Weierstraß-Institut
für Angewandte Analysis und Stochastik
Leibniz-Institut im Forschungsverbund Berlin e. V.

Preprint ISSN 2198-5855

Oracle complexity separation in convex optimization

Anastasiya Ivanova[1,2], Alexander Gasnikov[1,2,3], Pavel Dvurechensky[4], Darina Dvinskikh[4], Alexander Tyurin[2], Evgeniya Vorontsova[1,5], Dmitry Pasechnyuk[6]

submitted: April 2, 2020

[1] Moscow Institute of Physics and Technology
Dolgoprudny, Russia
E-Mail: anastasiya.s.ivanova@phystech.edu
gasnikov@yandex.ru
vorontsovaea@gmail.com

[2] Higher School of Economics
Moscow, Russia
E-Mail: anastasiya.s.ivanova@phystech.edu
alexandertiurin@gmail.com
gasnikov@yandex.ru

[3] Institute for Information Transmission Problems
Moscow, Russia
E-Mail: gasnikov@yandex.ru

[4] Weierstrass Institute
Mohrenstr. 39
10117 Berlin, Germany
E-Mail: darina.dvinskikh@wias-berlin.de
pavel.dvurechensky@wias-berlin.de

[5] Université Grenoble Alpes
Grenoble, France
E-Mail: vorontsovaea@gmail.com

[6] Presidential Physics and Mathematics Lyceum No.239
St. Petersburg, Russia
E-Mail: pasechnyuk2004@gmail.com

No. 2711
Berlin 2020

2010 *Mathematics Subject Classification.* 90C30, 90C25, 68Q25, 65K15.

Key words and phrases. Convex optimization, composite optimization, proximal method, acceleration, random coordinate descent, variance reduction.

Edited by
Weierstraß-Institut für Angewandte Analysis und Stochastik (WIAS)
Leibniz-Institut im Forschungsverbund Berlin e. V.
Mohrenstraße 39
10117 Berlin
Germany

Fax: +49 30 20372-303
E-Mail: preprint@wias-berlin.de
World Wide Web: http://www.wias-berlin.de/

Oracle complexity separation in convex optimization

Anastasiya Ivanova, Alexander Gasnikov, Pavel Dvurechensky, Darina Dvinskikh, Alexander Tyurin, Evgeniya Vorontsova, Dmitry Pasechnyuk

Abstract

Ubiquitous in machine learning regularized empirical risk minimization problems are often composed of several blocks which can be treated using different types of oracles, e.g., full gradient, stochastic gradient or coordinate derivative. Optimal oracle complexity is known and achievable separately for the full gradient case, the stochastic gradient case, etc. We propose a generic framework to combine optimal algorithms for different types of oracles in order to achieve separate optimal oracle complexity for each block, i.e. for each block the corresponding oracle is called the optimal number of times for a given accuracy. As a particular example, we demonstrate that for a combination of a full gradient oracle and either a stochastic gradient oracle or a coordinate descent oracle our approach leads to the optimal number of oracle calls separately for the full gradient part and the stochastic/coordinate descent part.

1 Introduction

The complexity of an optimization problem usually depends on the parameters of the objective, such as the Lipschitz constant of the gradient and the strong convexity parameter. In Machine Learning applications the objective is constructed from many building blocks, a typical example of a block being the individual loss for an example or the different regularizers in supervised machine learning. Standard theoretical results for optimization algorithms for such problems provide *iteration* complexity, namely the number of iterations to achieve a given accuracy. Unlike these results, in this paper, we address the question of *oracle* complexity, focusing on the number of oracle calls. Moreover, the goal is to study what number of oracle calls *for each* building block of the objective is sufficient to obtain the required accuracy. Indeed, typically the finite-sum part of the objective is much more computationally expensive than the regularizer, which motivates the usage of a randomized oracle for the finite-sum part and a proximal oracle for the regularizer. Further on, some components in the finite-sum part may be more expensive than others and it is desirable to call the gradient oracle of the former less frequently than the gradient oracle of the latter. Moreover, some of the building blocks of the objective may be available with their gradient, while for the other block only the value of the objective may be available. In this case, one would prefer to call the gradient oracle for the former less frequently than the zero-order oracle of the latter. To the best of our knowledge, the current optimization theory does not provide a convincing answer to the question of how to do this.

To be more precise, we consider minimization problem in the form

$$\min_{x\in\mathbb{R}^n} h(x) + g(x), \tag{1}$$

where the full objective is μ-strongly convex, $h(x)$ has L_h-Lipschitz continuous gradient and is available via its full gradients. The part g is available through different types of oracle, e.g. full gradient,

DOI 10.20347/WIAS.PREPRINT.2711 Berlin 2020

stochastic gradient, coordinate derivative, objective value, etc. The goal is to separate oracle complexity for h and g and call the oracle of h less frequently than that of g. To motivate this goal we consider two particular examples.

Kernel SVM [17]. The learning problem in Kernel SVM is to solve optimization problem

$$\min_x \sum_{i=1}^m (\alpha - b_i a_i^T K x)_+ + \frac{\lambda}{2} x^T K x,$$

where K is the Kernel matrix, (b_i, a_i^T) is the data. The standard approach is to make a transition to a basis in which the Kernel K is diagonal, which can be prohibitive in high dimensions. At the same time, composite versions of standard variance reduction methods [21, 19, 1] need to evaluate proximal operator of the quadratic term on each iteration, which is equivalent to inversion of the Kernel matrix and can be expensive in high dimensions. I our approach we use cheap first-order oracle for the quadratic term and cheap stochastic gradient oracle for the loss term. Moreover, the full gradient oracle for the quadratic term is called much more rarely than the stochastic gradient for the loss. This opens the way of using non-proximal friendly regularizers for ERM problem.

Log-density estimation with Gaussian Prior [20]. This problem has the form

$$\max_x \left\{ \langle c, x \rangle - m \log \left(\sum_{k=1}^p \exp \left(\langle A_k, x \rangle \right) \right) - \frac{1}{2} \|Gx\|^2 \right\}.$$

This problem is not in the standard form of a ERM problem and the standard method of choice is the full gradient method. At the same time, coordinate descent is an efficient method for quadratic functions and, in some cases, for problems related to smoothing [16]. Our approach allows to combine coordinate descent method for the quadratic part and full gradient for the log-sum-exp function.

The literature on combining different types of oracles to propose more efficient methods is quite sparse to the best of our knowledge. The most popular combination is known as composite optimization [14], in which first-order, second-order or stochastic gradient oracle for h is combined with proximal oracle for g, which allows complexity not to depend on g. Yet, proximal oracle is needed on each iteration of the method, so the complexity is not separated. An example of combination of first-order oracles for h and g is the paper [9], where the complexity is separated in this case. Recently a separation of oracle complexities was also introduced in the context of higher-order methods [7].

Let us briefly describe the main idea of the proposed approach. Assume that we have to solve a smooth μ-strongly convex problem

$$\min_{x \in \mathbb{R}^n} h(x) + g(x), \tag{2}$$

where $h(x)$ has L_h-Lipschitz continuous gradient and we have an algorithm that can solve the problem

$$\min_{x \in \mathbb{R}^n} g(x) + \frac{\tilde{L}}{2} \|x - \tilde{x}^k\|_2^2 \tag{3}$$

with (O_g-oracle) complexity $\tilde{O}\left(\sqrt{\tilde{L}_g / \tilde{L}}\right)$, where $\tilde{L}_g \geq L_h$. We also assume that $\nabla g(x)$ may be computed in κ_g O_g-oracle calls. Then we can apply an accelerated proximal method [6] with parameter L satisfying $\mu \leq L \leq L_h$ to (2). This method requires solving the auxiliary problem

$$\min_{x \in \mathbb{R}^n} h(x) + g(x) + \frac{L}{2} \|x - x^k\|_2^2 \tag{4}$$

$\tilde{O}(\sqrt{L/\mu})$ times. To solve (4) we may then use a non-accelerated composite gradient method with $g(x) + \frac{L}{2}\|x - x^k\|_2^2$ as the composite [14]. During each of the $\tilde{O}(L_h/L)$ iterations of this method we need to solve auxiliary problem (3) with $\tilde{L} = L + L_h$. So the total number of $\nabla h(x)$-oracle calls will be $\tilde{O}(\sqrt{L/\mu})$ and the total number of O_g-oracle calls will be

$$\tilde{O}(\sqrt{L/\mu}) \cdot \left[\tilde{O}(L_h/L) \cdot \tilde{O}\left(\sqrt{\tilde{L}_g/(L+L_h)}\right) + C_n\right].$$

Minimizing this expression over $L \in [\mu, L_h]$ and assuming that $\kappa_g = \tilde{O}\left(\sqrt{\tilde{L}_g/L_h}\right)$, we obtain

$$L \simeq L_h.$$

Thus, we can solve problem (2) via

$$\tilde{O}\left(\sqrt{L_h/\mu}\right) \nabla h(x)\text{-oracle calls}$$

and

$$\tilde{O}\left(\sqrt{\tilde{L}_g/\mu}\right) O_g\text{-oracle calls.}$$

In case when the O_g-oracle is the standard $\nabla g(x)$-oracle, this result corresponds to the accelerated sliding [9]. But our approach significantly differs from [9]. We use an accelerated proximal envelope with the non-accelerated composite gradient as an outer envelope instead of a special bulky accelerated outer method that was used in [9]. First of all, this simplifies the approach. Second, our approach allows to deal with different types of O_g-oracles, not only $\nabla g(x)$. For example, when the O_g-oracle comes from block-coordinate descent, directional search, derivative-free methods [3] or incremental methods [1, 10, 8].

