

Adaptive Catalyst for Smooth Convex Optimization

Anastasiya Ivanova^{a,b} and Dmitry Pasechnyuk^a and Dmitry Grishchenko^c and Egor Shulgin^{a,d} and Alexander Gasnikov^{a,b,e} and Vladislav Matyukhin^a

^aMoscow Institute of Physics and Technology, Moscow, Russia;

^bNational Research University Higher School of Economics, Moscow, Russia;

^cUniversité Grenoble Alpes, Grenoble, France;

^dKing Abdullah University of Science and Technology, Thuwal, Saudi Arabia;

^eInstitute for Information Transmission Problems, Moscow, Russia

ABSTRACT

In this paper, we present a generic framework that allows accelerating almost arbitrary non-accelerated deterministic and randomized algorithms for smooth convex optimization problems. The major approach of our *envelope* is the same as in *Catalyst* [43]: an accelerated proximal outer gradient method, which is used as an envelope for a 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 can be tuned along the way. As a result, the main contribution of our work 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 [43, 44].

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 [51]. Until 2015 acceleration schemes for different convex optimization problems seem to be quite different to unify them. But starting from the work [43] in which universal acceleration technique (*Catalyst*) was proposed, there appears a stream of subsequent works [40, 44, 55, 56] that allows spreading Catalyst on monotone variational inequalities, non-convex problems, stochastic optimization problems. In all these works, the basic idea is to use an accelerated proximal algorithm as an outer envelope [60] 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 envelope [57] was proposed in the paper [47]. The main difference with standard accelerated proximal envelopes is the adaptability of

The research of A. Gasnikov and V. Matyukhin was supported by the Ministry of Science and Higher Education of the Russian Federation (Goszadaniye) №075-00337-20-03, project No. 0714-2020-0005.

the scheme [47]. Note, that this scheme allows also to build (near) optimal tensor (high-order) accelerated methods [20, 21, 23, 51, 66]. 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 [44, 57] on the adaptive accelerated proximal envelope from [47].**

In Section 2, we describe adaptive Catalyst envelope – Algorithm 1 and generalized Monteiro-Svaiter theorem from [47] to set out how to do this envelope work. We emphasize that the proof of the theorem contains as a byproduct the new theoretical analysis of the stopping criterion for the inner algorithm (9). This stopping criterion allows one to show that the proposed envelope in non-adaptive mode is log-times better (see Corollary 1) in the total number of oracle calls of the inner method (we’ll measure the complexity of the envelope in such sense) in comparison with all other envelopes known for us.

By using this adaptive accelerated proximal envelope, we propose in Section 3 an accelerated variant of steepest descent [20, 59] as an alternative to A. Nemirovski accelerated steepest descent (see [8, 52] and references therein), adaptive accelerated variants of alternating minimization procedures [3] as an alternative to [7, 28, 65] and adaptive accelerated coordinate descent [49]. 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 [17] 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, adaptive coordinate descent, alternating minimization and local SGD. We try to accelerate these methods by the envelope, described in Algorithm 1.

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 [24, 26, 27].

The first draft of this paper appeared in arXiv in November 2019. Since that time, this paper has developed (and cited) in different aspects. The main direction is a convenient (from the practical (6) and theoretical (9) point of view) criterion of stopping the inner algorithm that is wrapped in accelerated proximal envelope. We emphasize that our contribution in this part is not a new accelerated proximal envelope (we use the well-known envelope [47]), but we indicate that this envelope is better than the other ones due to the new theoretical analysis of its inner stopping criteria that lead us from (6) to (9). Although this calculation looks simple enough, to the best of our knowledge, this was the first time when it was provably developed an accelerated proximal envelope that required to solve the auxiliary problem with prescribed relative accuracy in argument $\simeq 1/5$. Since the auxiliary problem is smooth and strongly convex, this observation eliminates the logarithmic factor (in the desired accuracy) in the complexity estimate for such an envelope in comparison with all known analogues. Note that this small observation will have a remarkable influence on the development of accelerated algorithms. A close stopping condition, for instance, arises in the following papers [9, 10] that developed (sub-)optimal accelerated tensor method based on accelerated proximal envelopes. The proposed “logarithm-free” envelope allows one to

improve the best known bounds [45] for strongly convex-concave saddle-point problems (with different constants of strong convexity and concavity) on logarithmic factor [12]. Composite variant of this envelope also allows one to develop “logarithm-free” gradient sliding-type methods² [12, 13, 32] and its tensor generalizations [34]. Moreover, one of the variants of the hyper-fast second-order method was also developed based on this envelope [35]. Though this envelope had known before, it seems that the original idea of this paper to use this envelope in Catalyst type procedures and new (important from the theoretical point of view) reformulation of stopping criteria for inner algorithm (9) has generated a large number of applications, some of them mentioned in this paper, the others can be found in the literature cited above. As an important example, we show in section 3.2 that the developed envelope with non-accelerated coordinate descent method for auxiliary problem works much better in theory (and better in practice) than all known direct accelerated coordinate-descent algorithms for sparse soft-max type problem. Before this article, this was an open problem, how to beat standard accelerated coordinate-descent algorithms that don’t allow one to take into account sparsity of the problem for soft-max type functional [18, 58].

The other contribution of this paper is an adaptive choice of the smooth parameter. Since our approach requires two inputs (lower bound L_d and upper bound L_u for the unknown smoothness parameter L), it’s hardly possible to call it “adaptive”. Moreover, the greater the discrepancy between these two parameters, the worthier is our adaptive envelope in theory. But almost all of our experiments demonstrate low sensitivity to these parameters rather than to real smoothness parameter. But even for such a “logarithm-free” and adaptive envelope, we expect that typically the direct adaptive accelerated procedures will work better than its Catalyst type analogues. It was recently demonstrated in the following work [64] for accelerated alternating minimization procedure. But even to date, there are problems in which one can expect that firstly optimal accelerated algorithms will be developed by using Catalyst type procedures rather than direct acceleration. Recent advances in saddle-point problems [12, 45, 70] and decentralized distributed optimization³ [29, 41] confirm this thought. We expect that for Homogeneous Federated Learning architectures, Accelerated Local SGD can be developed (see [67] for the state of the art approach) by using Catalyst-type envelope with SCAFFOLD version of local SGD algorithm [37]. As far as we know, it’s still an open problem to build Accelerated local SGD as an inner algorithm. In this paper, we demonstrate some optimistic experiments in this direction.

2. The Main Scheme

Let us consider the following minimization problem

$$\min_{y \in \mathbb{R}^n} f(y), \tag{1}$$

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

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

²Note, that [13] contains variance reduction [1, 61] generalization (with non proximal-friendly composite) of proposed in this paper scheme.

