Adaptive Catalyst for Smooth Convex Optimization

Anastasiya Ivanova[1,2] Dmitry Pasechnyuk[3] Dmitry Grishchenko[4]

Egor Shulgin[1,5] Alexander Gasnikov[1,2,6]

February 24, 2020

Abstract

In this paper, we present the generic framework that allows accelerating almost arbitrary non-accelerated deterministic and randomized algorithms for smooth convex optimization problems. The main approach of our *envelope* is the same as in *Catalyst* (Lin et al., 2015): an accelerated proximal outer gradient method, which is used as an envelope for the non-accelerated inner method for the ℓ_2 regularized auxiliary problem. Our algorithm has two key differences: 1) easily verifiable stopping criteria for inner algorithm; 2) the regularization parameter is capable of modification. As a result, the main contribution of this paper is a new framework that applies to adaptive inner algorithms: Steepest Descent, Adaptive Coordinate Descent, Alternating Minimization. Moreover, in the non-adaptive case, our approach allows obtaining Catalyst without a logarithmic factor, which appears in the standard Catalyst (Lin et al., 2015, 2018).

Keywords— Adaptive Methods, Catalyst, Accelerated Methods, Steepest Descent, Coordinate Descent, Alternating Minimization, Distributed Methods, Stochastic Methods.

1 Introduction

One of the main achievements in numerical methods for convex optimization is the development of accelerated methods (Nesterov, 2018). Until 2015 acceleration schemes for different convex optimization problems seem to be quite different to unify them. But starting from the work (Lin et al., 2015) in which universal acceleration technique (*Catalyst*) was proposed, there appears a stream of subsequent works (Lin et al., 2018; Palaniappan & Bach, 2016; Paquette et al., 2017; Kulunchakov & Mairal, 2019) that allows spreading Catalyst on monotone variational inequalities, non-convex problems, stochastic

[1]Moscow Institute of Physics and Technology, Moscow, Russia
[2]National Research University Higher School of Economics, Moscow, Russia
[3]Saint Petersburg Lyceum 239, Russia
[4]Université Grenoble Alpes, Grenoble, France
[5]King Abdullah University of Science and Technology, Thuwal, Saudi Arabia
[6]Institute for Information Transmission Problems, Moscow, Russia

*The research of A. Gasnikov was supported by Russian Science Foundation project 18-71-00048 mol-a-ved and was partially supported by Yahoo! Research Faculty Engagement Program.

optimization problems. In all these works, the basic idea is to use an accelerated proximal algorithm as an outer envelope (Rockafellar, 1976) with non-accelerated algorithms for inner auxiliary problems. The main practical drawback of this approach is the requirement to choose a regularization parameter such that the conditional number of the auxiliary problem becomes became $O(1)$. To do that, we need to know the smoothness parameters of the target that are not typically free available.

An alternative accelerated proximal envelop (Parikh et al., 2014) was proposed in the paper (Monteiro & Svaiter, 2013). The main difference with standard accelerated proximal envelops is the adaptability of the scheme (Monteiro & Svaiter, 2013). Note, that this scheme allows also to build (near) optimal tensor (high-order) accelerated methods (Gasnikov, 2017; Nesterov, 2018; Gasnikov et al., 2019a,b; Wilson et al., 2019). That is, the 'acceleration' potential of this scheme seems to be the best known for us for the moment. So the main and rather simple idea of this paper can be formulated briefly as follows: **To develop adaptive Catalyst, we replace the accelerated proximal envelope with a fixed regularization parameter (Parikh et al., 2014; Lin et al., 2018) on the adaptive accelerated proximal envelope from (Monteiro & Svaiter, 2013)**.

This replacement described in Section 2.

By using this adaptive accelerated proximal envelope, we propose in Section 3 an accelerated variant of steepest descent (Polyak, 1987; Gasnikov, 2017) as an alternative to A. Nemirovski accelerated steepest descent (see (Nesterov et al., 2018; Diakonikolas & Orecchia, 2019) and references therein), adaptive accelerated variants of alternating minimization procedures (Beck, 2017) as an alternative to (Diakonikolas & Orecchia, 2018; Guminov et al., 2019; Tupitsa et al., 2019) and adaptive accelerated coordinate descent (Nesterov, 2012). For the last example, as far as we know, there were no previously complete adaptive accelerated coordinate descent. The most advanced result in this direction is the work (Fercoq & Richtárik, 2015) that applies only to the problems with increasing smoothness parameter along the iteration process. For example, for the target function like $f(x) = x^4$, this scheme doesn't recognize that smoothness parameters (in particular Lipschitz gradient constant) tend to zero along the iteration process.

In Section 4 we describe numerical experiments with the steepest descent and adaptive coordinate descent.

We hope that the proposed approach allows accelerating not only adaptive on their own procedures, but also many other different non-accelerated non-adaptive randomized schemes by settings on general smoothness parameters of target function that can be difficult to analyze patently (Gower et al., 2019; Gazagnadou et al., 2019; Gorbunov et al., 2019b).

2 The Main Scheme

Let us consider the following minimization problem

$$\min_{y \in Q} f(y),$$

where $f(y)$ is a convex function, and its gradient is Lipschitz continuous w.r.t. $\|\cdot\|_2$ with the constant L_f:

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L_f \|x - y\|_2.$$

To propose the main scheme of the algorithm we need to define the following functions:

$$\begin{aligned} F_{L,x}(y) &= f(y) + \tfrac{L}{2}||y - x||_2^2, \\ f_L(x) &= \min_{y \in Q} F_{L,x}(y) = F_{L,x}(y_L(x)), \end{aligned}$$

then the function $F_{L,x}(y)$ is L–strongly convex, and its gradient is Lipschitz continuous w.r.t. $\|\cdot\|_2$ with the constant $(L + L_f)$. So, the following inequality holds

$$||\nabla F_{L,x}(y_2) - \nabla F_{L,x}(y_1)||_2 \leq (L + L_f)||y_1 - y_2||_2.$$

Due to this definition, for all $L \geq 0$ we have that $f_L(x) \leq f(x)$ and the convex function $f_L(x)$ has a Lipschitz-continuous gradient with the Lipschitz constant L. Moreover, according to (Polyak, 1987) [Theorem 5, ch. 6], since

$$x_\star \in \operatorname*{Argmin}_x f_L(x) = \operatorname*{Argmin}_{x \in Q} f_L(x),$$

we obtain

$$x_\star \in \operatorname*{Argmin}_{x \in Q} f(x) \quad \text{and } f_L(x_\star) = f(x_\star).$$

Thus, instead of the initial problem (1), we can consider the Moreau–Yosida regularized problem

$$\min_{x \in Q} f_L(x).$$

Note that the problem (3) is an ordinary problem of smooth convex optimization. Then the complexity of solving the problem (3) up to the accuracy ε with respect to the function using the Fast Gradient Method (FGM) (Nesterov, 2018) can be estimated as follows $O\left(\sqrt{\frac{LR^2}{\varepsilon}}\right)$. The 'complexity' means here the number of oracle calls. Each oracle call means calculation of $\nabla f_L(x) = L(x - y_L(x))$, where $y_L(x)$ is the exact solution of the auxiliary problem $\min\limits_{y \in Q} F_{L,x}(y)$.

Note that the smaller the value of the parameter L we choose, the smaller is the number of oracle calls (outer iterations). However, at the same time, this increases the complexity of solving the auxiliary problem at each iteration.

At the end of this brief introduction to standard accelerated proximal point methods, let us describe the step of ordinary (proximal) gradient descent (for more details see (Parikh et al., 2014))

$$x^{k+1} = x^k - \tfrac{1}{L}\nabla f_L(x) = x^k - \tfrac{L}{L}(x - y_L(x)) = y_L(x).$$

To develop an adaptive proximal accelerated envelop, we should replace standard FGM (Nesterov, 2018) on the following adaptive variant of FGM Algorithm 1, introduced by (Monteiro & Svaiter, 2013) for smooth convex optimization problems. We also assume that $Q = \mathbb{R}^n$.