Below in the paper we describe the scheme above (and its non-strongly convex variant) in detail, by controlling with what accuracy we have to solve the auxiliary problems.

2 Main result

Consider the problem

$$\min_{x\in\mathbb{R}^n} f(x) := h(x) + g(x).$$

We introduce the following assumptions about this problem:

Assumption 1. *We assume that $f(\cdot)$ has Lipschitz continuous gradient with the Lipschitz constant L_f and is μ-strongly convex w.r.t. $||\cdot||_2$.*

Assumption 2. *We assume that $h(\cdot)$ has Lipschitz continuous gradient with the Lipschitz constant L_h w.r.t. $||\cdot||_2$ and there is an oracle O_h which in one call produces the gradient $\nabla h(\cdot)$.*

Assumption 3. *We assume that $g(\cdot)$ has Lipschitz continuous gradient with Lipschitz constant L_g w.r.t. $||\cdot||_2$ and there is a basic oracle O_g which in κ_g calls produces the gradient $\nabla g(\cdot)$.*

Algorithm 1 Monteiro–Svaiter algorithm MS(x^0, L, N)

Parameters: Starting point $x^0 = y^0 = z^0$; parameter $L \in (0, L_h]$; number of iterations N.
for $k = 0, 1, \dots, N-1$ **do**
Compute

$$\begin{array}{rcl} a_{k+1} & = & \frac{1/L+\sqrt{1/L^2+4A_k/L}}{2}, \\ A_{k+1} & = & A_k + a_{k+1}, \\ x^{k+1} & = & \frac{A_k}{A_{k+1}} y^k + \frac{a_{k+1}}{A_{k+1}} z^k. \end{array}$$

Compute

$$y^{k+1} = \text{GMCO}(x^{k+1}, F_{L,x^{k+1}}(\cdot)). \tag{6}$$

Compute

$$z^{k+1} = z^k - a_{k+1} \nabla f\left(y^{k+1}\right). \tag{7}$$

end for
Output: y^N

Algorithm 2 Restarting Strategy for MS

Parameters: Starting point η^0; strong convexity parameter $\mu > 0$; parameter $L > 0$; accuracy $\varepsilon > 0$.
Compute $T = \left\lceil \log\left(\frac{||\eta^0 - \eta^*||_2^2 \cdot \mu}{\varepsilon} \right) \right\rceil, \quad N_0 = \sqrt{\frac{8L}{\mu}}$.
for $t = 1, \dots, T$ **do**
η^t = MS(η^{t-1}, L, N_0).
end for
Output: η^T

Moreover, we need the following assumption to state the main result.

Assumption 4. *We assume what there is a method $\mathcal{M}_{inn}(\varphi(\cdot), N(\tilde{\varepsilon}))$, which takes as input an objective function with the structure $\varphi(v) = \langle \beta, v \rangle + \frac{\alpha}{2}||v||_2^2 + g(v)$ and returns a point $\hat{v}$ such that*

$$\mathbb{E}\left(\varphi(\hat{v}) - \varphi(v^*)\right) \le \tilde{\varepsilon},$$

in $N(\tilde{\varepsilon}) = O\left(\frac{\tau_g}{\sqrt{\alpha}} \ln \frac{C||v^0 - v^*||_2^2}{\tilde{\varepsilon}}\right)$ basic oracle calls, where τ_g is a parameter dependent on the function $g(\cdot)$ and the method $\mathcal{M}_{inn}$ and independent of α, such that $\tau_g > \sqrt{\alpha}$, and C is a constant satisfying $C > 0$.

To solve the problem (5) we introduce the Monteiro–Svaiter Accelerated Proximal Method [11], which in non-adaptive case is presented as Algorithm 1, where

$$F_{L,y}(x) := f(x) + \frac{L}{2}||x - y||_2^2.$$

Note that the parameter L must be chosen so that $0 < L \le L_h$. If $\mu > 0$, to recover the acceleration through strong-convexity we apply a restarting strategy (Algorithm 2) to Algorithm 1.

Note that on each iteration of the MS algorithm in step (6) we solve the minimization problem: $\min_y F_{L,\,x^{k+1}}(y)$. We consider this problem as a composite optimization problem with the composite

Algorithm 3 Gradient method for Composite Optimization GMCO$(\zeta^0, F_{\zeta^0,L}(\cdot))$

Parameters: starting point $\zeta^0 \in \mathbb{R}^n$, objective function $F_{L,\zeta^0}(\zeta) = f(\zeta) + \frac{L}{2}||\zeta - \zeta^0||_2^2 = h(\zeta) + g(\zeta) + \frac{L}{2}||\zeta - \zeta^0||_2^2$.
Set $k := 0$
repeat
  Set $k := k + 1$.
  Set

$$\begin{aligned}\varphi_k(\zeta) \;&:=\; \langle \nabla h(\zeta^{k-1}), \zeta - \zeta^{k-1}\rangle + g(\zeta)\\ &+\; \tfrac{L}{2}\|\zeta - \zeta^0\|_2^2 + \tfrac{L_h}{2}\|\zeta - \zeta^{k-1}\|_2^2,\end{aligned}$$

  Compute

$$\zeta^k := \mathcal{M}_{inn}(\varphi_k(\zeta), N_{\mathcal{M}_{inn}}), \tag{9}$$

  where $N_{\mathcal{M}_{inn}}$ is defined as in (8).
until $\|\nabla F_{L,\zeta^0}(\zeta^k)\|_2 \le \frac{L}{2}\|\zeta^k - \zeta^0\|_2$.
Output: ζ^k.

$g(y) + \frac{L}{2}||y - x^{k+1}||_2^2$. To solve this problem we use the Gradient method for Composite Optimization (Algorithm 3) [14].

So, on each iteration k of the Algorithm 1 we use GMCO$(x^{k+1}, F_{x^{k+1},L}(\cdot))$. Note that we don't assume the proximal-friendliness of the function $g(x)$. Hence, it is necessary to take into account the complexity of the problem $\min\limits_{v\in\mathbb{R}^n} \varphi_k(v)$ which arises at each iteration of the Algorithm 3. To solve this problem we consider $\frac{L}{2}\|\zeta - \hat{\zeta}^0\|_2^2 + \frac{L_h}{2}\|\zeta - \hat{\zeta}^{k-1}\|_2^2$ as the composite and use the inner method $\mathcal{M}_{inn}$ from the Assumption 4, where $\alpha = L + L_h$ and

$$N_{\mathcal{M}_{inn}} = \begin{cases} O\left(\frac{\tau_g}{\sqrt{L+L_h}} \ln \frac{C_1 L_h}{\delta\sqrt{\mu L}}\right) & \text{if } \mu \ge 0,\\ O\left(\frac{\tau_g}{\sqrt{L+L_h}} \ln \frac{C_1 L_h R}{\delta\sqrt{\varepsilon L}}\right) & \text{if } \mu = 0,\end{cases} \tag{8}$$

where $R \ge ||x^0 - x^*||_2$, $C_1 > 0$ and $\delta \in (0,1)$.

So, for this scheme we can state the following main result:

Theorem 1. *Under the Assumptions 1-4 with probability at least $1 - \delta$ we can obtain $\hat{x}$ such that $f(\hat{x}) - f(x^*) \le \varepsilon$ in*

a) $O\left(\sqrt{\frac{LR^2}{\varepsilon}} \cdot \left(1 + \frac{L_h}{L}\right)\right)$ *Oracle calls for $h(\cdot)$ and*

$$O\left(\sqrt{\frac{LR^2}{\varepsilon}} \cdot \left(\kappa_g + \frac{L_h}{L} \cdot \left(\frac{\tau_g}{\sqrt{L+L_h}} \ln \frac{R}{\delta\sqrt{\varepsilon}}\right)\right)\right)$$

Oracle calls for $g(\cdot)$, if $\mu = 0$, and

b) $O\left(\sqrt{\frac{L}{\mu}} \log\left(\frac{\mu R^2}{\varepsilon}\right) \cdot \left(1 + \frac{L_h}{L}\right)\right)$ *Oracle calls for $h(\cdot)$ and*

$$O\left(\sqrt{\frac{L}{\mu}} \log\left(\frac{\mu R^2}{\varepsilon}\right) \cdot \left(\kappa_g + \frac{L_h}{L} \cdot \left(\frac{\tau_g}{\sqrt{L+L_h}} \ln \frac{1}{\delta}\right)\right)\right)$$

Oracle calls for $g(\cdot)$, if $\mu > 0$.