³Note, that the results of these papers were further reopened by using direct acceleration [39, 42].

We denote x_\star a solution of (1).

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

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

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

$$\|\nabla F_{L,x}(y_2) - \nabla F_{L,x}(y_1)\|_2 \leq (L + L_f)\|y_1 - y_2\|_2. \quad (2)$$

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 [59] [Theorem 5, ch. 6], since

$$x_\star \in \underset{x \in \mathbb{R}^n}{\text{Argmin}} f_L(x),$$

we obtain

$$x_\star \in \underset{x \in \mathbb{R}^n}{\text{Argmin}} f(x) \quad \text{and} \quad f_L(x_\star) = f(x_\star).$$

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

$$\min_{x \in \mathbb{R}^n} f_L(x). \quad (3)$$

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) [51] 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 $\min_{y \in \mathbb{R}^n} F_{L,x}(y)$.

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 [57])

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

To develop an adaptive proximal accelerated envelop, we should replace standard FGM [51] on the following adaptive variant of FGM Algorithm 1, introduced by [47] for smooth convex optimization problems.

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

Algorithm 1 Monteiro–Svaiter algorithm

Parameters: $z^0, y^0, A_0 = 0$

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

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

$$z^{k+1} = z^k - a_{k+1} \nabla f(y^{k+1})$$

end for

Output: y^N

Theorem 1 (Theorem 3.6 [47]). Let sequence (x^k, y^k, z^k) , $k \geq 0$ be generated by Algorithm 1 and define $R := \|y^0 - x_\star\|_2$. Then, for all $N \geq 0$,

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

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

$$\sum_{k=1}^N A_k L_k \|y^k - x^k\|_2^2 \leq 2R^2. \quad (4)$$

We also need the following Lemma.

Lemma 1 (Lemma 3.7a [47]). Let sequences $\{A_k, L_k\}$, $k \geq 0$ be generated by Algorithm 1. Then, for all $N \geq 0$,

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

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

Assumption 1. The convergence rate (after t iterations / oracle calls) 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)⁴

$$F(y_t) - F(y_\star) = O\left(L_F R_y^2 \ln \frac{t}{\delta}\right) \min\left\{\frac{C_n}{t}, \exp\left(-\frac{\mu_F t}{C_n L_F}\right)\right\},$$

where y_\star is the solution of the problem, $R_y = \|y^0 - y_\star\|_2$, 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, [6, 59] for the steepest descent and [7, 36, 65] 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 see [2, 11, 14, 15, 19, 62], for coordinate descent methods see [49, 53, 69] and for more general situations see [27].

Remark 2. Note that in Assumption 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. Taking into account that $\tilde{O}(\cdot)$ means the same as $O(\cdot)$ up to a logarithmic factor, based on the Monteiro–Svaiter Theorem 1 we can introduce the following theorem:

Theorem 2. Consider Algorithm 2 with $0 < L_d < L_u$ for solving problem (1), where $Q = \mathbb{R}^n$, with auxiliary (inner) non-accelerated algorithm (method) \mathcal{M} that satisfy Assumption 1 with constants C_n and L_f such that $L_d \leq L_f \leq L_u$.

Then the total complexity⁵ 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

$$\|\nabla F_{L,x}^{k+1}(y^{k+1})\|_2 \leq \frac{L_{k+1}}{2} \|y^{k+1} - x^{k+1}\|_2 \quad (6)$$

⁴For deterministic algorithms we can skip “with probability at least $1 - \delta$ ” and factor “ $\ln \frac{N}{\delta}$ ”.

⁵The number of oracle calls (iterations) of auxiliary method \mathcal{M} that required to find ε solution of (1) in terms of functions value.

Algorithm 2 Adaptive Catalyst

Parameters: Starting point $x^0 = y^0 = z^0$; initial guess $L_0 > 0$; parameters $\alpha > \beta \gtrsim \gamma > 1$; optimization method \mathcal{M} , $A_0 = 0$.

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

$$L_{k+1} = \beta \cdot \min \{ \alpha L_k, L_u \}$$

$$r = 0$$

repeat

$$r := r + 1$$

$$L_{k+1} := \max \{ L_{k+1} / \beta, L_d \}$$

Compute

$$a_{k+1} = \frac{1/L_{k+1} + \sqrt{1/L_{k+1}^2 + 4A_k/L_{k+1}}}{2},$$

$$A_{k+1} = A_k + a_{k+1},$$

$$x^{k+1} = \frac{A_k}{A_{k+1}} y^k + \frac{a_{k+1}}{A_{k+1}} z^k.$$

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

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

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

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

until $r > 1$ and $N_r \geq \gamma \cdot N_{r-1}$ or $L_{k+1} = L_d$

$$z^{k+1} = z^k - a_{k+1} \nabla f(y^{k+1})$$

end for

Output: y^N

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

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

Using the triangle inequality we have

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

Since r.h.s. of the inequality (8) coincide with the r.h.s. of the M-S condition and l.h.s. of the inequality (7) coincide with the l.h.s. of the M-S condition up to a multi-

plicative factor $L_{k+1}/2$, one can conclude that if the inequality

$$\|y^{k+1} - y_\star^{k+1}\|_2 \leq \frac{L_{k+1}}{3L_{k+1}+2L_f} \|x^{k+1} - y_\star^{k+1}\|_2 \quad (9)$$

holds, the M-S condition holds too.

To solve the auxiliary problem $\min_y F_{L_{k+1}, x^{k+1}}(y)$ we use non-accelerated method \mathcal{M} .

Using Assumption 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 (after t iterations of \mathcal{M} , see Assumption 1)

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

Note, that $R_{k+1} = \|x^{k+1} - y_\star^{k+1}\|_2$ since x^{k+1} is a starting point.

Since $F_{L,x}^{k+1}(\cdot)$ is L_{k+1} -strongly convex function, the following inequality holds [51]

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

Thus,

$$\|y_t^{k+1} - y_\star^{k+1}\|_2 \leq O\left(\sqrt{\frac{(L_f + L_{k+1})R_{k+1}^2}{L_{k+1}} \ln \frac{Nt}{\delta}}\right) \exp\left(-\frac{L_{k+1}t}{2C_n(L_f + L_{k+1})}\right). \quad (10)$$

From (9), (10) and the fact that we start \mathcal{M} at x^{k+1} , we obtain that the complexity T (number of iterations of \mathcal{M}) of solving the auxiliary problem with probability at least $1 - \frac{\delta}{N}$ is determined from

$$O\left(R_{k+1} \sqrt{\frac{(L_f + L_{k+1})}{L_{k+1}} \ln \frac{NT}{\delta}}\right) \exp\left(-\frac{L_{k+1}T}{2C_n(L_f + L_{k+1})}\right) \simeq \frac{L_{k+1}}{3L_{k+1}+2L_f} R_{k+1}, \quad (11)$$

hence

$$T = \tilde{O}\left(C_n \frac{(L_{k+1} + L_f)}{L_{k+1}}\right). \quad (12)$$

Since we use in (12) $\tilde{O}(\cdot)$ notation, we can consider T to be the estimate that corresponds to the total complexity of auxiliary problem including all inner restarts on L_{k+1} .

