

# Restarting the accelerated coordinate descent method with a rough strong convexity estimate\*

Olivier Fercoq<sup>†</sup>      Zheng Qu<sup>‡</sup>

September 10, 2018

## Abstract

We propose new restarting strategies for the accelerated coordinate descent method. Our main contribution is to show that for a well chosen sequence of restarting times, the restarted method has a nearly geometric rate of convergence. A major feature of the method is that it can take profit of the local quadratic error bound of the objective function without knowing the actual value of the error bound. We also show that under the more restrictive assumption that the objective function is strongly convex, any fixed restart period leads to a geometric rate of convergence. Finally, we illustrate the properties of the algorithm on a regularized logistic regression problem and on a Lasso problem.

## 1 Introduction

### 1.1 Motivation

We consider in this paper the minimization of composite convex functions of the form

$$F(x) = f(x) + \psi(x), \quad x \in \mathbb{R}^n$$

where  $f$  is differentiable with Lipschitz gradient and  $\psi$  may be nonsmooth but is separable, and has an easily computable proximal operator. Coordinate descent methods are often considered in this context thanks to the separability of the proximal operator of  $\psi$ . These are optimization algorithms that update only one coordinate of the vector of variables at each iteration, hence using partial derivatives rather than the whole gradient.

Similarly to what he had done for the gradient method, Nesterov introduced, for smooth functions, the randomized accelerated coordinate descent method with an improved guarantee on the iteration complexity [Nes12]. Indeed, for a mild additional computational cost, accelerated methods transform the proximal coordinate descent method, for which the optimality gap  $F(x_k) - F^*$  decreases as  $O(1/k)$ , into an algorithm with “optimal”  $O(1/k^2)$  complexity [Nes83]. [LS13] introduced an efficient implementation of the method and [FR15] developed the accelerated parallel and proximal coordinate descent method (APPROX) for the minimization of composite functions.

When solving a strongly convex problem, the classical (non-accelerated) gradient and coordinate descent methods automatically have a linear rate of convergence, i.e.  $F(x_k) - F^* \in O((1 - \mu)^k)$  for a problem

---

\*The first author’s work was supported by the EPSRC Grant EP/K02325X/1 *Accelerated Coordinate Descent Methods for Big Data Optimization*, the Centre for Numerical Algorithms and Intelligent Software (funded by EPSRC grant EP/G036136/1 and the Scottish Funding Council), the Orange/Telecom ParisTech think tank Phi-TAB, the ANR grant ANR-11-LABX-0056-LMH, LabEx LMH as part of the Investissement d’avenir project. The second author’s work was supported by Hong Kong Research Grant Council 27302016. This research was partially conducted using the HKU Information Technology Services research computing facilities that are supported in part by the Hong Kong UGC Special Equipment Grant (SEG HKU09).

<sup>†</sup>Olivier Fercoq, LTCI, Télécom ParisTech, Université Paris-Saclay, Paris, France, [olivier.fercoq@telecom-paristech.fr](mailto:olivier.fercoq@telecom-paristech.fr)

<sup>‡</sup>Zheng Qu, Department of Mathematics, The University of Hong Kong, Hong Kong, China, [zhengqu@maths.hku.hk](mailto:zhengqu@maths.hku.hk)

dependent  $0 < \mu < 1$ , whereas one needs to know explicitly the strong convexity parameter in order to set accelerated gradient and accelerated coordinate descent methods to have a linear rate of convergence, see for instance [LS13, LMH15, LLX14, Nes12, Nes13]. Setting the algorithm with an incorrect parameter may result in a slower algorithm, sometimes even slower than if we had not tried to set an acceleration scheme [OC12]. This is a major drawback of the method because in general, the strong convexity parameter is difficult to estimate.

In the context of accelerated gradient method with unknown strong convexity parameter, Nesterov [Nes13] proposed a restarting scheme which adaptively approximate the strong convexity parameter. The same idea was exploited in [LX15] for sparse optimization. [Nes13] also showed that, instead of deriving a new method designed to work better for strongly convex functions, one can restart the accelerated gradient method and get a linear convergence rate. It was later shown in [LY17, FQ17] that a local quadratic error bound is sufficient to get a global linear rate of convergence.

The adaptive restart of randomized accelerated coordinate descent methods is more complex than in the deterministic case. As the complexity bound holds in expectation only, one cannot rely on this bound to estimate whether the rate of convergence is in line with our estimate of the local error bound, as was done in the deterministic case. Instead, [FQ16] proposed a fixed restarting scheme. They needed to restart at a point which is a convex combination of all previous iterates and required stronger assumptions than in the present work.

## 1.2 Contributions

In this paper, we show how restarting the accelerated coordinate descent method can help us take profit of the local quadratic error bound of the objective, when this property holds.

We consider three setups:

1. If the local quadratic error bound coefficient  $\mu$  of the objective function is known, then we show that setting a restarting period as  $O(1/\sqrt{\mu})$  yields an algorithm with optimal rate of convergence. More precisely restarted APPROX admits the same theoretical complexity bound as the accelerated coordinate descent methods for strongly convex functions developed in [LLX14], is applicable with milder assumptions and exhibits better performance in numerical experiments.
2. If the objective function is strongly convex, we show that we can restart the accelerated coordinate descent method at *any* frequency and get a linearly convergent algorithm. The rate depends on an estimate of the local quadratic error bound and we show that for a wide range of this parameter, one obtains a faster rate than without acceleration. In particular, we do not require the estimate of the error bound coefficient to be smaller than the actual value. The difference with respect to [FQ16] is that in this section, we show that there is no need to restart at a complex combination of previous iterates.
3. If the local error bound coefficient is not known, we introduce a variable restarting periods and show that up to a  $\log(\log 1/\epsilon)$  term, the algorithm is as efficient as if we had known the local error bound coefficient.

In Section 2 we recall the main convergence results for the accelerated proximal coordinate descent method (APPROX) and present restarted APPROX. In Section 3, we study restart for APPROX with a fixed restart period and in Section 4 we give the algorithm with variable restarting periods. Finally, we present numerical experiments on the lasso and logistic regression problem in Section 6.

## 2 Problems, assumptions and algorithms

In this section, we present in detail the problem we are studying. We also recall basic facts about the accelerated coordinate descent method that will be useful in the analysis.

## 2.1 Problem and assumptions

For simplicity we present the algorithm in coordinatewise form. The extension to blockwise setting follows naturally (see for instance [FR15]). We consider the following optimization problem:

$$\begin{aligned} & \text{minimize} && F(x) := f(x) + \psi(x) \\ & \text{subject to} && x = (x^1, \dots, x^n) \in \mathbb{R}^n, \end{aligned} \tag{1}$$

where  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  is a differentiable convex function and  $\psi : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$  is a closed convex and separable function:

$$\psi(x) = \sum_{i=1}^n \psi^i(x^i).$$

Note that this implies that each function  $\psi^i : \mathbb{R} \rightarrow \mathbb{R}$  is closed and convex. We denote by  $F^*$  the optimal value of (1) and assume that the optimal solution set  $\mathcal{X}^*$  is nonempty. For any positive vector  $v \in \mathbb{R}_+^n$ , we denote by  $\|\cdot\|_v$  the weighted Euclidean norm:

$$\|x\|_v^2 \stackrel{\text{def}}{=} \sum_{i=1}^n v_i (x^i)^2,$$

and  $\text{dist}_v(x, \mathcal{X}^*)$  the distance of  $x$  to the set  $\mathcal{X}^*$  with respect to the norm  $\|\cdot\|_v$ .

Throughout the paper, we will assume that the objective function satisfies the following local error bound condition.

**Assumption 1.** *For any  $x_0 \in \text{dom}(F)$ , there is  $\mu > 0$  such that*

$$F(x) \geq F^* + \frac{\mu}{2} \text{dist}_v(x, \mathcal{X}^*)^2, \quad \forall x \in [F \leq F(x_0)], \tag{2}$$

where  $[F \leq F(x_0)]$  denotes the set of all  $x$  such that  $F(x) \leq F(x_0)$ .

We denote by  $\mu(v, x_0)$  the largest  $\mu > 0$  satisfying (2) and by  $\mu(x_0)$  the value of  $\mu(\vec{e}, x_0)$  where  $\vec{e}$  is the unit vector. Note that

$$\min_i \mu(x_0)/v_i \leq \mu(v, x_0) \leq \max_i \mu(x_0)/v_i. \tag{3}$$

The fact that we assume that (2) holds for any  $x_0 \in \text{dom}(F)$  is not restrictive. The proposition below shows that, if it holds for a given  $x_0$ , it will hold on any bounded set.

**Proposition 1.** *If  $F$  is convex and satisfies the local error bound Assumption 1 then for all  $M \geq 1$ , if  $F(x') - F^* = M(F(x_0) - F^*)$ , then  $\mu(v, x') \geq \mu(v, x_0)/M$ .*

*Proof.* Suppose there exists  $\mu, v$  and  $r = F(x_0) - F^*$  such that  $\forall x \in [F - F^* \leq r]$ ,

$$F(x) \geq F^* + \frac{\mu}{2} \text{dist}_v(x, \mathcal{X}^*)^2.$$

Let  $y \in \text{dom}(F)$  such that  $r \leq F(y) - F^* \leq rM$ ,  $p$  be the orthogonal projection of  $y$  onto  $\mathcal{X}^*$  and  $x = p + \frac{r}{F(y) - F^*} (y - p)$ . As  $F$  is convex,  $F(x) \leq \frac{r}{F(y) - F^*} F(y) + (1 - \frac{r}{F(y) - F^*}) F(p)$ . Note in particular that  $F(x) - F^* \leq r$ . Rearranging, we get

$$\begin{aligned} F(y) &\geq \frac{F(y) - F^*}{r} F(x) - \left( \frac{F(y) - F^*}{r} - 1 \right) F^* \\ &\geq F^* + \frac{F(y) - F^*}{r} \frac{\mu}{2} \text{dist}_v(x, \mathcal{X}^*)^2 \\ &= F^* + \frac{r}{F(y) - F^*} \frac{\mu}{2} \text{dist}_v(y, \mathcal{X}^*)^2 \geq F^* + \frac{1}{M} \frac{\mu}{2} \text{dist}_v(y, \mathcal{X}^*)^2 \end{aligned}$$