The analysis of the algorithm is based on the following theorem.

Theorem 1. *(Monteiro & Svaiter, 2013)[Theorem 3.6] Let sequence (x^k, y^k, z^k), $k \geq 0$ be generated by Algorithm 1 and define $R := \left\|y^0 - x_\star\right\|_2$. Then, for all $N \geq 0$,*

$$\tfrac{1}{2}\left\|z^N - x_\star\right\|_2^2 + A_N \cdot \left(f\left(y^N\right) - f\left(x_\star\right)\right) + \tfrac{1}{4}\sum_{k=1}^{N} A_k L_k \left\|y^k - x^k\right\|_2^2 \leq \tfrac{R^2}{2},$$

$$f\left(y^N\right) - f\left(x^\star\right) \leq \tfrac{R^2}{2A_N}, \quad \left\|z^N - x_\star\right\|_2 \leq R,$$

$$\sum_{k=1}^{N} A_k L_k \left\|y^k - x^k\right\|_2^2 \leq 2R^2. \tag{4}$$

Algorithm 1 Monteiro–Svaiter algorithm

Parameters: $z^0, y^0, A_0 = 0$
for $k = 0, 1, \ldots, N-1$ **do**
Choose L_{k+1} and y^{k+1} such that

$$\|\nabla F_{L_{k+1}, x^{k+1}}(y^{k+1})\|_2 \leq \frac{L_{k+1}}{2}\|y^{k+1} - x^{k+1}\|_2,$$

where

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

$z^{k+1} = z^k - a_{k+1}\nabla f\left(y^{k+1}\right)$
end for
Output: y^N

We also need the following Lemma.

Lemma 1. *(Monteiro & Svaiter, 2013)[Lemma 3.7a)] Let sequences $\{A_k, L_k\}$, $k \geq 0$ be generated by Algorithm 1. Then, for all $N \geq 0$,*

$$A_N \geq \frac{1}{4}\left(\sum_{k=1}^{N} \frac{1}{\sqrt{L_k}}\right)^2. \tag{5}$$

Let us define non-accelerated method $\mathcal{M}$ that we will use to solve auxiliary problem.

Proposition 1. *The convergence rate for the method $\mathcal{M}$ for problem*

$$\min_{y \in \mathbb{R}^n} F(y)$$

can be written in the general form as follows: with probability at least $1 - \delta$ holds (for randomized algorithms, like Algorithm 4, this estimates holds true with high probability)

$$F(y^N) - F(y_\star) = O\left(L_F R^2 \ln \tfrac{1}{\delta}\right) \min\left\{\tfrac{C_n}{N}, \exp\left(-\tfrac{\mu_F N}{C_n L_F}\right)\right\},$$

where $y_\star$ is the solution of the problem, $R = ||y^0 - y_\star||_2$, function F is μ_F–strongly convex and L_F is a constant which characterized smoothness of function F.

Typically $C_n = O(1)$ for the standard full gradient first order methods, $C_n = O(p)$, where p is a number of blocks, for alternating minimization with p blocks and $C_n = O(n)$ for gradient free or coordinate descent methods, where n is dimension of y. See the references in next Remark for details.

Remark 1. *Let us clarify what we mean by a constant L_F which characterized smoothness of function F. Typically for the first order methods this is just the Lipschitz constant of gradient F (see, (Polyak, 1987; De Klerk et al., 2017) for the steepest descent and (Karimi et al., 2016; Diakonikolas & Orecchia, 2018; Tupitsa et al., 2019) for alternating minimization); for gradient free methods like Algorithm 4 this constant is the average value of the directional smoothness parameters, for gradient free methods*

Algorithm 2 Adaptive Catalyst

Parameters: Starting point $x^0 = y^0 = z^0$; initial guess $L_0 > 0$; parameters $\alpha > \beta > \gamma > 0$; optimization method $\mathcal{M}$.
for $k = 0, 1, \ldots, N-1$ **do**
$L_{k+1} = \beta \cdot \min\{\alpha L_k, L_u\}$
$t = 0$
repeat
$t := t + 1$
$L_{k+1} := \max\{L_{k+1}/\beta, L_d\}$
Compute

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

Compute an approximate solution of the following problem with auxiliary non-accelerated method $\mathcal{M}$

$$y^{k+1} \approx \operatorname*{argmin}_y F_{L,x}^{k+1}(y)$$

By running $\mathcal{M}$ with starting point x^{k+1} and output point y^{k+1} we wait N_t iterations to fulfill adaptive stopping criteria

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

until $t > 1$ and $N_t \geq \gamma \cdot N_{t-1}$ or $L_{k+1} = L_d$
$z^{k+1} = z^k - a_{k+1}\nabla f\left(y^{k+1}\right)$
end for
Output: y^N

see (Duchi et al., 2015; Gasnikov et al., 2016; Shamir, 2017; Bayandina et al., 2018; Dvurechensky et al., 2018a,b), for coordinate descent methods see (Nesterov, 2012; Wright, 2015; Nesterov & Stich, 2017) and for more general situations see (Gower et al., 2019).

Remark 2. *Note that in proposition 1 the first estimate corresponds to the estimate of the convergence rate of the method $\mathcal{M}$ for convex problems. And the second estimate corresponds to the estimate for strongly convex problems.*

Our main goal is to propose a scheme to accelerate methods of this type. But note that we apply our scheme only to degenerate convex problems since it does not take into account the strong convexity of the original problem.

Denote $F_{L,x}^{k+1}(\cdot) \equiv F_{L_{k+1},x^{k+1}}(\cdot)$. Based on Monteiro–Svaiter accelerated proximal method we propose Algorithm 2.

Now let us prove the main theorem about the convergence rate of the proposed scheme. So, based on the Monteiro–Svaiter Theorem 1 we can introduce the following theorem

Theorem 2. *Consider Algorithm 2 with $L_d < L_u$ for solving problem (1), where $Q = \mathbb{R}^n$, with auxiliary (inner) non-accelerated algorithm (method) $\mathcal{M}$ that satisfy Proposition 1 with constants C_n and L_f such that $L_d \leq L_f \leq L_u$.*

Then the total complexity[*] *of the proposed Algorithm 2 with inner method $\mathcal{M}$ is*

$$\tilde{O}\left(C_n \cdot \max\left\{\sqrt{\frac{L_u}{L_f}}, \sqrt{\frac{L_f}{L_d}}\right\} \cdot \sqrt{\frac{L_f R^2}{\varepsilon}}\right)$$

with probability at least $1 - \delta$.

Proof. Note that the Monteiro–Svaiter (M-S) condition

$$\|\nabla F_{L,x}^{k+1}(y^{k+1})\|_2 \leq \frac{L_{k+1}}{2}\|y^{k+1} - x^{k+1}\|_2$$

instead of the exact solution $y_\star^{k+1} = y_{L_{k+1}}(x^{k+1})$ of the auxiliary problem, for which

$$\|\nabla F_{L,x}^{k+1}(y_\star^{k+1})\|_2 = 0,$$

allows to search inexact solution that satisfies the condition (6).

Since $y_\star^{k+1}$ the solution of the problem $\min\limits_y F_{L,x}^{k+1}(y)$, the $\nabla F_{L,x}^{k+1}(y_\star^{k+1}) = 0$. Then, using inequality (2) we obtain

$$||\nabla F_{L,x}^{k+1}(y^{k+1})||_2 \leq (L_{k+1} + L_f)||y^{k+1} - y_\star^{k+1}||_2. \tag{7}$$

Using the triangle inequality we have

$$||x^{k+1} - y_\star^{k+1}||_2 - ||y^{k+1} - y_\star^{k+1}||_2 \leq ||y^{k+1} - x^{k+1}||_2. \tag{8}$$

Since r.h.s. of the inequality (7) coincide with the r.h.s. of the M-S condition and l.h.s. of the inequality (8) coincide with the l.h.s. of the M-S condition up to a multiplicative factor $L_{k+1}/2$, one can conclude that if the inequality

$$||y^{k+1} - y_\star^{k+1}||_2 \leq \frac{L_{k+1}}{3L_{k+1}+2L_f}||x^{k+1} - y_\star^{k+1}||_2$$

holds, the M-S condition holds too. Note, that $||x^{k+1} - y_\star^{k+1}||_2 = R$ since x^{k+1} is a starting point.

