numerical scheme is the total computational time. As we can see, ACDM always outperforms FGM. Its domination is less impressive with respect to the theoretical prediction. However, this can be explained by the ability of method (4.2) to use much smaller estimate of the constant \( L(f_{\mu}) \) than the worst-case theoretical value.

To conclude, we can see that potentially, ACDM is a promising computational scheme, which has good chances to outperform FGM on many important real-life problems. At this moment, as compared with FGM, ACDM has four main drawbacks:

- absence of version with separable constraints;
- impossibility to adjust the worst-case estimates for \( L_i(f) \) during the minimization process;
- absence of a reliable stopping criterion;
- impossibility to generate good primal-dual solutions.

In our opinion, any advancement in these directions will be very interesting.
References