3 Proof of the main result

The proof of the main result consists of four steps:

1 Estimating the number of iterations of the inner method $\mathcal{M}_{inn}$.

2 Estimating the number of iterations of Algorithm 3.

3 Estimating the number of iterations of Algorithm 1 with the restarting strategy as in Algorithm 2.

4 Obtaining a final estimate of the number of iterations of oracles O_f and O_g based on estimates from steps $1-3$.

3.1 Step 1.

On each iteration of Algorithm 3 we need to solve the problem

$$\begin{aligned}\min_{v\in\mathbb{R}^n}\varphi(v) &:= \langle\nabla h(v^k), v-v^k\rangle + g(v)\\ &+ \tfrac{L}{2}\|v-v^0\|_2^2 + \tfrac{L_h}{2}\|v-v^k\|_2^2.\end{aligned}$$

Applying the method $\mathcal{M}_{inn}$ to (10) with $\alpha = L+L_h$ we obtain that in $N_{\mathcal{M}}(\tilde\varepsilon) = O\left(\frac{\tau_g}{\sqrt{L+L_h}}\ln\frac{C||v^0-v^*||_2^2}{\tilde\varepsilon}\right)$ Oracle calls we can find $v^{N_{\mathcal{M}}(\tilde\varepsilon)}$ such that

$$\mathbb{E}(\varphi(v^{N_{\mathcal{M}}(\tilde\varepsilon)}) - \varphi(v^*)) \le \tilde\varepsilon.$$

Since $\varphi(v^{N_{\mathcal{M}}(\tilde\varepsilon)}) - \varphi(v^*) \ge 0$, with an arbitrary $\tilde\delta \in (0,1)$ we can apply the Markov inequality:

$$\mathbb{P}\left(\varphi(v^{N_{\mathcal{M}}(\tilde\delta\tilde\varepsilon)}) - \varphi(v^*) \ge \tilde\varepsilon\right) \le \frac{\mathbb{E}(\varphi(v^{N_{\mathcal{M}}(\tilde\delta\tilde\varepsilon)})-\varphi(v^*))}{\tilde\varepsilon} \le \tilde\delta.$$

We have shown that with probability at least $1-\tilde\delta$ in $N_{\mathcal{M}}(\tilde\delta\tilde\varepsilon) = O\left(\frac{\tau_g}{\sqrt{L+L_h}}\ln\frac{C||v^0-v^*||_2^2}{\tilde\delta\tilde\varepsilon}\right)$ Oracle calls we can find $\hat v$ such that $\varphi(\hat v)-\varphi(v^*) \le \tilde\varepsilon$. Since $\varphi(\cdot)$ is $L+L_h$ strongly convex, we have

$$\tfrac{L+L_h}{2}||\hat v - v^*||_2^2 \le \varphi(\hat v) - \varphi(v^*) \le \tilde\varepsilon.$$

Moreover, since $\varphi(\cdot)$ is L_g-smooth and $\nabla\varphi(v^*)=0$

$$||\nabla\varphi(\hat v)||_2 \le L_g||\hat v - v^*||_2.$$

Using these two inequalities, we obtain

$$\begin{aligned}\langle\nabla\varphi(\hat v), \hat v - v^*\rangle &\le ||\nabla\varphi(\hat v)||_2\cdot||\hat v - v^*||_2\\ &\le L_g||\hat v - v^*||_2^2 \le L_g\tfrac{2\tilde\varepsilon}{L+L_h}.\end{aligned}$$

This leads to the following lemma

Lemma 1. *Applying the method $\mathcal{M}_{inn}$ to* (10) *we have that with probability at least $1-\tilde\delta$ in $N_{\mathcal{M}}(\tilde\varepsilon) = O\left(\frac{\tau_g}{\sqrt{L+L_h}}\ln\frac{2CL_g||v^0-v^*||_2^2}{\tilde\delta\tilde\varepsilon(L+L_h)}\right)$ Oracle calls we can find $\hat v$ such that*

$$\begin{aligned}\langle\nabla\varphi(\hat v), \hat v - v^*\rangle &\le \tilde\varepsilon,\\ ||\hat v - v^*||_2^2 &\le \tfrac{1}{L_g}\tilde\varepsilon.\end{aligned}$$

3.2 Step 2.

To estimate the number of iterations of Algorithm 3 note that the MS condition

$$\|\nabla F_{L,\zeta^0}(\hat{\zeta}^k)\|_2 \le \tfrac{L}{2}\|\hat{\zeta}^k - \zeta^0\|_2$$

instead of the exact solution ζ^* of the auxiliary problem in the Algorithm 1, for which

$$\|\nabla F_{L,\zeta^0}(\zeta^*)\|_2 = 0,$$

allows to search inexact solution $\hat{\zeta}^k$.

Since the function $F_{L,\zeta^0}(\cdot)$ is $(L + L_f)$-smooth, we have

$$||\nabla F_{L,\zeta^0}(\hat{\zeta}^k)||_2 \le (L + L_f)||\hat{\zeta}^k - \zeta^*||_2. \tag{13}$$

Using the triangle inequality we have

$$||\zeta^0 - \zeta^*||_2 - ||\hat{\zeta}^k - \zeta^*||_2 \le ||\hat{\zeta}^k - \zeta^0||_2. \tag{14}$$

Since r.h.s. of the inequality (13) coincide with the r.h.s. of the M-S condition and l.h.s. of the inequality (14) coincide with the l.h.s. of the M-S condition up to a multiplicative factor $L/2$, one can conclude that if the inequality

$$||\hat{\zeta}^k - \zeta^*||_2 \le \tfrac{L}{3L+2L_f}||\zeta^0 - \zeta^*||_2$$

holds, the M-S condition holds too, where ζ^0 is a starting point.

We assume that on each iteration of the Algorithm 3 we solve an auxiliary problem (9) in the sense of (11). Then, we provide the following convergence rate theorem for the Algorithm 3:

Theorem 2. *Assume that $\frac{\mu+L}{2L_h} \le 1$. After N iterations of Algorithm 3 we have*

$$F_{L,\zeta^0}(\zeta^N) - F_{L,\zeta^0}(\zeta^*) \le \exp\left(-\tfrac{N(\mu+L)}{4L_h}\right)(F_{L,\zeta^0}(\zeta_0) - F_{L,\zeta^0}(\zeta^*)) + \tfrac{4L_h}{\mu+L}\tilde{\varepsilon},$$

$$\tfrac{1}{2}||\zeta^* - \zeta^N||_2^2 \le \tfrac{L_h}{2(\mu+L)}||\zeta^* - \zeta_0||_2^2 + \tfrac{4L_h}{(\mu+L)^2}\tilde{\varepsilon}.$$

The proof of this Theorem is given in the appendix.

Now we consider the function $F_{L,\zeta^0}(\cdot)$ as an L-strongly convex function, not taking μ into account. From Theorem 2 we obtain that

$$F_{L,\zeta^0}(\zeta^N) - F_{L,\zeta^0}(\zeta^*) \le \tfrac{L_h||\zeta^0-\zeta^*||_2^2}{2}\exp\left(\tfrac{-NL}{4L_h}\right) + \tfrac{4L_h}{L}\tilde{\varepsilon}.$$

From strong convexity of $F_{L,\zeta^0}(\cdot)$, the following inequality holds. [15]

$$\tfrac{L}{2}||\hat{\zeta}^N - \zeta^*||_2^2 \le F_{L,\zeta^0}(\hat{\zeta}^N) - F_{L,\zeta^0}(\zeta^*).$$

Thus, for condition (15) to be satisfied, it is necessary that

$$\tfrac{L_h||\zeta^0-\zeta^*||_2^2}{2}\exp\left(\tfrac{-NL}{L_h}\right) + \tfrac{4L_h}{L}\tilde{\varepsilon} \le \tfrac{L^3}{2(3L+2L_f)^2}||\zeta^0 - \zeta^*||_2^2.$$

Equating each term of l.h.s. to half of the r.h.s. we obtain that the number of iterations of Algorithm 3 is

$$N_{\text{GMCO}} \overset{def}{=} O\Big(\frac{L_h}{L}\ln\Big(\frac{(3L+2L_f)^2 L_h}{L^3}\Big)\Big)$$

and $\tilde{\varepsilon} = \varepsilon_{\mathcal{M}} \overset{def}{=} \frac{L^4}{8L_h(3L+2L_f)^2}||\zeta^0 - \zeta^*||_2^2$.