To solve the auxiliary problem $\min\limits_y F_{L_{k+1},x^{k+1}}(y)$ we use non-accelerated method $\mathcal{M}$. Using proposition (1) with probability $\geq 1 - \frac{\delta}{N}$ (where N is the total number of the Catalyst's steps), we obtain that the convergence rate for these methods can be written as follows

$$F_{L,x}^{k+1}(y_t^{k+1}) - F_{L,x}^{k+1}(y_\star^{k+1}) = O\left((L_f + L_{k+1})R^2 \ln\frac{N}{\delta}\right)\exp\left(-\frac{L_{k+1}t}{C_n(L_f+L_{k+1})}\right).$$

Since $F_{L,x}^{k+1}(\cdot)$ is L_{k+1}-strongly convex function, the following inequality holds (Nesterov, 2018)

$$\frac{L_{k+1}}{2}||y_t^{k+1} - y_\star^{k+1}||_2^2 \leq F_{L,x}^{k+1}(y_t^{k+1}) - F_{L,x}^{k+1}(y_\star^{k+1}).$$

Thus,

$$||y_t^{k+1} - y_\star^{k+1}||_2 \leq O\left(\sqrt{\frac{(L_f+L_{k+1})R^2}{L_{k+1}}\ln\frac{N}{\delta}}\right)\exp\left(-\frac{L_{k+1}t}{2C_n(L_f+L_{k+1})}\right).$$

[*]The number of oracle calls (iterations) of auxiliary method $\mathcal{M}$ that required to find ε solution of (1) in terms of functions value.

From this, we obtain that the complexity of solving the auxiliary problem with probability at least $1 - \frac{\delta}{N}$ is

$$T = \tilde{O}\left(C_n \frac{(L_{k+1}+L_f)}{L_{k+1}}\right).$$

Note, that $\tilde{O}(\cdot)$ means the same as $O(\cdot)$ up to a logarithmic factor. Since that we can consider T to be the estimate that include total complexity of auxiliary problem including all inner restarts on L_{k+1}.

Substituting inequality (5) into estimation (4) we obtain

$$f(y^N) - f(x_\star) \leq \frac{2R^2}{\left(\sum\limits_{k=1}^{N} \frac{1}{\sqrt{L_k}}\right)^2}.$$

Since the complexity of the auxiliary problem is with probability at least $1 - \frac{\delta}{N}$ T we assume that in the worst case all L_{k+1} are equal. Then the worst case we can estimate as the following optimization problem

$$\max_{L_d \leq L \leq L_u} \frac{L+L_f}{L} \sqrt{\frac{LR^2}{\varepsilon}},$$

Obviously, the maximum is reached at the border. So, using union bounds inequality over all N iterations of the Catalyst we can estimate the complexity in the worst two cases as follows:

- If all $L_{k+1} = L_d \leq L_f$ (at each iteration we estimate the regularization parameter as lower bound), then $\frac{(L_{k+1}+L_f)}{L_{k+1}} \approx \frac{L_f}{L_{k+1}}$ and total complexity with probability $\geq 1 - \delta$ is

$$\tilde{O}\left(C_n \frac{L_f}{L_d} \sqrt{\frac{L_d R^2}{\varepsilon}}\right) = \tilde{O}\left(C_n \sqrt{\frac{L_f}{L_d}} \cdot \sqrt{\frac{L_f R^2}{\varepsilon}}\right).$$

- If all $L_{k+1} = L_u \geq L_f$ (at each iteration we estimate the regularization parameter as upper bound), then $\frac{(L_{k+1}+L_f)}{L_{k+1}} \approx 1$ and total complexity with probability $\geq 1 - \delta$ is

$$\tilde{O}\left(C_n \sqrt{\frac{L_u R^2}{\varepsilon}}\right) = \tilde{O}\left(C_n \sqrt{\frac{L_u}{L_f}} \cdot \sqrt{\frac{L_f R^2}{\varepsilon}}\right).$$

Then, using these two estimations we obtain the result of the theorem. □

Note that this result shows that such a procedure will works not worse than standard Catalyst (Lin et al., 2015, 2018) up to a factor $\tilde{O}\left(\max\left\{\sqrt{\frac{L_u}{L_f}}, \sqrt{\frac{L_f}{L_d}}\right\}\right)$ independent on the stopping criteria in the restarts on L_{k+1}.

Since the complexity of solving the auxiliary problem is proportional to $\frac{(L_{k+1}+L_f)C_n}{L_{k+1}}$, when we reduce the parameter L_{k+1} so that $L_{k+1} < L_f$ the complexity of solving an auxiliary problem became growth exponentially. Therefore, as the stopping criterion of the inner method, we select the number of iterations N_t compared to the number of iterations N_{t-1} at the previous restart $t-1$. This means that if $N_t \leq \gamma N_{t-1}$ then the complexity begins to grow exponentially and it is necessary to go to the next iteration of the external method. By using such adaptive rule we try to recognize the best possible value of $L_{k+1} \simeq L_f$. The last facts are basis of standard Catalyst approach (Lin et al., 2015, 2018) and have very simple explanation. To minimize the total complexity we should take parameter $L_{k+1} \equiv L$ such that

$$\min_L \sqrt{\frac{LR^2}{\varepsilon}} \cdot \tilde{O}\left(\frac{L_f+L}{L}\right).$$

This leads us to $L_{k+1} \simeq L_f$.

Note that also in non-adaptive case (if we choose all $L_{k+1} \equiv L_f$) we can obtain the following corollary from the Theorem 2.

Corollary 1. *If we consider Algorithm 2 with $L_{k+1} \equiv L_f$ for solving problem (1), then the total complexity of the proposed Algorithm 2 with inner method $\mathcal{M}$ is*

$$O\left(C_n\sqrt{\frac{L_f R^2}{\varepsilon}}\right). \tag{11}$$

Proof. Using inequality (9) one can obtain that the complexity of the auxiliary problem is (10), i.e.

$$T = O\left(C_n \frac{(L_{k+1}+L_f)}{L_{k+1}} \cdot \ln\left(\frac{3L_{k+1}+2L_f}{L_{k+1}}\right)\right)$$

And since $L_d \leq L_f \leq L_u$ we can estimate $\frac{3L_{k+1}+2L_f}{L_{k+1}} \leq \frac{3L_d+2L_f}{L_u} \leq C$, where the constant C does not depend on the accuracy ε (accuracy of solving an auxiliary problem). Then the complexity of the auxiliary problem is $T = O\left(C_n \frac{(L_{k+1}+L_f)}{L_{k+1}}\right)$. Using this estimate and that $L_{k+1} \equiv L_f$, we obtain that the total complexity is (11). □

Note that in the standard Catalyst approach the total complexity is $\tilde{O}\left(C_n\sqrt{\frac{L_f R^2}{\varepsilon}}\right)$. From this we get that choosing the stopping criterion for the inner method as the criterion from the Algorithm 2 we can get the Catalyst without a logarithmic cost. It seems that such variant of Catalyst can be useful in many applications. For example, as universal envelope for non accelerated asynchronized centralized distributed algorithms (Mishchenko et al., 2018).

3 Applications

In this section, we present a few examples of algorithms that we consider as inner solvers. All of them have an adaptive structure, so it makes little sense to use Catalyst algorithm to accelerate them.

3.1 Steepest Descent

Consider the following problem

$$\min_{x\in\mathbb{R}^n} f(x),$$

where $f(x)$ is a L_f–smooth convex function (its gradient is Lipschitz continuous w.r.t. $\|\cdot\|_2$ with the constant L_f).