□

Condition (2) is sometimes referred to as quadratic (functional) growth condition, as it controls the growth of the objective function value by the squared distance between any point in the sublevel set and the optimal set. This geometric property is equivalent to the Łojasiewicz gradient inequality with exponent 1/2 [BNPS17], as well as to the so-called first-order error bound condition which bounds the distance by the norm of the proximal residual [DL18]. A wide range of structured optimization models that arise in application [NNG18] satisfy Assumption 1. The constant  $\mu(x_0)$  embodies in some sense the geometrical complexity of the problem. We recall a prototype for which lower bounds for  $\mu(x_0)$  can be deduced. It is a composition of strongly convex function with linear term under polyhedral constraint:

$$\min_x F_1(x) \equiv g(Ax) + c^\top x + 1_{Cx \leq d} \quad (4)$$

where  $A$ ,  $C$ ,  $c$  and  $d$  are matrices with appropriate dimensions and  $g$  is a smooth and strongly convex function. In this case the optimal set is polyhedral and can be written as:

$$\mathcal{X}^* = \{x : Ax = t^*, c^\top x = s^*, Cx \leq d\}, \quad (5)$$

for uniquely determined  $t^*$  and  $s^*$ . It was shown in [NNG18] that

$$\mu_{F_1}(x_0) \geq \frac{\sigma_g}{\theta^2(1 + (F_1(x_0) - F_1^*)\sigma_g + 2\|\nabla g(t^*)\|^2)} \quad (6)$$

where  $\sigma_g$  is the strong convexity parameter of  $g$  and  $\theta$  is the Hoffman constant for the polyhedral optimal set  $\mathcal{X}^*$ . The dependence on  $F_1^*$  and  $\mathcal{X}^*$  in the bound (6) can be further replaced by known constants if the constraint set  $\{Cx \leq d\}$  is compact, see for example [BS17]. Hence the estimation of  $\mu_{F_1}(x_0)$  mainly relies on the computation of the Hoffman constant  $\theta$  for the linear inequality system (5). Assuming that  $A$  has full row rank and  $\{x : Ax = 0, c^\top x = 0, Cx < 0\} \neq \emptyset$ , then, we have [KT95]:

$$\theta = \min\{\|A^\top u + c^\top w + C^\top v\| : \|u\|^2 + \|v\|^2 + \|w\|^2 = 1, v \geq 0\}. \quad (7)$$

Many problems fall into the category of (4), including the  $L^1$  regularized least square problem (26) and the logistic regression problem (24) that we shall consider later for numerical illustration. Indeed the  $L^1$  regularization term can be written equivalently as a system of linear inequalities  $\{Cx \leq d\}$ , as shown in [BNPS17]. However, the size of the matrix  $C$  shall be of the same order as the number of variables  $n^1$ . Therefore, the computation of  $\theta$  and estimation of  $\mu(x_0)$  is far from trivial when the problem dimension is high and can sometimes be as hard as solving the original optimization problem. Note that under Assumption 1, a broad class of first order methods, including the proximal gradient method and its coordinatewise extensions [LT93, WL14], converge linearly without requiring a priori knowledge on the error bound constant  $\mu(x_0)$ .

## 2.2 APPROX and its properties

In the following,  $\nabla f(y_k)$  denotes the gradient of  $f$  at point  $y_k$  and  $\nabla_i f(y_k)$  denotes the partial derivative of  $f$  at point  $y_k$  with respect to the  $i$ th coordinate.  $\hat{S}$  is a random subset of  $[n] := \{1, 2, \dots, n\}$  with the property that  $\mathbf{P}(i \in \hat{S}) = \mathbf{P}(j \in \hat{S})$  for all  $i, j \in [n]$  and  $\tau = \mathbf{E}[|\hat{S}|]$ .

We recall the definition of APPROX (Accelerated Parallel PROXimal coordinate descent method) in Algorithm 1. The algorithm reduces to the accelerated proximal gradient (APG) method [Tse08] when  $\hat{S} = [n]$  with probability one. It employs a positive parameter vector  $v \in \mathbb{R}^n$ . To guarantee the convergence of the algorithm, the positive vector  $v$  should satisfy the so-called expected separable overapproximation (ESO) assumption, developed in [FR13, RT16] for the study of parallel coordinate descent methods.

**Assumption 2 (ESO).** We write  $(f, \hat{S}) \sim \text{ESO}(v)$  if

$$\mathbf{E} \left[ f(x + h_{[\hat{S}]}) \right] \leq f(x) + \frac{\tau}{n} \left( \langle \nabla f(x), h \rangle + \frac{1}{2} \|h\|_v^2 \right), \quad x, h \in \mathbb{R}^n. \quad (8)$$

---

<sup>1</sup>In [BNPS17], the  $L^1$  norm is rewritten by a matrix of  $2^n$  rows.

where for  $h = (h^1, \dots, h^n) \in \mathbb{R}^n$  and  $S \subset [n]$ ,  $h_{[S]}$  is defined as:

$$h_{[S]} \stackrel{\text{def}}{=} \sum_{i \in S} h^i e_i,$$

with  $e_i$  being the  $i$ th standard basis vectors in  $\mathbb{R}^n$ .

We require that the positive vector  $v$  used in APPROX satisfy (8) with respect to the sampling  $\hat{S}$  used. When in each step we update only one coordinate, we have  $\tau = 1$  and (8) reduces to:

$$\frac{1}{n} \sum_{i=1}^n f(x + h^i e_i) \leq f(x) + \frac{1}{n} \left( \langle \nabla f(x), h \rangle + \frac{1}{2} \|h\|_v^2 \right), \quad x, h \in \mathbb{R}^n. \quad (9)$$

It is easy to see that in this case the vector  $v$  corresponds to the coordinate-wise Lipschitz constants of  $\nabla f$ , see e.g. [Nes12]. Explicit formulas for computing admissible  $v$  with respect to more general sampling  $\hat{S}$  can be found in [RT16, FR13, QR16].

---

**Algorithm 1** APPROX( $f, \psi, x_0, K$ ) [FR15]

---

Set  $\theta_0 = \frac{\tau}{n}$  and  $z_0 = x_0$ .

**for**  $k \in \{0, \dots, K-1\}$  **do**

$$y_k = (1 - \theta_k)x_k + \theta_k z_k$$

Randomly generate  $S_k \sim \hat{S}$

**for**  $i \in S_k$  **do**

$$z_{k+1}^i = \arg \min_{z \in \mathbb{R}} \left\{ \langle \nabla_i f(y_k), z - y_k^i \rangle + \frac{\theta_k n v_i}{2\tau} |z - z_k^i|^2 + \psi^i(z) \right\}$$

**end for**

$$x_{k+1} = y_k + \frac{n}{\tau} \theta_k (z_{k+1} - z_k)$$

$$\theta_{k+1} = \frac{\sqrt{\theta_k^4 + 4\theta_k^2} - \theta_k^2}{2}$$

**end for**

**return**  $x_K$

---

The rest of this section recalls some basic results about APPROX that we shall need.

**Lemma 1.** *The sequence  $(\theta_k)$  defined by  $\theta_0 \leq 1$  and  $\theta_{k+1} = \frac{\sqrt{\theta_k^4 + 4\theta_k^2} - \theta_k^2}{2}$  satisfies*

$$\frac{(2 - \theta_0)}{k + (2 - \theta_0)/\theta_0} \leq \theta_k \leq \frac{2}{k + 2/\theta_0} \quad (10)$$

$$\frac{1 - \theta_{k+1}}{\theta_{k+1}^2} = \frac{1}{\theta_k^2}, \quad \forall k = 0, 1, \dots \quad (11)$$

Part of the results are proved in [FR15]. We give the complete proof in the appendix.

**Proposition 2** ([FR15]). *The iterates of APPROX (Algorithm 1) satisfy for all  $k \geq 1$  and any  $x_* \in \mathcal{X}^*$ ,*

$$\frac{1}{\theta_{k-1}^2} \mathbf{E}[F(x_k) - F^*] + \frac{1}{2\theta_0^2} \mathbf{E}[\|z_k - x_*\|_v^2] \leq \frac{1 - \theta_0}{\theta_0^2} (F(x_0) - F^*) + \frac{1}{2\theta_0^2} \|x_0 - x_*\|_v^2$$

### 2.3 Restarted APPROX

Under Assumption 1, restarted APG [Tse08] or FISTA [BT09] enjoys linear convergence [FQ17] and can have improved complexity bound than proximal gradient method with appropriate restart periods [OC12]. Our goal is to design restarted APPROX with similar properties based on the results in the previous section. Having defined a set of integers  $\{K_0, K_1, \dots\}$ , at which frequencies one wishes to restart the method, we can write the restarted APPROX as in Algorithm 2.

---

**Algorithm 2** APPROX with restart

---

Choose  $x_0 \in \text{dom } \psi$  and set  $\bar{x}_0 = x_0$ .  
Choose restart periods  $\{K_0, \dots, K_r, \dots\} \subseteq \mathbb{N}$ .  
**for**  $r \geq 0$  **do**  
     $\bar{x}_{r+1} = \text{APPROX}(f, \psi, \bar{x}_r, K_r)$   
**end for**

---

In order to take profit of the local error bound, the proofs for restarted FISTA and APG used the non-blowout property guaranteeing that the function value and distance to the optimal points never go above their initial value [FQ17]. For APPROX the complexity result holds in expectation only. This has two consequences:

- Even if the initial point is in a set where the local error bound holds, we are not guaranteed that the next iterates will remain to the set. To overcome this issue, at the time of restart, we will check whether the function value has increased and if this is unfortunately the case, we will instead restart at the initial point.
- Designing an adaptive restart scheme is much more complex than in the deterministic case studied in [FQ17, Nes13]. In the deterministic case, one can compare the actual progress of the method with a theoretical bound and, if the actual progress violates the theoretical bound, one has a certificate that the estimate of the local error bound was not correct. In the random case, this does not hold any more: one can only know that either the local error bound was not correct or we have fallen into a small probability event. Instead of looking for certificates, we introduce in Section 4 a sequence of variable restart periods that allows us to try several restart periods at an expense that we control.

To partially overcome the above mentioned difficulties, it will be convenient to force a decrease in function value when a restart takes place. This leads to Algorithm 3. Note that we only check function values at the time of each restart: thus this means negligible overhead.