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

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

Since the complexity of the auxiliary problem with probability at least $1 - \frac{\delta}{N}$ is 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. \square

Note that this result shows that such a procedure will work not worse than standard Catalyst [43, 44] 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 [43, 44] and have very simple explanation. To minimize the total complexity we should take parameter $L_{k+1} \equiv L$ such that

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

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 non-randomized method \mathcal{M} is

$$O\left(C_n \sqrt{\frac{L_f R^2}{\varepsilon}}\right). \quad (13)$$

Proof. Using (11) without $\ln \frac{NT}{\delta}$ factor (since \mathcal{M} is non-randomized) we derive that

the complexity of the auxiliary problem is (see also (12))

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

And since we choose $L_{k+1} \equiv L_f$,

$$\frac{3L_{k+1} + 2L_f}{L_{k+1}} = 5$$

Then the complexity of the auxiliary problem is $T = O(C_n)$. Using this estimate, we obtain that the total complexity is (13). \square

If method \mathcal{M} is randomized we have the additional factor $\ln \frac{NT}{\delta} \simeq \ln\left(\frac{1}{\delta\varepsilon}\right)$. Hence, (13) changes: with probability at least $1 - \delta$

$$O\left(C_n \ln\left(\frac{1}{\delta\varepsilon}\right) \sqrt{\frac{L_f R^2}{\varepsilon}}\right).$$

Note that in the standard Catalyst approach [43, 44] the total complexity is $O\left(C_n \ln\left(\frac{1}{\delta\varepsilon}\right) \sqrt{\frac{L_f R^2}{\varepsilon}} \cdot \ln\left(\frac{1}{\varepsilon'}\right)\right)$, where $\varepsilon' = \text{Poly}(\varepsilon)$ is the relative accuracy of solving the auxiliary problem at each iteration. 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 $\ln\left(\frac{1}{\varepsilon'}\right)$. 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 [46].

3. Applications

In this section, we present a few examples of algorithms that we consider as inner solvers. Most of them have an adaptive structure. It's natural to apply adaptive envelope to adaptive algorithms since the developed methods keep adaptability.

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 [59] it was proposed an adaptive way to select h_k as following (see also [6] for precise rates of convergence)

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

Algorithm 3 Steepest descent

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

In contrast with the standard selection $h_k \equiv \frac{1}{L_f}$ for L -smooth functions f , in this method there is 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 Assumption 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 β_1, \dots, β_n such that for any $x \in \mathbb{R}^n, u \in \mathbb{R}$

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

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 [49, 69].

Note that for Algorithm 4 the Assumption 1 holds with $C_n = O(n)$ (for $x \in \mathbb{R}^n$) and⁶ $L_f = \bar{L}_f := \frac{1}{n} \sum_{i=1}^n \beta_i$ (the average value of the directional smoothness parameters).

As one of the motivational example, consider the following minimization problem

$$\min_{x \in \mathbb{R}^n} f(x) = \gamma \ln \left(\sum_{i=1}^m \exp \left(\frac{[Ax]_i}{\gamma} \right) \right) - \langle b, x \rangle,$$

where $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$. We denote the i^{th} row of the matrix A by A_i . A is sparse, i.e. average number of nonzero elements in A_i is less than s . f is L_f -smooth w.r.t.

⁶Strictly speaking, such a constant takes place for non-adaptive variant of the CDM with specific choice of i_k [49]: $\pi(i_k = j) = \frac{\beta_j}{\sum_{j'=1}^n \beta_{j'}}$. For described RACDM the analysis is more difficult [58].

Algorithm 4 RACDM

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

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

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

$\|\cdot\|_2$ with $L_f = \max_{i=1, \dots, m} \|A_i\|_2^2$ and its gradient is component-wise β_j -continuous with $\beta_j = \max_{i=1, \dots, m} |A_{ij}|$.

Fast Gradient Method (FGM) [52] requires $O\left(\sqrt{\frac{L_f R^2}{\varepsilon}}\right)$ iterations with the complexity of each iteration $O(ns)$.

Coordinate Descent Method (CDM) [4] requires $O\left(n\frac{\bar{L}_f R^2}{\varepsilon}\right)$ iterations with the complexity of each iteration⁷ $O(s)$.

Accelerated Coordinate Descent Method (ACDM) [22, 53] requires $O\left(n\sqrt{\frac{\tilde{L}_f R^2}{\varepsilon}}\right)$ iterations with the complexity of each iteration $O(n)$, where $\tilde{L}_f = \frac{1}{n} \sum_{j=1}^n \sqrt{\beta_j}$.

For proposed in this paper approach we have $O\left(n\sqrt{\frac{\bar{L}_f R^2}{\varepsilon}}\right)$ iterations of CGM with complexity of each inner iteration $O(s)$ and complexity of each outer iteration $O(ns)$. However, outer iteration executes ones per $\sim n$ inner iterations, so average-case iteration complexity is $O(s)$.

We combine all these results in the table below. From the table one can conclude that if $\bar{L}_f < L_f$, then our approach has better theoretical complexity.

Algorithm	Complexity	Reference
FGM	$O\left(ns\sqrt{\frac{L_f R^2}{\varepsilon}}\right)$	[52]
CDM	$O\left(ns\frac{\bar{L}_f R^2}{\varepsilon}\right)$	[4, 49]
ACDM	$O\left(n^2\sqrt{\frac{\tilde{L}_f R^2}{\varepsilon}}\right)$	[53]
Catalyst CDM	$O\left(ns\sqrt{\frac{\bar{L}_f R^2}{\varepsilon}}\right)$	this paper

⁷Here one should use a following trick in recalculation of $\ln(\sum_{i=1}^m \exp([Ax]_i))$ and its gradient (partial derivative). From the structure of the method we know that $x^{new} = x^{old} + \delta e_i$, where e_i is i -th orth. So if we've already calculate Ax^{old} then to recalculate $Ax^{new} = Ax^{old} + \delta A_i$ requires only $O(s)$ additional operations independently of n and m .