Assuming that on each iteration of Algorithm 3 we solve the auxiliary problem (10) with probability at least $1 - \delta_{\text{MS}}/N_{\text{GMCO}}$ in the sense of (11) with $\tilde{\varepsilon} = \varepsilon_{\mathcal{M}}$, using the union bound over all N_{GMCO} iterations we obtain

Lemma 2. *In N_{GMCO} iterations of Algorithm 3 with probability at least $1 - \delta_{MS}$ we find $\hat{\zeta}$ such that*

$$\|\nabla F_{L,\zeta^0}(\hat{\zeta})\|_2 \le \tfrac{L}{2}\|\hat{\zeta} - \zeta^0\|_2.$$

3.3 Step 3.

To estimate the number of iterations of Algorithm 1 note that in (6) we apply Algorithm 3 and, according to the stopping criterion of GMCO$(x^{k+1}, F_{L,x^{k+1}})$ obtain y^{k+1} such that

$$\|\nabla F_{L,x^{k+1}}(y^{k+1})\|_2 \le \tfrac{L}{2}\|y^{k+1} - x^{k+1}\|_2. \tag{16}$$

So we can apply the Theorem 3.6 from [11] for Algorithm 1 and obtain that for all $N \ge 0$

$$f\left(y^N\right) - f\left(x^*\right) \le \tfrac{R^2}{2A_N}, \quad \left\|z^N - x^*\right\|_2 \le R, \tag{17}$$

where $R \ge \left\|y^0 - x^*\right\|_2$. Moreover, from Lemma 3.7 a) of [11] for all $N \ge 0$

$$A_N \ge \tfrac{N^2}{4L}. \tag{18}$$

Substituting the inequality (18) into the estimate (17) we obtain that after N iterations of Algorithm 1 the following inequality holds.

$$f(y^N) - f(x_\star) \le \tfrac{2L||x^0 - x^*||_2^2}{N^2}.$$

Thus, if $\mu = 0$, then the total number of iterations of MS is $T^{\text{c}}_{\text{MS}}(\varepsilon) \overset{def}{=} \sqrt{\frac{L||x^0-x^*||_2^2}{\varepsilon}}$.

If $\mu > 0$, to recover the acceleration through strong-convexity we need to apply the restarting strategy. In light of the definition of strong convexity of $f(\cdot)$ and the estimate (19), we have

$$\tfrac{\mu}{2}||y^N - x^*||_2 \le f(y^N) - f(x^*) \le \tfrac{2L}{N^2}||x^0 - x^*||_2^2.$$

In particular, in every $N = N_0 = \sqrt{\frac{8L}{\mu}}$ iterations, we can halve the distance $||y^N - x^*||_2 \le \frac{1}{2}||x^0 - x^*||_2^2$. And if we repeatedly invoke MS$(\cdot, L, N_0)$ t times, each time choosing the initial point x_0 as the previous output y^{N_0}, then in the last run of N_0 iterations, we have

$$f(y^{N_0}) - f(x^*) \le \tfrac{4L}{2^t N_0^2}||x^0 - x^*||_2^2 = \tfrac{1}{2^{t+1}}||x^0 - x^*||_2^2.$$

By choosing $t = \log\left(\frac{||x_0 - x^*||_2^2 \cdot \mu}{\varepsilon}\right)$, we conclude that

Lemma 3. *If $f(\cdot)$ is μ-strongly convex w.r.t. $||\cdot||_2$, then after $T(\varepsilon) \stackrel{def}{=} \log\left(\frac{||\eta_0-\eta^*||_2^2\cdot\mu}{\varepsilon}\right)$ iterations of the Algorithm 2 we obtain some η^T such that $f(\eta^T) - f(\eta^\star) \leq \varepsilon$.*

From this lemma we obtain that the total number of iterations of MS is $T^{\text{sc}}_{\text{MS}}(\varepsilon) \stackrel{def}{=} O\left(\sqrt{\frac{L}{\mu}}\log\left(\frac{||\eta_0-\eta^*||_2^2\cdot\mu}{\varepsilon}\right)\right)$.

Assume that on each iteration of the Algorithm 1 we find y^{k+1} satisfying (16) with probability at least $1-\delta_{\text{MS}}$ with $\delta_{\text{MS}} = \delta/T_{\text{MS}}(\varepsilon)$, where $\delta \in (0,1)$ and

$$T_{\text{MS}}(\varepsilon) = \begin{cases} T^{\text{SC}}_{\text{MS}}(\varepsilon) & \text{if } \mu \geq 0, \\ T^{\text{C}}_{\text{MS}}(\varepsilon) & \text{if } \mu = 0. \end{cases}$$

Using the union bound over all iterations of MS and Lemma 3 for the strongly convex case, we obtain the following lemma.

Lemma 4. *If on each iteration of Algorithm 1 we find y^{k+1} satisfying* (16) *with probability at least $1-\delta_{MS}$ with $\delta_{MS} = \delta/T_{MS}(\varepsilon)$, then*

a) after T iterations of Algorithm 2 for the case $\mu > 0$

b) after T^c_{MS} iterations of Algorithm 1 for the case $\mu = 0$

we obtain that with probability as least $1-\delta$ we find $\hat\eta$ such that $f(\hat\eta) - f(\eta^\star) \leq \varepsilon$.

3.4 Step 4.

Before we give the estimates of the number of oracle calls for $h(\cdot)$ and $g(\cdot)$, we will explain how we plan on obtaining them.

For $h(\cdot)$ we heed to compute the gradient at each step of Algorithm 3, which we run $T_{\text{MS}}(\varepsilon)$ times. Moreover, at each iteration of Algorithm 1 in step (7) we compute the gradient of $f(\cdot)$, so we also need to compute the gradient of $h(\cdot)$.

For $g(\cdot)$ we heed to compute the gradient at each step of the inner algorithm $\mathcal{M}_{inn}$, which we run at each iteration of Algorithm 3, and at each iteration of Algorithm 1 in step (7) we also need to compute the gradient of $g(\cdot)$.

Note that using the triangle inequality we have

$$||v^0 - v^*||_2 \leq ||v^0 - \hat v||_2 + ||\hat v - v^*||_2. \tag{20}$$

And at each iteration of Algorithm 3 we use the method $\mathcal{M}_{inn}$ with starting point ζ^k to compute the point ζ^{k+1}. So for the k-th iteration of Algorithm 3 we have $v^0 \equiv \zeta^k$ and $\hat v \equiv \zeta^{k+1}$. Using the triangle inequality and Theorem 2, we have

$$\begin{aligned} ||\zeta^k - \zeta^{k+1}||_2 &\leq ||\zeta^k - \zeta^*||_2 + ||\zeta^{k+1} - \zeta^*||_2 \\ &\leq 2\sqrt{\frac{L_h}{L}}||\zeta^0 - \zeta^*||_2 + 2\sqrt{\frac{2L_h}{L^2}\tilde\varepsilon}. \end{aligned}$$

Then, using (12), from (20) we have

$$\begin{aligned} &||v^0 - v^*||_2 \leq ||v^0 - \hat v||_2 + \sqrt{\tilde\varepsilon/L_g} \\ \leq\ & \sqrt{\frac{4L_h}{L}}||\zeta^0 - \zeta^*||_2 + \sqrt{\tilde\varepsilon}\left(\sqrt{\frac{8L_h}{L^2}} + \sqrt{\frac{1}{L_g}}\right). \end{aligned}$$

Choosing the $\tilde{\varepsilon} = \varepsilon_{\mathcal{M}}$ and using Lemma 1 we obtain that we need $N_{\mathcal{M}}(\varepsilon_{\mathcal{M}}) = O\left(\frac{\tau_g}{\sqrt{L+L_h}} \ln \frac{2CL_g\|v^0 - v^*\|_2^2}{\tilde{\delta}\varepsilon_{\mathcal{M}}(L+L_h)}\right)$ Oracle calls of $\mathcal{M}_{inn}$. And using (21) we obtain that in $\tilde{N}_{\mathcal{M}} = O\left(\frac{\tau_g}{\sqrt{L+L_h}} \ln \frac{C_1}{\tilde{\delta}}\right) \geq N_{\mathcal{M}}(\varepsilon_{\mathcal{M}})$ Oracle calls the Lemma 1 holds with $\tilde{\varepsilon} = \varepsilon_{\mathcal{M}}$ and $C_1 = \frac{2C(32L_h^2 L_g(3L+2L_f)^2 + (8L_hL_g+L^2)L^3}{L^4(L+L_h)}$.