---

**Algorithm 3** APPROX with restart and guaranteed function value decrease

---

Choose  $x_0 \in \text{dom } \psi$  and set  $\tilde{x}_0 = x_0$ .  
Choose restart periods  $\{K_0, \dots, K_r, \dots\} \subseteq \mathbb{N}$ .  
**for**  $r \geq 0$  **do**  
     $\bar{x}_{r+1} = \text{APPROX}(f, \psi, \tilde{x}_r, K_r)$   
     $\tilde{x}_{r+1} \leftarrow \bar{x}_{r+1} \mathbb{1}_{F(\bar{x}_{r+1}) \leq F(\tilde{x}_r)} + \tilde{x}_r \mathbb{1}_{F(\bar{x}_{r+1}) > F(\tilde{x}_r)}$   
**end for**

---

### 3 Linear convergence with constant restart periods

In this section we consider constant restart periods, i.e.  $K_i = K$  for all  $i \in \mathbb{N}$ . We present two types of convergence result for restated APPROX. The first one asserts that linear convergence (in expectation) can be obtained for restarted APPROX with long enough restart period  $K$ , similar to the classical results about restarted APG or FISTA [Nes13, OC12]. The second one claims that linear convergence (in expectation) is guaranteed for arbitrary restart period, if strong convexity condition holds. The basic tool upon which we build our analysis is a contraction property.

#### 3.1 Contraction for long enough periods

The first result is an extension of the “optimal fixed restart” of [Nes13, OC12] to APPROX.

**Proposition 3** (Conditional restarting at  $x_k$ ). *Let  $(x_k, z_k)$  be the iterates of APPROX applied to (1). Denote:*

$$\tilde{x} = x_k \mathbf{1}_{F(x_k) \leq F(x_0)} + x_0 \mathbf{1}_{F(x_k) > F(x_0)}.$$

*We have*

$$\mathbf{E}[F(\tilde{x}) - F^*] \leq \theta_{k-1}^2 \left( \frac{1 - \theta_0}{\theta_0^2} + \frac{1}{\theta_0^2 \mu(v, x_0)} \right) (F(x_0) - F^*). \quad (12)$$

*Moreover, given  $\alpha < 1$ , if*

$$k \geq \frac{2}{\theta_0} \left( \sqrt{\frac{1 + \mu(v, x_0)}{\alpha \mu(v, x_0)}} - 1 \right) + 1, \quad (13)$$

*then  $\mathbf{E}[F(\tilde{x}) - F^*] \leq \alpha(F(x_0) - F^*)$ .*

*Proof.* By Proposition 2, the following holds for the iterates of APPROX:

$$\begin{aligned} \mathbf{E}[F(x_k) - F^*] &\leq \theta_{k-1}^2 \left( \frac{1 - \theta_0}{\theta_0^2} (F(x_0) - F^*) + \frac{1}{2\theta_0^2} \text{dist}_v(x_0, \mathcal{X}^*)^2 \right) \\ &\stackrel{(2)}{\leq} \theta_{k-1}^2 \left( \frac{1 - \theta_0}{\theta_0^2} + \frac{1}{\mu(v, x_0) \theta_0^2} \right) (F(x_0) - F^*). \end{aligned}$$

As with probability one,

$$F(\tilde{x}) - F^* \leq F(x_k) - F^*,$$

we obtain (12). Condition (13) is equivalent to:

$$\frac{4}{(k - 1 + 2/\theta_0)^2} \left( \frac{1}{\theta_0^2} + \frac{1}{\mu(v, x_0) \theta_0^2} \right) \leq \alpha,$$

and we have the contraction using (10).  $\square$

**Remark 1.** Notice that the condition (13) requires to know a lower bound on the error bound constant  $\mu(v, x_0)$ .

**Corollary 1.** *Denote  $K(\alpha) = \left\lceil \frac{2}{\theta_0} \left( \sqrt{\frac{1 + \mu(v, x_0)}{\alpha \mu(v, x_0)}} - 1 \right) + 1 \right\rceil$ . If the restart periods  $\{K_0, \dots, K_r, \dots\}$  are all equal to  $K(\alpha)$ , then the iterates of Algorithm 3 satisfy*

$$\mathbf{E}[F(\tilde{x}_r) - F^*] \leq \alpha^r (F(x_0) - F^*).$$

*Proof.* By the definition of  $\tilde{x}_r$ , we know that for all  $r$ ,  $F(\tilde{x}_r) \leq F(x_0)$ . Hence, for all  $r$ ,  $\mu(v, \tilde{x}_r) \geq \mu(v, x_0)$ . We can thus apply Proposition 3 recursively and obtain the linear convergence.  $\square$

The iterate  $\tilde{x}_r$  is obtained after running  $r$  times APPROX (Algorithm 1) of  $K(\alpha)$  iterations. Said otherwise we have a linear rate equal to

$$\alpha^{1/K(\alpha)} \approx \alpha^{\sqrt{\alpha} \frac{\theta_0}{2} \sqrt{\mu(v, x_0)/(1 + \mu(v, x_0))}}, \quad (14)$$

which suggests choosing  $\alpha = \exp(-2)$ , or equivalently a fixed restart period

$$K^* = \left\lceil \frac{2e}{\theta_0} \left( \sqrt{\frac{1 + \mu(v, x_0)}{\mu(v, x_0)}} - 1 \right) + 1 \right\rceil. \quad (15)$$

Then, to obtain  $\mathbf{E}[F(\tilde{x}_r) - F^*] \leq \epsilon$ , the total number of iterations is bounded by

$$N^* := \ln \left( \frac{F(x_0) - F^*}{\epsilon} \right) K^*. \quad (16)$$

**Remark 2.** We compare the two extreme cases of the complexity bound (16) when  $\tau = 1$  and  $\tau = n$ . In view of (3) and (9), we have,

$$O\left(\sqrt{\frac{L}{\mu(x_0)}} \ln(1/\epsilon)\right) \text{ for } \tau = n \text{ V.S. } O\left(n\sqrt{\frac{\max_i L_i}{\mu(x_0)}} \ln(1/\epsilon)\right) \text{ for } \tau = 1,$$

where  $L$  is the Lipschitz constant of  $\nabla f$  and  $L_i$  is the Lipschitz constant of  $\nabla_i f$  with respect to the  $i$ th coordinate. Note that  $\tau$  is the number of coordinates to update at each iteration. Hence for serial computation, choosing  $\tau = 1$  can be more advantageous than restarted APG or FISTA since we always have  $\max_i L_i \leq L$ .

**Remark 3.** It is unknown whether Algorithm 2 admits the same convergence bound as Algorithm 3. Indeed, without forcing decrease of the objective value, we can not guarantee  $\mathbf{E}[F(\bar{x}_{r+1}) - F^*] \leq \alpha \mathbf{E}[F(\bar{x}_r) - F^*]$  as  $\bar{x}_r$  is not necessarily in the sublevel set  $[F \leq F(x_0)]$ . Clearly, when either a global error bound condition holds or  $\hat{S} = [n]$  with probability one, there is no need to ensure the decrease of  $F$ . The latter two cases were respectively considered in our two previous papers [FQ16] and [FQ17].

### 3.2 Contraction for any period under strong convexity condition

In this subsection, we assume that the function  $F$  is  $\mu$ -strongly convex such that  $\mathcal{X}^*$  contains a unique element  $x^*$  and

$$F(x) \geq F^* + \frac{\mu}{2} \|x - x^*\|_v^2, \quad \forall x \in [F < +\infty]. \quad (17)$$

In [FQ16], we showed that under the condition (17), linear convergence is guaranteed for any restart period by restarting at a particular convex combination of all past iterates. Here we show that the same conclusion holds by simply restarting at the last iterate.

**Theorem 1.** Assume that  $F$  is  $\mu$ -strongly convex. Denote

$$\Delta(x) = \frac{1 - \theta_0}{\theta_0^2} (F(x) - F^*) + \frac{1}{2\theta_0^2} \text{dist}_v(x, \mathcal{X}^*)^2.$$

Then the iterates of APPROX satisfy

$$\mathbf{E}[\Delta(x_k)] \leq \frac{1 + (1 - \theta_0)\mu}{1 + \frac{\theta_0^2}{2\theta_{k-1}^2}\mu} \Delta(x_0), \quad \forall k \geq 1. \quad (18)$$

*Proof.* The proof is organised in 4 steps. Firstly, we derive a one-iteration inequality, secondly, we identify conditions under which this inequality is a supermartingale inequality, thirdly, we give a solution to the set of inequalities and finally, we bound the rate we obtain by a simpler formula.

Because of its length, we postpone the proof to the appendix.  $\square$

**Remark 4.** The deterministic special case of (18) when  $\hat{S} = [n]$  with probability one, was proved in [FQ17], under the local error bound condition. The more general case with random  $\hat{S}$  turns out to be more complex as the one iteration inequality involves both  $\|x_k - x^*\|_v$  and  $\|z_k - x^*\|_v$  and uniqueness of  $\mathcal{X}^*$  is required.

When strong convexity condition holds, we do not force decrease of the function value after a restart. Moreover, unlike Corollary 1, we could show a linear rate of convergence for any restart period  $K \geq n/\tau$ .

**Corollary 2.** Under condition (17), if the restart periods  $\{K_0, \dots, K_r, \dots\}$  are all equal to  $K \geq n/\tau$ , then the iterates of Algorithm 2 satisfy

$$\mathbf{E}[\Delta(\bar{x}_r)] \leq \left(\frac{1 + (1 - \theta_0)\mu}{1 + \frac{\theta_0^2}{2\theta_{K-1}^2}\mu}\right)^r \Delta(x_0).$$

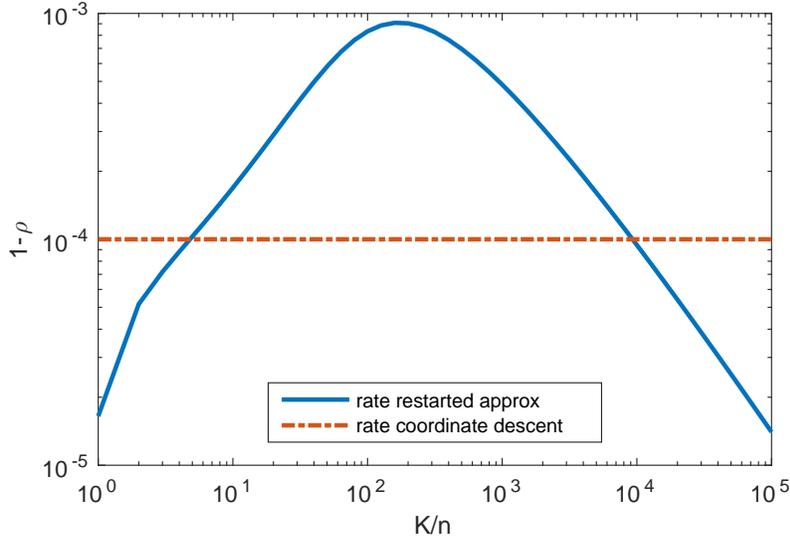


Figure 1: Comparison of the worst case rate of convergence  $\rho$  of restarted APPROX and coordinate descent for a  $\mu$ -strongly convex objective with  $\mu = 10^{-3}$ ,  $n = 10$ ,  $\tau = 1$  and various choices of the restart period  $K$ . The larger  $1 - \rho$  is, the faster we expect the algorithm to be. We can see that if  $K \in [5n, 9.10^3n]$ , then restarted APPROX has a better convergence rate than coordinate descent.