To solve this problem, let us consider the general gradient descent update rule

$$x^{k+1} = x^k - h_k \nabla f(x^k).$$

In (Polyak, 1987) it was proposed an adaptive way to select h_k as following (see also (De Klerk et al., 2017) for precise rates of convergence)

$$h_k = \operatorname*{argmin}_{h\in\mathbb{R}} f(x^k - h\nabla f(x^k)).$$

In contrast with the standard selection $h_k \equiv \frac{1}{L_f}$ for L-smooth functions f, in this method there is

Algorithm 3 Steepest descent

Parameters: Starting point x^0.
for $k = 0, 1, \ldots, N-1$ **do**
  Choose $h_k = \text{argmin}_{h \in \mathbb{R}} f(x^k - h\nabla f(x^k))$
  Set $x^{k+1} = x^k - h_k \nabla f(x^k)$
end for
Output: x^N

no need to know smoothness constant of the function. It allows to use this method for the smooth functions f when L_f is unknown (or expensive to compute) or when the global L_f is much bigger than the local ones along the trajectory.

On the other hand, as far as we concern, there is no direct acceleration of the steepest descent algorithm. Moreover, it is hard to use Catalyst with it as far as acceleration happens if L_k (κ in Catalyst article notations) is selected with respect to L_f and the scheme does not support adaptivity out of the box. Even if global L_f is known, the local smoothness constant could be significantly different from it that will lead to the worse speed of convergence.

Note that for Algorithm 3 the proposition 1 holds with $C_n = O(1)$ and L_f is the Lipschitz constant of the gradient of function f.

3.2 Random Adaptive Coordinate Descent Method

Consider the following unconstrained problem

$$\min_{x \in \mathbb{R}^n} f(x).$$

Now we assume directional smoothness for f, that is there exists $\beta_1, \ldots, \beta_n$ such that for any $x \in \mathbb{R}^n, u \in \mathbb{R}$

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

where $\nabla_i f(x) = \partial f(x)/\partial x_i$. For twice differentiable f it equals to $(\nabla^2 f(x))_{i,i} \leq \beta_i$. Due to the fact that we consider the situation when smoothness constants are not known, we use such a dynamic adjustment scheme from (Nesterov, 2012; Wright, 2015).

Note that for Algorithm 4 the proposition 1 holds with $C_n = O(n)$ (for $x \in \mathbb{R}^n$) and $L_f = \overline{L}_f := \frac{1}{n}\sum\limits_{i=1}^{n} \beta_i$ (the average value of the directional smoothness parameters).

As one of the motivational example, we can consider the following problem

$$\min_{x \in R^n} \ln \left(\sum_{k=1}^{m} \exp(a_k^T x) \right),$$

where the matrix $A = (a_1, \ldots, a_m)^T$ is a sparse matrix and nnz $= s$ (number of non-zero elements), $s \ll m = O(n)$. As far as we know all existing accelerated coordinate descent algorithms for this problem are not completely adaptive and don't allow to make iteration faster then with $O(n)$ arithmetic operations. The proposed envelope solve both of these problems (average iteration complexity will be $O(s)$).

Algorithm 4 RACDM

Parameters: Starting point x^0;
lower bounds $\hat{\beta}_i := \beta_i^0 \in (0, \beta_i], i = 1, \ldots, n$
for $k = 0, 1, \ldots, N-1$ **do**
Sample $i_k \sim \mathcal{U}[1, \ldots, n]$
Set $x^{k+1} = x^k - \hat{\beta}_{i_k}^{-1} \cdot \nabla_{i_k} f\left(x^k\right) \cdot e_{i_k}$
While $\nabla_{i_k} f(x^k) \cdot \nabla_{i_k} f(x^{k+1}) < 0$ **do**

$$\left\{\hat{\beta}_{i_k} = 2\hat{\beta}_{i_k}, \quad x^{k+1} = x^k - \hat{\beta}_{i_k}^{-1} \cdot \nabla_{i_k} f\left(x^k\right) \cdot e_{i_k}\right\}$$

Set $\beta_{i_k} = \frac{1}{2}\beta_{i_k}$
end for
Output: x^N

Algorithm 5 Alternating Minimization

Parameters: Starting point x^0.
for $k = 0, 1, \ldots, N-1$ **do**
Choose $i \in \{1, \ldots, p\}$
Set $x^{k+1} = \underset{x \in S_i(x^k)}{\operatorname{argmin}} f(x)$
end for
Output: x^N

3.3 Alternating Minimization

Consider the following problem

$$\min_{x \in Q \subseteq E} f(x),$$

where $f(x)$ is a L_f–smooth convex function (its gradient is Lipschitz continuous w.r.t. $\|\cdot\|_2$ with the constant L_f), $Q = \otimes_{i=1}^p Q_i \subseteq E$, with $Q_i \subseteq E_i$, $i = 1, ..., p$ being closed convex sets.

For the general case of number of blocks $p \geqslant 2$ the Alternating Minimization algorithm may be written as Algorithm 5. There are multiple common block selection rules, such as the cyclic rule or the Gauss–Southwell rule (Karimi et al., 2016; Beck, 2017; Diakonikolas & Orecchia, 2018; Tupitsa et al., 2019).

Note that for Algorithm 5 the proposition 1 holds with $C_n = O(p)$ (p – number of blocks) and L_f is the Lipschitz constant of the gradient of function f.

3.4 Local Stochastic Gradient Descent

Local SGD (Khaled et al., 2019) becomes popular first of all due to the application in federated learning (Kairouz et al., 2019). In the core of the approach lies a very simple idea (Nesterov & Vial, 2008; Dvurechensky et al., 2018c; Stich, 2018; Khaled et al., 2019): to solve considered stochastic convex optimization problem in parallel on M nodes via Adaptive Stochastic Gradient Descent (SGD) (Ogaltsov et al., 2019) with synchronization after each τ iterations of SGD and sharing an average point. The large we want to choose M the smaller we should choose τ to conserve the total (optimal) number of oracle T (stochastic gradient) calls (on M nodes). Say, for strongly convex case $M\tau \simeq T$

(Khaled et al., 2019). It is well known that for stochastic convex optimization problems from the oracle complexity point of view there is no difference between accelerated schemes and non-accelerated ones (Nemirovsky & Yudin, 1983). On the other hand, if we reduce the variance by batching acceleration could play a significant role (Woodworth et al., 2018; Gorbunov et al., 2019a). That is in parallelized architecture the accelerated schemes are dominant. So, the natural question: Can we accelerate local SGD? Below we'll try to demonstrate the numerical possibility of acceleration local SGD by proposed M-S Catalyst scheme.

3.5 Theoretical Guarantees

	non-accelerated	M-S accelerated
Steepest Descent	$\frac{L_f R^2}{\varepsilon}$	$\alpha\sqrt{\frac{L_f R^2}{\varepsilon}}$
Random Adaptive Coordinate Descent Method	$n \cdot \frac{\overline{L}_f R^2}{\varepsilon}$	$n \cdot \alpha\sqrt{\frac{\overline{L}_f R^2}{\varepsilon}}$
Alternating Minimization	$p \cdot \frac{L_f R^2}{\varepsilon}$	$p \cdot \alpha\sqrt{\frac{L_f R^2}{\varepsilon}}$

Let us present the table that establishes the comparison of rates of convergence for the above algorithms before and after acceleration via Algorithm 2. In non-accelerated case these algorithms apply to the convex but non-strongly convex problem, therefore, we use estimates for the convex case from proposition 1. But in the case of acceleration of these methods, we apply them to a regularized function which is strongly convex. Denote $\alpha = \max\left(\sqrt{\frac{L_u}{L_f}}, \sqrt{\frac{L_f}{L_d}}\right)$, then we represent the following table.

4 Experiments

In this section, we perform experiments for justifying the acceleration of the aforementioned methods in practice. For all figures below, the vertical axis measures functional suboptimality $f(x) - f(x_\star)$ in the logarithmic scale.