Each time choosing $\tilde{\delta} \approx \delta_{\mathsf{MS}}/N_{\mathsf{GMCO}} = \delta/(N_{\mathsf{GMCO}} \cdot T_{\mathsf{MS}})$ we obtain that we need $N_{\mathcal{M}}^{\mathsf{SC}} = O\left(\frac{\tau_g}{\sqrt{L+L_h}} \ln \frac{C_1 L_h}{\delta\sqrt{\mu L}}\right)$ Oracle calls of $\mathcal{M}_{inn}$ for the strongly convex case and $N_{\mathcal{M}}^{\mathsf{C}}(\varepsilon) = O\left(\frac{\tau_g}{\sqrt{L+L_h}} \ln \frac{C_1 L_h R}{\delta\sqrt{\varepsilon L}}\right)$ Oracle calls for the convex case, where $R \geq \|x^0 - x^*\|_2$.

Using the union bound over all launches of $\mathcal{M}_{inn}$, we obtain that with probability at least $1-\delta$ we can find such $\hat{x}$ that $f(\hat{x}) - f(x^*) \leq \varepsilon$, and to do this we need $O\left(\sqrt{\frac{L}{\mu}} \log\left(\frac{\mu R^2}{\varepsilon}\right) \cdot \left(1 + \frac{L_h}{L}\right)\right)$ Oracle calls for $h(\cdot)$ and

$$O\left(\sqrt{\frac{L}{\mu}} \log\left(\frac{\mu R^2}{\varepsilon}\right) \cdot \left(\kappa_g + \frac{L_h}{L} \cdot \left(\frac{\tau_g}{\sqrt{L+L_h}} \ln \frac{1}{\delta}\right)\right)\right)$$

Oracle calls for $g(\cdot)$, if $\mu > 0$,

and $O\left(\sqrt{\frac{LR^2}{\varepsilon}} \cdot \left(1 + \frac{L_h}{L}\right)\right)$ Oracle calls for $h(\cdot)$ and

$$O\left(\sqrt{\frac{LR^2}{\varepsilon}} \cdot \left(\kappa_g + \frac{L_h}{L} \cdot \left(\frac{\tau_g}{\sqrt{L+L_h}} \ln \frac{R}{\delta\sqrt{\varepsilon}}\right)\right)\right)$$

Oracle calls for $g(\cdot)$, if $\mu = 0$.

4 Applications

In this section, we present a few examples of algorithms that we consider as $\mathcal{M}_{inn}$.

4.1 Accelerated Gradient Method for Composite Optimization

Consider the following unconstrained problem

$$\min_{x \in \mathbb{R}^n} f(x) := h(x) + g(x).$$

We assume that the function $g(\cdot)$ is L_g-smooth w.r.t. $\|\cdot\|_2$. To solve this problem we consider the Accelerated Gradient Method for Composite Optimization from [14]. For this method the Assumption 4 holds with $\tau_g = \sqrt{L_g}$ if $L_g \geq L_h$.

As the basic oracle O_g we have a first order oracle which computes the full gradient $\nabla g(\cdot)$ in one oracle call, so, for this case $\kappa_g = 1$.

Minimizing the number of Oracle calls for $g(\cdot)$, we obtain that the optimal value of L is L_h. We can then state the following corollary of Theorem 1:

Corollary 1. *Using the Accelerated Gradient Method for Composite Optimization as $\mathcal{M}_{inn}$ we can obtain $\hat{x}$ such that $f(\hat{x}) - f(x^*) \leq \varepsilon$ in* *a)* $O\left(\sqrt{\frac{L_h R^2}{\varepsilon}}\right)$ *Oracle calls for* $h(\cdot)$, $O\left(\sqrt{\frac{L_g R^2}{\varepsilon}}\right)$ *Oracle calls for* $g(\cdot)$, *if* $\mu = 0$, *and*

b) $\tilde{O}\left(\sqrt{\frac{L_h}{\mu}}\right)$ *Oracle calls for* $h(\cdot)$ *and* $\tilde{O}\left(\sqrt{\frac{L_g}{\mu}}\right)$ *Oracle calls for* $g(\cdot)$, *if* $\mu > 0$.

4.2 Accelerated Proximal Coordinate Descent Method

Consider the following unconstrained problem

$$\min_{x \in \mathbb{R}^n} f(x) := h(x) + g(x).$$

Now we assume the directional smoothness for $g(\cdot)$, that is that there exist$\beta_1, \ldots, \beta_n$ such that for any $x \in \mathbb{R}^n, u \in \mathbb{R}$

$$|\nabla_i g\left(x + ue_i\right) - \nabla_i g(x)| \leq \beta_i |u|, \quad i = 1, \ldots, n,$$

where $\nabla_i g(x) = \partial g(x) / \partial x_i$. For twice differentiable $g(\cdot)$ it is equivalent to the condition $(\nabla^2 g(x))_{i,i} \leq \beta_i$. In this case we consider the Accelerated Proximal Coordinate Gradient Method from [13, 16, 4, 5] as the inner method $\mathcal{M}_{inn}$. For this method Assumption 4 holds with $\tau_g = n\sqrt{\overline{L}_g}$, where $\sqrt{\overline{L}_g} = \frac{1}{n}\sum_{i=1}^{n}\sqrt{\beta_i}$, if $\overline{L}_g \geq L_h$.

As the basic oracle O_g we have an oracle which computes a partial derivative $\nabla_i g(\cdot)$ in one iteration. For this case we need $\kappa_g = n$ calls to O_g to compute the full gradient $\nabla g(\cdot)$.

Minimizing the number of Oracle calls for $g(\cdot)$, we obtain that the optimal $L = L_h$, so we can state the following corollary from Theorem 1:

Corollary 2. *Using the Accelerated Gradient Method for Composite Optimization as* $\mathcal{M}_{inn}$ *we can obtain* $\hat{x}$ *such that* $f(\hat{x}) - f(x^*) \leq \varepsilon$ *in*

a) $O\left(\sqrt{\frac{L_h R^2}{\varepsilon}}\right)$ *Oracle calls for* $h(\cdot)$, $O\left(n \cdot \sqrt{\frac{\overline{L}_g R^2}{\varepsilon}}\right)$ *Oracle calls for* $g(\cdot)$, *if* $\mu = 0$, *and*

b) $\tilde{O}\left(\sqrt{\frac{L_h}{\mu}}\right)$ *Oracle calls for* $h(\cdot)$ *and* $\tilde{O}\left(n \cdot \sqrt{\frac{\overline{L}_g}{\mu}}\right)$ *Oracle calls for* $g(\cdot)$, *if* $\mu > 0$.

Note, that if $\mathcal{M}_{inn}$ is a directional search or a derivative-free method such as in [3], then the main conclusions of corollary 2 remain valid after replacing $\overline{L}_g$ on L_g.

4.3 Accelerated Stochastic Variance Reduced Algorithm

Consider the following minimization problem

$$\min_{x \in \mathbb{R}^n} f(x) := h(x) + \frac{1}{m}\sum_{k=1}^{m} g_k(x).$$

We assume that each component $g_k(\cdot)$ is smooth with the constant L_{g_k}. To solve this problem we can use the Katyusha [1] and other Accelerated Stochastic Variance Reduced Algorithms [10, 8] in place of the inner method $\mathcal{M}_{inn}$. Note that for the Accelerated Stochastic Variance Reduced Algorithms the estimate of the number of oracle calls for problem (4) is $\tilde{O}\left(m + \sqrt{\frac{m\hat{L}_g}{L}}\right)$ if $\hat{L}_g \geq L$, where

$\hat{L}_g = \max_k L_{g_k}$. If we additionally assume that $L_h m \leq \hat{L}_g$, then for this method Assumption 4 holds with $\tau_g = \sqrt{m\hat{L}_g}$.

As the basic oracle O_g we have an oracle which computes $\nabla g_k(\cdot)$ in one iteration. Hence, in this case we need $\kappa_g = m$ basic oracle O_g calls to compute the full gradient $\nabla g(\cdot)$.

Corollary 3. *Using the Accelerated Gradient Method for Composite Optimization as $\mathcal{M}_{inn}$ we can obtain $\hat{x}$ such that $f(\hat{x}) - f(x^*) \leq \varepsilon$ in*

a) $O\left(\sqrt{\frac{L_h R^2}{\varepsilon}}\right)$ Oracle calls for $h(\cdot)$, $O\left(\sqrt{\frac{m\hat{L}_g R^2}{\varepsilon}}\right)$ Oracle calls for $g(\cdot)$, if $\mu = 0$, and

b) $\tilde{O}\left(\sqrt{\frac{L_h}{\mu}}\right)$ Oracle calls for $h(\cdot)$ and $\tilde{O}\left(\sqrt{\frac{m\hat{L}_g}{\mu}}\right)$ Oracle calls for $g(\cdot)$, if $\mu > 0$.