*Proof.* This is a direct application of Theorem 1. The bound (18) is a strict contraction when  $2\theta_{k-1}^2 < \theta_0^2/(1 - \theta_0)$ , which holds when  $k \geq 1/\theta_0 = n/\tau$  by (10).  $\square$

We here obtain a linear rate:

$$\left( \frac{1 + (1 - \theta_0)\mu}{1 + \frac{\theta_0^2}{2\theta_{K-1}^2}\mu} \right)^{\frac{1}{K}}, \quad (19)$$

which is slightly worse than that suggested by (14) for large  $K$  but implies the same order of complexity bound as (16) if  $K = \Theta(K^*)$ .

In Figure 1, we show in a numerical application that the convergence rate (19) of restarted APPROX is smaller than the convergence rate of coordinate descent for a wide range of restart periods  $K$ .

## 4 Variable restart

In this section, we are going to show that by choosing properly the sequence of restart times, we can ensure a nearly linear rate, even if we know in advance nothing about the local quadratic error bound. For this, we consider a sequence of restart periods  $\{K_0, K_1, \dots\}$  satisfying the following assumption:

**Assumption 3.**

1.  $K_0 \in \mathbb{N} \setminus \{0\}$ ;
2.  $K_{2^j-1} = 2^j K_0$  for all  $j \in \mathbb{N}$ ;
3.  $|\{0 \leq r < 2^J - 1 \mid K_r = 2^j K_0\}| = 2^{J-1-j}$  for all  $j \in \{0, 1, \dots, J-1\}$  and  $J \in \mathbb{N}$ .

For instance, we may take:

$$\begin{aligned}
K_0 &= K_0 & K_1 &= 2^1 K_0 & K_2 &= K_0 & K_3 &= 2^2 K_0 \\
K_4 &= K_0 & K_5 &= 2^1 K_0 & K_6 &= K_0 & K_7 &= 2^3 K_0 \\
K_8 &= K_0 & K_9 &= 2^1 K_0 & K_{10} &= K_0 & K_{11} &= 2^2 K_0 \\
K_{12} &= K_0 & K_{13} &= 2^1 K_0 & K_{14} &= K_0 & K_{15} &= 2^4 K_0
\end{aligned}$$

**Theorem 2.** *Consider Algorithm 3 with restart periods satisfying Assumption 3. Then*

$$\mathbf{E}[F(\tilde{x}_{2^J-1}) - F^*] \leq \epsilon, \quad (20)$$

where  $J = \lceil \max(\log_2(K^*/K_0), 0) \rceil + \lceil \log_2(\ln(\delta_0/\epsilon)/2) \rceil$ ,  $\delta_0 = F(x_0) - F^*$  and  $K^*$  is defined as in (15). To obtain (20), the total number of APPROX iterations  $K_0 + \dots + K_{2^J-1}$  is bounded by

$$(\lceil \max(\log_2(K^*/K_0), 0) \rceil + \lceil \log_2(\ln(\delta_0/\epsilon)) \rceil + 1) \ln(\delta_0/\epsilon) \max(K^*, K_0). \quad (21)$$

*Proof.* Define

$$i = \lceil \max(\log_2(K^*/K_0), 0) \rceil.$$

Denote  $\delta_r = \mathbf{E}[F(\tilde{x}_r) - F^*]$  and

$$c_i(J) := |\{l < 2^J - 1 \mid K_l \geq 2^i K_0\}| + 1 = 1 + \sum_{k=i}^{J-1} 2^{J-1-k} = 2^{J-i}.$$

$c_i(J)$  is the number of restarts such that  $K_l \geq K^*$ , i.e. the number of restarts for which Proposition 3 applies (the “+1” comes from the restart number  $2^J - 1$ ). Then it follows from Proposition 3 that

$$\delta_{2^J-1} \leq e^{-2c_i(J)} \delta_0 = e^{-(2^{J+1-i})} \delta_0.$$

Since

$$J = \lceil \max(\log_2(K^*/K_0), 0) \rceil + \lceil \log_2(\ln(\delta_0/\epsilon)) \rceil - 1 \geq i - 1 + \log_2(\ln(\delta_0/\epsilon)),$$

we deduce that  $\delta_{2^J-1} \leq \epsilon$ . This proves the first assertion.

By Assumption 3,

$$\begin{aligned}
K_0 + \dots + K_{2^J-1} &= \sum_{j=0}^{J-1} |\{0 \leq r < 2^j - 1 \mid K_r = 2^j K_0\}| 2^j K_0 + K_{2^J-1} \\
&= \sum_{i=0}^{J-1} 2^{J-1-i} 2^i K_0 + 2^J K_0 = (J+2) 2^{J-1} K_0.
\end{aligned}$$

Since

$$2^{J-1} \leq 2^{\max(\log_2(K^*/K_0), 0) + \log_2(\ln(\delta_0/\epsilon))} = \max(K^*/K_0, 1) \ln(\delta_0/\epsilon),$$

we obtain the bound defined as in (21).  $\square$

**Remark 5.** *If we have an estimate of  $\log(\delta_0/\epsilon)$ , then we may modify the sequence of restarts in order to stop considering restarts with  $K_0$  iterations after  $\log(\delta_0/\epsilon)$  restarts. Indeed, if  $K_0$  were greater than  $K^*$  then the algorithm should already have terminated. Similarly, one can stop considering restarts with  $2K_0$  iterations after  $2 \log(\delta_0/\epsilon)$  restarts,  $2^j K_0$  after  $(j+1) \log_2(\delta_0/\epsilon)$  restarts.*

A priori knowledge	Theorem	Assumption	Complexity bound
$\mu(v, x_0)$	Corollary 1	Assumption 1	$N^*$
$\mu(v, x_0) \geq \underline{\mu}$	Proposition 3	Assumption 1	$O\left(N^* \sqrt{\frac{\mu(v, x_0)}{\underline{\mu}}}\right)$
$\mu_0 > 0$	Theorem 1	(17)	$O\left(N^* \left(\sqrt{\frac{\underline{\mu}}{\mu_0}} + \sqrt{\frac{\mu_0}{\underline{\mu}}}\right)\right)$
$\mu(v, x_0) \leq \bar{\mu}$	Theorem 2	Assumption 1+ 3	$O\left(N^* \log_2 \left(\ln \left(\frac{F(x_0) - F^*}{\epsilon}\right) \sqrt{\frac{\bar{\mu}}{\mu(v, x_0)}}\right)\right)$

Table 1: Complexity bound under different assumptions and a priori knowledge of the error bound constant. Here  $N^* = O\left(\frac{\ln(1/\epsilon)}{\theta_0 \sqrt{\mu(v, x_0)}}\right)$  is defined in (16).

We summarize our theoretical findings in Table 1 in four different regimes: when we know a lower bound  $\underline{\mu}$ , an upper bound  $\bar{\mu}$ , exactly the value of or nothing about  $\mu(v, x_0)$ . We recall from (6) and (7) that a non-trivial lower bound  $\underline{\mu}$  can be much harder to be obtained than an upper bound  $\bar{\mu}$ . In addition, the complexity bound based on an upper bound  $\bar{\mu}$  differs from the optimal scheme by a logarithm term. For comparison purpose, we recall the log-scale grid search schedule proposed in [RD17] for restart of APG or FISTA. Fixing an integer  $N > 0$ , the restart periods proposed in [RD17] can be described as follows:

$$\begin{aligned}
K_0 &= \dots = K_{\lceil N/2 \rceil} = 2 \\
K_{\lceil N/2 \rceil + 1} &= \dots = K_{\lceil N/2 \rceil + \lceil N/2^2 \rceil} = 2^2 \\
&\vdots \\
K_{\lceil N/2 \rceil + \dots + \lceil N/2^{i-1} \rceil + 1} &= \dots = K_{\lceil N/2 \rceil + \dots + \lceil N/2^i \rceil} = 2^i
\end{aligned}$$

To ensure a nearly linear time convergence with logarithm difference, the inner number of iterations  $N$  is required to be larger than  $2e\sqrt{L/\mu(x_0)}$ . Therefore a lower bound  $\underline{\mu} \leq \mu(x_0)$  is needed.

## 5 Extension to other randomized accelerated methods

Proposition 3 and Theorem 2 only rely on the fact that APPROX guarantees

$$\mathbf{E}[F(x_k) - F^*] \leq \frac{1}{(k+a)^2} \left( C_F(F(x_0) - F^*) + \frac{C_d}{2} \text{dist}(x_0, \mathcal{X}^*)^2 \right) \quad (22)$$

A couple of other algorithms have such convergence guarantees. Instead of assuming that  $\psi$  is separable, they assume that  $f$  is a sum of functions  $f(x) = \frac{1}{2n_f} \sum_{j=1}^{n_f} f_j(x)$ :

- Katyusha<sup>ns</sup> [AZ17] (an accelerated stochastic variance reduced gradient method) satisfies for its output vector  $\tilde{x}^S$

$$\mathbf{E}[F(\tilde{x}^S) - F^*] \leq \frac{8}{(S+3)^2} \left( 2(F(x_0) - F^*) + \frac{3L}{2m} \text{dist}(x_0, \mathcal{X}^*)^2 \right)$$

where  $m = 2n_f$  is the number of SVRG steps between each momentum step,  $n_f$  is the number of summands in the definition of  $f$  and the other symbols follow the notation of this paper.