4.1 RADCM Acceleration

Let us consider quadratic optimization problem

$$\min_{x \in \mathbb{R}^n} \quad f(x) = \frac{1}{2} x^\top A x, \tag{12}$$

for Hilbert matrix (Hilbert, 1894) with such entries $A_{ij} = \frac{1}{i+j-1}$. This is an example of a Hankel matrix and is known to be very ill-conditioned (e.g. for $n = 6$ condition number $\approx 1.5 \cdot 10^7$ (Polyak, 1987)). It leads to a degenerate optimization problem, typically very hard for gradient methods.

In Figure 1 we compare the performance of the method 4 and its M-S accelerated version with different sets of parameters (α, β, γ) for problem (12). For the horizontal axis we use number of partial

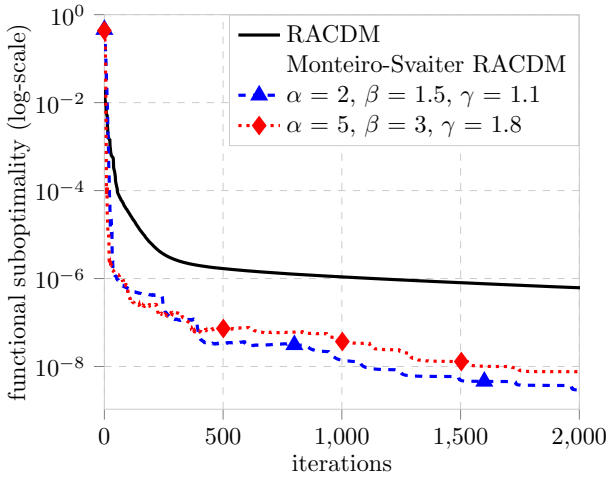


Figure 1: Quadratic problem (12) with Hilbert matrix, $n = 1000$.

derivative evaluations divided by dimensionality n of the problem. Our warm start strategy includes running inner method from the last point y_k and with estimates $\hat{\beta}_i$ of smoothness constants obtained from the previous outer (M-S) iteration. The initial points $y_0 = z_0$ entries were sampled from the standard uniform distribution $\mathcal{U}(0,1)$. L_0 was initialized as $0.5L_f$ and $L_d = 0.001L_f, L_u = 100L_f, \beta_i^0 = 1/L_0$.

4.2 Alternating Least Squares Acceleration

Consider the typical collaborative filtering problem: completion of the user-item preferences matrix with estimated values based on a little count of observed ratings made by other users on other items. The considered being accelerated AM algorithm is induced by the idea of preferences matrix factorization and estimating unknown rating r_{ui} associated with the user u and the item i as a product $x_u^\top y_i$, where the vectors x_u and y_i are being optimized variables. Following the approach set out in (Hu et al., 2008), formulate such an optimization problem:

$$\min_{x,y} \quad F(x,y) = \sum_{u,i} c_{ui} \left(p_{ui} - x_u^\top y_i\right)^2 + \lambda \left(\sum_u ||x_u||_2^2 + \sum_i ||y_i||_2^2\right), \tag{13}$$

where c_{ui} is confidence in observing r_{ui}, in our case expressed as $c_{ui} = 1 + 5r_{ui}$, p_{ui} is binarized rating:

$$p_{ui} = \begin{cases} 1 & r_{ui} > 0, \\ 0 & r_{ui} = 0, \end{cases}$$

and $\lambda \left(\sum_u ||x_u||_2^2 + \sum_i ||y_i||_2^2\right)$ — regularization term preventing overfitting during the learning process (in our case, the regularization coefficient is set to $\lambda = 0.1$).

For described objective functional optimization we used modified Algorithm 5 in that on every iteration functional optimizing with relation to x and y consistently (for that functional we can get the explicit expression for the solutions of equations $\nabla_x f(x,y) = 0$ and $\nabla_y f(x,y) = 0$, computational effective matrix expressions for solutions of these auxiliary problems are presented in (Hu et al., 2008)).

The considered objective function is not convex, so instead of the described Adaptive Catalyst scheme for accelerating should be used the modified one, in which the step of updating variable y_k replaced with such construction:

$$\tilde{y}_{k+1} \approx \operatorname*{argmin}_y F_{L,x}^{k+1}(y)$$
$$y_{k+1} = \arg\min \{f(y) \mid y \in \{y_k, \tilde{y}_{k+1}\}\}$$

Used in experiments sparse matrix $\{r_{ui}\}_{u,i}$ generated from *radio* dataset* with ratings made by listeners on compositions of certain artists. Count of considered users was 70, count of artists — 100, and sparsity coefficient of the matrix was near the 2%.

In Figure 2 we compare the performance of the modified Algorithm 5 and their accelerated via Algorithm 2 versions (with a different choice of hyperparameters (α, β, γ)). The horizontal axis measures the number of variables recomputations. The plot show that there was the acceleration of the base algorithm and both parameters β and γ had an impact on the convergence rate.

*https://www.upf.edu/web/mtg/lastfm360k

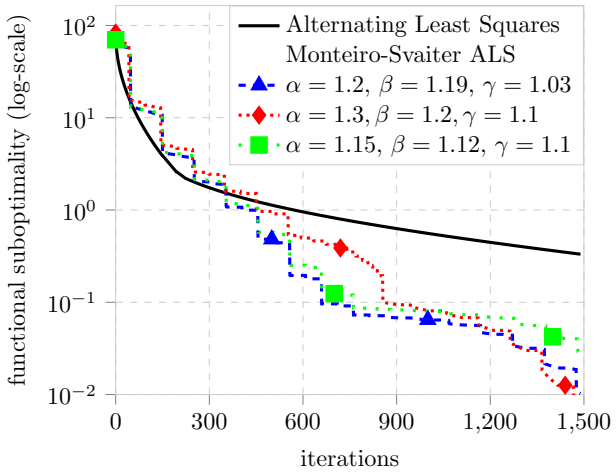


Figure 2: Matrix completion problem (13) with different (α, β, γ).

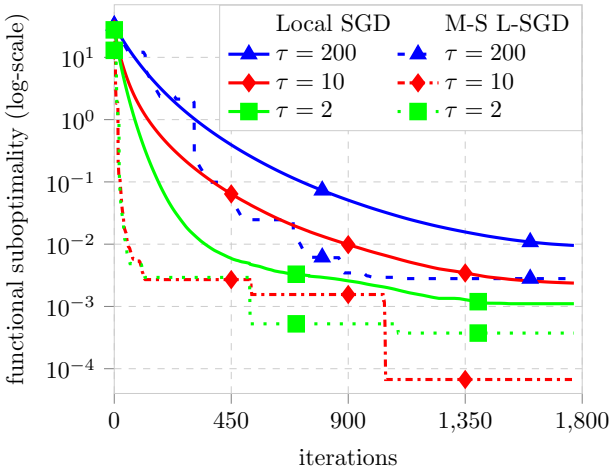


Figure 3: Regularized logistic loss (14) for different synchronization intervals τ.

4.3 Local SGD Acceleration

Consider the following l2-regularized logistic loss minimization problem:

$$\min_{x \in \mathbb{R}^n} \quad F(x) = \frac{1}{m} \sum_{j=1}^{m} \log(1 + \exp(-y_j x^\top p_j)) + g(x) \tag{14}$$

where the features matrix $P \in \mathbb{R}^{m \times n}$, labels $y \in \{0, 1\}^m$ and g is a regularization term, aggregated by two different regularizers for the sparse features $I_S \subset [1, n]$ (with coefficient λ_1) and the dense features $I_D \subset [1, n], I_S \cap I_D = \varnothing$ (with coefficient λ_2):

$$g(x) = \lambda_1 \sum_{i \in I_S} x_i^2 + \lambda_2 \sum_{i \in I_D} x_i^2.$$

In this experiment, we use the *adults* dataset with one-hot encoded categorial features and binarized 'work class' feature as a label, $m = 40000$, $n = 100$, $\lambda_1 = 1.1$, $\lambda_2 = 2.1$, the sparsity coefficient (percentage of features with a fraction of zeros less than 0.2) is equal to 4%. Also, the initial value of the model's weights randomly generated from the uniform distribution $x_{0,i} \sim \mathcal{U}(0, 1)\ \forall i \in [1, n]$.