Note that the use of Component Descent Method allows us to improve convergence estimate by factor \sqrt{n} compared to Fast Gradient Method. Indeed, for this problem we have $L_f = \max_{i=1,\dots,m} \|A_i\|_2^2 = O(n)$, and on the other hand $\bar{L}_f = \frac{1}{n} \sum_{j=1,\dots,n} \max_{i=1,\dots,m} |A_{ij}| = O(1)$. Therefore, the total convergence estimate for Fast Gradient Method can be written as

$$O\left(ns \cdot \sqrt{n} \cdot \sqrt{\frac{R^2}{\varepsilon}}\right),$$

and for proposed in this paper method factor \sqrt{n} is reduced to $O(1)$ and could be omitted:

$$O\left(ns \cdot \sqrt{\frac{R^2}{\varepsilon}}\right).$$

The best complexity improvement is achieved if $L_f = n$, which means there is at least one row in the matrix such that $A_i = \mathbb{1}^n$, even though all other rows can be arbitrary sparse.

3.3. Alternating Minimization

Consider the following problem

$$\min_{x=(x_1,\dots,x_p)^T \in \otimes_{i=1}^p \mathbb{R}^{n_i}} f(x),$$

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).

For the general case of number of blocks $p \geq 2$ 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 [3, 7, 36, 65].

Algorithm 5 Alternating Minimization

Parameters: Starting point x^0 .
for $k = 0, 1, \dots, N - 1$ **do**
 Choose $i \in \{1, \dots, p\}$
 Set $x^{k+1} = \underset{x_i}{\operatorname{argmin}} f(x_1^k, \dots, x_{i-1}^k, x_i, x_{i+1}^k, \dots, x_p^k)$
end for
Output: x^N

Note that for Algorithm 5 the Assumption 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 [38] becomes popular first of all due to the application in federated learning [33]. In the core of the approach lies a very simple idea [16, 38, 50, 63]: to solve considered stochastic convex optimization problem in parallel on M nodes via Adaptive

Stochastic Gradient Descent (SGD) [54] with synchronization after each τ iterations of SGD and sharing an average point. The larger 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 $M\tau \simeq T$ [38]. 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 [48]. On the other hand, if we reduce the variance by batching acceleration could play a significant role [25, 68]. 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. From the theoretical perspective, an acceleration of local SGD is still an open problem [67] rather than acceleration of ordinary SGD (see, for example, [18] and reference therein).

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 $k = 0, 1, \dots, N - 1$ **do**
 for $h \in \{1, \dots, w\}$ **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: $\hat{x}^N = \frac{1}{wS_N} \sum_{h=1}^w \sum_{k=0}^{N-1} \xi^k x_h^k$, where $\xi^k = (\max\{16L/\mu, \tau\} + k)^2$, $S_N = \sum_{k=0}^{N-1} \xi^k$.

3.5. Theoretical Guarantees

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 Assumption 1. But in the case of acceleration of these methods, we apply them to a regularized function which is strongly convex.

Denote $\chi = \max \left(\sqrt{\frac{L_u}{L_f}}, \sqrt{\frac{L_f}{L_d}} \right)$, 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 $f(x) - f(x_*)$ in the logarithmic scale.

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

4.1. Steepest Descent Acceleration

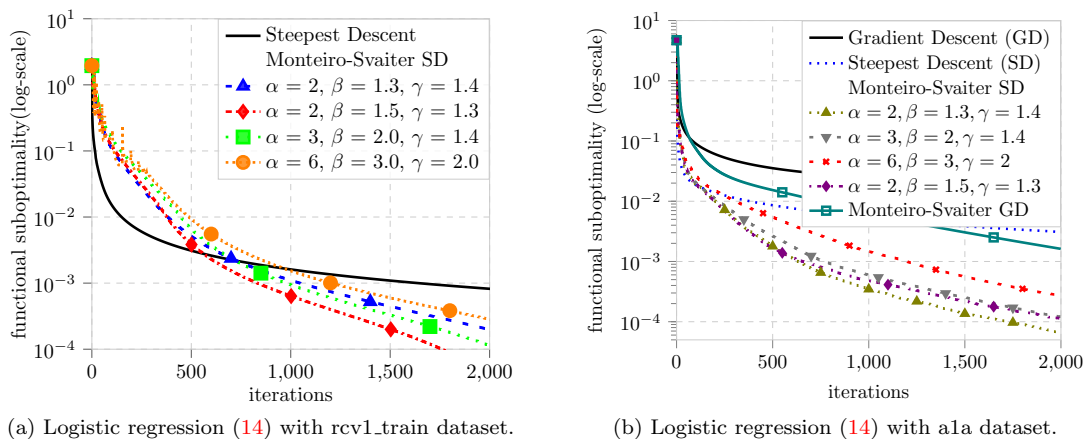


Figure 1. Results of Steepest Descent acceleration for different problems

In this experimental setup we consider logistic loss minimization problem

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

with two different datasets from LIBSVM [5] 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.⁸

In Figure 1a 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+1} . 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.

⁸For all runs with steepest descent we used $L_d = 0.0001L_f$ 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 but is highly dependent on the parameters of Monteiro–Svaiter algorithm. For instance, big α and β make it harder to algorithm to adapt to the current “optimal” value of L_{k+1} that makes algorithm slower. Second, selecting big γ is not reasonable too as far as it corresponds to the big fluctuation of L_{k+1} during every restart. Moreover, selecting α and β close to each other also tends to slow down the convergence process.

Let us give some intuition why this happens. Let us recall, that parameters α and β impact on the speed of adaptation. More precisely, the decrease of estimation L_{k+1} after the one iterate is by factor of β^p/α , where $p \in \mathbb{Z}_+$. This implies two different things:

- if $L_{k+1} < L_f$ the “optimistic” amount of adaptation rounds is $\log_\alpha \frac{L_f}{L_{k+1}}$, that is very big if α is close to 1;
- if $L_f \in \left(\frac{L_{k+1}}{\beta}, L_{k+1}\right)$ the “worst” amount of adaptation rounds is $\log_{\frac{\beta}{\alpha}} \beta$, that is very big if α is close to β .⁹

Combining these one can find the explanation of the dependence between α , β , and the rate of convergence.

In Figure 1b, 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+1} . It means, that fixed constant $L_{k+1} = L_f$ 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.

4.2. RACDM Acceleration

Let us consider quadratic optimization problem

$$\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{2} x^\top A x, \quad (15)$$

for Hilbert matrix [30] 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 $\approx 1.5 \cdot 10^7$ [59]). It leads to a degenerate optimization problem, typically very hard for gradient methods.

In Figure 2 we compare the performance of the method 4 and its M-S accelerated version with different sets of parameters (α, β, γ) for problem (15).