- DASVRDA<sup>ns</sup> [MS17] (another accelerated stochastic variance reduced gradient method) has a similar

guarantee

$$\mathbf{E}[F(\tilde{x}^S) - F^*] \leq \frac{4}{(S+2)^2} \left( (F(x_0) - F^*) + \frac{2(1+\gamma(m+1))L}{(1-\gamma^{-1})m(m+1)} \text{dist}(x_0, \mathcal{X}^*)^2 \right)$$

where  $\gamma \geq 3$  is a parameter of the method.

From (22) we obtain directly:

$$\mathbf{E}[F(x_k) - F^*] \leq \frac{1}{(k+a)^2} \left( C_F + \frac{C_d}{\mu(x_0)} \right) (F(x_0) - F^*).$$

which is a strict contraction if

$$k \geq K^* := \left\lceil \frac{1}{\epsilon} \sqrt{C_F + \frac{C_d}{\mu(x_0)}} - a \right\rceil. \quad (23)$$

Then the same conclusion as Corollary 1 and Theorem 2 can be obtained.

**Theorem 3.** *Let  $\mathcal{A}$  be an algorithm satisfying (22). Consider restarted  $\mathcal{A}$  in the fashion of Algorithm 3. Then, to obtain*

$$\mathbf{E}[F(\tilde{x}_r) - F^*] \leq \epsilon,$$

the number of inner iterations is bounded by

$$\ln \left( \frac{F(x_0) - F^*}{\epsilon} \right) K^*,$$

if the restart periods are all equal to  $K^*$  defined in (23), and by

$$(\lceil \max(\log_2(K^*/K_0), 0) \rceil + \lceil \log_2(\ln(\delta_0/\epsilon)) \rceil + 1) \ln(\delta_0/\epsilon) \max(K^*, K_0)$$

if the restart periods satisfy Assumption 3.

*Proof.* The proof arguments are the same as in the coordinate descent case.  $\square$

**Remark 6.** *The proof of Theorem 1 requires to go deeper into the specificities of accelerated coordinate descent. Hence, extending restart at any fixed frequency to any randomized accelerated method is not trivial, and beyond the scope of this paper.*

## 6 Numerical experiments

### 6.1 Logistic regression

We solve the following logistic regression problem:

$$\min_{x \in \mathbb{R}^N} \frac{\lambda_1}{\|A^\top b\|_\infty} \sum_{j=1}^m \log(1 + \exp(b_j a_j^\top x)) + \|x\|_1 \quad (24)$$

We consider

$$f(x) = \frac{\lambda_1}{\|A^\top b\|_\infty} \sum_{j=1}^m \log(1 + \exp(b_j a_j^\top x)),$$

and

$$\psi(x) = \|x\|_1.$$

In particular, for serial sampling ( $\tau = 1$ ), (8) is satisfied for

$$v_i = \frac{\lambda_1}{4\|A^\top b\|_\infty} \sum_{i=1}^m (b_j A_{ij})^2, \quad i = 1, \dots, n. \quad (25)$$

Even if the logistic loss is not strongly convex, we expect that the local curvature around the optimum is nonzero and so, restarting APPROX may be useful.

We run our numerical comparison on the dataset RCV1 [CL11] with regularization parameter  $\lambda_1 = 10^4$ . We compare the following algorithms.

- Randomized coordinate descent [RT14].
- APPROX [FR15].
- APCG [LLX14]: we run APCG using three different settings for the parameter  $\mu$  in the algorithm:  $10^{-3}$ ,  $10^{-5}$  and  $10^{-7}$ . We stop the program when the duality gap is lower than  $10^{-8}$  or the running time is larger than 6,000s. The results are reported in Figure 2.
- Prox-Newton: we modified the implementation of [JG15] in order to deactivate feature selection, which is not the topic of the present paper, and we increased the maximum number of inner iterations to be able to obtain high accuracy solutions. Each prox-Newton step is solved approximately using coordinate descent.
- APPROX-restart (Algorithm 3) with fixed frequency set as if we knew the error bound constant. As for APCG, we tried values equal to  $10^{-3}$ ,  $10^{-5}$  and  $10^{-7}$ .
- APPROX-restart (Algorithm 3) with the restart sequence given in Section 4 and  $K_0 = \lceil 20e/\theta_0 \rceil$ . Most problems encountered in practice have a conditioning larger than 100, hence this choice leads to  $K_0 \leq K^*$  in most cases.

Note that the convergence of APCG is only proved for  $\mu$  smaller than the strong convexity coefficient in [LLX14]. In our experiments, we observed numerical issues when running APCG for several cases when taking larger  $\mu$  (we were not able to compute the  $i$ th partial derivative at  $y_k = \rho_k w_k + z_k$  because  $\rho_k$  had reached the double precision float limit). Such a case can be identified in the plot when the line corresponding to APCG stops abruptly before the time limit (6000s) with a precision worse than  $10^{-8}$ .

Fixed restart can give the best performance among all tested algorithms but the setting of the restart period has a large influence on the performance. Note that the objective function is not strongly convex, so the linear convergence is guaranteed only if the restart period is large enough.

On the other hand, variable restart APPROX is nearly as fast as the best among the fixed restart, although  $K_0$  was set with a clearly under optimal value. Moreover, it has a clear theoretical guarantee for any initial restarting period  $K_0$ .

## 6.2 Lasso path

We then present experiments on the  $L^1$ -regularised least squares problem (Lasso)

$$\min_{x \in \mathbb{R}^N} \frac{1}{2} \|Ax - b\|_2^2 + \lambda \|x\|_1. \quad (26)$$

We consider solving a set of such problems for  $\lambda \in \{\lambda_0, \lambda_1, \dots, \lambda_T\}$ , where  $T = 10$ ,  $\lambda_0 = \|A^\top b\|_\infty$ ,  $\lambda_t = \lambda_0 \alpha^t$  and  $\alpha^T = 10^{-3}$ . This procedure is called pathwise optimization [FHH<sup>+</sup>07] and is often considered for statistical problems with hyper-parameters.

We selected 6 data sets from the LibSVM dataset repository [CL11]. We centered and normalized the columns.

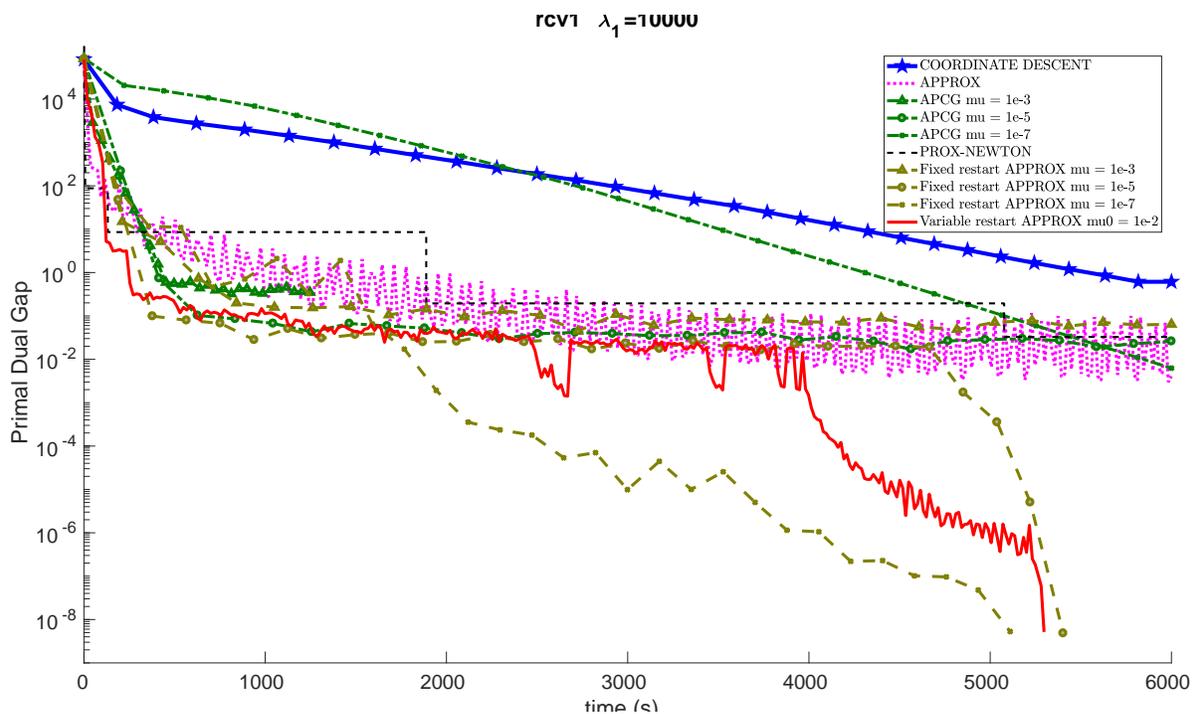


Figure 2: Comparison of (accelerated) coordinate descent algorithms for the logistic regression problem on the dataset RCV1: coordinate descent, APPROX, APCG with  $\mu \in \{10^{-3}, 10^{-5}, 10^{-7}\}$ , Prox-Newton, fixed frequency restarted APPROX with  $\mu \in \{10^{-3}, 10^{-5}, 10^{-7}\}$  and APPROX with variable restart initiated with  $K_0 = \lceil 20e/\theta_0 \rceil$ . APCG failed for  $\mu = 10^{-3}$ .

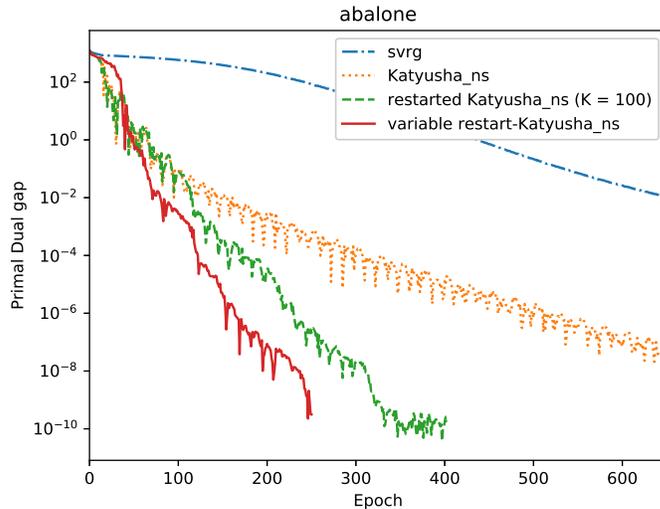


Figure 3: Comparison of vanilla, accelerated and restarted versions of SVRG. An epoch corresponds to  $m$  SVRG steps.