Condition $L_h m \leq \hat{L}_g$ might seem very restrictive, but there exists a class of problems with non-smooth g_k that is well suited to this condition. Assume that the convex conjugates g_k^* are proximal-friendly. In particular, this is the case for generalized linear model [18] $g_k(x) := g_k(\langle a_k, x\rangle)$. In this case we can apply the Nesterov's smoothing technique [12, 2] and regularize the convex conjugate functions g_k^* with coefficient $\sim \varepsilon$. Since all g_k^* are proximal-friendly, we can efficiently compute the conjugate function to the resulting regularized function. This allows us to build an ε-approximation of initial problem with $\hat{L}_g \sim 1/\varepsilon$. In Section **??** we demonstrate how this approach works on the *Kernel SVM* example.

5 Experiments

In this section, we present experimental results of applying Algorithm 1 to the real-world machine learning problems, and demonstrate its effectiveness. More detailed theoretical explanation of described below results see in Appendix B.

5.1 Log-density estimation with Gaussian Prior

To estimate the log-density of some measure P [20] we suppose that we observe only m random observations $\tilde{z}_1, \dots, \tilde{z}_m \in \mathcal{Z}$ generated from this measure. Without loss of generality, we assume that $\mathcal{Z}$ has finite support $\{z_k\}_{k=1}^p$ of size p, then

$$\sum_{k=1}^{p} f(z_k) = 1. \tag{22}$$

We parameterize the log-density by the linear model

$$\log f(z) = l(z, x^*) = \sum_{i=1}^{n} x_i^* a_i(z) - c(x^*),$$

where $a_1(z), a_2(z), \dots, a_n(z)$ are given basis functions and $x^* \in \mathbb{R}^n$ is an unknown vector, corresponding to the actual density. The normalization constant $c(x^*)$ is determined using (22):

$$c(x) = \log\left(\sum_{k=1}^{p} \exp\left(\langle A_k, x\rangle\right)\right),$$

where $A_k = a(z_k) = (a_1(z_k), \ldots, a_n(z_k))^T$ is the k-th column of $A = [a_j(z_k)]_{j,k=1}^{n,p}$. From [20] it's known that x^* can be alternatively defined as

$$x^* = \arg\max_{x\in Q}\{\langle \mathbb{E}_z[a(z)], x\rangle - mc(x)\}.$$

It's also known (Fisher theorem) that Maximum Likelihood Estimation (MLE)

$$\tilde{x} = \arg\max_{x\in\mathbb{R}^n}\{\sum_{k=1}^{m}\langle a(\tilde{z}_k), x\rangle - mc(x)\}.$$

will be a good estimation of x^*. Moreover, if we introduce Gaussian prior $\mathcal{N}(0, G^2)$ for x^*, MLE changes as follows

$$\tilde{x}_G = \arg\max_{x\in\mathbb{R}^n}\{\sum_{k=1}^{m}\langle a(\tilde{z}_k), x\rangle - mc(x) - \frac{1}{2}\|Gx\|^2\}. \qquad (23)$$

Bernstein–von Mises theorem claims [20], that $\tilde{x}_G$ is a good estimation of x^* in Bayesian set up.

Particular case when matrix A is sparse and all elements of G^2 are from the interval [1,2], is considered in the paper. Modern Accelerated Coordinate Descent algorithms don't allow to take into account sparsity of matrix A [4], so for the first two terms in $\arg\max$ of RHS of (23) it'd be better to use common accelerated method [15]. The third (last) term in (23) is vice versa very friendly for Accelerated Coordinate Descent [16]. So this problem formulation for relatively small m (or relatively large G^2) is well suited for splitting scheme with $\mathcal{M}_{inn}$ to be Accelerated Coordinate Descent.

Based on the problem statement, let us consider the optimization problem with the following objective function:

$$f(x) = \log\left(\sum_{k=1}^{p}\exp\left(\langle A_k, x\rangle\right)\right) + \frac{1}{2}\|Gx\|_2^2 \to \min_{x\in\mathbb{R}^n}.$$

In our case, $n = 500$, $p = 6000$, A is a sparse $p \times n$ matrix with sparsity coefficient 0.001, whose non-zero elements are drawn randomly from $\mathcal{U}(-1, 1)$, and matrix G^2 generated as follows:

$$G^2 = \sum_{i=1}^{n}\lambda_i \tilde{e}_i^\top \tilde{e}_i,$$

where $\sum_{i=1}^{n}\lambda_i = 1$ and $[\tilde{e}_i]_j \sim \mathcal{U}(1, 2)$ for every i, j.

The Lipschitz constant for the first term of f calculated according to the following formula:

$$L_h = \max_{i=1,\ldots,n}\|A^{\langle k\rangle}\|_2^2,$$

where $A^{\langle k\rangle}$ denotes the k-th column of A, $L = 25L_h$ and directional Lipschitz constants for the φ from (10) are $L_i = G_{ii}^2 + L + L_h$.

Below there are given the result of experiments[1] for Fast Coordinate Descent [16] with $\beta = 1/2$ being restarted every 300 iterations as $\mathcal{M}_{inn}$. The vertical axis of the both 1 and 2 figures measures function value $f(x^i)$ in logarithmic scale, the horizontal axis of figures 1 and 2 measures physical working time.

However, although for some problems, as above, it is possible to achieve convergence acceleration using a Monteiro-Svaiter envelope, many additional experiments have shown that, in the general case,

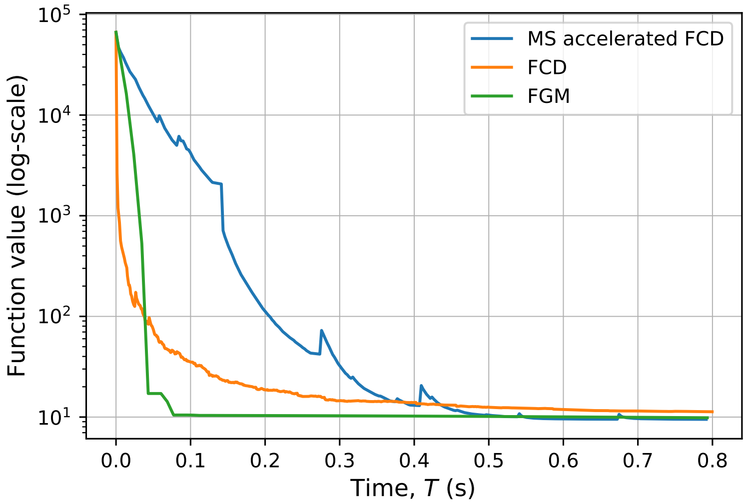

Figure 1: M-S accelerated Fast Coordinate Descent, function value $f(x^i)$ vs working time

it does not give a significant improvement in the performance of the Fast Gradient Method. Nevertheless, experiments show that the convergence of the proposed method, in practice, corresponds to the estimates obtained in the article.

We also compare the methods by the number of $\nabla h_i(\cdot)$ and $\nabla g_i(\cdot)$ oracles calls. Figure 3 shows a three-dimensional plot of the function value $f(x^i)$ in logarithmic scale vs the number of $\nabla h_i(\cdot)$ and $\nabla g_i(\cdot)$ oracles calls and two-dimensional projections of this plot for the $\nabla h_i(\cdot)$ and $\nabla g_i(\cdot)$ oracles respectively. Since some of the methods involve calculating the full gradient ($\nabla h(\cdot)$ or $\nabla g(\cdot)$), the oracles calls, in this case, accounted for with a weight of $t_1/t_2 \approx 2$, where t_1 — is the average full gradient computing time, t_2 — is the average time to calculate only the component of the gradient.

As can be noted from the plots, M-S accelerated version of FCD requires a significantly smaller number of $\nabla h_i(\cdot)$ oracle calls than FCD and FGM.