In Figure 3 we compare the performance of the Local SGD (see Appendix, Section Local SGD) and its accelerated via Algorithm 2 versions. Parameters used: $w = 20$ (amount of computing nodes), varying τ (number of iterations between two consequent synchronization steps), $\alpha = 1.15, \beta = 1.12, \gamma = 1.1$ (for Monteiro-Svaiter). The horizontal axis measures the number of outer iterations (one outer iteration includes recomputation of the variables in all computing nodes). The plot shows that there was the acceleration of the base algorithm and synchronization interval had an impact on the convergence rate.

Conclusion

In this work we present the universal framework for accelerating the non-accelerated adaptive methods such as Steepest Descent, Alternating Least Squares Minimization and RACDM and show that acceleration works in practice (code is available online on GitHub). Moreover, we show theoretically that for nonadaptive run proposed in this paper acceleration has a better rate than via Catalyst.

Acknowledgements

We would like to thank Soomin Lee (Yahoo), Erik Ordentlich (Yahoo), César A. Uribe (MIT), Pavel Dvurechensky (WIAS, Berlin) and Peter Richtarik (KAUST) for useful remarks.

References

Bayandina, A., Gasnikov, A., and Lagunovskaya, A. Gradient-free two-points optimal method for non smooth stochastic convex optimization problem with additional small noise. *Automation and remote control*, 79(7), 2018. arXiv:1701.03821.

Beck, A. *First-order methods in optimization*, volume 25. SIAM, 2017.

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

De Klerk, E., Glineur, F., and Taylor, A. B. On the worst-case complexity of the gradient method with exact line search for smooth strongly convex functions. *Optimization Letters*, 11(7):1185–1199, 2017.

Diakonikolas, J. and Orecchia, L. Alternating randomized block coordinate descent. *arXiv preprint arXiv:1805.09185*, 2018.

Diakonikolas, J. and Orecchia, L. Conjugate gradients and accelerated methods unified: The approximate duality gap view. *arXiv preprint arXiv:1907.00289*, 2019.

Duchi, J. C., Jordan, M. I., Wainwright, M. J., and Wibisono, A. Optimal rates for zero-order convex optimization: The power of two function evaluations. *IEEE Trans. Information Theory*, 61(5): 2788–2806, 2015.

Dvurechensky, P., Gasnikov, A., and Gorbunov, E. An accelerated method for derivative-free smooth stochastic convex optimization. *arXiv:1802.09022*, 2018a.

Dvurechensky, P., Gasnikov, A., and Gorbunov, E. An accelerated directional derivative method for smooth stochastic convex optimization. *arXiv:1804.02394*, 2018b.

Dvurechensky, P., Gasnikov, A., and Lagunovskaya, A. Parallel algorithms and probability of large deviation for stochastic convex optimization problems. *Numerical Analysis and Applications*, 11(1): 33–37, 2018c.

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

Gasnikov, A. Universal gradient descent. *arXiv preprint arXiv:1711.00394*, 2017.

Gasnikov, A., Dvurechensky, P., Gorbunov, E., Vorontsova, E., Selikhanovych, D., Uribe, C. A., Jiang, B., Wang, H., Zhang, S., Bubeck, S., et al. Near optimal methods for minimizing convex functions with lipschitz p-th derivatives. In *Conference on Learning Theory*, pp. 1392–1393, 2019a.

Gasnikov, A., Gorbunov, E., Kovalev, D., Mokhammed, A., and Chernousova, E. Reachability of optimal convergence rate estimates for high-order numerical convex optimization methods. In *Doklady Mathematics*, volume 99, pp. 91–94. Springer, 2019b.

Gasnikov, A. V., Lagunovskaya, A. A., Usmanova, I. N., and Fedorenko, F. A. Gradient-free proximal methods with inexact oracle for convex stochastic nonsmooth optimization problems on the simplex. *Automation and Remote Control*, 77(11):2018–2034, Nov 2016. ISSN 1608-3032. doi: 10.1134/S0005117916110114. URL http://dx.doi.org/10.1134/S0005117916110114. arXiv:1412.3890.

Gazagnadou, N., Gower, R. M., and Salmon, J. Optimal mini-batch and step sizes for saga. *arXiv preprint arXiv:1902.00071*, 2019.

Gorbunov, E., Dvinskikh, D., and Gasnikov, A. Optimal decentralized distributed algorithms for stochastic convex optimization. *arXiv preprint arXiv:1911.07363*, 2019a.

Gorbunov, E., Hanzely, F., and Richtarik, P. A unified theory of sgd: Variance reduction, sampling, quantization and coordinate descent, 2019b.

Gower, R. M., Loizou, N., Qian, X., Sailanbayev, A., Shulgin, E., and Richtárik, P. Sgd: General analysis and improved rates. *arXiv preprint arXiv:1901.09401*, 2019.

Guminov, S., Dvurechensky, P., and Gasnikov, A. Accelerated alternating minimization. *arXiv preprint arXiv:1906.03622*, 2019.

Hilbert, D. Ein beitrag zur theorie des legendre'schen polynoms. *Acta Math.*, 18:155–159, 1894. doi: 10.1007/BF02418278. URL https://doi.org/10.1007/BF02418278.

Hu, Y., Koren, Y., and Volinsky, C. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*, pp. 263–272. Ieee, 2008.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

Karimi, H., Nutini, J., and Schmidt, M. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 795–811. Springer, 2016.

Khaled, A., Mishchenko, K., and Richtárik, P. Better communication complexity for local sgd. *arXiv preprint arXiv:1909.04746*, 2019.

Kulunchakov, A. and Mairal, J. A generic acceleration framework for stochastic composite optimization. *arXiv preprint arXiv:1906.01164*, 2019.

Lin, H., Mairal, J., and Harchaoui, Z. A universal catalyst for first-order optimization. In *Advances in neural information processing systems*, pp. 3384–3392, 2015.

Lin, H., Mairal, J., and Harchaoui, Z. Catalyst acceleration for first-order convex optimization: from theory to practice. *arXiv preprint arXiv:1712.05654*, 2018.

Mishchenko, K., Iutzeler, F., Malick, J., and Amini, M.-R. A delay-tolerant proximal-gradient algorithm for distributed learning. In *International Conference on Machine Learning*, pp. 3587–3595, 2018.

Monteiro, R. D. and Svaiter, B. F. 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.

Nemirovsky, A. S. and Yudin, D. B. Problem complexity and method efficiency in optimization. 1983.

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

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

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

Nesterov, Y. and Vial, J.-P. Confidence level solutions for stochastic programming. *Automatica*, 44(6): 1559–1568, 2008.

Nesterov, Y., Gasnikov, A., Guminov, S., and Dvurechensky, P. Primal-dual accelerated gradient descent with line search for convex and nonconvex optimization problems. *arXiv preprint arXiv:1809.05895*, 2018.

Ogaltsov, A., Dvinskikh, D., Dvurechensky, P., Gasnikov, A., and Spokoiny, V. Adaptive gradient descent for convex and non-convex stochastic optimization. *arXiv preprint arXiv:1911.08380*, 2019.

Palaniappan, B. and Bach, F. Stochastic variance reduction methods for saddle-point problems. In *Advances in Neural Information Processing Systems*, pp. 1416–1424, 2016.

Paquette, C., Lin, H., Drusvyatskiy, D., Mairal, J., and Harchaoui, Z. Catalyst acceleration for gradient-based non-convex optimization. *arXiv preprint arXiv:1703.10993*, 2017.

Parikh, N., Boyd, S., et al. Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3): 127–239, 2014.

Polyak, B. T. *Introduction to optimization.* No. 04; QA402. 5, P6., 1987.

Rockafellar, R. T. Monotone operators and the proximal point algorithm. *SIAM journal on control and optimization*, 14(5):877–898, 1976.

Shamir, O. An optimal algorithm for bandit and zero-order convex optimization with two-point feedback. *Journal of Machine Learning Research*, 18:52:1–52:11, 2017.