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 outer (M-S) iteration. The initial points $y_0 = z_0$ entries

⁹If $L_f < L_{k+1}/\beta$ then the scaling by factor of β give us the situation described in this case.

were sampled from the standard uniform distribution $\mathcal{U}(0,1)$. L_0 was initialized as $0.5L_f$ and $L_d = 0.001L_f, L_u = 100L_f, \beta_i^0 = 1/L_0$. Here we introduce the relationship between L_d, L_u , and L_f only from the theoretical interest; however, the dependence between L_d, L_u , and L_f is never used in the algorithm.

Consider a simple case of quadratic optimization problem (15) with matrix $A = S^\top DS$ such 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).¹⁰

In Figure 3a we compare the performance of RACDM and its M-S accelerated version with different sets of parameters (α, β, γ) .

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 envelope 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.

Consider logistic loss minimization problem

$$\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{m} \sum_{j=1}^m \log(1 + \exp(-y_j z_j^\top x)). \quad (16)$$

¹⁰Frankly 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

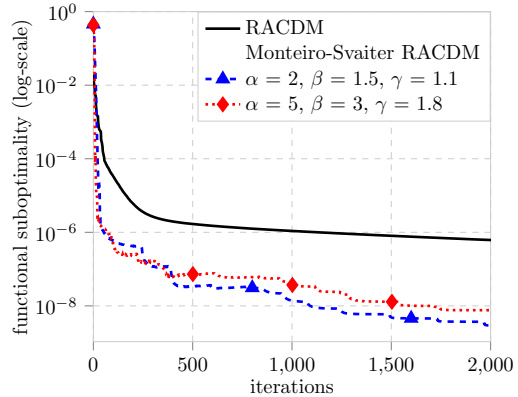


Figure 2. Results of RACDM acceleration for quadratic problem (15) with Hilbert matrix, $n = 1000$.

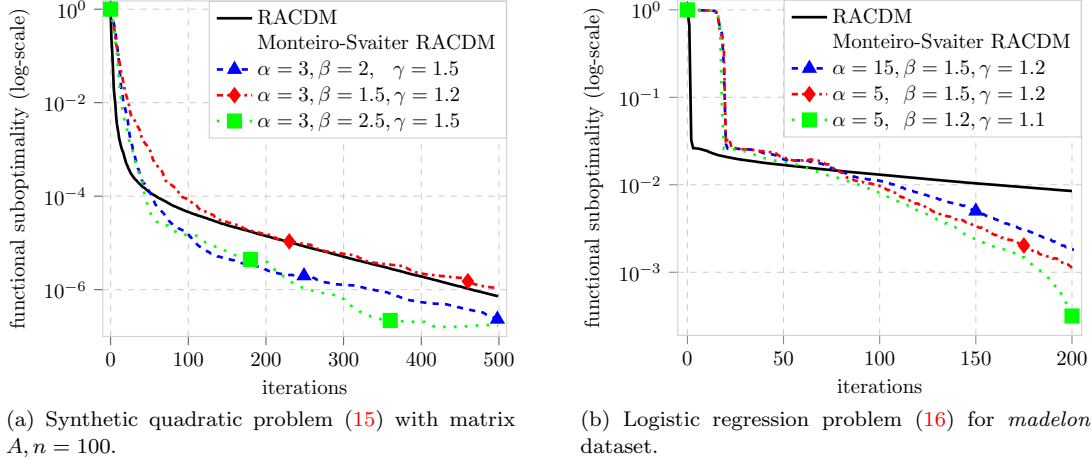


Figure 3. Results of RACDM acceleration for different problems

In Figure 3a we compare the performance of RACDM and its M-S accelerated version with different sets of parameters (α, β, γ) for logistic regression problem with *madelon* dataset from LIBSVM [5] 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 $\left\| \nabla F_{L,x}^{k+1}(y^{k+1}) \right\|_2 \leq \frac{L_{k+1}}{2} \|y^{k+1} - x^{k+1}\|_2$ and full gradient step from the outer loop $z^{k+1} = z^k - a_{k+1} \nabla f(y^{k+1})$ (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.

Consider also the following softmax function minimization problem

$$\min_{x \in \mathbb{R}^n} f(x) = \gamma \ln \left(\sum_{i=1}^m \exp \left(\frac{[Ax]_i}{\gamma} \right) \right) - \langle b, x \rangle, \quad (17)$$

where matrix $A \in \mathbb{R}^{m \times n}$ is such that average number of nonzero elements in A_i is less than $s \ll m$ and one of the rows A_k is non-sparse. f is Lipschitz smooth with the constant $L_f = \max_{i=1, \dots, m} \|A_i\|_2^2$ and has component-wise Lipschitz continuous gradient with constants $\beta_j = \max_{i=1, \dots, m} |A_{ij}|$.

In figures 4 and 5 we compare the performance of the Accelerated Coordinate Descent Method (ACDM) [53], Gradient Method (GM) [51], Fast Gradient Method (FGM) [51], and proposed approach – accelerated via Algorithm 2 variant of non-adaptive Coordinate Descent Method (Catalyst CDM) for strongly convex w.r.t. $\|\cdot\|_2$ auxiliary problem, in which i_k is drawn from the distribution π defined by [49]

$$\pi(i_k = j) = \frac{\beta_j}{\sum_{j'=1}^n \beta_{j'}}.$$

In the case of randomly-generated A with $A_{ij} \in \{0, 1\}$, $s \approx 0.2m$, non-sparse row A_k

with uniformly random components and $\gamma = 0.6$ proposed method converges faster (in terms of working time) than all comparable methods except FGM. However, in other setting, for the smaller count of nonzero elements in the matrix ($s \approx 0.75m$), non-sparse row $A_k = \mathbb{1}^n$, and with higher variation in the sparsity of the rows of A ($0.9m$ rows with $0.1n$ nonzero elements and $0.1m$ rows with $0.9n$ nonzero elements) — proposed method converges faster than FGM due to $\bar{L}_f \ll L_f$.

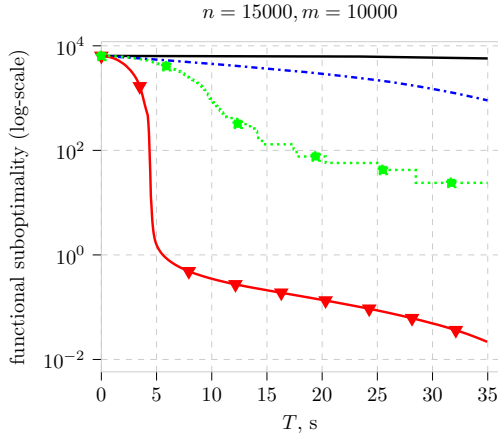


Figure 4. Softmax problem (17) with random sparse matrix.