[1] Source code of these experiments are on GitHub: `https://github.com/ICML2020-OracleComplexitySeparation/Oracle-Complexity-Separation`

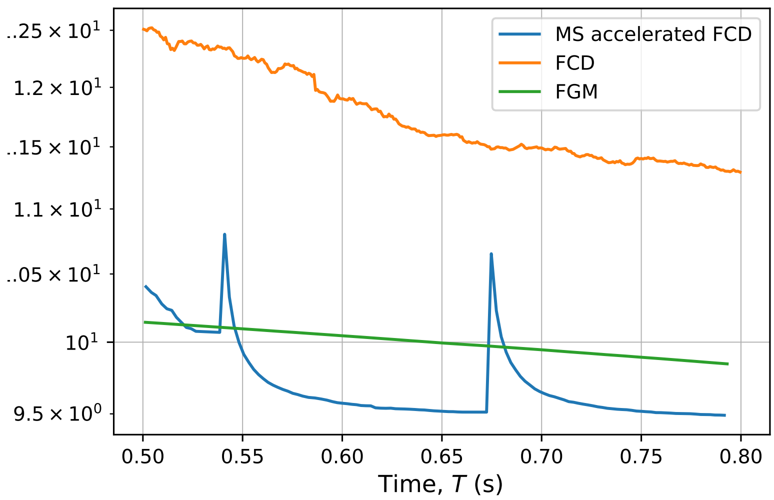

Figure 2: M-S accelerated Fast Coordinate Descent, function value $f(x^i)$ vs working time (from 0.5 s)

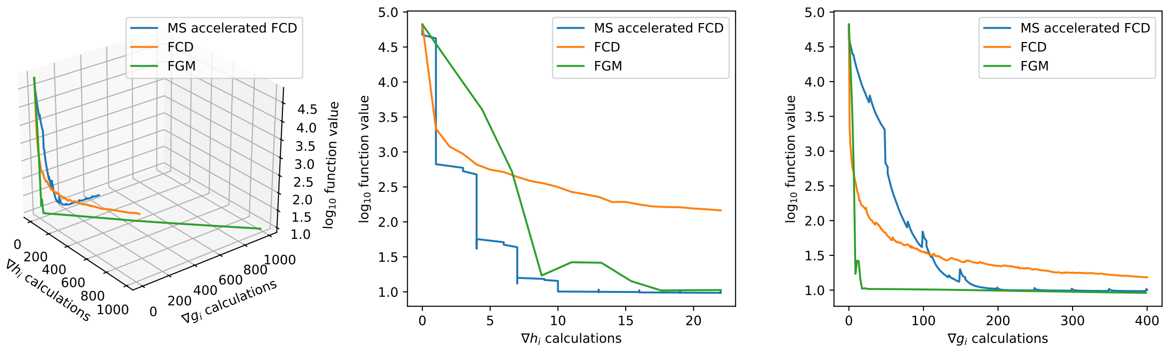

Figure 3: M-S accelerated Fast Coordinate Descent, function value $f(x^i)$ vs number of $\nabla h_i(\cdot)$ and $\nabla g_i(\cdot)$ oracles calls

References

[1] Z. Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. *The Journal of Machine Learning Research*, 18(1):8194–8244, 2017.

[2] Z. Allen-Zhu and E. Hazan. Optimal black-box reductions between optimization objectives. In *Advances in Neural Information Processing Systems*, pages 1614–1622, 2016.

[3] P. Dvurechensky, A. Gasnikov, and A. Tiurin. Randomized similar triangles method: A unifying framework for accelerated randomized optimization methods (coordinate descent, directional search, derivative-free method). *arXiv:1707.08486*, 2017.

[4] O. Fercoq and P. Richtárik. Accelerated, parallel, and proximal coordinate descent. *SIAM Journal on Optimization*, 25(4):1997–2023, 2015.

[5] A. Gasnikov, P. Dvurechensky, and I. Usmanova. About accelerated randomized methods. *Proceedings of the Moscow Institute of Physics and Technology*, 8(2 (30)), 2016.

[6] A. Ivanova, D. Grishchenko, A. Gasnikov, and E. Shulgin. Adaptive catalyst for smooth convex optimization. *arXiv preprint arXiv:1911.11271*, 2019.

[7] D. Kamzolov, A. Gasnikov, and P. Dvurechensky. On the optimal combination of tensor optimization methods. *arXiv:2002.01004*, 2020.

[8] G. Lan, Z. Li, and Y. Zhou. A unified variance-reduced accelerated gradient method for convex optimization. In *Advances in Neural Information Processing Systems*, pages 10462–10472, 2019.

[9] G. Lan and Y. Ouyang. Accelerated gradient sliding for structured convex optimization. *arXiv preprint arXiv:1609.04905*, 2016.

[10] G. Lan and Y. Zhou. An optimal randomized incremental gradient method. *Mathematical programming*, 171(1-2):167–215, 2018.

[11] R. D. Monteiro and B. F. Svaiter. An accelerated hybrid proximal extragradient method for convex optimization and its implications to second-order methods. *SIAM Journal on Optimization*, 23(2):1092–1125, 2013.

[12] Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 2005.

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

[14] Y. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013. First appeared in 2007 as CORE discussion paper 2007/76.

[15] Y. Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018.

[16] Y. Nesterov and S. U. Stich. Efficiency of the accelerated coordinate descent method on structured optimization problems. *SIAM Journal on Optimization*, 27(1):110–123, 2017.

[17] B. Schölkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning)*. MIT Press, 2001.

[18] S. Shalev-Shwartz, O. Shamir, N. Srebro, and K. Sridharan. Stochastic convex optimization. In *COLT*, 2009.

[19] S. Shalev-Shwartz and T. Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. In E. P. Xing and T. Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 64–72, Bejing, China, 22–24 Jun 2014. PMLR. First appeared in arXiv:1309.2375.

[20] V. Spokoiny and M. Panov. Accuracy of gaussian approximation in nonparametric bernstein–von mises theorem. *arXiv preprint arXiv:1910.06028*, 2019.

[21] Y. Zhang and L. Xiao. Stochastic primal-dual coordinate method for regularized empirical risk minimization. In F. Bach and D. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 353–361, Lille, France, 07–09 Jul 2015. PMLR.

A Proof of Theorem

Definition 1. *For a convex optimization problem* $\min_{x\in Q}\Psi(x)$*, we denote by Arg* $\min_{x\in Q}^{\widetilde{\delta}}\Psi(x)$ *a set of such* $\widetilde{x}$ *that*

$$\exists h \in \partial\Psi(\widetilde{x}) : \forall x \in Q \ \rightarrow \ \langle h, x - \widetilde{x}\rangle \geq -\widetilde{\delta}.$$

We denote by $\operatorname{argmin}_{x\in Q}^{\widetilde{\delta}}\Psi(x)$ *some element of Arg* $\min_{x\in Q}^{\widetilde{\delta}}\Psi(x)$.

Algorithm 4 Gradient method for Composite Optimization GMCO$(x_0, F(\cdot))$

1: **Parameters:** starting point $x_0 \in \mathbb{R}^n$, objective function $F(x) = f(x)+p(x)$, constant L (function f with L Lipschitz gradient w.r.t. the $||\cdot||_2$), error $\widetilde{\delta}$.

2: **for** $k = 0, \dots, N-1$ **do**

3: Set

$$\phi_{k+1}(x) := \langle\nabla f(x_k), x - x_k\rangle + p(x) + \tfrac{L}{2}\|x - x_k\|_2^2,$$

4: Compute

$$x_{k+1} := \operatorname*{argmin}_{x\in Q}{}^{\widetilde{\delta}}(\phi_{k+1}(x)) \tag{24}$$

5: **end for**

6: **Output:** x_N

Lemma 5. *Let* $\psi(x)$ *be a convex function and*

$$y = \operatorname*{argmin}_{x\in Q}{}^{\widetilde{\delta}}\{\psi(x) + \tfrac{\beta}{2}||z - x||_2^2\},$$

where $\beta \geq 0$*. Then*

$$\psi(x) + \tfrac{\beta}{2}||z - x||_2^2 \geq \psi(y) + \tfrac{\beta}{2}||z - y||_2^2 + \tfrac{\beta}{2}||x - y||_2^2 - \widetilde{\delta}, \ \ \forall x \in Q.$$

Proof. By Definition 1:

$$\exists g \in \partial\psi(y), \ \ \langle g + \tfrac{\beta}{2}\nabla_y||y - x||_2^2, x - y\rangle = \langle g + \beta(y - z), x - y\rangle \geq -\widetilde{\delta}, \ \ \forall x \in Q.$$

From β–strong convexity of $\psi(x) + \frac{\beta}{2}||z - x||_2^2$ we have

$$\psi(x) + \tfrac{\beta}{2}||z - x||_2^2 \geq \psi(y) + \tfrac{\beta}{2}||z - y||_2^2 + \langle g + \tfrac{\beta}{2}\nabla_y||y - x||_2^2, x - y\rangle + \tfrac{\beta}{2}||x - y||_2^2$$

The last two inequalities complete the proof. □

The next theorem proves convergence rate of Algorithm 4 for optimization problem

$$\min_{x\in\mathbb{R}^n} F(x) := f(x) + p(x),$$

where function f is convex function with L Lipschitz gradient w.r.t. the $||\cdot||_2$ norm, function p is convex function and function F is μ–strongly convex.