Stich, S. U. Local sgd converges fast and communicates little. *arXiv preprint arXiv:1805.09767*, 2018.

Tupitsa, N., Dvurechensky, P., and Gasnikov, A. Alternating minimization methods for strongly convex optimization. *arXiv preprint arXiv:1911.08987*, 2019.

Wilson, A. C., Mackey, L., and Wibisono, A. Accelerating rescaled gradient descent: Fast optimization of smooth functions. In *Advances in Neural Information Processing Systems*, pp. 13533–13543, 2019.

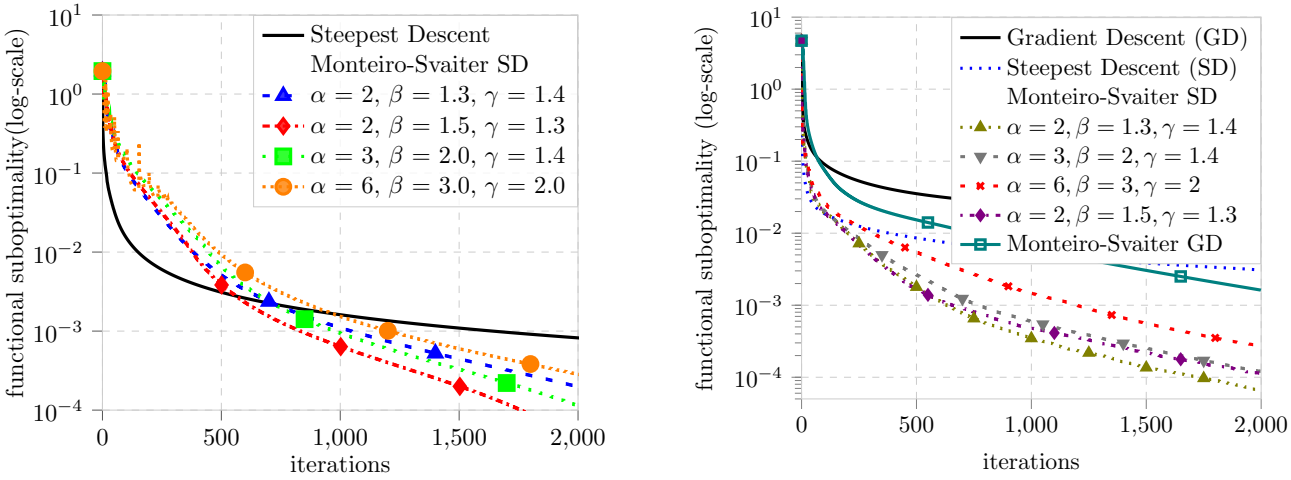
Woodworth, B. E., Wang, J., Smith, A., McMahan, B., and Srebro, N. Graph oracle models, lower bounds, and gaps for parallel stochastic optimization. In *Advances in neural information processing systems*, pp. 8496–8506, 2018.

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

A Additional experimental results

In this section, we provide additional numerical experiments for our algorithm. For all figures below, the vertical axis measures functional suboptimality $f(x) - f(x_\star)$ in the logarithmic scale.

A.1 Steepest Descent Acceleration



(a) rcv1_train (b) a1a

Figure 4: Logistic regression (15)

In this experimental setup we consider logistic loss minimization problem

$$\min_{x \in \mathbb{R}^n} \quad f(x) = \frac{1}{m} \sum_{j=1}^{m} \log(1 + \exp(-y_j z_j^\top x)) \tag{15}$$

with two different datasets from LIBSVM (Chang & Lin, 2011) repository (*rcv1_train* and *a1a*) We selected logistic regression as the objective as far as logistic regression converges with sub-linear rate like general non-strongly function that is important assumption for accelerated versions of algorithm to be provable.

In this setup we present our experimental results for *Steepest Descent* (Algorithm 3) and its accelerated via Algorithm 2 versions with different tuples of parameters (α, β, γ) to show the dependence of the algorithm on parameters.*

In Figure 4a we present functional suboptimality vs aggregated amount of gradient computations (oracle calls) in logarithmic scale. To be more precise, we present the following: for every "restart" we split all the points into two groups. First group - points from the inner algorithm run with "optimal" L_k. Second group is for the points, that are extra (cost of adaptation) and for this points we plot the value in point y^k from the previous restart (to have the horizontal lines on plots in cases when adaptation takes so long). In the end of the day, for each restart we, first, plot "horizontal line" for all points from the second group and after we present points from the first group with the corresponding to them values.

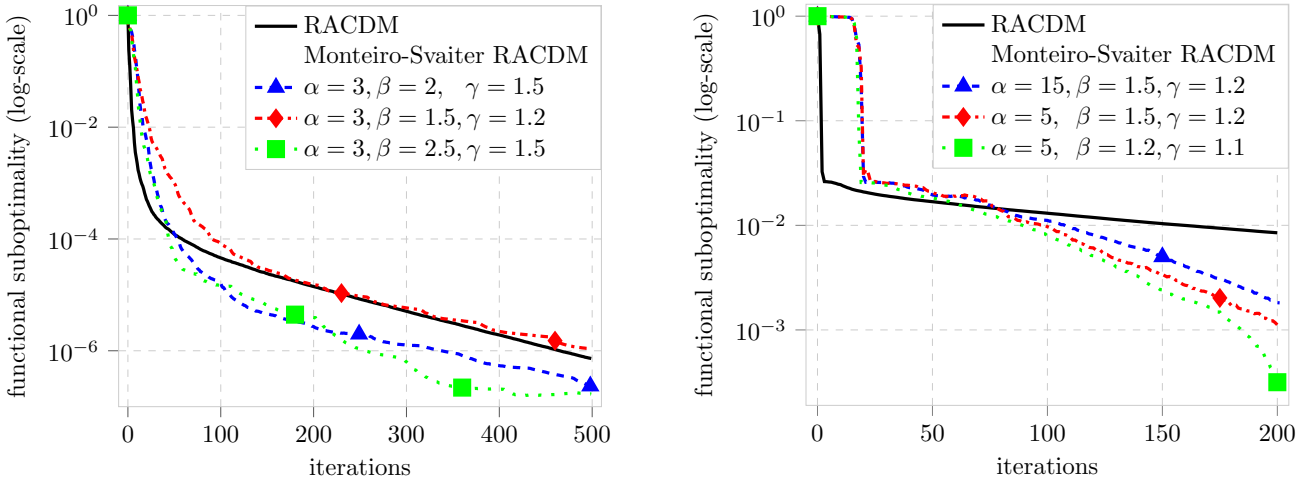
*For all runs with steepest descent we used $L_d = 0.0001 L_f$ and $L_u = L_f$, where L_f is a real Lipschitz constant of ∇f.

As we could see from the plot, acceleration happens when M-S acceleration scheme is used together with steepest descent algorithm but is highly dependent on the parameters of Monteiro-Svaiter algorithm. For instance, big α and β makes it harder to algorithm to adapt to the current "optimal" value of L_k that makes algorithm slower. Second, selecting big γ is not reasonable too as far as it corresponds to the big fluctuation of L_k during every restart. Moreover, selecting α and β close to each other also tends to slow down the convergence process.

In Figure 4b, we add also Gradient Descent algorithm to the list of presented algorithms. To be precise enough, we use Monteiro-Svaiter acceleration without any adaptation for L_k. It means, that fixed constant $L_k = L_f$ is used during algorithm run.

As we could see from the set of hyper parameters $(\alpha, \beta, \gamma) = (6, 3, 2)$ again leads to the slowest version of accelerated algorithm. All the other set ups, also have roughly the same behavior. Finally, for Gradient Descent acceleration also takes place and even makes it faster than Steepest Descent without acceleration. An important thing to notice is the following: Monteiro-Svaiter acceleration for adaptive algorithms (Steepest Descent) makes them faster than acceleration of non-adaptive (Gradient Descent) in spite of cost for adaptation.

A.2 Random Adaptive Coordinate Descent Method (RACDM) Acceleration



(a) Synthetic quadratic problem (12) with matrix $A, n = 100$.