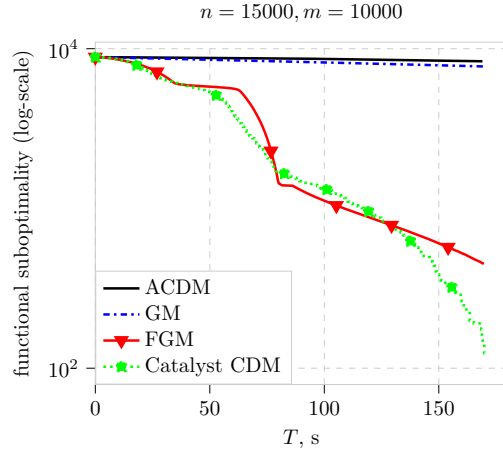


Figure 5. Softmax problem (17) with heterogeneous sparse matrix.

4.3. 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 [31], formulate such an optimization problem:

$$\min_{x,y} F(x,y) = \sum_{u,i} c_{ui} (p_{ui} - x_u^\top y_i)^2 + \lambda \left(\sum_u \|x_u\|_2^2 + \sum_i \|y_i\|_2^2 \right), \quad (18)$$

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 $\lambda (\sum_u \|x_u\|_2^2 + \sum_i \|y_i\|_2^2)$ — 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 [31]).

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:

$$\tilde{y}_{k+1} \approx \underset{y}{\operatorname{argmin}} F_{L,x}^{k+1}(y)$$

$$y_{k+1} = \operatorname{argmin} \{f(y) \mid y \in \{y_k, \tilde{y}_{k+1}\}\}$$

Used in experiments sparse matrix $\{r_{ui}\}_{u,i}$ generated from *radio* dataset¹¹ 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 6 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.

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

In Figure 7 we compare the performance of the Alternating Least Squares algorithm and its accelerated via Monteiro–Svaiter algorithm versions for problem 18 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.

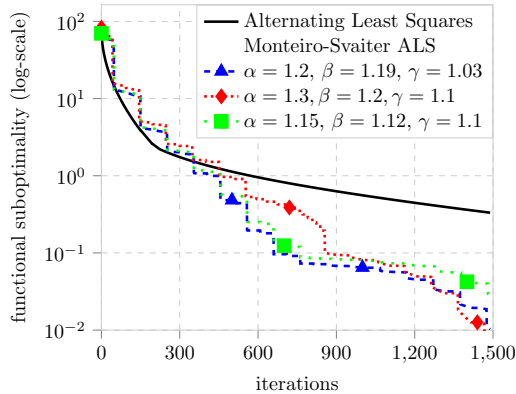


Figure 6. Matrix completion problem (18) with different (α, β, γ) .

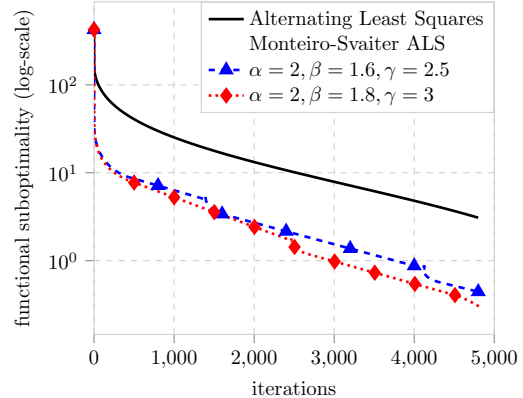


Figure 7. Matrix completion problem (18) with big matrix.

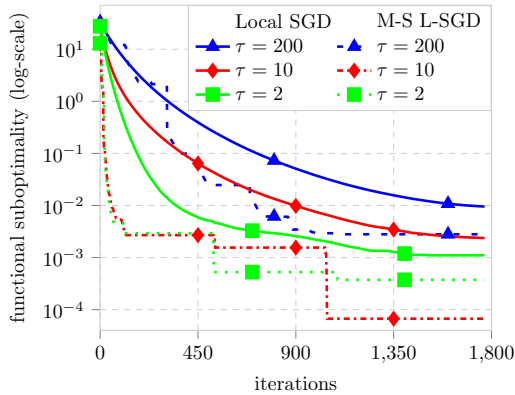


Figure 8. Regularized logistic loss (19) for different synchronization intervals τ .

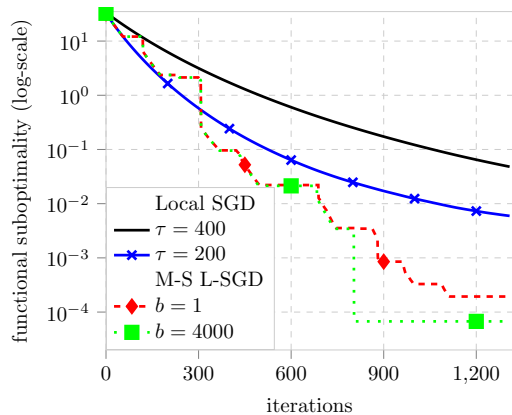


Figure 9. Logistic loss (19) minimization with minibatching.

4.4. Local SGD Acceleration

Consider the following ℓ_2 -regularized logistic loss minimization problem:

$$\min_{x \in \mathbb{R}^n} F(x) = \frac{1}{m} \sum_{j=1}^m \log(1 + \exp(-y_j x^\top p_j)) + g(x) \quad (19)$$

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 $I_D \subset [1, n]$, $I_S \cap I_D = \emptyset$ (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 categorical 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 is randomly generated from the uniform distribution $x_{0,i} \sim \mathcal{U}(0, 1) \forall i \in [1, n]$.

In Figure 8 we compare the performance of the 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.

In Figure 9 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 (19) with applying the minibatch technique (where parameter b controls the batch size). The horizontal axis measures the number

¹¹<https://www.upf.edu/web/mtg/lastfm360k>

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.

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 the non-adaptive run proposed in this paper, acceleration has in a log-factor better rate than via Catalyst. Note, that this “fight” for the log-factor in accelerated procedure’s become popular in the last time, see [39, 42] for concrete examples. In this paper, we eliminate log-factor in a rather big generality.

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. We also would like to thank anonymous reviewers for their fruitful comments.