Theorem 3. *Let us assume that $\frac{\mu}{2L} \leq 1$. After N iterations of Algorithm 4 we have*

$$F(x_N) - F(x_*) \leq \exp\left(-\frac{N\mu}{4L}\right)(F(x_0) - F(x_*)) + \frac{4L}{\mu}\widetilde{\delta},$$

$$\frac{1}{2}||x_* - x_N||_2^2 \leq \frac{L}{2\mu}||x_* - x_0||_2^2 + \frac{4L}{\mu^2}\widetilde{\delta}.$$

Proof of Theorem 3. Since gradient of function F is L Lipschitz w.r.t. the $||\cdot||_2$ norm, we have

$$F(x_N) \leq f(x_{N-1}) + \langle \nabla f(x_{N-1}), x_N - x_{N-1}\rangle + p(x_N) + \frac{L}{2}||x_{N-1} - x_N||_2^2.$$

From Lemma 5 and auxiliary problem (24) we get

$$F(x_N) \leq f(x_{N-1}) + \langle \nabla f(x_{N-1}), x - x_{N-1}\rangle$$
$$+ p(x) + \frac{L}{2}||x - x_{N-1}||_2^2 + \widetilde{\delta}.$$

In view of convexity of function f, we obtain

$$F(x_N) \leq F(x) + \frac{L}{2}||x - x_{N-1}||_2^2 + \widetilde{\delta}.$$

We rewrite the last inequality for $x = \alpha x_* + (1-\alpha)x_{N-1}$ ($\alpha \in [0,1]$) as

$$F(x_N) \leq F(\alpha x_* + (1-\alpha)x_{N-1}) + \frac{L\alpha^2}{2}||x_* - x_{N-1}||_2^2 + \widetilde{\delta}.$$

In view of convexity of function f, we have

$$F(x_N) \leq F(x_{N-1}) - \alpha(F(x_{N-1}) - F(x_*)) + \frac{L\alpha^2}{2}||x_* - x_{N-1}||_2^2 + \widetilde{\delta}.$$

From μ–strong convexity of function F we have $F(x_{N-1}) \geq F(x_*) + \frac{\mu}{2}||x_* - x_{N-1}||_2^2$, this yields inequality:

$$F(x_N) \leq F(x_{N-1}) - \alpha\left(1 - \alpha\frac{L}{\mu}\right)(F(x_{N-1}) - F(x_*)) + \widetilde{\delta}.$$

The minimum of the right part of the last inequality is achieved with $\alpha = \min(1, \frac{\mu}{2L})$. Due to $\frac{\mu}{2L} \leq 1$ with $\alpha = \frac{\mu}{2L}$ we have

$$F(x_N) - F(x_*) \leq \left(1 - \frac{\mu}{4L}\right)(F(x_{N-1}) - F(x_*)) + \widetilde{\delta}.$$

and

$$F(x_N) - F(x_*) \leq \left(1 - \frac{\mu}{4L}\right)^N (F(x_0) - F(x_*)) + \frac{4L}{\mu}\widetilde{\delta}$$
$$\leq \exp\left(-\frac{N\mu}{4L}\right)(F(x_0) - F(x_*)) + \frac{4L}{\mu}\widetilde{\delta}.$$

From μ–strong convexity of function F and the fact that gradient of function F is L Lipschitz we obtain

$$\frac{1}{2}||x_* - x_N||_2^2 \leq \frac{L}{2\mu}\exp\left(-\frac{N\mu}{4L}\right)||x_* - x_0||_2^2 + \frac{4L}{\mu^2}\widetilde{\delta}$$
$$\leq \frac{L}{2\mu}||x_* - x_0||_2^2 + \frac{4L}{\mu^2}\widetilde{\delta}.$$

□

B Motivation for examples

B.1 Kernel SVM

Let's consider the following function

$$f(x) = \frac{1}{m}\sum_{k=1}^{m} f_k(\langle A_k, x\rangle) + \frac{1}{2}\langle x, Cx\rangle.$$

We assume that $|f_k''(y)| = O(1/\epsilon)$, matrix $A = [A_1, ..., A_m]^T$ has ms nonzero elements, $\max_{k=1,...,m} \|A_k\|_2^2 = O(s)$, where $1 \ll s \leq n$ and C is positive semidefinite matrix,[2] with $\lambda_{\max}(C) \leq 1/(\varepsilon m)$. Fast Gradient Method [15] requires

$$O\left(\sqrt{\frac{(s/\varepsilon + \lambda_{\max}(C))\, R^2}{\varepsilon}}\right)$$

iterations with the complexity of each iteration[3]

$$O\left(ms + n^2\right).$$

For proposed in this paper approach we have

$$\tilde{O}\left(\sqrt{\frac{\lambda_{\max}(C) R^2}{\varepsilon}}\right)$$

iterations of FGM for the second term in target function with the complexity of each iteration

$$O(n^2)$$

and

$$\tilde{O}\left(\sqrt{\frac{(ms/\varepsilon)\, R^2}{\varepsilon}}\right)$$

iterations of variance reduction algorithm [1] with the complexity of each

$$O(s).$$

We combine all these results in the table below. From the table one can conclude that since $s \gg 1$, $\lambda_{\max}(C) \leq 1/(\varepsilon m) \ll s/\varepsilon$, then our approach has better theoretical complexity.

Algorithm	Complexity	Reference
FGM	$O\left(\frac{R}{\varepsilon}\sqrt{s}\,(ms + n^2)\right)$	[15]
Our approach	$\tilde{O}\left(\frac{R}{\varepsilon}\sqrt{ms} \cdot s\right) + \tilde{O}\left(\sqrt{\frac{\lambda_{\max}(C)R^2}{\varepsilon}} \cdot n^2\right)$	this paper

[2] Here an below we also assume after 'semidefinite' that valuable part of the spectrum of dens matrix C lies in a small (right) vicinity of zero point on real line.

[3] To obtain the part of the complexity $O(ms)$ one should use special representation of matrix A in the memory – adjacency list. The same requirements take place for other algorithms.

B.2 Soft-max plus quadratic form

Let's consider the following function

$$f(x) = \log\left(\sum_{k=1}^{p} \exp\left(\langle A_k, x\rangle\right)\right) + \frac{1}{2}\|Gx\|_2^2 \to \min_{x\in\mathbb{R}^n}.$$

We introduce matrix $A = [A_1, ..., A_p]^T$, is such that $\max_{ij}|A_{ij}| = O(1)$, $\max_{j=1,...,n}\|A^{<j>}\|_2^2 = O(n)$ and A has $O(ps)$ nonzero elements; G^2 is positive semidefinite matrix with $\lambda_{\max}(G^2) = O(n)$ and $\frac{1}{n}\sum_{i=1}^n \sqrt{G_{ii}^2} = O(1)$.

Fast Gradient Method [15] requires

$$O\left(\sqrt{\frac{\left(\max_{j=1,...,n}\|A^{<j>}\|_2^2 + \lambda_{\max}(G^2)\right)R^2}{\varepsilon}}\right)$$

iterations with the complexity of each iteration

$$O\left(ps + n^2\right).$$

Coordinate Fast Gradient Method [16] requires

$$O\left(n\sqrt{\frac{\left(\max_{ij}|A_{ij}|^2 + \left(\frac{1}{n}\sum_{i=1}^n\sqrt{G_{ii}^2}\right)^2\right)R^2}{\varepsilon}}\right)$$

iterations with the complexity of each iteration[4]

$$O\left(p + n\right).$$

For proposed in this paper approach we have

$$\tilde{O}\left(\sqrt{\frac{\left(\max_{j=1,...,n}\|A^{<j>}\|_2^2 + \lambda_{\max}(G^2)\right)R^2}{\varepsilon}}\right)$$

iterations of FGM for the first term in target function with complexity of each iteration

$$O(ps)$$

and

$$\tilde{O}\left(n\sqrt{\frac{\left(\frac{1}{n}\sum_{i=1}^n\sqrt{G_{ii}^2}\right)^2 R^2}{\varepsilon}}\right)$$

iterations of coordinate FGM for the second term in target function with complexity of each iteration

$$O(n).$$

We combine all these results in the table below. From the table one can conclude that if $n \ll p$, $s \ll \min\{n^2/p, \sqrt{n}\}$, then our approach has better theoretical complexity.

[4]Here one should use a following trick in recalculation of $\log\left(\sum_{k=1}^p \exp\left(\langle A_k, x\rangle\right)\right)$ and its gradient (partial derivative). From the structure of the method we know that $x^{new} = \alpha x^{old} + \beta e_i$, where e_i is i-th orth. So if we've already calculate $\langle A_k, x^{old}\rangle$ then to recalculate $\langle A_k, x^{new}\rangle = \alpha\langle A_k, x^{old}\rangle + \beta[A_k]_i$ requires only $O(1)$ additional operations independently of n and s.

Algorithm	Complexity	Reference
FGM	$O\left(\sqrt{\frac{nR^2}{\varepsilon}}\,(ps+n^2)\right)$	[15]
coordinate FGM	$O\left(n\sqrt{\frac{R^2}{\varepsilon}}\,(p+n)\right)$	[16]
Our approach	$\tilde{O}\left(\sqrt{\frac{nR^2}{\varepsilon}}\cdot ps\right)+\tilde{O}\left(n\sqrt{\frac{R^2}{\varepsilon}}\cdot n\right)$	this paper