We did not run APCG on this problem because, as shown on the logistic regression experiment, the strong convexity estimate has a dramatic consequence on the behaviour of the algorithm and setting it to a too high value may lead to major numerical issues.

We chose to restart APPROX with variable restart set as follows. In the beginning, we start with  $10n$  iterations of non-accelerated coordinate descent. After that, we run variable restart APPROX with  $K_0 = 10n$  and we double  $K_0$  after each  $\log_2(1/\epsilon)$  iterations. When the duality gap at  $\lambda_t$  reaches  $\epsilon$ , we switch to  $\lambda_{t+1}$ . We perform a warm start on the optimization variable but we also set  $K_0$  to the last value it had when solving the problem at  $\lambda_t$ .

We then compare coordinate descent [RT14], APPROX [FR15] and restarted APPROX (Algorithm 3) on the 6 pathwise optimization lasso problems.

APPROX is sometimes getting into trouble when high accuracy is requested. This does not happen with variable restart APPROX. In all the experiments, variable restart APPROX is at most twice as slow as coordinate descent. On the other hand, variable restart APPROX is much faster on problems requiring more computational resources.

### 6.3 Lasso with SVRG

Our last experiment illustrate the restart of accelerated stochastic variance reduced gradient on a Lasso problem (26). We took the `abalone` dataset and  $\lambda = \|A^\top b\|_\infty/1000$  and solved the problem using SVRG [AZY16], Katyusha<sup>ns</sup> [AZ17] and restarted variants of Katyusha<sup>ns</sup> described in Section 5. As shown on Figure 3, restarted accelerated SVRG is able to solve the problem faster than previously proposed methods.

## A Proof of Lemma 1 and Theorem 1

*Proof of Lemma 1.* The equation (11) holds because  $\theta_{k+1}$  is the unique positive square root to the polynomial  $P(X) = X^2 + \theta_k^2 X - \theta_k^2$ . (??) is a direct consequence of (11).

Let us prove (10) by induction. It is clear that  $\theta_0 \leq \frac{2}{0+2/\theta_0}$ . Assume that  $\theta_k \leq \frac{2}{k+2/\theta_0}$ . We know that

Data set	Accuracy	Coordinate descent	APPROX	Variable restart APPROX
abalone	$10^{-2}$	0.066	0.045	0.047
$n = 8$	$10^{-6}$	0.211	<b>0.126</b>	0.133
$m = 4,177$	$10^{-10}$	0.404	0.271	<b>0.257</b>
triazines	$10^{-2}$	0.194	<b>0.038</b>	0.060
$n = 60$	$10^{-6}$	$>2.442$	0.989	<b>0.379</b>
$m = 186$	$10^{-10}$	$>3.788$	$>9.850$	<b>1.038</b>
leukemia	$10^{-2}$	<b>0.113</b>	0.174	0.189
$n = 7129$	$10^{-6}$	10.367	11.735	<b>5.132</b>
$m = 72$	$10^{-10}$	97.532	$>435.002$	<b>23.223</b>
blogfeedback	$10^{-2}$	6.732	<b>2.561</b>	3.868
$n = 280$	$10^{-6}$	452.177	63.832	<b>54.962</b>
$m = 52,397$	$10^{-10}$	$>1617.721$	$>1845.049$	<b>208.319</b>
rcv1	$10^{-2}$	<b>2.898</b>	4.381	4.323
$n = 47236$	$10^{-6}$	227.590	108.778	<b>69.616</b>
$m = 20242$	$10^{-10}$	2986.070	943.178	<b>180.206</b>
news20	$10^{-2}$	<b>33</b>	64	67
$n = 1,355,191$	$10^{-6}$	8161	2594	<b>1869</b>
$m = 19,996$	$10^{-10}$	$>36000$	$>82000$	<b>7173</b>

Table 2: Time is seconds to compute the Lasso path over a grid. The  $>$  sign means that the algorithm had not reach the target accuracy on the duality gap after 40,000  $n$  coordinate updates, for at least one  $\lambda_t$  in the grid. We put the number in boldface when an algorithm is at most 50% faster than another.

$P(\theta_{k+1}) = 0$  and that  $P$  is an increasing function on  $[0, +\infty]$ . So we just need to show that  $P\left(\frac{2}{k+1+2/\theta_0}\right) \geq 0$ .

$$P\left(\frac{2}{k+1+2/\theta_0}\right) = \frac{4}{(k+1+2/\theta_0)^2} + \frac{2}{k+1+2/\theta_0}\theta_k^2 - \theta_k^2$$

As  $\theta_k \leq \frac{2}{k+2/\theta_0}$  and  $\frac{2}{k+1+2/\theta_0} - 1 \leq 0$ ,

$$\begin{aligned} P\left(\frac{2}{k+1+2/\theta_0}\right) &\geq \frac{4}{(k+1+2/\theta_0)^2} + \left(\frac{2}{k+1+2/\theta_0} - 1\right) \frac{4}{(k+2/\theta_0)^2} \\ &= \frac{4}{(k+1+2/\theta_0)^2(k+2/\theta_0)^2} \geq 0. \end{aligned}$$

For the other inequality,  $\frac{(2-\theta_0)}{0+(2-\theta_0)/\theta_0} \leq \theta_0$ . We now assume that  $\theta_k \geq \frac{(2-\theta_0)}{k+(2-\theta_0)/\theta_0}$  but that  $\theta_{k+1} < \frac{(2-\theta_0)}{k+1+(2-\theta_0)/\theta_0}$ . Remark that  $(x \mapsto (1-x)/x^2)$  is strictly decreasing for  $x \in (0, 2)$ . Then, using (11), we have

$$\frac{(k+1+(2-\theta_0)/\theta_0)^2}{(2-\theta_0)^2} - \frac{k+1+(2-\theta_0)/\theta_0}{2-\theta_0} < \frac{1-\theta_{k+1}}{\theta_{k+1}^2} \stackrel{(11)}{=} \frac{1}{\theta_k^2} \leq \frac{(k+(2-\theta_0)/\theta_0)^2}{(2-\theta_0)^2}.$$

This is equivalent to

$$(2 - (2 - \theta_0))(k + (2 - \theta_0)/\theta_0) + 1 < (2 - \theta_0)$$

which obviously does not hold for any  $k \geq 0$ . So  $\theta_{k+1} \geq \frac{(2-\theta_0)}{k+1+(2-\theta_0)/\theta_0}$ .  $\square$

*Proof of Theorem 1.* The proof is organised in 4 steps. Firstly, we derive a one-iteration inequality, secondly, we identify conditions under which this inequality is a supermartingale inequality, thirdly, we give a solution to the set of inequalities and finally, we bound the rate we obtain by a simpler formula.

*Step 1: Derive a one-iteration inequality.*

Let us denote  $\hat{F}_k = f(x_k) + \sum_{l=0}^k \gamma_k^l \psi(z_l) \geq F(x_k)$  where  $\gamma_k^l$  are the same constants as defined in [FR15]. By [FR15], for any  $x_* \in \mathcal{X}^*$ ,

$$\mathbf{E}_k[\hat{F}_{k+1} - F^*] \leq (1 - \theta_k)(\hat{F}_k - F^*) + \frac{\theta_k^2}{2\theta_0^2} (\|z_k - x_*\|_v^2 - \mathbf{E}_k[\|z_{k+1} - x_*\|_v^2]) \quad (27)$$

Using

$$x_{k+1} = (1 - \theta_k)x_k + \frac{\theta_k}{\theta_0}z_{k+1} - \frac{\theta_k}{\theta_0}(1 - \theta_0)z_k$$

and the equality

$$\|(1 - \lambda)a + \lambda b\|^2 = (1 - \lambda)\|a\|^2 + \lambda\|b\|^2 - \lambda(1 - \lambda)\|a - b\|^2$$

which is valid for any vectors  $a$  and  $b$  and any  $\lambda \in \mathbb{R}$ , we get, denoting  $x_*$  the unique element of  $\mathcal{X}^*$ ,

$$\begin{aligned} \|x_{k+1} - x_*\|_v^2 &= (1 - \theta_k)\|x_k - x_*\|_v^2 + \frac{\theta_k}{\theta_0}\|z_{k+1} - x_*\|_v^2 - \frac{\theta_k}{\theta_0}(1 - \theta_0)\|z_k - x_*\|_v^2 \\ &\quad + \frac{(1 - \theta_k)\theta_k/\theta_0(1 - \theta_0)}{1 - \theta_k/\theta_0}\|x_k - z_k\|_v^2 - \frac{\theta_k/\theta_0}{1 - \theta_k/\theta_0}\|x_{k+1} - z_{k+1}\|_v^2 \end{aligned} \quad (28)$$

Let us consider nonnegative sequences  $(a_k)$ ,  $(b_k)$ ,  $(c_k)$ ,  $(d_k)$  and  $\sigma_k^K$  and study the quantity

$$\begin{aligned} C_{k+1} &= a_{k+1}(\hat{F}_{k+1} - F^*) + \frac{b_{k+1}}{2}\|x_{k+1} - x_*\|_v^2 + \frac{c_{k+1}}{2}\|z_{k+1} - x_*\|_v^2 \\ &\quad + \frac{d_{k+1}}{2}\|x_{k+1} - z_{k+1}\|_v^2 \end{aligned}$$

By the strong convexity condition of  $F$ , we have:

$$\frac{1}{2}\|x_{k+1} - x_*\|_v^2 \leq \frac{1}{\mu}(F(x_{k+1}) - F^*).$$

Hence, we get for any  $\sigma_{k+1}^K \in [0, 1]$ , (the usefulness of superscript  $K$  will become clear later)

$$\begin{aligned} \mathbf{E}_k[C_{k+1}] &\stackrel{(28)}{\leq} \mathbf{E}_k \left[ \left( a_{k+1} + \frac{b_{k+1}\sigma_{k+1}^K}{\mu} \right) (\hat{F}_{k+1} - F^*) \right. \\ &\quad + \frac{b_{k+1}}{2}(1 - \sigma_{k+1}^K)(1 - \theta_k)\|x_k - x_*\|_v^2 \\ &\quad + (c_{k+1} + b_{k+1}(1 - \sigma_{k+1}^K)\frac{\theta_k}{\theta_0})\frac{1}{2}\|z_{k+1} - x_*\|_v^2 \\ &\quad - \frac{b_{k+1}}{2}(1 - \sigma_{k+1}^K)\frac{\theta_k}{\theta_0}(1 - \theta_0)\|z_k - x_*\|_v^2 \\ &\quad + \frac{b_{k+1}}{2}(1 - \sigma_{k+1}^K)\frac{(1 - \theta_k)\theta_k/\theta_0(1 - \theta_0)}{1 - \theta_k/\theta_0}\|x_k - z_k\|_v^2 \\ &\quad \left. + (d_{k+1} - b_{k+1}(1 - \sigma_{k+1}^K)\frac{\theta_k/\theta_0}{1 - \theta_k/\theta_0})\frac{1}{2}\|x_{k+1} - z_{k+1}\|_v^2 \right] \end{aligned}$$