References

- [1] Z. Allen-Zhu and E. Hazan, *Optimal black-box reductions between optimization objectives*, arXiv preprint arXiv:1603.05642 (2016).
- [2] A. Bayandina, A. Gasnikov, and A. Lagunovskaya, *Gradient-free two-points optimal method for non smooth stochastic convex optimization problem with additional small noise*, Automation and remote control 79 (2018). arXiv:1701.03821.
- [3] A. Beck, *First-order methods in optimization*, Vol. 25, SIAM, 2017.
- [4] S. Bubeck, *Convex optimization: Algorithms and complexity*, Foundations and Trends® in Machine Learning 8 (2015), pp. 231–357.
- [5] C.C. Chang and C.J. Lin, *Libsvm: a library for support vector machines*, ACM Transactions on Intelligent Systems and Technology (TIST) 2 (2011), p. 27.
- [6] E. De Klerk, F. Glineur, and A.B. Taylor, *On the worst-case complexity of the gradient method with exact line search for smooth strongly convex functions*, Optimization Letters 11 (2017), pp. 1185–1199.
- [7] J. Diakonikolas and L. Orecchia, *Alternating randomized block coordinate descent*, arXiv preprint arXiv:1805.09185 (2018).
- [8] J. Diakonikolas and L. Orecchia, *Conjugate gradients and accelerated methods unified: The approximate duality gap view*, arXiv preprint arXiv:1907.00289 (2019).
- [9] N. Doikov and Y. Nesterov, *Contracting proximal methods for smooth convex optimization*, SIAM Journal on Optimization 30 (2020), pp. 3146–3169.
- [10] N. Doikov and Y. Nesterov, *Inexact tensor methods with dynamic accuracies*, arXiv preprint arXiv:2002.09403 (2020).
- [11] J.C. Duchi, M.I. Jordan, M.J. Wainwright, and A. Wibisono, *Optimal rates for zero-order convex optimization: The power of two function evaluations*, IEEE Trans. Information Theory 61 (2015), pp. 2788–2806.

- [12] D. Dvinskikh, D. Kamzolov, A. Gasnikov, P. Dvurechensky, D. Pasechnyuk, V. Matykhin, and A. Chernov, *Accelerated meta-algorithm for convex optimization*, Computational Mathematics and Mathematical Physics 61 (2021).
- [13] D. Dvinskikh, S. Omelchenko, A. Gasnikov, and A. Tyurin, *Accelerated Gradient Sliding for Minimizing a Sum of Functions*, in *Doklady Mathematics*, Vol. 101. Springer, 2020, pp. 244–246.
- [14] P. Dvurechensky, A. Gasnikov, and E. Gorbunov, *An accelerated directional derivative method for smooth stochastic convex optimization*, arXiv:1804.02394 (2018).
- [15] P. Dvurechensky, A. Gasnikov, and E. Gorbunov, *An accelerated method for derivative-free smooth stochastic convex optimization*, arXiv:1802.09022 (2018).
- [16] P. Dvurechensky, A. Gasnikov, and A. Lagunovskaya, *Parallel algorithms and probability of large deviation for stochastic convex optimization problems*, Numerical Analysis and Applications 11 (2018), pp. 33–37.
- [17] O. Fercoq and P. Richtárik, *Accelerated, parallel, and proximal coordinate descent*, SIAM Journal on Optimization 25 (2015), pp. 1997–2023.
- [18] A. Gasnikov, *Universal gradient descent*, MCCME, Moscow, 2021.
- [19] A. Gasnikov, A. Lagunovskaya, I. Usmanova, and F. Fedorenko, *Gradient-free proximal methods with inexact oracle for convex stochastic nonsmooth optimization problems on the simplex*, Automation and Remote Control 77 (2016), pp. 2018–2034. Available at <http://dx.doi.org/10.1134/S0005117916110114>, arXiv:1412.3890.
- [20] A. Gasnikov, *Universal gradient descent*, arXiv preprint arXiv:1711.00394 (2017).
- [21] A. Gasnikov, P. Dvurechensky, E. Gorbunov, E. Vorontsova, D. Selikhanovych, C.A. Uribe, B. Jiang, H. Wang, S. Zhang, S. Bubeck, *et al.*, *Near Optimal Methods for Minimizing Convex Functions with Lipschitz p -th Derivatives*, in *Conference on Learning Theory*. 2019, pp. 1392–1393.
- [22] A. Gasnikov, P. Dvurechensky, and I. Usmanova, *On accelerated randomized methods*, Proceedings of Moscow Institute of Physics and Technology 8 (2016), pp. 67–100. In Russian, first appeared in arXiv:1508.02182.
- [23] A. Gasnikov, E. Gorbunov, D. Kovalev, A. Mokhammed, and E. Chernousova, *Reachability of optimal convergence rate estimates for high-order numerical convex optimization methods*, in *Doklady Mathematics*, Vol. 99. Springer, 2019, pp. 91–94.
- [24] N. Gazagnadou, R.M. Gower, and J. Salmon, *Optimal mini-batch and step sizes for saga*, arXiv preprint arXiv:1902.00071 (2019).
- [25] E. Gorbunov, D. Dvinskikh, and A. Gasnikov, *Optimal decentralized distributed algorithms for stochastic convex optimization*, arXiv preprint arXiv:1911.07363 (2019).
- [26] E. Gorbunov, F. Hanzely, and P. Richtarik, *A unified theory of sgd: Variance reduction, sampling, quantization and coordinate descent* (2019).
- [27] R.M. Gower, N. Loizou, X. Qian, A. Sailanbayev, E. Shulgin, and P. Richtárik, *Sgd: General analysis and improved rates*, arXiv preprint arXiv:1901.09401 (2019).
- [28] S. Guminov, P. Dvurechensky, and A. Gasnikov, *Accelerated alternating minimization*, arXiv preprint arXiv:1906.03622 (2019).
- [29] H. Hendrikx, F. Bach, and L. Massoulié, *Dual-free stochastic decentralized optimization with variance reduction*, Advances in Neural Information Processing Systems 33 (2020).
- [30] D. Hilbert, *Ein Beitrag zur Theorie des Legendre’schen Polynoms*, Acta Math. 18 (1894), pp. 155–159. Available at <https://doi.org/10.1007/BF02418278>.
- [31] Y. Hu, Y. Koren, and C. Volinsky, *Collaborative filtering for implicit feedback datasets*, in *2008 Eighth IEEE International Conference on Data Mining*. Ieee, 2008, pp. 263–272.
- [32] A. Ivanova, A. Gasnikov, P. Dvurechensky, D. Dvinskikh, A. Tyurin, E. Vorontsova, and D. Pasechnyuk, *Oracle complexity separation in convex optimization*, arXiv preprint arXiv:2002.02706 (2020).
- [33] P. Kairouz, H.B. McMahan, B. Avent, A. Bellet, M. Bennis, A.N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, *et al.*, *Advances and open problems in federated learning*, arXiv preprint arXiv:1912.04977 (2019).
- [34] D. Kamzolov, P. Dvurechensky, and A. Gasnikov, *Optimal Combination of Tensor Op-*