(b) Logistic regression problem (15) for *madelon* dataset.

Figure 5: Results of RACDM acceleration for different problems

A.2.1 Quadratic Problem

Let us consider a simple case of quadratic optimization

$$\min_{x \in \mathbb{R}^n} \quad f(x) = \frac{1}{2} x^\top A x,$$

where matrix $A = S^\top D S$ is synthetically generated in such a way that S is a random orthogonal matrix. The elements of diagonal matrix D are sampled from standard uniform distribution $\mathcal{U}(0, 1)$

and one random D_{ii} is assigned to zero to guarantee that the smallest eigenvalue of the resulting matrix A is smaller than 10^{-15} and thus the optimization problem is convex but not strongly-convex (up to machine precision).*

In Figure 5a we compare the performance of RACDM and its M-S accelerated version with different sets of parameters (α, β, γ) for problem A.2.1. For the horizontal axis we use number of partial derivative evaluations divided by dimensionality n of the problem. Our warm start strategy includes running inner method from the last point y_k and with estimates $\hat{\beta}_i$ of smoothness constants obtained from the previous iteration. The initial points $y_0 = z_0$ are sampled from the standard uniform distribution $\mathcal{U}(0, 1)$. L_0 was initialized as $1.6L_f$ and $L_d = 0.005L_f, L_u = 10L_f, \beta_i^0 = 1/L_0$. We observe that clear acceleration can be achieved not for all sets of parameters. Concretely, both β and γ affected convergence as one can see from the plot. Besides, we find out that for higher accuracy the proposed method can show unstable performance.

Note, that we can obtain provable acceleration by the proposed Adaptive Catalyst procedure only for convex problems. For strongly convex problems, this is no longer true either in theory or in our experiments. The reason is that the proposed M-S accelerated envelop doesn't take in to account possible strong convexity. Moreover, as far as we know, this is still an open problem, to propose a fully adaptive accelerated algorithm for strongly convex problems. The problem is in the strong convexity parameter. In practice, we met this problem in different places. For example, when we choose the restart parameter for conjugate gradient methods or try to propose accelerated (fast) gradient methods that do not require any information about strong convexity parameter but know all other characteristics of the problem.

A.2.2 Logistic Regression

In Figure 5b we present the results for logistic regression problem (15) with *madelon* dataset from LIBSVM (Chang & Lin, 2011) repository. Initial parameters for this set up are $L_0 = L_f, L_d = 10^{-5}, L_u = 100L_f$. Warm start strategy and depicted values for the horizontal axis are the same as for the quadratic problem. It is important to mention that gradient norm computation for checking Monteiro-Svaiter condition $\left\|\nabla F_{L,x}^{k+1}\left(y^{k+1}\right)\right\|_2 \leq \frac{L_{k+1}}{2}\left\|y^{k+1} - x^{k+1}\right\|_2$ and full gradient step from the outer loop $z^{k+1} = z^k - a_{k+1}\nabla f\left(y^{k+1}\right)$ (according to Algorithm 2) were counted as evaluation of n partial derivative. For this case, we also noted that L_0 initialization affects the convergence significantly at the beginning and it has to be chosen lower than in the previous cases.

A.3 Alternating Least Squares Acceleration

In addition to the results presented in paragraph 5.2, consider the performance of the Alternating Least Squares algorithm for the problem with a larger being estimated matrix $\{r_{ui}\}_{u,i}$.

In Figure 6 we compare the performance of the Alternating Least Squares algorithm and its accelerated via Monteiro-Svaiter algorithm versions for problem 13 with $\lambda = 5$ and matrix $\{r_{ui}\}_{u,i}$ of the size 150 users $\times$ 300 artists, generated from radio dataset. The horizontal axis measures the number of variables recomputations. The plot shows that there was the acceleration of the base algorithm and both parameters β and γ had an impact on the convergence rate.

*Frankly speaking, for such objective functions we observe that non-accelerated gradient descent based algorithms converge with linear rate, because of the quadratic nature of the problem and specificity of spectrum. Since $n = 100$ in these experiments we typically have that the next eigenvalue after zero is about 0.01. This value determined the real rate of convergence

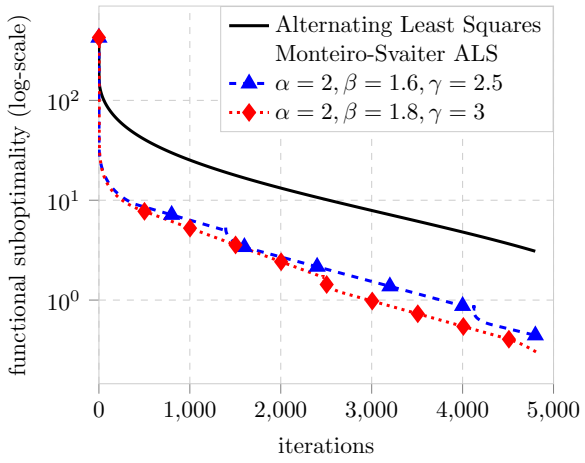


Figure 6: Matrix completion problem (13) with different (α, β, γ).

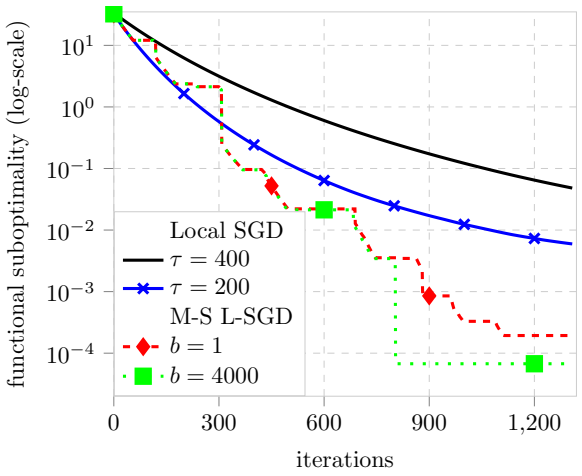


Figure 7: Logistic loss (14) minimization with minibatching.

A.4 Local SGD Acceleration

Algorithm 6 Local SGD algorithm

Parameters: $x^0 \in \mathbb{R}^n, w$ — number of workers, L, μ,
$\mathcal{S}_N$ — set of synchronization steps indices
τ — maximum difference between two consequent integers in $\mathcal{S}_N$
Initialize variables $x_h^0 = x^0$ for $h \in [1, w]$
for $k = 0, 1, \ldots, N-1$ **do**
  for $h \in \{1, \ldots, w\}$ **do in parallel**
    Sample i_h^k uniformly in $[1, m]$
    if $k+1 \in \mathcal{S}_N$ **then**
      $x_h^{k+1} = \frac{1}{w} \sum_{j=1}^{w} \left(x_j^k - \eta_k \nabla f_{i_h^k}(x_j^k) \right)$
    else
      $x_h^{k+1} = x_h^k - \eta_k \nabla f_{i_h^k}(x_h^k)$
    end if
  end for
end for
Output: $\hat{x}^N = \frac{1}{w S_N} \sum_{h=1}^{w} \sum_{k=0}^{N-1} \xi^k x_h^k$, where $\xi^k = (\max\{16L/\mu, \tau\} + k)^2$, $S_N = \sum_{k=0}^{N-1} \xi^k$.

In addition to the results presented in paragraph 5.3, consider the performance of the Algorithm 6 with applying the minibatch technique.

In Figure 7 we compare the performance of the Algorithm 6 and its Monteiro-Svaiter (with parameters $\alpha = 1.15, \beta = 1.12, \gamma = 1.1$ and $\tau = 200$) accelerated versions ($w = 20$ and $\tau \in \{200, 400\}$) for problem (14). The horizontal axis measures the number of outer iterations (one outer iteration includes the recomputation of the variables in all the computing nodes). The plot shows that there was the acceleration of the base algorithm and, moreover, that using a batch of samples instead of one sample for calculating the stochastic gradient can improve the convergence rate.