Applying (27) we get:

$$\begin{aligned}
\mathbf{E}_k[C_{k+1}] &\leq (1 - \theta_k)(a_{k+1} + \frac{b_{k+1}\sigma_{k+1}^K}{\mu})(\hat{F}_k - F^*) \\
&+ \frac{b_{k+1}}{2}(1 - \sigma_{k+1}^K)(1 - \theta_k) \|x_k - x_*\|_v^2 \\
&+ \left( c_{k+1} + b_{k+1}(1 - \sigma_{k+1}^K) \frac{\theta_k}{\theta_0} - \left( a_{k+1} + \frac{b_{k+1}\sigma_{k+1}^K}{\mu} \right) \frac{\theta_k^2}{\theta_0^2} \right) \frac{1}{2} \mathbf{E}_k[\|z_{k+1} - x_*\|_v^2] \\
&\left( \left( a_{k+1} + \frac{b_{k+1}\sigma_{k+1}^K}{\mu} \right) \frac{\theta_k^2}{\theta_0^2} - b_{k+1}(1 - \sigma_{k+1}^K) \frac{\theta_k}{\theta_0} (1 - \theta_0) \right) \frac{1}{2} \|z_k - x_*\|_v^2 \\
&+ \frac{b_{k+1}}{2}(1 - \sigma_{k+1}^K) \frac{(1 - \theta_k)\theta_k/\theta_0(1 - \theta_0)}{1 - \theta_k/\theta_0} \|x_k - z_k\|_v^2 \\
&+ \left( d_{k+1} - b_{k+1}(1 - \sigma_{k+1}^K) \frac{\theta_k/\theta_0}{1 - \theta_k/\theta_0} \right) \frac{1}{2} \mathbf{E}_k[\|x_{k+1} - z_{k+1}\|_v^2]
\end{aligned}$$

*Step 2: Conditions to get a supermartingale.*

In order to have a bound, we need the following constraints on the parameters for  $k \in \{0, \dots, K-1\}$ :

$$a_k \geq (1 - \theta_k)(a_{k+1} + \frac{b_{k+1}\sigma_{k+1}^K}{\mu}) \quad (29)$$

$$b_k \geq b_{k+1}(1 - \sigma_{k+1}^K)(1 - \theta_k) \quad (30)$$

$$c_{k+1} \leq \left( a_{k+1} + \frac{b_{k+1}\sigma_{k+1}^K}{\mu} \right) \frac{\theta_k^2}{\theta_0^2} - b_{k+1}(1 - \sigma_{k+1}^K) \frac{\theta_k}{\theta_0} \quad (31)$$

$$c_k \geq \left( a_{k+1} + \frac{b_{k+1}\sigma_{k+1}^K}{\mu} \right) \frac{\theta_k^2}{\theta_0^2} - b_{k+1}(1 - \sigma_{k+1}^K) \frac{\theta_k}{\theta_0} (1 - \theta_0) \quad (32)$$

$$d_{k+1} \leq b_{k+1}(1 - \sigma_{k+1}^K) \frac{\theta_k/\theta_0}{1 - \theta_k/\theta_0} \quad (33)$$

$$d_k \geq b_{k+1}(1 - \sigma_{k+1}^K) \frac{(1 - \theta_k)\theta_k/\theta_0(1 - \theta_0)}{1 - \theta_k/\theta_0} \quad (34)$$

We also add the constraint

$$a_0 = (1 - \theta_0)(b_0 + c_0) = \frac{1 - \theta_0}{\theta_0^2} \quad (35)$$

in order to get a bound involving  $\Delta(x_0)$ .

If we can find a set of sequences satisfying (29)-(35), then we have

$$a_K(F(x_K) - F^*) + \frac{b_K}{2} \|x_{k+1} - x_*\|_v^2 \leq \Delta(x_0).$$

*Step 3: Explicit solution of the inequalities.*

By analogy to the gradient case, we are looking for such sequences saturating (29)-(32). For  $k \in \{1, \dots, K-1\}$ , equality in (31) and (32) can be fulfilled only if

$$\sigma_k^K = \frac{1 - \frac{\theta_k}{\theta_0} \frac{1 - \theta_0}{1 - \theta_k}}{1 + \frac{\theta_{k-1}}{\theta_0 \mu}}, \quad k \in \{1, \dots, K-1\}$$

For  $k = K$ , we would like to ensure  $c_K = 0$  and  $a_K = (1 - \theta_0)b_K$ . This can be done by taking

$$\sigma_K^K = \frac{1 - \frac{\theta_{K-1}}{\theta_0} (1 - \theta_0)}{1 + \frac{\theta_{K-1}}{\theta_0 \mu}}$$

Having found  $(\sigma_k^K)$ , we can get  $(b_k)$  as a function of  $b_0$  through the equality in (30)

$$b_k = \prod_{l=0}^{k-1} \frac{1}{(1-\theta_l)(1-\sigma_{l+1}^K)} b_0 = \frac{\theta_{-1}^2}{\theta_{k-1}^2} \prod_{l=1}^k \frac{1}{1-\sigma_l^K} b_0,$$

$(a_k)$  as a function of  $b_0$  through the equality in (29)

$$a_k = \frac{\theta_{-1}^2}{\theta_{k-1}^2} a_0 - \sum_{l=1}^k \frac{\sigma_l^K}{\mu} \frac{\theta_{-1}^2}{\theta_{l-1}^2} b_l.$$

Using equality in (31) and (32), we get the relation

$$c_{k+1} = c_k - \frac{\theta_k}{1-\theta_k} b_k$$

and so

$$c_k = c_0 - \sum_{l=0}^{k-1} \frac{\theta_l}{1-\theta_l} b_l$$

This gives us the opportunity to calculate  $b_0$  because

$$c_K = 0 = c_0 - \sum_{l=0}^{K-1} \frac{\theta_l}{1-\theta_l} b_l$$

Hence, we get two equalities:

$$\begin{aligned} c_0 + b_0 &= \frac{1}{\theta_0^2} \\ c_0 &= \sum_{l=0}^{K-1} \frac{\theta_l}{1-\theta_l} \frac{\theta_{-1}^2}{\theta_{l-1}^2} \prod_{j=1}^l \frac{1}{1-\sigma_j^K} b_0 \end{aligned}$$

and so since  $\theta_{l-1}^2 = \theta_l^2/(1-\theta_l)$

$$b_0 = \frac{1}{\theta_0^2} \frac{1}{1 + \sum_{l=0}^{K-1} \frac{\theta_{-1}^2}{\theta_l} \prod_{j=1}^l \frac{1}{1-\sigma_j^K}}$$

Finally, we need to check that there is a feasible sequence  $(d_k)$ . Indeed such a sequence exists because the upper and lower bound satisfy

$$\begin{aligned} b_k(1-\sigma_k^K) \frac{\theta_{k-1}/\theta_0}{1-\theta_{k-1}/\theta_0} &\geq d_k \geq b_{k+1}(1-\sigma_{k+1}^K) \frac{(1-\theta_k)\theta_k/\theta_0(1-\theta_0)}{1-\theta_k/\theta_0} \\ &\stackrel{\text{eq.in(30)}}{\Leftrightarrow} \sigma_k^K \leq 1 - (1-\theta_0) \frac{1/\theta_{k-1} - 1/\theta_0}{1/\theta_k - 1/\theta_0} \end{aligned}$$

which is always true because  $\sigma_k^K \leq \theta_0$  for all  $k$ .

Now that we have the sequence, we can compute the rate as

$$\begin{aligned} \rho_K &= \frac{b_0 + c_0}{b_K} = \frac{1}{\theta_0^2} \frac{\theta_{K-1}^2}{\theta_{-1}^2} \prod_{l=1}^K (1-\sigma_l^K) \theta_0^2 \left( 1 + \sum_{l=0}^{K-1} \frac{\theta_{-1}^2}{\theta_l} \prod_{j=1}^l \frac{1}{1-\sigma_j^K} \right) \\ &= \frac{\theta_{K-1}^2}{\theta_{-1}^2} \left( \prod_{l=1}^K (1-\sigma_l^K) + \sum_{l=0}^{K-1} \frac{\theta_{-1}^2}{\theta_l} \prod_{j=l+1}^K (1-\sigma_j^K) \right) \end{aligned}$$

Hence, we obtain that

$$\mathbf{E}[\Delta(x_K)] \leq \frac{\theta_{K-1}^2}{\theta_{-1}^2} \left( \prod_{l=1}^K (1 - \sigma_l^K) + \sum_{l=0}^{K-1} \frac{\theta_{-1}^2}{\theta_l} \prod_{j=l+1}^K (1 - \sigma_j^K) \right) \Delta(x_0) = \rho_K \Delta(x_0)$$

where  $\theta_{-1}^2 = \frac{\theta_0^2}{1-\theta_0}$  and

$$\sigma_k^K = \frac{1 - \frac{\theta_k - 1 - \theta_0}{\theta_{k-1} 1 - \theta_k}}{1 + \frac{\theta_{k-1}}{\theta_0 \mu}}, \quad k \in \{1, K-1\}$$

$$\sigma_K^K = \frac{1 - \frac{\theta_{K-1}}{\theta_0} (1 - \theta_0)}{1 + \frac{\theta_{K-1}}{\theta_0 \mu}}.$$

*Step 4: bound  $\rho_K$  by induction.*

$$\begin{aligned} \rho_1 &= \frac{\theta_0^2}{\theta_{-1}^2} \left( (1 - \sigma_1^1) + \frac{\theta_{-1}^2}{\theta_0} (1 - \sigma_1^1) \right) = (1 - \theta_0) \frac{\frac{\theta_0}{\theta_0 \mu} + \frac{\theta_0}{\theta_0} (1 - \theta_0)}{1 + \frac{\theta_0}{\theta_0 \mu}} \left( 1 + \frac{\theta_0}{1 - \theta_0} \right) \\ &= \frac{1 + (1 - \theta_0) \mu}{1 + \mu} \leq \frac{1 + (1 - \theta_0) \mu}{1 + \frac{\theta_0^2}{2\theta_0^2} \mu} \end{aligned}$$