- timization Methods*, in *Optimization and Applications: 11th International Conference, OPTIMA 2020, Moscow, Russia, September 28–October 2, 2020, Proceedings*. Springer Nature, p. 166.
- [35] D. Kamzolov and A. Gasnikov, *Near-optimal hyperfast second-order method for convex optimization and its sliding*, arXiv preprint arXiv:2002.09050 (2020).
 - [36] H. Karimi, J. Nutini, and M. Schmidt, *Linear convergence of gradient and proximal-gradient methods under the polyak-tojasiewicz condition*, in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2016, pp. 795–811.
 - [37] S.P. Karimireddy, S. Kale, M. Mohri, S.J. Reddi, S.U. Stich, and A.T. Suresh, *Scaffold: Stochastic controlled averaging for federated learning*, arXiv preprint arXiv:1910.06378 (2019).
 - [38] A. Khaled, K. Mishchenko, and P. Richtárik, *Better communication complexity for local sgd*, arXiv preprint arXiv:1909.04746 (2019).
 - [39] D. Kovalev, A. Salim, and P. Richtárik, *Optimal and practical algorithms for smooth and strongly convex decentralized optimization*, *Advances in Neural Information Processing Systems* 33 (2020).
 - [40] A. Kulunchakov and J. Mairal, *A generic acceleration framework for stochastic composite optimization*, arXiv preprint arXiv:1906.01164 (2019).
 - [41] H. Li and Z. Lin, *Revisiting extra for smooth distributed optimization*, arXiv preprint arXiv:2002.10110 (2020).
 - [42] H. Li, Z. Lin, and Y. Fang, *Optimal accelerated variance reduced extra and digging for strongly convex and smooth decentralized optimization*, arXiv preprint arXiv:2009.04373 (2020).
 - [43] H. Lin, J. Mairal, and Z. Harchaoui, *A universal catalyst for first-order optimization*, in *Advances in neural information processing systems*. 2015, pp. 3384–3392.
 - [44] H. Lin, J. Mairal, and Z. Harchaoui, *Catalyst acceleration for first-order convex optimization: from theory to practice*, arXiv preprint arXiv:1712.05654 (2018).
 - [45] T. Lin, C. Jin, and M. Jordan, *On gradient descent ascent for nonconvex-concave minimax problems*, in *International Conference on Machine Learning*. PMLR, 2020, pp. 6083–6093.
 - [46] K. Mishchenko, F. Iutzeler, J. Mallick, and M.R. Amini, *A delay-tolerant proximal-gradient algorithm for distributed learning*, in *International Conference on Machine Learning*. 2018, pp. 3587–3595.
 - [47] 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 (2013), pp. 1092–1125.
 - [48] A.S. Nemirovsky and D.B. Yudin, *Problem Complexity and Method Efficiency in Optimization*, A Wiley-Interscience publication, Wiley, 1983.
 - [49] Y. Nesterov, *Efficiency of coordinate descent methods on huge-scale optimization problems*, *SIAM Journal on Optimization* 22 (2012), pp. 341–362.
 - [50] Y. Nesterov and J.P. Vial, *Confidence level solutions for stochastic programming*, *Automatica* 44 (2008), pp. 1559–1568.
 - [51] Y. Nesterov, *Lectures on convex optimization*, Vol. 137, Springer, 2018.
 - [52] Y. Nesterov, A. Gasnikov, S. Guminov, and P. Dvurechensky, *Primal-dual accelerated gradient descent with line search for convex and nonconvex optimization problems*, arXiv preprint arXiv:1809.05895 (2018).
 - [53] Y. Nesterov and S.U. Stich, *Efficiency of the accelerated coordinate descent method on structured optimization problems*, *SIAM Journal on Optimization* 27 (2017), pp. 110–123.
 - [54] A. Ogaltsov, D. Dvinskikh, P. Dvurechensky, A. Gasnikov, and V. Spokoiny, *Adaptive gradient descent for convex and non-convex stochastic optimization*, arXiv preprint arXiv:1911.08380 (2019).
 - [55] B. Palaniappan and F. Bach, *Stochastic variance reduction methods for saddle-point problems*, in *Advances in Neural Information Processing Systems*. 2016, pp. 1416–1424.
 - [56] C. Paquette, H. Lin, D. Drusvyatskiy, J. Mairal, and Z. Harchaoui, *Catalyst acceleration*

- for gradient-based non-convex optimization, arXiv preprint arXiv:1703.10993 (2017).
- [57] N. Parikh, S. Boyd, et al., *Proximal algorithms*, Foundations and Trends® in Optimization 1 (2014), pp. 127–239.
 - [58] D. Pasechnyuk, A. Anikin, and V. Matyukhin, *Accelerated proximal envelopes: Application to the coordinate descent method*, arXiv preprint arXiv:2101.04706 (2021).
 - [59] B.T. Polyak, *Introduction to optimization*, Optimization Software, 1987.
 - [60] R.T. Rockafellar, *Monotone operators and the proximal point algorithm*, SIAM journal on control and optimization 14 (1976), pp. 877–898.
 - [61] S. Shalev-Shwartz and T. Zhang, *Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization*, in *International conference on machine learning*. 2014, pp. 64–72.
 - [62] O. Shamir, *An optimal algorithm for bandit and zero-order convex optimization with two-point feedback*, Journal of Machine Learning Research 18 (2017), pp. 52:1–52:11.
 - [63] S.U. Stich, *Local sgd converges fast and communicates little*, arXiv preprint arXiv:1805.09767 (2018).
 - [64] N. Tupitsa, *Accelerated alternating minimization and adaptability to strong convexity*, arXiv preprint arXiv:2006.09097 (2020).
 - [65] N. Tupitsa, P. Dvurechensky, and A. Gasnikov, *Alternating minimization methods for strongly convex optimization*, arXiv preprint arXiv:1911.08987 (2019).
 - [66] A.C. Wilson, L. Mackey, and A. Wibisono, *Accelerating Rescaled Gradient Descent: Fast Optimization of Smooth Functions*, in *Advances in Neural Information Processing Systems*. 2019, pp. 13533–13543.
 - [67] B. Woodworth, K.K. Patel, S.U. Stich, Z. Dai, B. Bullins, H.B. McMahan, O. Shamir, and N. Srebro, *Is local sgd better than minibatch sgd?*, arXiv preprint arXiv:2002.07839 (2020).
 - [68] B.E. Woodworth, J. Wang, A. Smith, B. McMahan, and N. Srebro, *Graph oracle models, lower bounds, and gaps for parallel stochastic optimization*, in *Advances in neural information processing systems*. 2018, pp. 8496–8506.
 - [69] S.J. Wright, *Coordinate descent algorithms*, Mathematical Programming 151 (2015), pp. 3–34.
 - [70] J. Yang, S. Zhang, N. Kiyavash, and N. He, *A catalyst framework for minimax optimization*, Advances in Neural Information Processing Systems 33 (2020).