Let us now assume that  $\rho_K \leq \frac{1 + (1 - \theta_0) \mu}{1 + \frac{\theta_0^2 \mu}{2\theta_{K-1}^2}}$ .

$$\begin{aligned} \rho_{K+1} &= \frac{\theta_K^2}{\theta_{-1}^2} \left( \prod_{l=1}^{K+1} (1 - \sigma_l^{K+1}) + \sum_{l=0}^K \frac{\theta_{-1}^2}{\theta_l} \prod_{j=l+1}^{K+1} (1 - \sigma_j^{K+1}) \right) \\ &= \frac{\theta_K^2}{\theta_{K-1}^2} \frac{\theta_{K-1}^2}{\theta_{-1}^2} (1 - \sigma_{K+1}^{K+1}) \frac{1 - \sigma_K^{K+1}}{1 - \sigma_K^K} \left( \prod_{l=1}^K (1 - \sigma_l^K) + \sum_{l=0}^{K-1} \frac{\theta_{-1}^2}{\theta_l} \prod_{j=l+1}^K (1 - \sigma_j^K) \right) \\ &\quad + \frac{\theta_{-1}^2}{\theta_K} \frac{1 - \sigma_K^K}{1 - \sigma_K^{K+1}} \\ &= (1 - \theta_K) (1 - \sigma_{K+1}^{K+1}) \frac{1 - \sigma_K^{K+1}}{1 - \sigma_K^K} \left( \rho_K + \frac{\theta_{K-1}^2}{\theta_K} \frac{1 - \sigma_K^K}{1 - \sigma_K^{K+1}} \right) \\ &= (1 - \theta_K) \frac{1 + \frac{\theta_0 \mu}{\theta_K} (1 - \theta_0)}{1 + \frac{\theta_0 \mu}{\theta_K}} \rho_K + \frac{1 + (1 - \theta_0) \mu}{1 + \frac{\theta_0 \mu}{\theta_K}} \theta_K \end{aligned}$$

We now use the induction hypothesis.

$$\begin{aligned} \rho_{K+1} &\leq (1 - \theta_K) \frac{1 + \frac{\theta_0 \mu}{\theta_K} (1 - \theta_0)}{1 + \frac{\theta_0 \mu}{\theta_K}} \frac{1 + (1 - \theta_0) \mu}{1 + \frac{\theta_0^2}{2\theta_{K-1}^2} \mu} + \frac{1 + (1 - \theta_0) \mu}{1 + \frac{\theta_0 \mu}{\theta_K}} \theta_K \\ &\leq \frac{1 + (1 - \theta_0) \mu}{1 + \frac{\theta_0^2 \mu}{2\theta_K^2}} \end{aligned}$$

The last inequality comes from the fact that

$$\begin{aligned}
& \frac{1 + \frac{\theta_0\mu}{\theta_K}(1 - \theta_0)}{1 + \frac{\theta_0\mu}{\theta_K}} \frac{1 - \theta_K}{1 + \frac{\theta_0^2\mu}{2\theta_{K-1}^2}} + \frac{\theta_K}{1 + \frac{\theta_0\mu}{\theta_K}} \leq \frac{1}{1 + \frac{\theta_0^2\mu}{2\theta_K^2}} \\
& \Leftrightarrow (1 - \theta_K)\left(1 + \frac{\theta_0\mu}{\theta_K}(1 - \theta_0)\right)\left(1 + \frac{\theta_0^2\mu}{2\theta_{K-1}^2}\right) + \theta_K\left(1 + \frac{\theta_0\mu}{2\theta_K^2}\right)\left(1 + \frac{\theta_0^2\mu}{2\theta_{K-1}^2}\right) \\
& \leq \left(1 + \frac{\theta_0^2\mu}{2\theta_{K-1}^2}\right)\left(1 + \frac{\theta_0\mu}{\theta_K}\right) \\
& \Leftrightarrow 0 + \mu \left[ (1 - \theta_K) \frac{\theta_0(1 - \theta_0)}{\theta_K} + \frac{(1 - \theta_K)\theta_0^2}{2\theta_K^2} + \frac{\theta_K\theta_0^2}{2\theta_{K-1}^2} + \frac{\theta_K\theta_0^2}{2\theta_K^2} - \frac{\theta_0^2}{2\theta_{K-1}^2} - \frac{\theta_0}{\theta_K} \right] \\
& \quad + \mu^2 \left[ \frac{(1 - \theta_K)\theta_0^3(1 - \theta_0)}{2\theta_K^2\theta_K} + \frac{\theta_K\theta_0^4}{4\theta_{K-1}^2\theta_K^2} - \frac{\theta_0^3}{2\theta_{K-1}^2\theta_K} \right] \leq 0
\end{aligned}$$

and both terms in the brackets are nonpositive for all  $K$ . □

## References

- [AZ17] Zeyuan Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1200–1205. ACM, 2017.
- [AZY16] Zeyuan Allen-Zhu and Yang Yuan. Improved svrg for non-strongly-convex or sum-of-non-convex objectives. In *International conference on machine learning*, pages 1080–1089, 2016.
- [BNPS17] Jérôme Bolte, Trong Phong Nguyen, Juan Peypouquet, and Bruce W. Suter. From error bounds to the complexity of first-order descent methods for convex functions. *Mathematical Programming*, 165(2):471–507, Oct 2017.
- [BS17] Amir Beck and Shimrit Shtern. Linearly convergent away-step conditional gradient for non-strongly convex functions. *Mathematical Programming*, 164(1):1–27, Jul 2017.
- [BT09] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.
- [CL11] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [DL18] Dmitriy Drusvyatskiy and Adrian S. Lewis. Error bounds, quadratic growth, and linear convergence of proximal methods. *Mathematics of Operations Research*, 43(3):919–948, 2018.
- [FHH<sup>+</sup>07] Jerome Friedman, Trevor Hastie, Holger Höfling, Robert Tibshirani, et al. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 1(2):302–332, 2007.
- [FQ16] Olivier Fercoq and Zheng Qu. Restarting accelerated gradient methods with a rough strong convexity estimate. *arXiv preprint arXiv:1609.07358*, 2016.
- [FQ17] Olivier Fercoq and Zheng Qu. Adaptive restart of accelerated gradient methods under local quadratic growth condition. *arXiv preprint arXiv:1709.02300*, 2017.
- [FR13] Olivier Fercoq and Peter Richtárik. Smooth minimization of nonsmooth functions by parallel coordinate descent. *Preprint arXiv:1309.5885*, 2013.

- [FR15] Olivier Fercoq and Peter Richtárik. Accelerated, parallel and proximal coordinate descent. *SIAM Journal on Optimization*, 25(4):1997–2023, 2015.
- [JG15] Tyler Johnson and Carlos Guestrin. Blitz: A principled meta-algorithm for scaling sparse optimization. In *International Conference on Machine Learning*, pages 1171–1179, 2015.
- [KT95] Diethard Klatte and Gisbert Thiere. Error bounds for solutions of linear equations and inequalities. *Zeitschrift für Operations Research*, 41(2):191–214, Jun 1995.
- [LLX14] Qihang Lin, Zhaosong Lu, and Lin Xiao. An accelerated proximal coordinate gradient method. In *Advances in Neural Information Processing Systems*, pages 3059–3067, 2014.
- [LMH15] Hongzhou Lin, Julien Mairal, and Zaid Harchaoui. A universal catalyst for first-order optimization. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3384–3392. Curran Associates, Inc., 2015.
- [LS13] Yin Tat Lee and Aaron Sidford. Efficient accelerated coordinate descent methods and faster algorithms for solving linear systems. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 147–156. IEEE, 2013.
- [LT93] Zhi-Quan Luo and Paul Tseng. Error bounds and convergence analysis of feasible descent methods: a general approach. *Annals of Operations Research*, 46(1):157–178, Mar 1993.
- [LX15] Qihang Lin and Lin Xiao. An adaptive accelerated proximal gradient method and its homotopy continuation for sparse optimization. *Computational Optimization and Applications*, 60(3):633–674, 2015.
- [LY17] Mingrui Liu and Tianbao Yang. Adaptive accelerated gradient converging method under holden error bound condition. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3104–3114. Curran Associates, Inc., 2017.
- [MS17] Tomoya Murata and Taiji Suzuki. Doubly accelerated stochastic variance reduced dual averaging method for regularized empirical risk minimization. In *Advances in Neural Information Processing Systems*, pages 608–617, 2017.
- [Nes83] Yurii Nesterov. A method of solving a convex programming problem with convergence rate  $O(1/k^2)$ . *Soviet Mathematics Doklady*, 27(2):372–376, 1983.
- [Nes12] Yurii Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.
- [Nes13] Yurii Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.
- [NNG18] I. Necoara, Yu. Nesterov, and F. Glineur. Linear convergence of first order methods for non-strongly convex optimization. *Mathematical Programming*, Jan 2018.
- [OC12] Brendan O’Donoghue and Emmanuel Candes. Adaptive restart for accelerated gradient schemes. *Foundations of Computational Mathematics*, pages 1–18, 2012.
- [QR16] Zheng Qu and Peter Richtárik. Coordinate descent with arbitrary sampling II: expected separable overapproximation. *Optimization Methods and Software*, 31(5):858–884, 2016.
- [RD17] Vincent Roulet and Alexandre D’Aspremont. Sharpness, restart and acceleration. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 1119–1129. Curran Associates, Inc., 2017.

- [RT14] Peter Richtárik and Martin Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming* (doi: 10.1007/s10107-012-0614-z), 2014.
- [RT16] Peter Richtárik and Martin Takáč. Parallel coordinate descent methods for big data optimization. *Mathematical Programming*, 156(1-2):433–484, 2016.
- [Tse08] Paul Tseng. On accelerated proximal gradient methods for convex-concave optimization. *Submitted to SIAM Journal on Optimization*, 2008.
- [WL14] Po-Wei Wang and Chih-Jen Lin. Iteration complexity of feasible descent methods for convex optimization. *Journal of Machine Learning Research*, 15:1523–1548, 2014.